# Visualisation de champs scalaires guidée par la topologie

Léo Allemand-Giorgis

UNIVERSITÉ DE GRENOBLE

# THÈSE

Pour obtenir le grade de

## DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques-Informatique**

Arrêté ministériel : 7 août 2006


Présentée par

## Léo ALLEMAND-GIORGIS

Thèse dirigée par **Georges-Pierre BONNEAU**
et codirigée par **Stefanie HAHMANN**

préparée au sein **du Laboratoire Jean Kuntzmann (LJK)**
et de **l'École doctorale EDMSTII**


# Visualisation de champs scalaires guidée par la topologie

Thèse soutenue publiquement le **16 juin 2016**,
devant le jury composé de :

**Géraldine Morin**
Maître de Conférences HDR, INP de Toulouse, Rapporteure
**Jean-Michel Dischler**
Professeur, Université de Strasbourg, Rapporteur
**Julien Tierny**
Chargé de Recherche CNRS, Sorbonne Universites UPMC - LIP6, Examinateur
**Luc Biard**
Professeur, Université de Grenoble-Alpes, Examinateur
**Georges-Pierre Bonneau**
Professeur, Université Grenoble-Alpes, Directeur de thèse
**Stefanie Hahmann**
Professeure, Grenoble-INP, Directrice de thèse

# RÉSUMÉ

Les points critiques d'une fonction scalaire (minima, points col et maxima) sont des caractéristiques importantes permettant de décrire de gros ensembles de données, comme par exemple les données topographiques. L'acquisition de ces données introduit souvent du bruit sur les valeurs. Un grand nombre de points critiques sont créés par le bruit, il est donc important de supprimer ces points critiques pour faire une bonne analyse de ces données. Le complexe de Morse-Smale est un objet mathématique qui est étudié dans le domaine de la Visualisation Scientifique car il permet de simplifier des fonctions scalaires tout en gardant les points critiques les plus importants de la fonction étudiée, ainsi que les liens entre ces points critiques. Nous proposons dans cette thèse une méthode permettant de construire une fonction qui correspond à un complexe de Morse-Smale d'une fonction définie sur $\mathbb{R}^2$ après suppression de paires de points critiques dans celui-ci.

Tout d'abord, nous proposons une méthode qui définit une surface interpolant des valeurs de fonction aux points d'une grille de façon monotone, c'est-à-dire en ne créant pas de point critique. Cette surface est composée d'un ensemble de patchs de Bézier triangulaires cubiques assemblés de telle sorte que la surface soit globalement $C^1$. Nous donnons des conditions suffisantes sur les valeurs de fonction et les valeurs de dérivées partielles aux points de la grille afin que la surface soit croissante dans la direction $(x + y)$. Il n'est pas évident de créer des valeurs de dérivées partielles en chaque point de la grille vérifiant ces conditions. C'est pourquoi nous introduisons deux algorithmes : le premier permet de modifier des valeurs de dérivées partielles données en entrée afin que celles-ci vérifient les conditions et le second calcule des valeurs de dérivées partielles à partir des valeurs de fonctions aux points de la grille.

Ensuite, nous décrivons une méthode de reconstruction de champs scalaires à partir de complexes de Morse-Smale simplifiés. Pour cela, nous commençons par approximer les 1-cellules (les liens entre les points critiques dans le complexe de Morse-Smale , ceux-ci sont décrits par des polylignes) par des courbes composées de courbes de Bézier cubiques. Nous décrivons ensuite comment notre interpolation monotone de valeurs aux points d'une grille est utilisée pour construire des surfaces monotones interpolant les courbes construites précédemment. De plus, nous montrons que la fonction reconstruite contient tout les points critiques du complexe de Morse-Smale simplifié et n'en contient aucun autre.

# ABSTRACT

Critical points of a scalar function (minima, saddle points and maxima) are important features to characterize large scalar datasets, like topographic data. But the acquisition of such datasets introduces noise in the values. Many critical points are caused by the noise, so there is a need to delete these extra critical points. The Morse-Smale complex is a mathematical object which is studied in the domain of Visualization because it allows to simplify scalar functions while keeping the most important critical points of the studied function and the links between them. We propose in this dissertation a method to construct a function which corresponds to a Morse-Smale complex defined on $\mathbb{R}^2$ after the suppression of pairs of critical points.

Firstly, we propose a method which defines a monotone surface (a surface without critical points).This surface interpolates function values at a grid points. Furthermore, it is composed of a set of triangular cubic Bézier patches which define a $C^1$ continuous surface. We give sufficient conditions on the function values at the grid points and on the partial derivatives at the grid points so that the surface is increasing in the $(x + y)$ direction. It is not easy to compute partial derivatives values which respect these conditions. That's why we introduce two algorithms : the first modifies the partial derivatives values on input such that they respect the conditions and the second computes these values from the function values at the grid points.

Then, we describe a reconstruction method of scalar field from simplified Morse-Smale complexes. We begin by approximating the 1-cells of the complex (which are the links between the critical points, described by polylines) by curves composed of cubic Bézier curves. We then describe how our monotone interpolant of values at grid points is used to construct monotone surfaces which interpolate the curves we computed before. Furthermore, we show that the function we compute contains all the critical points of the simplified Morse-Smale complex and has no others.

# PLAN

**Plan** **5**

**1 Introduction** **9**
Contributions . . . 11
Organisation du mémoire . . . 11

**2 Fondements mathématiques** **13**
2.1 Introduction . . . 14
2.1.1 Espaces topologiques . . . 14
2.1.2 CW-Complexes . . . 15
2.2 Complexes de Morse . . . 17
2.2.1 Fonctions de Morse . . . 17
2.2.2 Lignes intégrales . . . 18
2.2.3 Complexes de Morse . . . 20
2.3 Complexes de Morse-Smale . . . 21
2.4 Complexes de Morse-Smale en discret . . . 23
2.4.1 Définition d'une fonction sur un CW-complexe . . . 23
2.4.2 Gradient d'une fonction de Morse discrète . . . 26
2.4.3 Simplification . . . 28
2.5 Problématique . . . 29

**3 État de l'art** **31**
3.1 État de l'art sur l'interpolation et la préservation de la monotonie . . . 32
3.1.1 Interpolation de données . . . 32
3.1.2 Préservation de la monotonie . . . 40
3.1.3 Conclusion . . . 44
3.2 État de l'art sur les complexes de Morse-Smale . . . 44
3.2.1 Calcul des complexes de Morse-Smale . . . 44
3.2.2 Reconstruction par lissages Laplacien itératifs . . . 45

3.2.3 Reconstruction par résolution d'une minimisation sur l'ensemble du maillage . . . 46
3.2.4 Conclusion . . . 49
3.3 Conclusion . . . 50

**4 Interpolation monotone de données sur une grille 53**
4.1 Introduction . . . 54
4.2 Interpolation de Sibson . . . 55
4.2.1 Surfaces de Bézier triangulaires . . . 55
4.2.2 Interpolant de Sibson . . . 57
4.3 Monotonie axiale et diagonale . . . 60
4.3.1 Monotonie axiale . . . 60
4.3.2 Monotonie relâchée . . . 61
4.3.3 Problème à résoudre . . . 62
4.4 Interpolation de Sibson modifiée . . . 62
4.5 Conditions suffisantes de monotonie . . . 66
4.6 Calcul de dérivées partielles admissibles . . . 69
4.6.1 Algorithme 1 . . . 69
4.6.2 Algorithme 2 . . . 75
4.7 Résultats . . . 77
4.8 Conclusion . . . 80

**5 Reconstruction polynomiale par morceaux de champs scalaires à partir de complexes de Morse-Smale simplifiés 83**
5.1 Introduction . . . 84
5.2 Données en entrée . . . 84
5.3 Lissage monotone des 1-cellules . . . 85
5.3.1 Points de jonctions : définition et traitement . . . 86
5.3.2 Lissage des 1-cellules dans le plan $(x, y)$ . . . 89
5.3.3 Lissage monotone du champs scalaire le long des 1-cellules . . . 95
5.4 Interpolation monotone à l'intérieur des 2-cellules . . . 99
5.4.1 Reparamétrisation . . . 100
5.4.2 Champs scalaire monotone à l'intérieur du carré unité et à l'intérieur de la 2-cellule . . . 103
5.5 Cas particuliers . . . 111
5.5.1 Cellules poches . . . 111
5.5.2 Quasi-cellules poches . . . 112
5.6 Résultats . . . 113
5.6.1 Moyens informatiques . . . 113
5.6.2 Visualisations . . . 114
5.6.3 Les données . . . 114
5.7 Conclusion . . . 120

**6 Conclusion 121**
6.1 Bilan . . . 122
6.2 Perspectives . . . 122
6.2.1 Interpolation monotone de données sur une grille . . . 122

6.2.2 Reconstruction polynomiale par morceaux de champs scalaires à partir de complexes de Morse-Smale simplifiés . . . . . 123

**Annexes** **125**

**A Optimisation ayant un objectif en somme de valeurs absolues** **127**

**B Dérivation des coordonnées barycentriques d'un triangle par rapport aux coordonnées *x* et *y*** **129**

**C Intersection de deux courbes de Bézier cubiques par la méthode d'impliticisation** **131**

**Bibliographie** **134**

CHAPITRE

# 1

# INTRODUCTION

Dans cette thèse, nous allons adresser le problème de la visualisation de fonctions scalaires représentant des phénomènes complexes. Pour ce faire, nous introduirons une nouvelle méthode de construction de fonctions simplifiées à partir de complexes de Morse-Smale.

La taille grandissante de la mémoire des ordinateurs entraîne l'accroissement de la taille des données que l'on peut capturer et traiter à partir des phénomènes naturels les plus complexes. Par exemple, champs de compression des fluides, température atmosphère, topographie terrestre. Bien que beaucoup de ces données soient des fonctions scalaires, celles-ci dépendent de tellement de paramètres qu'elles ne sont plus interprétables directement par des êtres humains. Avec ces dernières avancées en acquisition de données est apparu un besoin croissant pour des outils automatiques permettant une analyse plus aisée des données.

Cette aide à l'analyse passe le plus souvent par l'affichage d'images mettant en valeur les informations pertinentes comprises dans les données. Ces méthodes sont communément rassemblées sous le terme de Visualisation scientifique. Parfois ces informations pertinentes sont tirées de données brutes en filtrant le bruit causé par l'acquisition des données. Par exemple en imagerie médicale, les méthodes de morphologies mathématiques permettent aux médecins de visualiser les données 3D issues d'IRM sans les fluctuations accidentelles venues des capteurs.

Cependant les informations pertinentes à mettre en valeur résident souvent dans les propriétés globales de la fonction scalaire, en particulier, sa topologie. Par exemple, dans le cas d'une fonction représentant la topographie d'un terrain, nous souhaitons visualiser où se trouvent les vallées ou encore les pics de ce terrain. Dans ces cas là, la plupart des méthodes utilisées en Visualisation scientifique sont mises en défaut.

Sur le versant mathématiques, l'analyse de la topologie des fonctions scalaires est faite grâce à la théorie de Morse. Cette théorie permet de relier les lignes de niveaux des fonctions scalaires aux comportements des points critiques et du gradient de la fonction. Un exemple d'objet mathématique tiré de la théorie de Morse est le graphe de Reeb. Dans cette thèse, nous allons utiliser les avancées récentes permettant d'appliquer Morse aux fonctions linéaires par morceaux et discrètes, c'est-à-dire à données mesurées en des points discrets de l'espace. Ces avancées (notamment [20]) ont débouché sur l'introduction du *complexe de Morse-Smale* permettant d'avoir une représentation multi-résolution de la topologie d'une fonction scalaire où les caractéristiques topologiques importantes sont gardées dans toutes les résolutions. En pratique, le complexe de Morse-Smale permet de décrire la topologie des lignes de niveaux de la fonction scalaire, mais aussi de partitionner son domaine de définition en régions monotones. Pour une fonction scalaire définie sur une variété à deux dimensions, cette partition comporte des points correspondant aux points critiques de la fonction, des courbes correspondant à des liens entre les points critiques et une partition du domaine de définition de la fonction.

Les complexes de Morse-Smale peuvent être *simplifiés* en supprimant des paires de points critiques liées par une courbe. Ce qui permet d'obtenir la représentation multi-niveaux. Dans le cas des données topographiques, les vallées sont alors les lignes de valeurs minimales. Celles-ci vont alors relier les minima aux points cols. Pour pouvoir analyser ces données, il faut tout d'abord appliquer une série de simplifications pour ne garder que les caractéristiques principales. En effet, la mesure des valeurs de hauteur du terrain engendre du bruit dans les données qui se manifeste par l'apparition de points critiques ayant une faible valeur de différence en hauteur avec les points auxquels ils sont reliés. Les étapes de simplification permettent de filtrer ces points. Ensuite, il nous suffit de récupérer les courbes reliant les minima et les points cols pour avoir les vallées du terrain. De plus, la segmentation du domaine en régions monotones nous permet aussi de récupérer le bassin versant de chaque vallée.

Les complexes de Morse-Smale sont une représentation de haut niveau idéale pour décrire des données et en extraire l'information pertinente, mais ils sont difficilement interprétables tels quels par un être humain. Pour en obtenir une visualisation utilisable, il faut afficher une nouvelle fonction mettant en valeur ces informations. Dans cette thèse, nous allons construire une fonction définie sur un domaine fermé de $\mathbb{R}^2$ correspondant au complexe de Morse-Smale simplifié. Par exemple, dans le cas de la topographie, cette nouvelle fonction n'affichera que les vallées qui ont été jugées pertinentes par la simplification. Trouver une telle fonction demande la résolution de plusieurs problèmes difficiles. Par exemple, la construction d'un interpolant de valeurs données sur une grille 2D sans créer de points critiques. Ce problème met en défaut la plupart des méthodes d'interpolation classiques. Un autre obstacle est la construction d'une fonction complètement monotone sur un domaine ayant un bord de géométrie arbitrairement complexe. Dans cette thèse, nous verrons comment résoudre ces problèmes. Nous verrons également que notre méthode permet l'utilisation d'autres visualisations telles que l'affichage de la représentation du complexe de Morse-Smale, c'est-à-dire une projection des points critiques et des courbes reliant les points critiques sur la représentation de la fonction.

## CONTRIBUTIONS

Dans le cadre de cette thèse, nous avons tout d'abord développé une méthode permettant d'interpoler des valeurs de fonction données sur une grille 2D sans créer de points critiques non désirés. La surface interpolante est $C^1$ et polynomiale par morceaux, elle est composée de surfaces de Bézier triangulaires cubiques. Les contributions à cette partie sont les suivantes :

- Une contrainte de monotonie plus générale que la contrainte standard de croissance le long des directions $x$ et $y$ est proposée.
- En accord avec cette contrainte de monotonie, nous avons introduit une modification de l'interpolant de Sibson cubique.
- Nous avons montré les conditions suffisantes sur les dérivées partielles aux points de la grille pour assurer la monotonie de la fonction interpolante.
- Nous avons développé deux algorithmes pour construire des dérivées partielles vérifiant les conditions.

La deuxième partie de la thèse propose une méthode de reconstruction de fonctions à partir de la connaissance d'une fonction initiale définie sur un domaine dans $\mathbb{R}^2$ et d'un niveau de simplification de son complexe de Morse-Smale. Pour cela, nous utilisons les contributions de la première partie. Nous construisons un champ scalaire qui est une approximation polynomiale par morceaux de la fonction initiale et qui est cohérent avec le complexe de Morse-Smale donné en entrée de la méthode. Être en cohérence avec un complexe de Morse-Smale signifie que les points critiques de la fonction sont uniquement ceux du complexe de Morse-Smale , que les points critiques sont agencés comme dans le complexe de Morse-Smale et qu'à l'intérieur de chaque cellule du complexe de Morse-Smale la fonction est monotone.

## ORGANISATION DU MÉMOIRE

Ce document comprend quatre parties.

Le chapitre 2 présente les fondements mathématiques de la théorie de Morse aux parties 2.1 et 2.2. Les complexes de Morse-Smale sont alors définis ainsi que leurs principales propriétés en section 2.4. Comme nous ne travaillons pas sur des données continues, nous montrons alors comment sont définis les complexes de Morse-Smale pour les fonctions scalaires définies sur

des complexes cellulaires. Dans la section 2.5, nous introduirons plus en détails la problématique de ce travail.

Ensuite, le chapitre 3 propose un état de l'art sur les problématiques d'interpolation de données en préservant la caractéristique de monotonie (section 3.1), puis présente les différents articles ayant pour sujet la construction d'une fonction cohérente avec une fonction initiale et un complexe de Morse-Smale donnés (section 3.2).

Le chapitre 4 propose une méthode d'interpolation de données échantillonnées sur une grille 2D. Notre méthode donne une surface $C^1$ polynomiale par morceaux. La section 2.1 permet d'introduire la problématique et les notations. Ensuite, la section 4.2 introduit les patchs triangulaires de Bézier (4.2.1), puis définit l'interpolant de Sibson (4.2.2). Nous présentons ensuite deux types de monotonie : la monotonie axiale en 4.3.1 et la monotonie diagonale en 4.3.2. Nous présentons ensuite notre méthode en partie 4.4. Nous donnons ensuite des conditions suffisantes pour que la surface reconstruite soit monotone (4.5), puis nous donnons deux algorithmes permettant d'avoir des dérivées aux points de la grille respectant les conditions de monotonie (4.6). Enfin, nous montrons des résultats de notre méthode en section 4.7.

Le chapitre 5 présente notre méthode de reconstruction de fonctions à partir de complexes de Morse-Smale simplifiés. La section 5.2 présente les données que nous avons en entrée de notre algorithme, c'est-à-dire la façon dont est décrite le complexe de Morse-Smale simplifié en entrée. Notre algorithme est composée de deux étapes : la première consiste à lisser les valeurs de la fonction le long des courbes reliant les points critiques et à les lisser dans le plan $(x, y)$ (5.3). La deuxième étape est la reconstruction de la fonction entre les courbes que nous venons de construire (5.4). Ensuite, nous présentons des résultats de notre méthode en section 5.6.

CHAPITRE

# 2

# FONDEMENTS MATHÉMATIQUES

Les complexes de Morse-Smale sont un concept mathématique venant de la topologie algébrique et de la topologie différentielle. Dans ce chapitre, nous allons définir ce que sont les complexes de Morse-Smale en posant les définitions de base.

Tout d'abord, en partie 2.1 nous introduirons les notions de base concernant les espaces sur lesquels sont définis les complexes de Morse-Smale, ce qui nous amène à définir les CW-complexes. Ensuite, dans la partie 2.2, la notion de Complexe de Morse sera introduite. Viendra alors la définition des complexes de Morse-Smale en partie 2.3. En partie 2.4, nous verrons la théorie sur les complexes de Morse-Smale en discret.

## 2.1 INTRODUCTION

La théorie de Morse est une branche de la topologie différentielle qui se base sur l'analyse des lignes de niveaux et des points critiques de fonctions définies sur des variétés différentielles. Les complexes de Morse-Smale viennent de cette théorie. Pour pouvoir les définir, nous allons ici donner les définitions principales en topologie. Tout d'abord, en section 2.1.1, nous rappelons les définitions des espaces topologiques. Ensuite, en section 2.1.2, nous verrons ce que sont les CW-complexes, qui permettent de rajouter une combinatoire entre des parties d'un espace topologique.

### 2.1.1 Espaces topologiques

Les complexes de Morse-Smale sont un cas particulier des CW-complexes. Avant d'aborder ces espaces dans la section suivante, il est nécessaire de rappeler les définitions de base des espaces topologiques.

**Définition 2.1** (Topologie)**.** Soit $E$ un ensemble. Une topologie $T$ sur $E$ est un ensemble de parties de $E$ tel que :

1. $\emptyset \in T$ et $E \in T$
2. Toute réunion quelconque de parties de $T$ est dans $T$ :
   Soit $\{S_i | i \in I\} \subseteq T$, alors $\bigcup_{i \in I} S_i \in T$
3. Toute intersection finie de parties de $T$ est dans $T$ :
   Soit $\{S_i | i \in \{1 \dots n\}\} \subseteq T$, alors $S_1 \cap \dots \cap S_n \in T$

Une topologie permet de décrire la connectivité à l'intérieur d'un ensemble. Un exemple de topologie pouvant être définie sur tout ensemble est la topologie grossière où $T = \{\{\emptyset\}, \{E\}\}$. C'est la topologie la moins fine pouvant être définie sur un ensemble.

Les topologies permettent de définir les notions d'ouverts et de fermés.

**Définition 2.2** (Ouverts et fermés)**.** Soit $E$ un ensemble et $T$ une topologie sur cet ensemble. Les ouverts de $E$ sont les parties $O \in T$. Les ensembles $U = E \setminus O \in E$ sont, quant à eux, les fermés de $E$.

**Définition 2.3** (Espaces topologiques)**.** Soit $E$ un espace et $T$ une topologie sur $E$. Le couple $(E, T)$ est appelé espace topologique.

Un espace topologique est donc un espace sur lequel on a la notion de parties ouvertes. On peut alors définir l'intérieur et la fermeture d'un sous-ensemble de $E$.

**Définition 2.4** (Intérieur et fermeture)**.** Soit $(E, T)$ un espace topologique et un ensemble $A \subseteq E$. L'intérieur de $A$, noté $int(A)$, est l'union de tous les ouverts contenus dans $A$.
La fermeture de $A$, notée $\bar{A}$ est l'intersection de tous les ouverts contenant $A$.

Il est à noter que l'intérieur d'un ensemble $A$ est ouvert alors que sa fermeture est fermée. Grâce à ces définitions, la notion de voisinage peut alors être donnée.

**Définition 2.5** (Voisinage)**.** Soit $(E, T)$ un espace topologique et $e$ un point quelconque de $E$. Tout ensemble $S \subseteq E$ tel que $e \in int(S)$ est un voisinage de $e$.

A partir de la notion de voisinage, on peut alors définir les espaces séparés.

**Définition 2.6** (Espaces séparés ou espace de Hausdorff)**.** Un espace topologique $E$ est dit séparé ou de Hausdorff si pour tout $u, v \in E$ tels que $u \neq v$ il existe un voisinage $U$ de $u$ et un voisinage $V$ de $v$ tels que $U \bigcap V = \emptyset$.

Un espace séparé est donc un espace topologique dans lequel deux points quelconques distincts possèdent des voisinages disjoints.

Une autre définition nécessaire est celle de frontière.

**Définition 2.7** (Frontières)**.** Soit $(E, T)$ un espace topologique et un ensemble $A \subseteq E$. La frontière de $A$, notée $\partial A$, est égale à :

$$\partial A = \overline{A} \cap \overline{E \setminus A}$$

Intuitivement, la frontière d'un ensemble correspond aux points qui sont sur le bord de cet ensemble.

**Définition 2.8** ($d$-variétés topologiques)**.** Un espace topologique $(E, T)$ séparé à base dénombrable est une $d$-variété topologique si pour tout $x \in E$, il existe un voisinage ouvert $U$ de $x$ et un ouvert $V$ de $\mathbb{R}^d$ tels que $U$ et $V$ soient homéomorphes.

Une $d$-variété topologique (appelée aussi variété) est un espace qui est localement homéomorphe à $\mathbb{R}^d$.

Dans la suite, les espaces topologiques seront des $d$-variétés topologiques.

### 2.1.2 CW-Complexes

Les définitions principales sur les espaces sur lesquels se basent les complexes de Morse-Smale ayant été données, nous pouvons maintenant présenter la notion de CW-complexe. Intuitivement, les CW-complexes permettent de structurer les espaces topologiques en sous-parties et de définir une combinatoire sur ces sous-parties.

#### Définitions préliminaires

Nous allons tout d'abord définir les disques et les sphères dans $\mathbb{R}^n$.

**Définition 2.9** (Disques, disques ouverts et sphères)**.** Soit $|\cdot| : \mathbb{R} \to [0, \infty[$ une norme sur $\mathbb{R}^n$. Le disque de dimension $n$ est définie par :

$$D^n = \{x \in \mathbb{R}^n : |x| \leqslant 1\}$$

C'est un sous-espace fermé de $\mathbb{R}^n$. Le disque ouvert est l'intérieur de $D^n$ :

$$int(D^n) = \{x \in \mathbb{R}^n : |x| < 1\}$$

Enfin, le bord de $D^n$ est la sphère de dimension $n-1$, notée $S^{n-1}$ :

$$S^{n-1} = \{x \in \mathbb{R}^n : |x| = 1\}$$

Une fonction est continue sur un espace topologique si la fonction possède une limite en tout point de l'espace topologique de définition.

**Définition 2.10** (Fonctions continues)**.** Soit $E$ et $F$ deux espaces topologiques, $f$ une application de $E$ dans $F$. La fonction $f$ est dite continue au point $a \in E$ si $f(a)$ est une limite de $f$ en ce point.

Le théorème qui suit permet de caractériser la continuité d'une fonction sur un espace topologique.

**Théorème 2.1.** *Soit $E$ et $F$ deux espaces topologiques, $f$ une application de $E$ dans $F$. Les propositions suivantes sont équivalentes :*

- *$f$ est continue en tout point de $E$ ;*
- *pour tout ouvert $O$ de $F$, $f^{-1}(O)$ est un ouvert de $E$ ;*
- *pour tout fermé $U$ de $F$, $f^{-1}(U)$ est un fermé de $E$ ;*
- *pour toute partie $A$ de $E$, $f(\overline{A}) \subseteq \overline{f(A)}$ ;*
- *pour toute partie $B$ de $F$, $\overline{f^{-1}(B)} \subseteq f^{-1}(\overline{B})$.*

Dans la suite, nous allons utiliser des espaces assimilables structurellement à des collections de disques. La définition qui suit définit une comparaison entre espaces topologiques en regardant leurs structures topologiques.

**Définition 2.11** (Homéomorphismes, espaces homéomorphes)**.** Un homéomorphisme $f : X \to Y$ est une fonction bijective telle que $f$ et l'inverse de $f$ soient continues. $X$ et $Y$ ont alors la même structure topologique et sont dits homéomorphes.

Nous pouvons maintenant définir la notion de cellules.

**Définition 2.12** ($n$-cellules, cellules)**.** Une $n$-cellule est un espace homéomorphe au disque ouvert de dimension $n \geqslant 0$. Lorsqu'il n'y a pas d'ambiguïté, on parle de cellule au lieu de $n$-cellule.

### Décompositions cellulaire

Un espace topologique peut être décomposé en un ensemble de cellules.

**Définition 2.13** (Décompositions cellulaire, $n$-squelettes)**.** Soit $(E, T)$ un espace topologique et $\mathscr{C} = \{c_a | a \in I\}$ une famille de sous-espaces disjoints de $E$ telle que chaque $c_a$ soit une cellule et $E = \bigcup_{a \in I} c_a$ . $\mathscr{C}$ est appelé une décomposition cellulaire de $E$.

Le $n$-squelette de $E$ est le sous-espace $E^n = \bigcup_{a \in I : dim(c_a) \leqslant n} c_a$.

Pour tout espace topologique il existe au moins une décomposition cellulaire. En effet, on peut décomposer tout espace en 0-cellules en prenant tout les points de l'espace comme des 0-cellules. De plus, il n'y a pas unicité de la décomposition cellulaire.

Nous pouvons maintenant donner une définition de CW-complexe comme une paire contenant un espace topologique et une décomposition cellulaire.

**Définition 2.14** (CW-complexes)**.** Soit $(E, \mathscr{C})$ un couple où $E$ est un espace séparé et $\mathscr{C}$ une décomposition cellulaire de $E$. Ce couple est un CW-complexe si les trois axiomes suivant sont respectés :

**Fonctions caractéristiques :** Pour toute $n$-cellule $c \in \mathscr{C}$, il existe une fonction $\Phi_c : D^n \to E$ telle que $\Phi_c \,|_{int(D^n)}$: $int(D^n) \to c$ soit un homéomorphisme et $\Phi_c(S^{n-1}) \subseteq E^{n-1}$.

**Fermeture finie :** Pour tout cellule $c \in \mathscr{C}$, $\bar{c}$ intersecte un nombre fini de cellules de $\mathscr{C}$.
**Topologie Faible :** Un sous-ensemble $A \subseteq E$ est fermé si et seulement si pour tout $c \in \mathscr{C}$, $A \bigcap \bar{c}$ est fermé.

Le premier axiome permet d'identifier le bord de chaque cellule à une union de cellules de dimension inférieure. Le deuxième axiome implique que cette union est finie. Le nom de cet objet est tiré des noms anglais des deuxième et troisième axiomes qui sont respectivement "Closure finitness" et "Weak topology".

Une notion importante permet de relier les cellules aux cellules qui les bordent dans le CW-complexe.

**Définition 2.15** (Face, co-face)**.** Soit $\alpha$ une cellule d'un CW-complexe $K$ et $\dot{\alpha} = \alpha \setminus int(\alpha)$ le bord de $\alpha$. Les cellules $\beta \in K \cap \dot{\alpha}$ sont appelées les faces de $\alpha$. Cette relation se note $\beta < \alpha$.

Soit $\beta$ une face de $\alpha$, alors $\alpha$ est appelée la co-face de $\beta$.

La notion de face et de co-face est donc utile pour analyser les relations dites d'adjacence entre cellules.

## 2.2 Complexes de Morse

Pour pouvoir définir les complexes de Morse-Smale , il est important de définir d'abord les complexes de Morse. Tout d'abord, nous allons voir les fonctions de Morse en section 2.2.1. Ensuite, en section 2.2.2, nous définirons les lignes intégrales. Finalement, nous verrons les complexes de Morse en section 2.2.3.

### 2.2.1 Fonctions de Morse

Dans la suite, $f$ est une fonction définie sur une variété $M$ et à valeurs dans $\mathbb{R}$, au moins $\mathscr{C}^2$. Nous allons d'abord voir la notion de points critiques puis ce qu'est une fonction de Morse.

#### Points critiques

À partir de la fonction $f$, un champ de vecteurs appelé gradient (noté $\bigtriangledown$) peut être calculé. En chaque point du domaine de définition de la fonction, le vecteur défini en ce point est dirigé vers la direction de pente la plus forte. En certains points, cette direction n'existe pas, le vecteur correspondant au gradient en ce point est donc nul. C'est alors un point critique de la fonction $f$.

**Définition 2.16** (Points critiques)**.** Un point $p \in M$ est un point critique de $f$ lorsque $\bigtriangledown f(p) = 0$.

Un point non critique est dit ordinaire.

**Définition 2.17** (Points ordinaires)**.** Un point $p \in M$ est un point ordinaire de $f$ si $p$ n'est pas un point critique.

À partir de la Hessienne de la fonction calculée en un point critique, on peut distinguer deux catégories de points critiques.

**Définition 2.18** (Points critiques dégénérés/non-dégénérés)**.** Un point critique $p \in M$ est dit dégénéré lorsque la matrice Hessienne $H_f$ de $f$ en $p$ est dégénérée, i.e. $det(H_f(p)) = 0$. Il est dit non-dégénéré dans le cas contraire.

Nous verrons par la suite que la Hessienne calculée en un point critique non dégénéré permet de connaître le type du point critique étudié. Dans le cas d'un point critique dégénéré, la Hessienne ne permet pas de connaître ce type.

### Fonctions de Morse

Les fonctions de Morse sont définies par les propriétés vérifiées par leurs points critiques.

**Définition 2.19** (Fonctions de Morse)**.** Une fonction $f : M \to \mathbb{R}$ de classe au moins $C^2$ est une fonction de Morse si tous ses points critiques sont non-dégénérés.

Les fonctions de Morse peuvent être approximées par un polynôme quadratique au voisinage de leurs points critiques.

**Théorème 2.2** (Lemme de Morse)**.** *Soit $f : M \to \mathbb{R}$ une fonction de Morse avec $dim(M) = n$ et $p$ un point critique de $f$. Il existe un système de coordonnées $(x_1, \ldots, x_n)$ au voisinage de $p$ avec $p$ comme origine tel que pour $x$ dans le voisinage de $p$, la fonction $f$ s'écrit :*

$$f(x) = f(p) - \sum_{i=1}^{\lambda} x_i^2 + \sum_{i=\lambda+1}^{n} x_i^2.$$

*Cette écriture est appelée la forme standard de $f$ en $p$.*

Ce lemme fait apparaître l'entier $\lambda$ qui permet de classer les points critiques en différents types.

**Définition 2.20** (Indice)**.** Soit $f : M \to \mathbb{R}$ une fonction de Morse avec $dim(M) = n$ et $p$ un point critique de $f$. Soit $f(x) = f(p) - \sum_{i=1}^{\lambda} x_i^2 + \sum_{i=\lambda+1}^{n} x_i^2$ la forme standard de $f$ en $p$. $\lambda$ est appelé l'indice de $p$.

L'indice d'un point critique $p$ d'une fonction $f$ correspond au nombre de valeurs propres négatives de la Hessienne de $f$. Il y a trois types de points critiques :

- Si l'indice de $p$ est 0, alors $p$ est un minimum.
- Si l'indice $\lambda$ de $p$ est compris entre 1 et $n-1$, alors $p$ est un $\lambda$-point col.
- Si l'indice de $p$ est $n$, alors c'est un maximum.

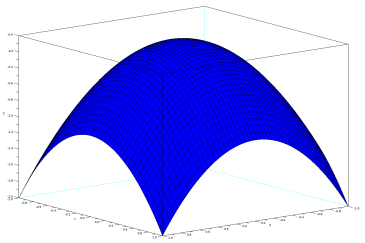
Dans le cas où le domaine de définition de la fonction est de dimension 2, l'indice ne peut être que 0, 1 ou 2. Il n'y a alors que trois types de points critiques : les minima, les maxima et les 1-points col. Dans ce cas, nous appellerons points col les 1-points col. La figure 2.1 montre un exemple pour chaque type de point critique.

## 2.2.2 Lignes intégrales

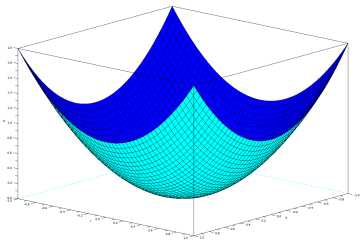
Un concept très important pour la suite est celui de ligne intégrale.

**Définition 2.21** (Lignes intégrales)**.** Soit $f : D \subset \mathbb{R}^n \to \mathbb{R}$, avec $D$ une $n$-variété topologique. Une courbe $l : \mathbb{R} \to D \in \mathbb{R}^n$ est une ligne intégrale de $f$ si pour tout $t \in \mathbb{R}$ on a $\frac{\partial}{\partial t} l(t) = \bigtriangledown f(l(t))$.

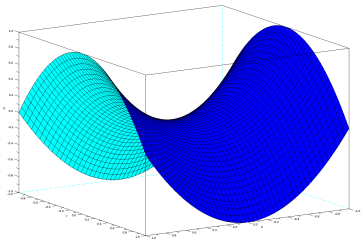
Une ligne intégrale est une courbe qui suit le gradient de la fonction $f$. Plus précisément, pour tout $t \in \mathbb{R}$, la tangente à la courbe $l$ en $t$ est le gradient de $f$ au point $l(t)$. Il est à noter que pour une condition initiale $x_0 \in D$, la ligne intégrale $l$ vérifiant les conditions données à la définition 2.21 et passant par $x_0$ au temps $t_0 \in \mathbb{R}$ est la solution maximale de l'équation différentielle définissant la ligne intégrale. En utilisant le théorème de Cauchy-Lipschitz, on peut alors montrer le théorème qui suit.



**(a)** *Maximum*



**(b)** *Minimum*



**(c)** *Point col*

**FIGURE 2.1 –** *Les trois différents types de points critiques pour les fonctions définies sur $D \subset \mathbb{R}^2$ et leur formes standard aux voisinages des points critiques.*

**Théorème 2.3** (Propriétés des lignes intégrales)**.** *Les lignes intégrales ont les propriétés suivantes :*

1. *Il existe une seule et unique courbe intégrale passant par un point du domaine de $f$.*
2. *$f \circ l$ est croissant.*

En résumé, les lignes intégrales suivent le gradient de $f$, et pour $t$ croissant, alors la valeur de $f$ le long d'une ligne intégrale va aussi être croissante.

Pour chaque ligne intégrale, on peut regarder le comportement aux limites pour le paramètre $t \in \mathbb{R}$.

**Définition 2.22** (Origine et destination d'une ligne intégrale)**.** L'origine $(org)$ d'une ligne intégrale $l : \mathbb{R} \to D \subset \mathbb{R}^n$ est la limite $\lim_{t \to -\infty} l(t) \in D$. De même, la destination $(dest)$ de cette ligne intégrale est la limite $\lim_{t \to +\infty} l(t) \in D$.

L'origine et la destination des lignes intégrales sont des points spécifiques du domaine de définition de la fonction. Ces points peuvent alors être reliés aux points critiques de la fonction.

**Proposition 2.1.** *L'origine et la destination d'une ligne intégrale sont des points critiques de $f$.*

De plus, comme $f \circ l$ est croissant, l'indice de l'origine est donc strictement inférieur à celui de la destination.

### 2.2.3 Complexes de Morse

**Notation.** *Soit $f$ une fonction de $E$ dans $F$. Nous noterons $Im(f) \subseteq F$ l'image de la fonction $f$ :*

$$Im(f) = \{y \in F \mid (\exists x \in E),\ f(x) = y\}$$

Nous avons vu à la partie 2.2.2 qu'en chaque point du domaine de définition passe une unique ligne intégrale qui a pour origine et pour destination des points critiques de $f$. On peut alors classifier les points du domaine en fonction de la destination de la ligne intégrale qui passe par ce point.

**Définition 2.23** (Variétés descendantes)**.** Soit $p$ un point critique de $f : M \to \mathbb{R}$. La variété descendante de $p$ est $D(p) = \{p\} \cup \{x \in M \mid (\exists l \in L),\ x \in Im(l)$ et $dest(l) = p\}$ avec $L$ l'ensemble des lignes intégrales de la fonction $f$.

Une variété descendante est donc une variété contenant un point critique $p$ ainsi que tout les points de $M$ étant dans l'image d'une ligne intégrale ayant pour destination le point $p$. De plus, une variété ascendante est une cellule.

**Proposition 2.2.** *Soit $f : M^n \to \mathbb{R}$ une fonction de Morse et $p$ un point critique d'indice $\lambda$, alors la variété descendante $D(p)$ est une $\lambda$-cellule.*

Cette proposition nous indique donc que l'on connaît la dimension de la cellule correspondante à la variété descendante d'un point critique à partir de la connaissance de l'indice du point critique. L'ensemble des variétés ascendantes d'une fonction de Morse peut donc être considéré comme un CW-complexe.

**Proposition 2.3** (Complexes de Morse)**.** *Soit $f$ une fonction de Morse. L'ensemble des variétés descendantes de $f$ forme un CW-complexe appelé complexe de Morse.*

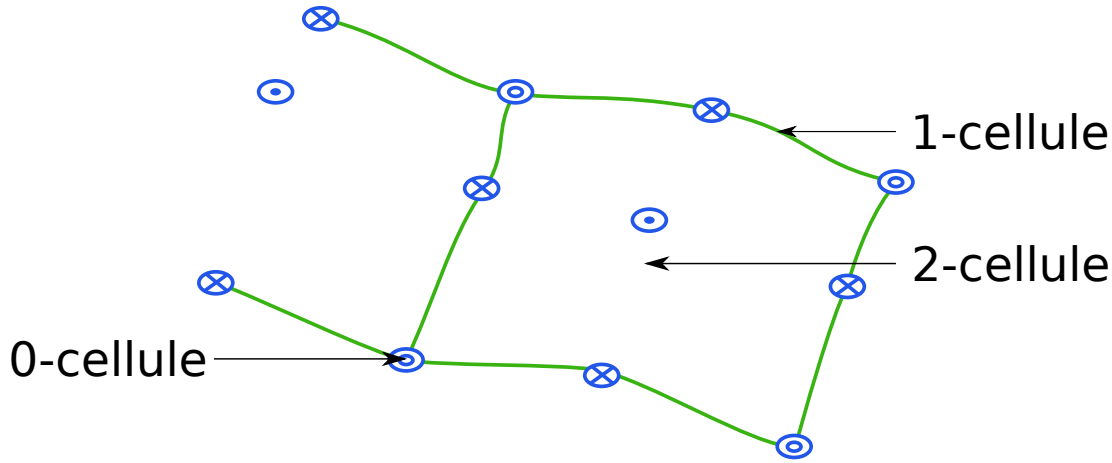



**FIGURE 2.2 –** *Exemple de complexe de Morse. Les lignes vertes décomposent le domaine en cellules : les morceaux de surfaces sont les 2-cellules, les lignes vertes sont les 1-cellules et les minima (cercles vide) sont les 0-cellules. Les différents points critiques sont indiqués : les maxima sont les points, les points cols sont les croix et les cercles sont les minima.*

La figure 2.2 montre la décomposition du domaine d'une fonction en un complexe de Morse. Les 1-cellules sont les lignes vertes qui relient les minima aux points col et les 0-cellules sont les minima. Quant aux 2-cellules, ce sont des morceaux de surfaces autour des

maxima. Les points du domaine sont donc classifiés en fonction de la destination de la ligne intégrale qui les traverse.

## 2.3 Complexes de Morse-Smale

De même qu'à la partie 2.2.3, nous allons découper le domaine de définition d'une fonction $f$ en un complexe cellulaire. Pour cela, nous allons définir les variétés ascendantes.

**Définition 2.24** (Variétés ascendantes)**.** Soit $p$ un point critique de $f : M^n \to \mathbb{R}$. La variété ascendante de $p$ (notée $A(p)$) est la variété descendante de $p$ pour la fonction $-f$. De plus, si $p$ est d'indice $\lambda$, alors $A(p)$ est une $(n-\lambda)$-cellule.

On peut alors calculer un complexe composé des variétés ascendantes de la fonction. Ce complexe correspond au complexe de Morse de la fonction $-f$.

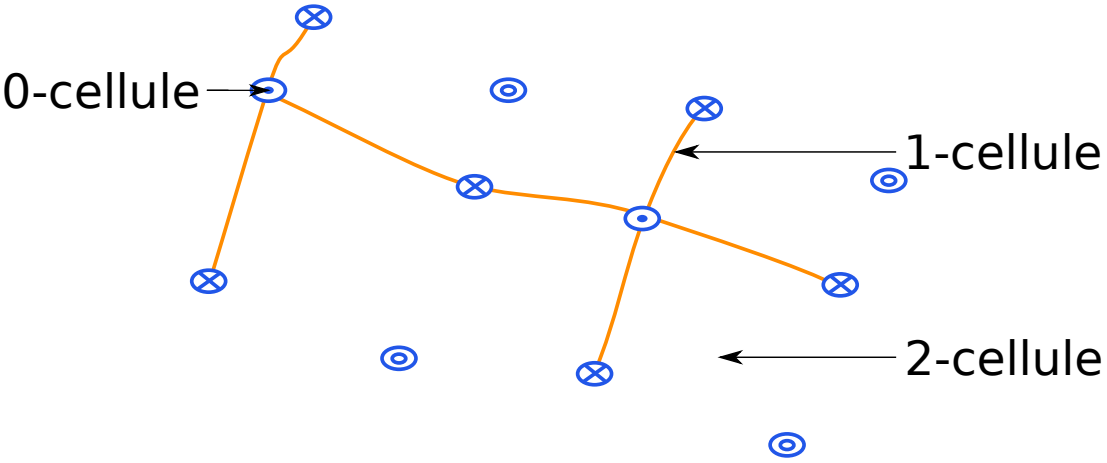



**Figure 2.3 –** *CW-complexe ayant pour cellules les variétés ascendantes de la fonction. Ce complexe correspond au complexe de Morse de la fonction $-f$.*

De même que pour les complexes de Morse, l'ensemble des variétés ascendantes forme un complexe. Un exemple est montré à la figure 2.3. Les 1-cellules sont les lignes orange reliant les maxima et les points col, les 2-cellules sont les morceaux de surface autour des minima et les 0-cellules sont les minima. Les variétés ascendantes partitionnent donc le domaine de définition de la fonction en fonction de l'origine des lignes intégrales.

Pour arriver au concept de complexe de Morse-Smale, il faut intersecter le complexe de Morse de $f$ et celui des variétés ascendantes de $f$.

**Définition 2.25** (Intersections transverses)**.** Soit deux sous-variétés $X$ et $Y$ de la variété $M$. $X$ et $Y$ sont dits transverses si pour tout point $x \in X \cap Y$ :

$$TX_x + TY_x = \{u + v \mid u \in TX_x,\ v \in TY_x\} = TM_x$$

avec $TX_x$, $TY_x$ et $TM_x$ les espaces tangents en $x$ respectivement de $X$, $Y$, et $M$.

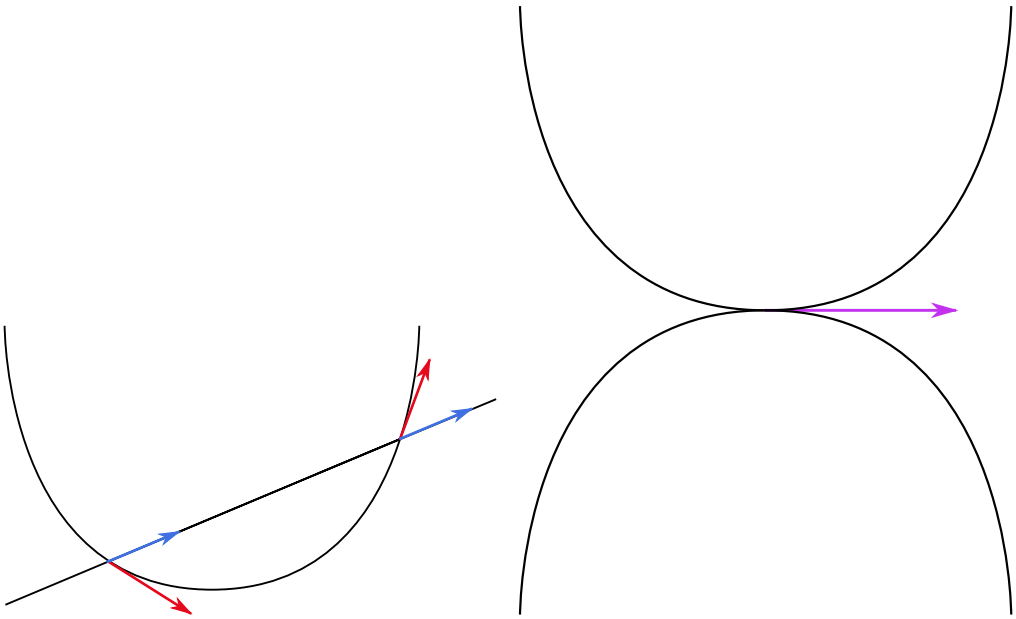
On peut alors déduire des informations sur l'intersection en fonction des dimensions des espaces.

**Proposition 2.4.** *Soit $X$ et $Y$ deux sous-variétés de $M$ avec $dim(X) = a$, $dim(Y) = b$ et $dim(M) = n$. On a alors :*

- *Si $a + b < n$, alors il y a intersection transverse seulement si $X \cap Y = \emptyset$*
- *Si $a + b = n$ et si pour tout $x \in X \cap Y$, $TX_x + TY_x = TM_x$, alors l'intersection est un point.*

Regardons comment cela se passe avec $dim(M) = 2$ :

- $dim(X) = 0$ **:**
  - $dim(Y) = 0$ : L'intersection est transverse si et seulement si $X \neq Y$.
  - $dim(Y) = 1$ : $X$ et $Y$ sont transverses si et seulement si $X \cap Y = \emptyset$.
  - $dim(Y) = 2$ : Les deux sous-espaces sont transverses.
- $dim(X) = 1$ **:**
  - $dim(Y) = 1$ : Pour savoir si les deux sous-variétés sont transverses, il faut vérifier la propriété de la définition 2.25. Un exemple de deux courbes étant transverses et un autre avec des courbes non transverses est à la figure 2.4.
  - $dim(Y) = 2$ : Les deux sous-variétés sont transverses.
- $dim(X) = 2$ **:**
  - $dim(Y) = 2$ : $X$ et $Y$ sont transverses.



**(a)** *Intersection transverse* **(b)** *Intersection non transverse*

**FIGURE 2.4 –** *Intersection de deux courbes dans $\mathbb{R}^2$. L'une des intersections est transverse car les tangentes aux intersections n'ont pas la même direction. Dans le deuxième cas, les tangentes ont la même direction, la somme des deux espaces tangents est alors égale à une droite et non à $\mathbb{R}^2$, l'intersection n'est donc pas transverse.*

Nous pouvons alors définir les fonctions de Morse-Smale :

**Définition 2.26** (Fonctions de Morse-Smale)**.** Une fonction de Morse est appelée fonction de Morse-Smale lorsque ses variétés ascendantes et ses variétés descendantes sont transverses.

De même qu'avec les fonctions de Morse, à partir des fonctions de Morse-Smale, on peut construire un complexe appelé complexe de Morse-Smale.

**Définition 2.27** (Complexes de Morse-Smale)**.** Soit $f$ une fonction de Morse-Smale. Le complexe de Morse-Smale de $f$ est l'intersection entre le complexe de Morse de $f$ et le complexe de Morse de $-f$.

Dans le cas où la fonction est définie sur une variété de dimension 2, nous avons vu que l'intersection de deux courbes peut poser problème. Lorsque la fonction est une fonction de Morse-Smale, alors toutes les 1-cellules sont transverses deux à deux puisqu'elles ne s'intersectent pas par la propriété d'unicité des lignes intégrales.

Le complexe de Morse-Smale partitionne le domaine de définition de la fonction $f$ à partir de l'origine et de la destination des lignes intégrales, c'est-à-dire, que pour chaque point d'une même cellule, toutes les lignes intégrales ont la même origine et la même destination. La figure 2.5 montre un exemple de complexe de Morse-Smale sur une fonction simple.

En utilisant le fait qu'une fonction est croissante le long d'une ligne intégrale, nous pouvons déduire certaines propriétés sur les complexes de Morse-Smale de fonctions définies sur des variétés de $\mathbb{R}^2$.

**Théorème 2.4.** *Soit $f : M \subset \mathbb{R}^2 \to \mathbb{R}$ une fonction de Morse-Smale et $(M, \mathscr{C})$ le complexe de Morse-Smale de $f$. Alors :*

- *Chaque 2-cellule de $\mathscr{C}$ possède quatre facettes (i.e. 1-cellules).*
- *Chaque 2-cellule de $\mathscr{C}$ possède quatre faces de degré 0 (i.e. 0-cellules). De plus, cet ensemble est composé de points critiques qui sont toujours un minimum, un maximum et deux points cols.*
- *Toute cellule de degré strictement supérieur à 0 est monotone, c'est-à-dire que la restriction de la fonction $f$ à la cellule $c$ de degré $n > 0$ ne contient pas de point critique en son intérieur.*

**Remarque :** Par abus de langage, nous allons écrire qu'une cellule d'un complexe de Morse-Smale est monotone ou qu'elle ne possède pas de point critique à la place de dire que la restriction de la fonction à la cellule ne possède pas de point critique.

Il existe un cas particulier que nous appellerons cellule poche dans la suite. Il apparaît lorsque les deux points cols sont fusionnés. Une image d'un tel cas est montré à l'image 2.6.

## 2.4 Complexes de Morse-Smale en discret

Nous avons présentée la théorie sur les complexes de Morse-Smale sur des espaces continus. En pratique, il y a deux façons principales de les calculer : la première est d'approximer les fonctions $C^2$ par des fonctions linéaire par morceaux et la seconde est de travailler à partir de la théorie de Morse discrète. Dans cette partie, nous allons voir cette dernière. Elle a été développé par Robin Forman dans l'article [20]. Le but de cette théorie est de refaire la théorie de Morse qui est valable pour des fonctions définies sur des variétés continues, pour qu'elle puisse s'appliquer à des fonctions définies sur des CW-complexes.

### 2.4.1 Définition d'une fonction sur un CW-complexe

En discret, le domaine de définition des fonctions de Morse sera un CW-complexe que nous supposerons fini et régulier.

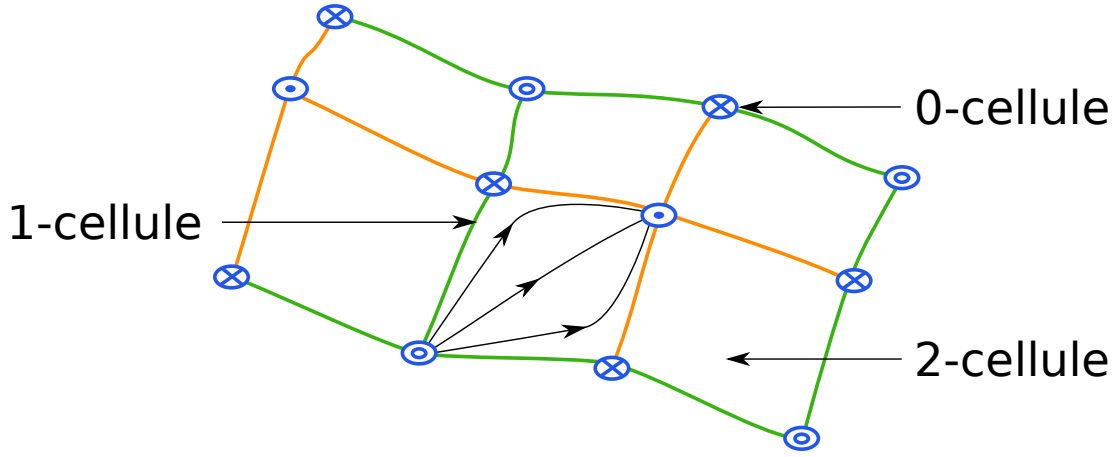



**FIGURE 2.5 –** *Exemple d'un complexe de Morse-Smale. C'est l'intersection entre le complexe de Morse (lignes vertes) et le complexe formé des variétés ascendantes (lignes orange). C'est donc l'intersection entre la figure 2.2 et la figure 2.3. À l'intérieur d'une cellule, les lignes intégrales ont toutes la même origine et la même destination. Trois exemples de lignes intégrales sont indiqués sur la figure par des lignes avec des flèches en leur milieu.*

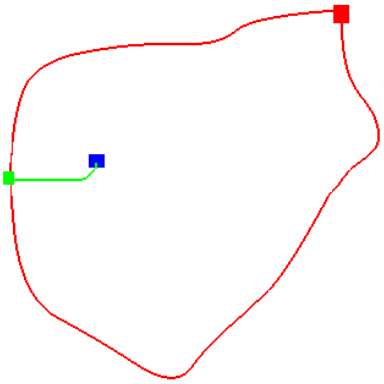


**FIGURE 2.6 –** *Exemple d'une cellule poche. Les deux points cols sont fusionnés, ainsi que les deux 1-cellules joignant le point col au minimum. Ceci est une cellule valide d'un complexe de Morse-Smale. Le point rouge représente un maximum, le point vert les point cols fusionnés et le point bleu le minimum.*

**Définition 2.28** (CW-complexes réguliers)**.** Soit $\alpha$ et $\beta$ deux cellules d'un CW-complexe $K$ telles que $dim(\alpha) - dim(\beta) = 2$. $K$ est régulier s'il existe exactement deux faces $\gamma_1$ et $\gamma_2$ de $\alpha$ telles que $\beta < \gamma_1 < \alpha$ et $\beta < \gamma_2 < \alpha$.

Une fonction $f : K \to \mathbb{R}$ est une fonction définie sur un CW-complexe. Cela veut dire que pour chaque cellule $\alpha$ du complexe une valeur $f(\alpha) \in \mathbb{R}$ lui est associée. Comme dans le cas continu, il faut définir ce qu'est une fonction de Morse.

**Notation.** *On note $\alpha^d \in K$ une cellule $\alpha$ de dimension $d$ d'un complexe $K$.*

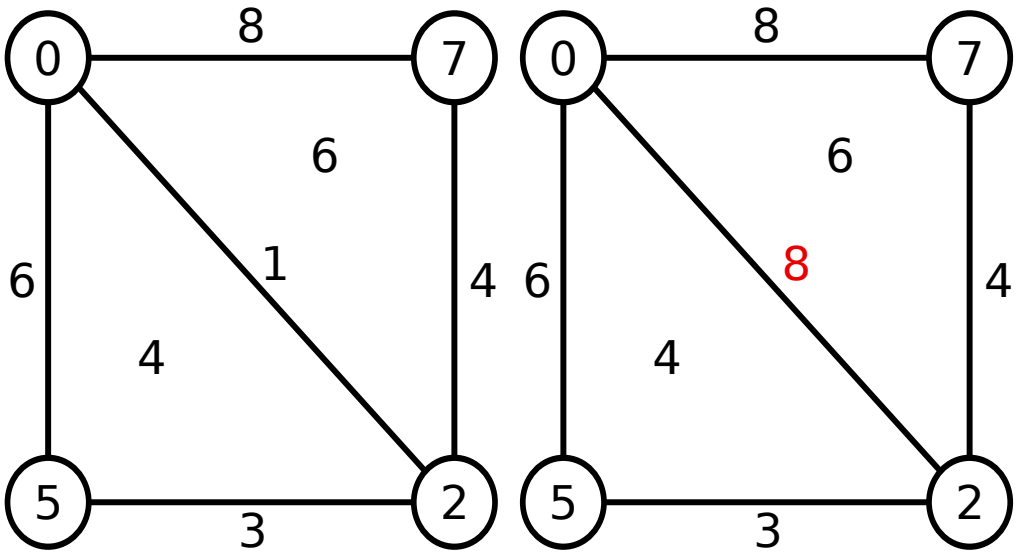
**Définition 2.29** (Fonctions de Morse discrètes)**.** Soit $K$ un complexe cellulaire fini et régulier représentant le domaine de définition de $f : K \to \mathbb{R}$. $f$ est une fonction de Morse discrète si :

$$(\forall \alpha^d \in K)\ \mathrm{Card}\{\beta^{d+1} \in K \mid \alpha < \beta \text{ et } f(\beta) \leqslant f(\alpha)\} \leqslant 1$$

et

$$(\forall \alpha^d \in K)\ \mathrm{Card}\{\gamma^{d-1} \in K \mid \gamma < \alpha \text{ et } f(\gamma) \geqslant f(\alpha)\} \leqslant 1$$

Une fonction de Morse discrète est donc une fonction qui assigne des valeurs plus hautes aux cellules de degrés supérieures. Pour une cellule, il ne peut exister qu'une seule cellule de dimension inférieure ayant une valeur plus haute et une cellule de dimension supérieure ayant une valeur inférieure. La figure 2.7 montre deux exemples de fonctions définies sur un complexe. La figure 2.7a est une fonction de Morse car en toute cellule, les inégalités sont respectées. Quant à la figure 2.7b, la valeur le long de l'arête diagonale a une valeur de fonction plus élevée que les deux triangles adjacents. Or les triangles sont des cellules de dimension 2 alors que l'arête est de dimension 1. Si la fonction était une fonction de Morse, il ne pourrait exister au plus qu'un seul triangle ayant une valeur inférieur à la valeur sur l'arête.




**(a)** *Fonction de Morse discrète.*

**(b)** *Fonction discrète n'étant pas une fonction de Morse.*

**FIGURE 2.7 –** *Deux fonction définies sur un complexe. L'image 2.7a montre une fonction qui respecte les conditions de la définition des fonctions de Morse discrètes. L'image 2.7b possède une arête ne respectant pas les conditions. En effet, la valeur de la fonction sur cette arête est plus grande que la valeur de la fonction sur les deux faces adjacentes.*

De la figure 2.7a, nous pouvons constater que dans tout les cas, l'un des ensembles définis à la définition 2.29 est vide. Ceci est évident pour les cellules de dimensions 0 et 2 puisqu'il n'y a pas de cellules de dimensions inférieures à 0 et que le complexe ne possède pas d'éléments de dimensions 3. Mais cela est aussi le cas pour les cellules de dimension 1. Regardons, par exemple, l'arête diagonale ayant pour valeur 1. Ses faces de dimensions 2 ont toutes une valeur de fonction supérieure, et seule une cellule de dimension 0 a une valeur de fonction supérieure. Ceci est une propriété des fonctions de Morse discrètes qui servira dans la partie 2.4.2.

**Proposition 2.5.** *Soit $K$ un complexe fini et régulier et $f$ une fonction de Morse définie sur celui-ci. Alors, pour toute cellule $\alpha$, au moins une des propriétés suivantes est vérifiée :*

$$\mathrm{Card}\{\beta^{d+1} \in K \mid \alpha < \beta \text{ et } f(\beta) \leqslant f(\alpha)\} = 0$$

*ou*

$$\mathrm{Card}\{\gamma^{d-1} \in K \mid \gamma < \alpha \text{ et } f(\gamma) \geqslant f(\alpha)\} = 0$$

Les deux ensembles définis précédemment peuvent être de cardinaux nuls, comme cela peut être vu à l'exemple de la figure 2.7a pour le point en haut à gauche ayant pour valeur de fonction 0. Ces points sont les points critiques de la fonction.

**Définition 2.30** (Cellules critiques)**.** Soit $K$ un complexe fini et régulier et $f$ une fonction de Morse définie sur celui-ci. Une cellule $\alpha \in K$ est dite critique si :

$$\mathrm{Card}\{\beta^{d+1} \in K \mid \alpha < \beta \text{ et } f(\beta) \leqslant f(\alpha)\} = 0$$

et

$$\mathrm{Card}\{\gamma^{d-1} \in K \mid \gamma < \alpha \text{ et } f(\gamma) \geqslant f(\alpha)\} = 0$$

Pour connaître l'indice d'une cellule critique d'une fonction de Morse discrète, il suffit de regarder la dimension de la cellule. En effet, si une $d$-cellule est une cellule critique, alors c'est une cellule critique d'indice $d$. Par exemple, une cellule de dimension 0 étant critique, n'est entourée que de cellules de dimensions 1 ayant des valeurs de fonction strictement supérieures à la fonction en cette cellule critique. Elle correspond donc à un minimum local de la fonction. Dans la fonction représentée à la figure 2.7a, il y a une unique cellule critique qui est la 0-cellule en haut à gauche ayant comme valeur 0. Celle-ci est un minimum. En dimension 2, les cellules critiques d'indice 1 (points cols) sont des arêtes et celles d'indice 2 (maxima) sont des faces.

### 2.4.2 Gradient d'une fonction de Morse discrète

À partir d'une fonction de Morse discrète, nous allons construire un champ de vecteurs correspondant à son champ de gradient. Pour cela, il faut donner une définition de la notion de vecteur dans ce cadre.

**Définition 2.31** (Vecteurs discrets)**.** Soit $K$ un complexe de dimension $n$ et $\alpha$ une $d$-cellule et $\beta$ une $(d+1)$-cellule de $K$, avec $d \in \{0, \ldots, n-1\}$. Un vecteur discret est une paire de cellules $\{\alpha, \beta\}$.

À partir de là, nous pouvons définir un champs de vecteur sur un complexe.

**Définition 2.32** (Champs de vecteurs discrets)**.** Soit $K$ un complexe. Un champ de vecteurs discrets $V$ sur $K$ est une collection de vecteurs discrets de $K$ telle que chaque cellule de $K$ soit dans au plus un vecteur de cette collection.

Pour construire un champ de gradient à partir d'une fonction de Morse discrète, on définit un vecteur contenant $\alpha^{(d)}$ et $\beta^{(d+1)}$ si et seulement si $\alpha$ est une face de $\beta$ et $f(\alpha) \geqslant f(\beta)$. Ceci définit bien un champ de vecteurs discrets puisque les propositions (2.29) et (2.5) impliquent qu'une cellule appartient à au plus un vecteur. Les cellules n'appartenant à aucune paire définissant un vecteur sont alors des cellules critiques.

À partir de la fonction de Morse de l'image 2.7a, il est alors possible de construire un champ de vecteurs discrets en utilisant la construction donnée ci-dessus. Ce champ de vecteurs peut être vu à l'image 2.8. Nous retrouvons que la seule cellule critique est la 0-cellule en haut à gauche ayant pour valeur 0 qui est un minimum.

Il faut ensuite définir un équivalent aux lignes intégrales. Cela s'appelle un $V$-chemin. C'est un ensemble de vecteurs d'un champ de vecteurs discrets.

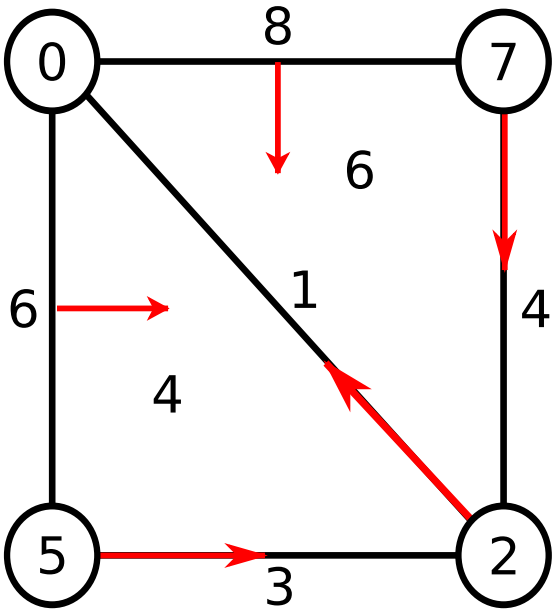



**FIGURE 2.8 –** *Champ de vecteurs discrets correspondant à la fonction de Morse discrète définie à l'image 2.7a.*

**Définition 2.33** ($V$-chemins)**.** Soit $V$ un champ de vecteurs discrets défini sur un complexe $K$ et $p \in \{0, \dots, n-1\}$. Un $V$-chemin est une suite de cellules

$$\alpha_0^{(p)},\ \beta_0^{(p+1)},\ \alpha_1^{(p)},\ \beta_1^{(p+1)}, \dots,\ \beta_r^{(p+1)},\ \alpha_{r+1}^{(p)}\ ^{1} \tag{2.1}$$

telle que pour tout $i \in \{0, \dots, r\}$, $\{\alpha^{(p)} < \beta^{(p+1)}\} \in V$ et $\alpha_i^{(p)} \neq \alpha_{i+1}^{(p)} < \beta_i^{(p+1)}$.

De plus, un $V$-chemin est un chemin fermé non-trivial si $r \geqslant 0$ et $\alpha_0 = \alpha_{r+1}$.

Un exemple de $V$-chemin pour l'exemple de la figure 2.8 commence à la 0-cellule ayant pour valeur 7 couplée avec la 1-cellule ayant pour valeur 4, ensuite le vecteur contenant la 0-cellule ayant pour valeur 2 et la 1-cellule correspondante à la valeur 1 et enfin la 0-cellule de valeur 0.

À partir de cette définition, il existe deux théorèmes concernant les champs de gradient discrets pour les fonctions de Morse.

**Théorème 2.5.** *Soit $V$ le champ de gradient d'une fonction de Morse discrète $f$. Alors une suite de cellules telle que définie en (2.1) est un $V$-chemin si et seulement si pour tout $i = 0, \dots, r$, $\alpha_i < \beta_i$ et $\alpha_{i+1} < \beta_i$ et si*

$$f(\alpha_0) \geqslant f(\beta_0) \geqslant f(\alpha_1) \geqslant \cdots \geqslant f(\beta_r) > f(\alpha_{r+1})$$

Le deuxième théorème est le suivant :

**Théorème 2.6.** *Un champ de vecteurs discrets $V$ est le champ de gradient d'une fonction de Morse discrète si et seulement s'il n'y a pas de $V$-chemin fermé non-trivial.*

La définition d'un tel champ de vecteurs est donc suffisante pour définir une fonction de Morse discrète. En assimilant les $V$-chemins des fonctions discrètes aux lignes intégrales des

1. La notation $\alpha^{(p)}$ indique que la cellule $\alpha$ est de dimension $p$.

fonctions continues, on peut alors transposer toutes les définitions des complexes de Morse-Smale au cas discret. On peut en particulier définir l'origine et la destination des $V$-chemins comme la première et la dernière cellule du $V$-chemin. On peut alors partitionner les cellules du complexe $K$ par rapport à l'origine et la destination du $V$-chemin passant par cette cellule.

Il est à noter que la propriété d'unicité de la ligne intégrale passant par un point donné n'est plus vérifiée dans le cas discret. Deux $V$-chemins peuvent être confondus sur une partie de leur séquence de vecteurs discrets. Le point où deux 1-cellules fusionnent est appelé point de jonction.

### 2.4.3 Simplification

Un complexe de Morse-Smale peut être simplifié en supprimant une paire de points critiques reliés par une 1-cellule dans le complexe de Morse-Smale. Il existe deux façons différentes de faire de la simplification. Celles-ci sont expliquées en détails et comparées dans l'article [24]. Nous nous intéressons ici à la simplification de complexe de Morse-Smale de fonctions définies sur $D \subset \mathbb{R}^2$. Il existe donc deux cas : la paire de points peut être une paire minimum-point col ou maximum-point col. Nous traiterons ici le cas d'une simplification d'un maximum et d'un point col, le deuxième cas étant similaire.

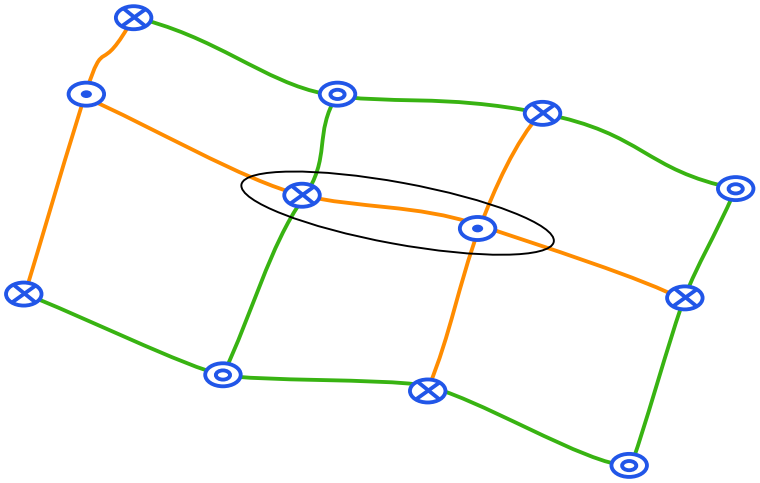
La première simplification est appelée simplification explicite. Le complexe de Morse-Smale est considéré comme un graphe ayant les points critiques comme des sommets et les 1-cellules comme les arcs du graphe. Pour faire la simplification, tous les arcs partant d'un des deux points que l'on veut simplifier sont supprimés. Ensuite, des arcs entre les points cols qui étaient reliés au maximum supprimé et le maximum qui était relié au point col supprimé sont créés. Les nouveaux arcs passent par les chemins des arcs supprimés.

La deuxième simplification est la simplification implicite. Elle consiste à inverser le sens du gradient de la fonction le long de la 1-cellule reliant les deux points que l'on souhaite supprimer. Le complexe de Morse-Smale obtenu à partir de ce nouveau champ de gradient discret est le nouveau complexe de Morse-Smale.

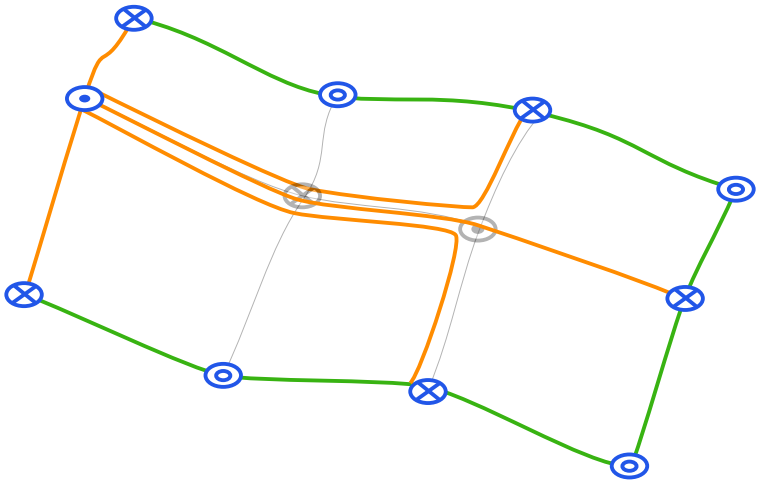
Lors d'une séquence de simplifications effectuée sur un complexe de Morse-Smale, les paires de points critiques seront supprimées par ordre croissant de persistance. Soit $p$ et $q$ une paire de points critiques liés par une 1-cellule dans le complexe de Morse-Smale de la fonction $f$, alors la persistance de la paire est $per_{p,q} = |f(p) - f(q)|$. Lorsque la persistance est faible, cela veut dire que les valeurs de la fonction en ces points sont proches. Cette notion de persistance est présentée plus en détails dans [13]. Après chaque simplification, il faut mettre à jour les persistances des paires de points critiques liées puisque certaines 1-cellules ont été modifiées.

Une suite de simplifications appliquée à un complexe de Morse-Smale d'une fonction définie sur une 2-variété donnera un résultat équivalent que l'on applique le premier ou le second procédé. Cela n'est pas le cas pour les fonction définies sur $D \subset \mathbb{R}^3$ à cause des changements de connectivité induits par des suppressions de paires de points critiques contenant un 1-point col et un 2-point col.

La figure 2.9 montre un complexe de Morse-Smale après une simplification. Les deux points critiques se trouvant au centre du domaine de définition de la fonction sont supprimés et de nouvelles 1-cellules sont créées. Initialement, le complexe de Morse-Smale contient 6 2-cellules, 17 1-cellules et 12 points critiques (0-cellules). Après le procédé de simplification, il reste 4 2-cellules, 13 1-cellules et 10 0-cellules. En supprimant des paires de points critiques dans le complexe de Morse-Smale, cela engendre aussi des suppressions de 1- et de 2-cellules.



**(a)** *Étape de simplification d'un complexe de Morse-Smale, la paire de points critiques à supprimer est entourée.*



**(b)** *Le complexe de Morse-Smale après la simplification. Les deux points critiques n'existent plus dans ce nouveau complexe de Morse-Smale.*

**FIGURE 2.9 –** *Une étape de simplification d'un complexe de Morse-Smale. Une paire de points critiques est supprimé, puis de nouvelles 1-cellules sont créées de telle sorte que le complexe en sortie de ce procédé soit un complexe de Morse-Smale.*

## 2.5 PROBLÉMATIQUE

Nous avons vu dans la partie précédente le déroulement d'un étape de simplification. Le processus de simplification d'un complexe de Morse-Smale permet de modifier le complexe de Morse-Smale en lui-même en appliquant des suppressions de points critiques et des suppressions et des unions de 1-cellules et de 2-cellules. Mais il ne va pas faire de modification sur la fonction initiale sur laquelle a été calculé le complexe de Morse-Smale avant simplification. Sur la figure 2.9, on peut voir un complexe de Morse-Smale simplifié projeté sur la surface correspondant à la fonction initiale. Ceux-ci ne sont alors plus cohérents, nous pouvons prendre comme exemple une surface possèdant un maximum et un point col dans l'intérieur de son

domaine de définition, alors que le complexe de Morse-Smale n'en contient seulement sur la frontière de son domaine de définition. Il est alors nécessaire d'essayer de trouver une surface qui va être cohérente avec le complexe de Morse-Smale donné et qui approxime la fonction initiale, comme la fonction montrée à la figure 2.10.

Le but de cette thèse est de construire une fonction polynomiale par morceaux cohérente avec un complexe de Morse-Smale donné, celui-ci correspondant à une simplification du complexe de Morse-Smale d'une fonction initiale qui est elle aussi donnée. Cela a aussi introduit le problème de la construction d'une surface monotone polynomiale par morceaux interpolant des données.

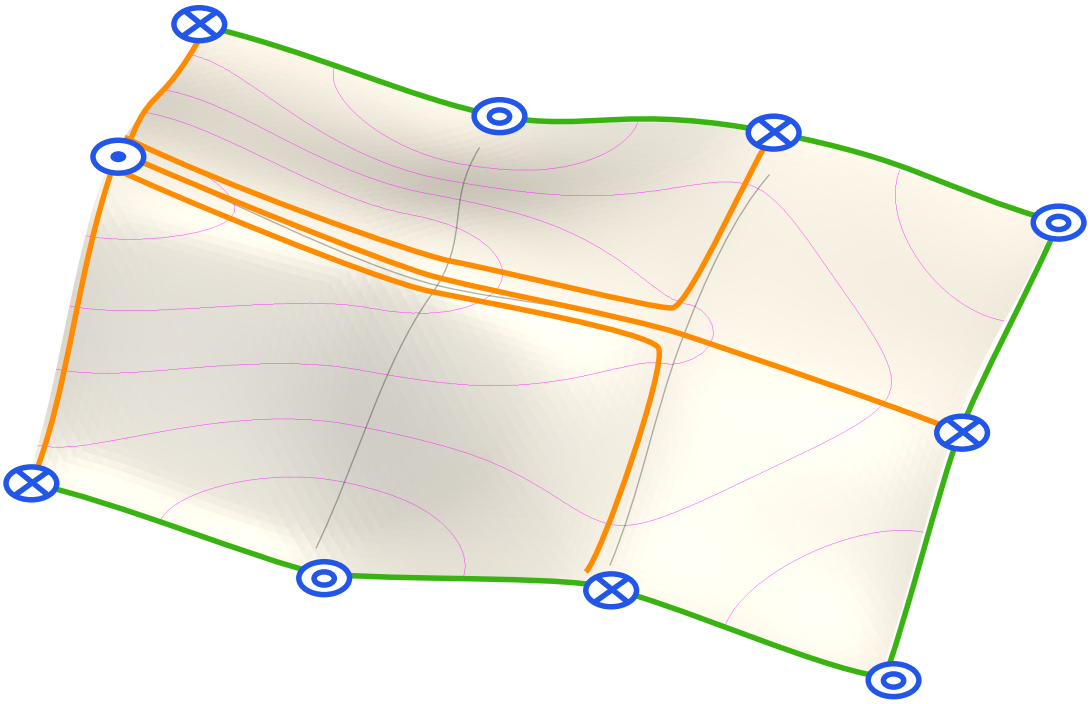


**FIGURE 2.10 –** *Construction d'une fonction cohérente avec un complexe de Morse-Smale simplifié.*

CHAPITRE

# 3

# ÉTAT DE L'ART

À partir d'une fonction initiale, le complexe de Morse-Smale partitionne le domaine de définition de cette fonction en régions sur lesquelles la fonction est monotone. Vouloir construire une fonction à partir d'un complexe de Morse-Smale simplifié correspond donc à définir une fonction monotone sur un ensemble donné. La reconstruction est effectuée par dimension de cellule croissante. En effet, les 0-cellules sont les points critiques que l'on veut garder de la fonction. Ceux-ci seront alors utilisés comme conditions aux deux extrémités des 1-cellules pour pouvoir définir une fonction monotone le long de ce type de cellule. Enfin, les valeurs définies le long des 1-cellules seront utilisées comme conditions aux bords des 2-cellules pour reconstruire ces derniers. Nous allons donc passer en revue un certain nombre de méthodes permettant d'interpoler des données en section 3.1.1. Puis nous présenterons les méthodes permettant d'interpoler des données de façon à conserver la monotonie de ces données (section 3.1.2). Nous verrons ensuite les différents travaux portant sur les complexes de Morse-Smale (section 3.2), tout d'abord en s'intéressant à la façon de les calculer, puis en regardant les différentes méthodes de reconstruction de fonction à partir d'un complexe de Morse-Smale simplifié.

## 3.1 État de l'art sur l'interpolation et la préservation de la monotonie

Dans cette section, nous allons voir les méthodes permettant d'interpoler des données en sous-section 3.1.1. Ces données pouvant être éparses ou bien structurées sur des grilles 2D. Les points que nous souhaitons interpoler peuvent avoir certaines caractéristiques comme la convexité ou encore la monotonie. Nous allons ici (sous-section 3.1.2) nous focaliser sur des méthodes permettant de préserver la monotonie, i.e. de créer une surface monotone à partir de données monotones. Les méthodes d'interpolation qui permettent de préserver ce type de caractéristiques des données en entrée s'appellent "Shape Preserving Interpolation".

### 3.1.1 Interpolation de données

Dans cette section, nous allons nous intéresser au problème suivant :

**Problème.** *Soit $n$ points $\left(P_i = (x_i, y_i, z_i)^t\right)_{i=1,\dots,n} \in \mathbb{R}^3$ tels que pour tout $i, j = 1, \dots, n$ avec $i \neq j$, $(x_i, y_i) \neq (x_j, y_j)$. Nous cherchons une surface $S : \mathbb{R}^2 \to \mathbb{R}$ telle que :*

$$(\forall i = 1, \dots, n)\ S(x_i, y_i) = z_i.$$

Nous allons voir deux aspects différents du problème : le premier, quand il n'y a pas de structures entre les données à interpoler, le second quand les points sont répartis sur une grille.

#### I. Données non structurées

Supposons qu'il n'y a pas de connaissances sur la structure entre les points à interpoler. Les méthodes que nous allons présenter sont donc aussi applicables aux données structurées. Dans un premier temps, nous allons présenter des méthodes basées sur une triangulation des données. Ensuite, nous verrons les méthodes de pondération inverse à la distance, puis les méthodes utilisant les fonctions de base radiale. Nous finirons par la méthode d'interpolation du voisinage naturel.

***(a) Méthodes basées sur une triangulation des données***

Le procédé de ces méthodes comporte deux étapes : une première étape de triangulation est tout d'abord appliquée sur les données dans le plan $(x, y)$. Pour cela, on peut utiliser n'importe quelle méthode de triangulation. La méthode la plus utilisée est la triangulation de Delaunay[12], celle-ci permet d'éviter au maximum les triangles mal formés, c'est-à-dire les triangles fins. Cette méthode nous garantie d'avoir l'angle minimal de la triangulation le plus grand possible. Ensuite, une surface va être construite sur les triangles interpolant les valeurs aux sommets des triangles. Nous allons voir ici différentes méthodes de construction de telles surfaces définies sur les triangles.

**Interpolation linéaire** Cette méthode est la plus simple des méthodes présentése. Elle consiste en la construction d'un plan défini sur le triangle passant par les trois sommets. Supposons que les points $P_i, P_j$ et $P_k$ soient les sommets d'un même triangle. Soit $P = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ un point du triangle. Il existe alors un triplet de réels $(\alpha, \beta, \gamma)$ tel que

$$\begin{aligned} \alpha, \beta, \gamma &\geqslant 0, \\ \alpha + \beta + \gamma &= 1. \end{aligned}$$

Ce triplet est appelé les coordonnées barycentriques de $P$ par rapport au triangle de sommets $P_i, P_j$ et $P_k$. La surface correspondant à l'interpolation linéaire des sommets du triangle est définie à l'intérieur du triangle par

$$S = \left\{ P = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mid P = \alpha P_i + \beta P_j + \gamma P_k,\ \alpha, \beta, \gamma \leqslant 0 \text{ et } \alpha + \beta + \gamma = 1 \right\}.$$

L'union des surfaces définies sur chaque triangle forme alors une surface interpolant les données.

Cette méthode possède deux avantages : le premier est que son implémentation est simple. Le deuxième est que la surface interpolante est continue. Mais la continuité est seulement $C^0$, cela a pour conséquence des artefacts visuels que sont les facettes de la triangulation. De plus, cette méthode ne permet pas de construire une surface ayant pour domaine de définition $\mathbb{R}^2$ entier, mais seulement une surface sur l'enveloppe convexe des points. Un autre problème vient du fait que la surface dépend de la triangulation donnée en entrée.

**Divisions des triangles (Méthodes de Splitting)** Ces méthodes se basent sur la construction de surfaces à l'intérieur de triangles par des patchs triangulaires de Bézier de degré 2 ou 3. Elles permettent de construire une surface globalement $C^1$ sur le domaine de définition qui est ici l'enveloppe convexe des données. Chaque triangle est d'abord subdivisé (splitting). Ensuite, un patch de Bézier sera calculé sur chaque sous-triangle. Cela est dû au fait que pour avoir une surface $C^1$ avec un seul patch de Bézier par triangle, il nous faudra donner des dérivées secondes. En effet, Farin montre en [16, page 106] les résultats suivants :

> « Théorème 5.1. Un interpolant local polynomial par morceaux qui interpole des dérivées d'ordres inférieurs ou égaux à $r$ d'une fonction $f$ aux sommets d'une triangulation et qui est globalement différentiable $r$ fois doit être de degré $n \geqslant 4r + 1$. »

Il faut donc donner des valeurs de dérivées d'ordre supérieur à l'ordre de continuité voulu. En utilisant la division du triangle, seul les dérivées d'ordre 1 sont nécessaires pour construire une surface $C^1$.

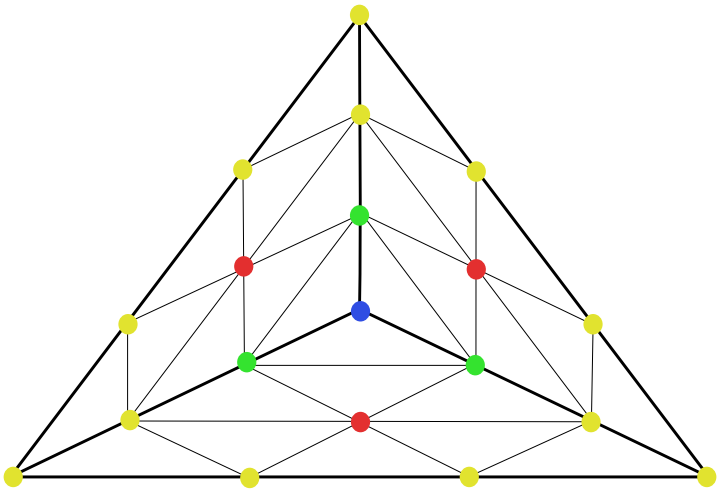


**FIGURE 3.1 –** *Division de Clough-Tocher.*

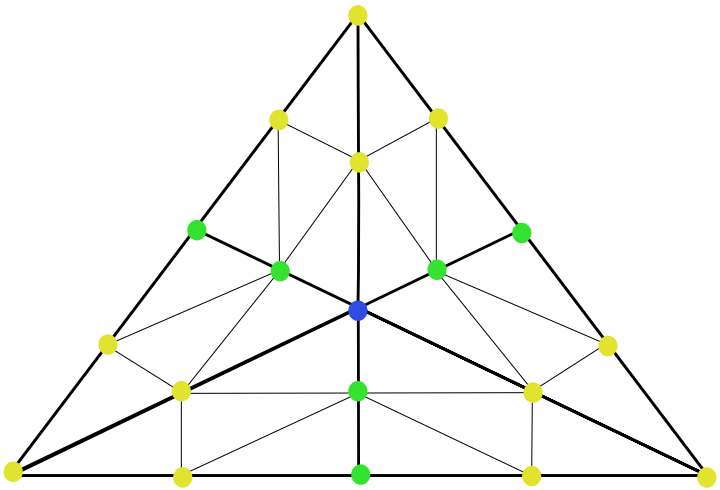


**FIGURE 3.2 –** *Division de Powell-Sabin en 6 sous-triangles.*

Les méthodes suivantes supposent donc connues les dérivées partielles dans les directions $x$ et $y$ en chaque sommet de la triangulation. De plus, elles nécessitent de connaître la valeur de la dérivée normale aux arêtes de la triangulation au milieu des arêtes. Si ces valeurs ne sont pas connues, il faudra alors en faire une estimation à partir des valeurs aux sommets.

La première division est montrée à la figure 3.1, elle est appelée division de *Clough-Tocher*. Elle a été introduite par Clough et Tocher [11]. Cette division consiste en la division du triangle en trois sous-triangles. Clough et Tocher définissent des surfaces triangulaires de Bézier de degré 3 sur cette subdivision de la façon suivante : les valeurs des points jaunes à la figure 3.1 sont données par les valeurs de fonction et de dérivées partielles aux sommets de la triangulation. Les valeurs aux points rouges sont données par les valeurs des dérivées transverses orthogonales aux arêtes. Les points verts valent la moyenne des points liés à ceux-ci par la condition de continuité $C^1$. Enfin, la valeur au point bleu est égale à la moyenne des valeurs aux points

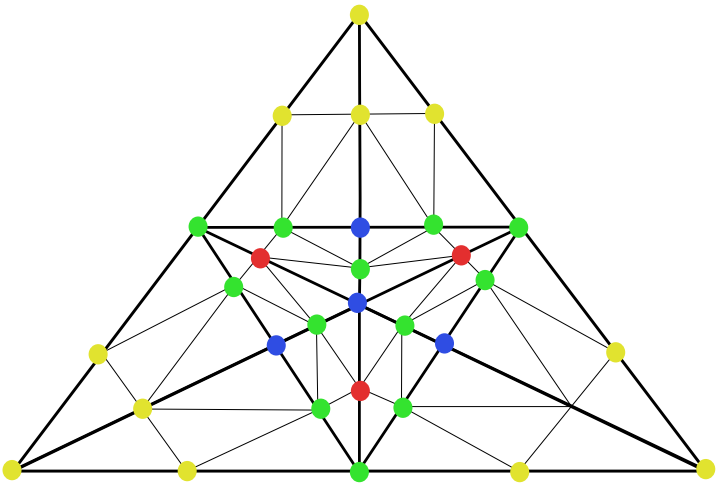


**FIGURE 3.3 –** *Division de Powell-Sabin en 12 sous-triangles.*

verts. Les valeurs que nous venons d'énoncer permettent d'avoir une surface $C^2$ sur le triangle et $C^1$ entre les triangles de la triangulation.

Un autre exemple de division de triangle est la méthode de *Powell-Sabin*[38]. Celle-ci est composée de deux divisions, l'une en 6 triangles et la deuxième en 12 triangles. L'utilisation de l'une ou l'autre des divisions dépend des angles du triangle considéré. Si les angles sont tous inférieurs à 75°, alors le triangle est découpé en 6 sous-triangles, sinon, on utilise la deuxième division.

La première division est montrée à la figure 3.2. Le triangle est découpé en six sous-triangles sur lesquels une surface de Bézier triangulaire de degré 2 sera définie. Pour ce cas, nous avons besoin des valeurs de fonction aux sommets du triangle ainsi que des valeurs de dérivées partielles dans les directions $x$ et $y$. Ces valeurs permettent de définir les valeurs des points de contrôles aux points jaunes. Ensuite, les points verts sont la moyenne des points jaunes alignés sur une même ligne. Et enfin, le point vert vaut la moyenne des trois points verts autour de lui.

La deuxième division est montrée à la figure 3.3. Le triangle est découpé en douze sous-triangles où, comme dans le cas précédent, une surface de Bézier triangulaire de degré 2 sera définie. Contrairement à la division précédente, nous aurons besoin des mêmes données d'entrées ainsi que des valeurs de dérivées transverses orthogonales aux arêtes aux milieu de celles-ci. Les valeurs de fonction et des dérivées partielles permettent de définir les valeurs aux points jaunes. Ensuite, les valeurs aux points rouges sont données par les valeurs des dérivées transverses. Les valeurs aux points verts sont calculées comme la moyenne entre les deux points alignés liés au point que l'on veut calculer. Enfin, les valeurs aux points bleus valent la moyenne des points auxquels ils sont liés.

Cette division des triangles permet d'avoir des surfaces globalement $C^1$ sur l'ensemble de définition.

Ces deux méthodes permettent d'avoir une surface $C^1$ sur l'enveloppe convexe des points donnés en entrée. De plus, elle sont locales, c'est-à-dire que le changement d'un point en entrée ne va impliquer des changements que sur les triangles ayant comme sommet le point modifié. Ces méthodes permettent aussi une évaluation rapide de la surface. Un désavantage de la méthode

de Powell-Sabin par rapport à celle de Clough-Tocher est le besoin de faire une disjonction de cas en fonction des angles des triangles. L'inconvénient des deux méthodes est que le résultat dépend de la triangulation choisie.

**Mélange** Cette méthode a été décrite par McLain dans [36]. Pour chaque point $P_i$ de la triangulation, on va considérer les 5 points les plus proches dans le plan $(x, y)$ de celui-ci. On cherche alors à faire passer un polynôme quadratique par ces six points $(P_j)_{j\in J}$ avec $J \subset \mathbb{N}$ et $Card(J) = 6$. Le polynôme s'écrit

$$f_i(x, y) = a_1x^2 + a_2xy + a_3y^2 + a_4x + a_5y + a_6$$

avec $(a_i)_{i=1,\dots,6} \in \mathbb{R}$. Ces coefficients sont donnés par la résolution du système linéaire

$$(\forall j \in J)\ f_i(x_j, y_j) = z_j.$$

Il suffit donc de résoudre un système linéaire de six équations à six inconnues pour chaque point.

Plaçons nous ensuite dans le triangle de sommets $P_1, P_2, P_3$. Posons la fonction $d_i$ avec $i = 1, 2, 3$ définie sur le triangle telle que $d_i(P_i) = 1$ et $d_i(P)|_{[P_{(i+1)\%3},P_{(i+2)\%3}]} = 0$. Elle vaut donc 1 au point considéré et 0 sur l'arête opposé au point. Posons maintenant le poids $\omega_i = \frac{d_i^k}{d_1^k+d_2^k+d_3^k}$, avec $k \in \mathbb{R}^+$ (souvent $k = 3$). La surface définie sur le triangle sera alors le mélange des trois fonctions $f_i$ combinées chacune avec le poids $\omega_i$ :

$$S(x, y)|_{P_1P_2P_3} = \omega_1 f_1(x, y) + \omega_2 f_2(x, y) + \omega_3 f_3(x, y)$$

Cette méthode garantie seulement la continuité $C^0$ entre les triangles, mais il existe des variantes donnant une continuité $C^1$ [35]. Elle est locale et rapide à évaluer. Elle a le même désavantage que les autres méthodes basées sur les triangulations : la triangulation n'est pas unique. Pour chaque triangulation, il y a donc un résultat différent. Par contre, contrairement aux autres méthodes, il existe des moyens de prolonger la surface en dehors de l'enveloppe convexe des données.

**Conclusion sur les méthodes basées sur les triangulations** Nous avons vu que toutes les méthodes se basant sur une triangulation des données ont pour avantage d'être locale. L'ordre de continuité de la surface interpolante dépend de la méthode. Elles ont toutes le même problème : la surface résultante dépend de la triangulation des données donnée en entrée. Si la triangulation est changée, la surface sera alors différente. Il faudra donc faire attention à la méthode de triangulation des données.

### *(b) Méthodes de pondération inverse à la distance*

Nous allons présenter ici la première méthode de pondération inverse à la distance définie par Shepard dans [43]. La valeur au point où l'on cherche à évaluer la surface correspond à une moyenne pondérée par la distance aux points donnés en entrée de leurs valeurs. Les valeurs des points les plus proches auront un poids plus fort que les valeurs des points éloignés. Les méthodes varient par leurs définitions des poids attribués aux points. La surface définie dans l'article [43] est donné par l'équation

$$S(x, y) = \sum_{i=1}^{n} \frac{w_i(x, y)}{\sum_{j=1}^{n} w_j(x, y)} z_i.$$

avec

$$w_i(x,y) = \| \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} x_i \\ y_i \end{pmatrix} \|^{-k},$$

où $k$ est un réel positif supérieur à 1. Si l'on prend $k = 1$, cela va créer des pics aux points que l'on veut interpoler, alors que si l'on prend $k > 1$, alors cela fait apparaître des plateaux (dérivées premières nulles) en ces points. La surface résultante est $C^0$ si $k = 1$ et $C^1$ sinon. De plus, cette méthode permet de créer une surface sur tout le domaine $\mathbb{R}^2$. Dans les inconvénients, nous pouvons citer que l'interpolation de crêtes, de vallées et de plans n'est pas possible. Cette méthode est globale : si l'on modifie un point dans les données, alors la surface interpolante change sur tout le domaine de définition.

#### *(c) Fonctions de base radiale*

Pour chaque point $P_i$ donné en entrée, on définit une fonction $h_i$. Ensuite, la surface interpolante est donnée par l'équation

$$S(x,y) = \sum_{i=1}^{n} a_i h_i(x,y)$$

avec $(a_i)_{i=1,\dots,n} \in \mathbb{R}$ tels que pour tout $i = 1,\dots,n$ on ait $S(x_i,y_i) = z_i$. On définit les fonctions $h_i$ comme des fonctions radiales, c'est-à-dire qu'elles vont dépendre seulement de la distance aux points auxquelles elles se rattachent. Posons le point $P = \begin{pmatrix} x \\ y \end{pmatrix}$ et $d_i = \|P - P_i\|$ la distance entre le point $P$ et le point $P_i$. Ainsi, on définit $h_i(x,y) = h_i(d_i)$.

La méthode originale définie par Hardy dans [29] définie la fonction $h_i$ par :

$$\begin{aligned} h_i(x,y) &= h_i(d_i) \\ &= \sqrt{(d_i^2 + c^2)} \end{aligned}$$

Le cas où $c \neq 0$ correspond à une hyperboloïde à deux nappes, sinon, cette équation décrit une conique.

Une fois que le coefficient $c$ est choisi, il suffit de résoudre le système linéaire à $n$ équations et $n$ inconnues permettant de connaître les valeurs des coefficients $a_i$. La particularité des fonctions $h_i$ définies de cette façon est que le système à résoudre est non singulier et bien conditionné, alors qu'un autre choix de fonctions ne garantit pas d'avoir une solution à cette équation.

La surface interpolante est $C^\infty$ lorsque $c \neq 0$ et $C^0$ sinon. Par contre, cette méthode nécessite en pré-calcul l'inversion d'une matrice $n \times n$ avec $n$ le nombre de points que l'on souhaite interpoler. De plus cette méthode est globale.

D'autres fonctions radiales se trouvent dans [28, 40].

#### *(d) Méthode d'interpolation du voisinage naturel*

Cette méthode, définie par Sibson dans [45], commence par calculer le diagramme de Voronoï des points donnés en entrée. Une cellule du diagramme contenant le point $P_i$ contient l'ensemble de tous les points du plan qui sont plus proches de $P_i$ que des points $P_j$ avec $j \neq i$ :

$$V_i = \{P \in \mathbb{R}^2 | (\forall j = 1,\dots,i-1,i+1,\dots,n)\ \|P - P_i\| \leq \|P - P_j\|\}$$

où $\|\cdot\|$ correspond à la distance euclidienne de $\mathbb{R}^2$. Pour un point $P = \begin{pmatrix} x \\ y \end{pmatrix}$, on calcule alors sa cellule de Voronoï en prenant comme germes pour le calcul ce point ainsi que les points d'entrée, on la note $V(P)$. On note alors l'intersection de cette cellule avec la cellule du point $P_i$ dans le diagramme initial $V_i(P) = V(P) \cap V_i$. Posons la fonction $\mathscr{A}(V)$ l'aire de l'ensemble $V$. Posons de plus la surface $S$ comme une somme pondérée des valeurs aux points d'entrée :

$$S(P) = \sum_{i=1}^{n} w_i(P) z_i$$

avec les poids

$$w_i(P) = \frac{\mathscr{A}(V_i(P))}{\mathscr{A}(V(P))}$$

De cette façon, la surface résultante sera de continuité d'ordre $C^0$.

Pour obtenir une surface de continuité $C^1$, il faut alors supposer que nous connaissons ou que nous pouvons évaluer le gradient de la fonction $\bigtriangledown z(P_i)$ aux points donnés en entrée. La surface est alors définie par

$$S(P) = \sum_{i=1}^{n} \omega_i(P)(z_i + \bigtriangledown z(P_i)^t (P - P_i))$$

avec les fonctions $\omega_i$ définies par

$$\omega_i(P) = \frac{w_i(P)\|P - P_i\|^{-1}}{\sum_{j=1}^{n} w_j(P)\|P - P_j\|^{-1}}$$

Cette méthode permet d'avoir une surface $C^1$. Le résultat est local. Cette méthode possède deux désavantages : le premier est qu'il faut faire un pré-calcul du diagramme de Voronoï, ce qui est équivalent à calculer une triangulation de Delaunay. Le deuxième est que l'évaluation de la surface est lente.

#### *(e) Conclusion sur les méthodes d'interpolation de données éparses*

En résumé, les méthodes permettant d'interpoler des données éparses permettent pour la plupart de créer une surface $C^1$. Une majorité de ces méthodes ont besoin d'un calcul d'une triangulation, la surface interpolante finale va alors dépendre de cette triangulation. Il faut aussi faire attention au caractère local ou global de la méthode, une méthode locale permet d'avoir une surface qui va n'être modifiée qu'au voisinage d'un point lorsque l'on modifie celui-ci.

### II. Données sur une grille

Nous venons de voir les méthodes permettant de construire une surface interpolant des données éparses. On se place ici dans le cas où les données sont sur une grille rectangulaire. Il existe donc $(x_i)_{i=1,\dots,n_x}$ et $(y_j)_{j=1,\dots,n_y}$, tels que nous connaissons les valeurs de fonction $z_{ij}$ en tout point $(x_i, y_j)$. Nous pouvons utiliser les méthodes précédentes telles que nous les avons définies dans la partie précédente.

Parmi les méthodes que nous avons vues précédemment, trois d'entre elles sont souvent utilisées dans le cas de données sur une grille. Celles-ci sont l'interpolation linéaire, l'interpolation cubique et la méthode des mélanges. Mais elles sont modifiées dans ce cas-ci pour

prendre en compte le fait que les données sont ordonnées. Ces méthodes étaient basées sur la triangulation des données, ici, elles vont rechercher une surface sur les rectangles composants la grille. Elles vont de plus utiliser des propriétés sur les triangles permettant de faire moins de calculs que dans le cas des triangles où il n'y avait pas de suppositions sur la forme des triangles. Pour plus de précisions concernant ces améliorations, voir [8].

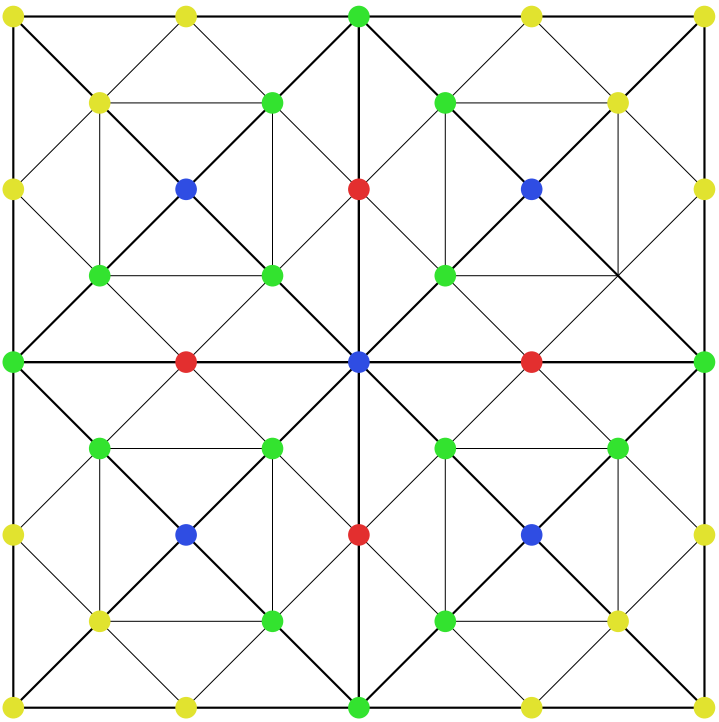


**FIGURE 3.4 –** *Division de Sibson de degré 2.*

Nous allons nous intéresser à deux divisions de rectangles définies par Sibson et Thomson [46]. Ces méthodes prennent partie du fait que les données sont ordonnées sur une grille et donc vont construire des surfaces sur des rectangles. La figure 3.4 représente la division contenant des surfaces de Bézier triangulaires de degré 2, alors qu'à la figure 3.5, les surfaces de Bézier sont de degré 3. Dans ces deux images, les points jaunes sont donnés par les valeurs de la fonction à interpoler aux sommets et par les valeurs de dérivées partielles dans les directions $x$ et $y$ aux sommets. Les valeurs aux points rouges sont obtenues par la condition que les dérivées transverses aux arêtes de bords soient linéaires. Ensuite, les valeurs aux points verts sont la moyenne des valeurs aux deux points rouges adjacents et enfin, la valeur au point bleu et la moyenne des valeurs aux points verts.

Ces deux constructions permettent de définir une surface globalement $C^1$ et de la définir de façon locale.

## III. Conclusion

Nous avons vu qu'il existe plusieurs catégories de méthodes.

– Premièrement, on peut faire la distinction entre les méthodes travaillant sur des données éparses et celles ayant en entrée des données structurées sur une grille rectangulaire. Les premières peuvent être utilisées dans tout les cas contrairement aux secondes.

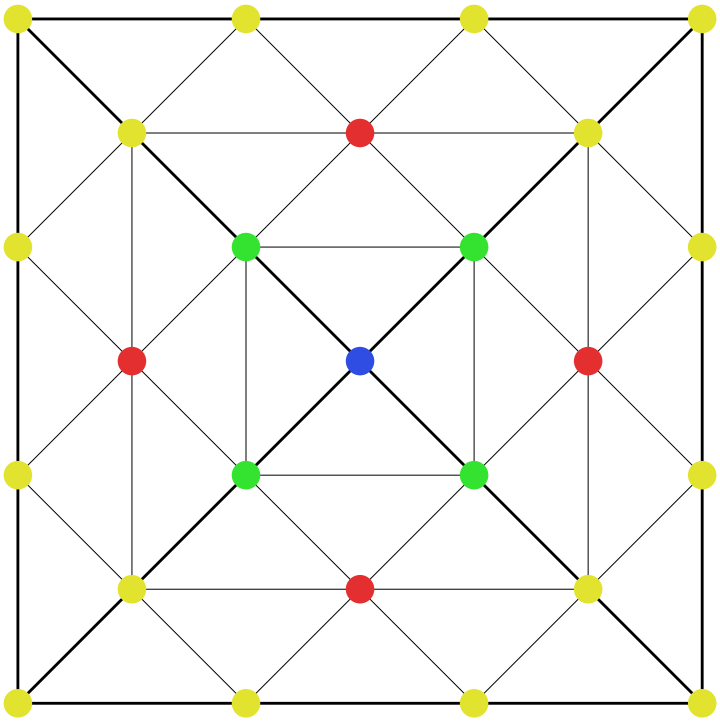


**FIGURE 3.5 –** *Division de Sibson de degré 3.*

– Ensuite, certaines méthodes sont locales et d'autres globales. Les méthodes locales permettent de faire des modifications dans les données et cela n'aura qu'une influence locale sur la surface résultante. Les méthodes locales sont donc à privilégier par rapport à celles globales.
– Enfin, la continuité de la fonction est importante pour les applications où l'on souhaite faire un retour visuel à un utilisateur. Les méthodes donnant des surfaces ayant une continuité $C^1$ ou supérieure sont donc à privilégier.

Les méthodes à privilégier sont donc les méthodes locales de continuité $C^1$ ou supérieure. Celles-ci sont les méthodes basées sur des divisions de triangles (les divisions de Clough-Tocher, Powell-Sabin et Sibson), la méthode des mélanges ainsi que la méthode d'interpolation du voisinage naturel.

### 3.1.2 Préservation de la monotonie

Nous avons vu à la partie précédente des méthodes d'interpolation de données. Les données que l'on cherche à interpoler peuvent venir de variables dont certaines caractéristiques sont connues. Il est alors intéressant d'avoir une surface interpolant ces données qui possède les mêmes caractéristiques. Il existe des méthodes, appelées "Shape Preserving", permettant de créer des surfaces interpolantes qui seront garanties d'être convexes ou encore monotones. Dans cette partie, nous allons nous concentrer sur la monotonie de la surface interpolante. Nous allons tout d'abord regarder une méthode travaillant sur des données éparses, puis nous nous intéresserons à des données structurées sur une grille.

### Données éparses

Soit $\Omega \subseteq \mathbb{R}^2$. Soient $n$ points de $\Omega \times \mathbb{R} \subseteq \mathbb{R}^3$ donnés par la connaissance de $t_1, \ldots, t_n \in \Omega$ et $z_1, \ldots, z_n \in \mathbb{R}$, les $(z_i)_{i=1,\ldots n}$ vérifiant un critère de monotonie. Nous cherchons alors une surface $S$ qui interpole les valeurs $(z_i)_{i=1,\ldots,n}$ aux points $(t_i)_{i=1,\ldots,n}$, c'est-à-dire :

$$(\forall i = 1, \ldots, n)\ S(t_i) = z_i$$

et qui soit monotone. Utreras et Varas [49] définissent la monotonie d'une surface ainsi :

**Définition 3.1.** Soit $d_1$ et $d_2$ deux vecteurs non colinéaires de $\mathbb{R}^2$. Notons $K = \{\lambda_1 d_1 + \lambda_2 d_2,\ \lambda_1, \lambda_2 \geqslant 0\}$. La surface $S$ est monotone selon la base $(d_1, d_2)$ si :

$$(\forall x, y \in \overline{\Omega})\ y - x \in K \Rightarrow S(y) \geqslant S(x) \tag{3.1}$$

Utreras et Varas choisissent d'utiliser les splines en plaque mince pour obtenir une technique d'interpolation monotone qui se généralise aisément aux cas où la dimension de l'espace $\Omega$ est plus élevée car celles-ci sont évaluées à partir de méthodes variationnelles. Soit l'énergie $E$ définie pour la surface $S$ telle que :

$$E(S(x,y)) = \iint_{\mathbb{R}^2} \left[ \left( \frac{\partial^2 S}{\partial x^2} \right)^2 + 2 \left( \frac{\partial^2 S}{\partial x \partial y} \right)^2 + \left( \frac{\partial^2 S}{\partial y^2} \right)^2 \right] dxdy$$

Cette énergie correspond à une énergie de flexion de la surface. La spline en plaque mince est alors la surface minimisant cette énergie passant par les points à interpoler.

Ce problème de minimisation ne possède qu'une unique solution sur l'ensemble des surfaces interpolant les données. Cette surface n'étant pas forcément monotone selon une base $(d_1, d_2)$ donnée, une méthode permettant de converger itérativement vers une solution monotone au problème d'interpolation est alors donnée en [49]. Pour cela, l'algorithme découpe l'espace des surfaces de façon itérative de manière à conserver seulement l'espace des fonctions monotones interpolant les points. Si une solution est trouvée lors d'une itération de l'algorithme, alors la surface est donnée en résultat, sinon les auteurs montrent que l'algorithme converge vers une spline en plaque mince monotone.

Cette méthode permet donc d'interpoler des données éparses avec un espace de départ de dimension finie quelconque. Les deux désavantages de cette méthode sont l'application d'une minimisation à chaque itération de l'algorithme et le fait que la monotonie de la surface finale doit être selon un nombre de directions égal à la dimension de l'espace. On ne peut donc pas créer une surface monotone selon une seule direction dans un espace de dimension strictement supérieure à 1.

D'autres méthodes, telles que [51], construisent une fonction monotone approximant des données éparses.

### Données structurées sur une grille

Nous allons maintenant nous intéresser au cas où les données sont structurées sur une grille 2D. Il existe deux approches différentes pour créer une surface monotone interpolant ces données : la première consiste à créer des surfaces produit tensoriel dans les rectangles de la grille. Quant à la seconde, elle va appliquer une division des rectangles en triangles afin de créer des surfaces à l'intérieur de ces triangles.

***Surfaces produit tensoriel***

La définition de monotonie donnée par Fritsch et Carlson [21] est la suivante :

**Définition 3.2.** Soit $S$ une surface définie sur $\Omega \subseteq \mathbb{R}^2$. $S$ est dite monotone si :

$$(\forall y^* \in \mathbb{R})\ (\forall x_1, x_2 \in \mathbb{R})\ s_1 x_1 \leqslant s_1 x_2 \Rightarrow s_1 S(x_1, y^*) \leqslant s_1 S(x_2, y^*)$$
$$(\forall x^* \in \mathbb{R})\ (\forall y_1, y_2 \in \mathbb{R})\ s_2 y_1 \leqslant s_2 y_2 \Rightarrow s_2 S(x^*, y_1) \leqslant s_2 S(x^*, y_2)$$

avec

$$s_1 = \begin{cases} +1 \text{ si la surface } S \text{ est monotone croissante selon la direction } x \\ -1 \text{ si la surface } S \text{ est monotone décroissante selon la direction } x \end{cases}$$
$$s_2 = \begin{cases} +1 \text{ si la surface } S \text{ est monotone croissante selon la direction } y \\ -1 \text{ si la surface } S \text{ est monotone décroissante selon la direction } y \end{cases}$$

deux constantes pour la surface $S$.

Pour que la surface soit monotone, il faut donc que la courbe obtenue par une coupe de la surface selon l'axe des $x$ ou selon l'axe des $y$ soit elle-même monotone.

Soit $[a, b] \times [c, d]$ un rectangle de la grille. Les auteurs posent alors la fonction $u$ sur ce rectangle par l'équation suivante pour $(x, y) \in [a, b] \times [c, d]$ :

$$\begin{aligned} u(x,y) =& u(a,c)H_1(x)G_1(y) + u(b,c)H_2(x)G_1(y) \\ &+ u(a,d)H_1(x)G_2(y) + u(b,d)H_2(x)G_2(y) \\ &+ u_x(a,c)H_3(x)G_1(y) + u_x(b,c)H_4(x)G_1(y) \\ &+ u_x(a,d)H_3(x)G_2(y) + u_x(b,d)H_4(x)G_2(y) \\ &+ u_y(a,c)H_1(x)G_3(y) + u_y(b,c)H_2(x)G_3(y) \\ &+ u_y(a,d)H_1(x)G_4(y) + u_y(b,d)H_2(x)G_4(y) \\ &+ u_{xy}(a,c)H_3(x)G_3(y) + u_{xy}(b,c)H_4(x)G_3(y) \\ &+ u_{xy}(a,d)H_3(x)G_4(y) + u_{xy}(b,d)H_4(x)G_4(y) \end{aligned}$$

où $(H_i, G_i)$ sont les fonctions d'Hermite de degré 3 telles que :

$$H_1(x) = \phi\left(\frac{b-x}{b-a}\right), \qquad G_1(y) = \phi\left(\frac{d-y}{d-c}\right),$$
$$H_2(x) = \phi\left(\frac{x-a}{b-a}\right), \qquad G_2(y) = \phi\left(\frac{y-c}{d-c}\right),$$
$$H_3(x) = -(b-a)\psi\left(\frac{b-x}{b-a}\right), \qquad G_3(y) = -(d-c)\psi\left(\frac{d-y}{d-c}\right),$$
$$H_4(x) = (b-a)\psi\left(\frac{x-a}{b-a}\right), \qquad G_4(y) = (d-c)\psi\left(\frac{y-c}{d-c}\right),$$

avec

$$\phi(t) = 3t^2 - 2t^3,$$
$$\psi(t) = t^3 - t^2.$$

Les auteurs donnent alors des conditions pour que la surface soit monotone dans la direction $y$. Tout d'abord, posons la fonction $\Delta_1 : [a, b] \to \mathbb{R}$ par $\Delta_1(x) = \frac{u(x,d)-u(x,c)}{d-c}$. Les trois conditions sont alors :

1. La fonction $signe(\Delta_1)$ est une constante et est égale à $signe(\frac{f(a,d)-f(a,c)}{d-c})$. Cela signifie que la surface est monotone en $y$ et qu'elle est du même type de monotonie que les points à interpoler.
2. Si $\Delta_1(x) = 0$ (ce qui signifie que les $u(x, d) = u(x, c)$), alors les dérivées partielles dans la direction $y$ aux deux extrémités sont nulles.
3. Les dérivées aux extrémités ne sont pas trop grandes par rapport à la pente correspondante à la droite passant par les deux extrémités.

Ces conditions peuvent être modifiées pour avoir la monotonie selon la direction $x$.

Cette méthode nécessite donc la connaissance ou l'estimation des dérivées partielles d'ordre 1 dans les directions $x$ et $y$ et aussi celle des dérivées partielles d'ordre 2 croisées en chaque point de la grille. Les auteurs donnent alors un algorithme permettant de calculer des dérivées partielles d'ordre 1 et 2 qui vérifient les conditions citées précédemment pour que la surface résultante soit monotone.

Cet algorithme permet seulement d'avoir une surface monotone dans les deux directions. L'article [9] propose alors une extension permettant d'avoir la monotonie de la fonction selon une unique direction qui est soit $x$ soit $y$.

#### *Décomposition en triangles*

Les articles [6] et [27] proposent d'utiliser les divisions de Sibson à l'intérieur des rectangles. Le premier utilise la division de Sibson montré à la figure 3.4 qui remplit les triangles par des surfaces de Bézier triangulaires quadratiques. Le deuxième utilise la division de Sibson utilisant des surfaces triangulaires de Bézier de degré 3 (illustrée à la figure 3.5). Ils utilisent la même définition de la monotonie que celle de la définition 3.2. Pour construire les surfaces de Bézier monotones, les deux méthodes ont besoin des valeurs monotones à interpoler aux sommets du rectangle, mais aussi des dérivées partielles d'ordre 1 dans les directions $x$ et $y$. Comme nous avons vu aux paragraphes précédents sur les surfaces produit tensoriel, pour pouvoir construire une surface monotone, il ne faut pas que les dérivées partielles aux sommets aient des valeurs trop élevées. Les deux articles proposent donc des conditions sur les dérivées partielles aux sommets pour que la surface interpolante soit monotone. Pour cela, ils calculent les dérivées partielles dans les directions $x$ et $y$ à partir des formules de leur interpolant. Cela leur permet ensuite de proposer des algorithmes estimant des dérivées partielles aux sommets de la grille qui définissent une surface monotone sur le rectangle.

Grâce aux divisions de Sibson, ces deux méthodes permettent de construire une surface $C^1$ sur l'ensemble du domaine de définition de la grille. Mais c'est deux méthodes se basent sur une définition de la monotonie qui oblige la surface à être monotone dans deux directions différentes.

### 3.1.3 Conclusion

Nous avons vu que les méthodes d'interpolation sont plus intéressantes lorsqu'elles sont locales et permettent d'avoir une surface résultante de continuité au moins $C^1$. Notre problème étant la construction d'une surface monotone à l'intérieur de chaque 2-cellule du complexe de Morse-Smale considéré, les méthodes dîtes de "Shape Preserving" qui concernent la préservation de la monotonie semblent correspondre à notre besoin. De plus, une fonction restreinte sur une 2-cellule de son complexe de Morse-Smale est contrainte seulement dans une seule direction de monotonie. Dans cette thèse, nous proposons une modification de la division de Sibson qui répondra bien aux critères énoncés.

## 3.2 État de l'art sur les complexes de Morse-Smale

Cette section porte sur l'état de l'art en ce qui concerne les travaux sur les complexes de Morse-Smale. Tout d'abord, en partie 3.2.1, nous verrons comment sont calculés les complexes de Morse-Smale. Ensuite, la sous-section 3.2.2 va s'intéresser au premier article traitant de la reconstruction de fonction à partir d'un complexe de Morse-Smale simplifié. Celui-ci applique une série de lissages Laplacien afin de reconstruire la fonction. Suivra en section 3.2.3 les méthodes de reconstructions de fonctions à partir de complexes de Morse-Smale en appliquant une optimisation sur le maillage.

### 3.2.1 Calcul des complexes de Morse-Smale

La problématique du calcul des complexes de Morse-Smale est d'avoir une représentation du gradient de la fonction afin que la géométrie et la connexité du complexe soient les meilleures possible. Le but a alors été d'améliorer cette représentation et de gérer des données dont la taille est de plus en plus grande. Le premier algorithme robuste a été présenté par Edelsbrunner *et al.* [14]. La méthode présentée permet de calculer des complexes de Morse-Smale pour des fonctions définies sur des variétés linéaires par morceaux. Dans l'article [7], l'algorithme permet d'avoir une géométrie plus précise, notamment grâce à la division des triangles. En effet, les 1-cellules du complexe de Morse-Smale pouvant être arbitrairement proches, leur proximité peut être inférieur à la taille des triangles où est définie la fonction. Dans l'algorithme précédent, cela a pour conséquence de joindre les 1-cellules qui sont trop proches, alors que cette méthode pose des conditions sous lesquelles deux 1-cellules peuvent fusionnées.

La deuxième approche pour le calcul des complexes de Morse-Smale est de faire le calcul sur des fonctions discrètes, comme nous avons vu à la partie 2.4. Tout d'abord, il a fallu construire des fonctions de Morse. L'article [34] permet de construire une fonction de Morse sur une 2-variété de façon optimale, c'est-à-dire que la fonction va avoir un nombre minimal de points critiques. Ensuite, une méthode pour définir une fonction de Morse sur un complexe à partir de valeurs données aux 0-cellules du complexe a été donnée dans l'article [33]. Ensuite sont venues des méthodes permettant de construire des complexes de Morse-Smale discrets. À partir de valeurs données aux sommets d'un maillage, Gyulassy *et al.* [25] calculent un gradient discret vérifiant les conditions pour qu'il corresponde à celui d'une fonction de Morse. Le complexe de Morse-Smale est ensuite déduit du gradient discret en utilisant une approche « diviser pour régner » sur le domaine de définition de la fonction. De même l'article [44] propose une méthode de définition d'une fonction de Morse discrète à partir des valeurs aux sommets du complexes. Les valeurs de la fonction sur le complexe ne sont pas définies explicitement, mais à partir d'un coefficient $\epsilon$strictement supérieur à 0. Cela leur permet alors de calculer un

gradient discret à la fonction ainsi que le complexe de Morse-Smale de la fonction. Ce calcul utilise aussi une approche diviser pour régner, mais la gestion des bords permet de paralléliser l'algorithme.

Le gradient discret d'une fonction de Morse tel que nous l'avons décrit à la partie 2.4 suit les directions données par le complexe sur lequel est définie la fonction. Par exemple si la fonction est définie sur une grille régulière découpant le domaine de définition en carrés, il y a alors deux directions possibles pour le gradient qui correspondent aux deux axes de la grille. Si la fonction échantillonnée a un gradient constant sur une partie connexe de son domaine et que le gradient n'est pas colinéaire aux directions de la grille, celui-ci sera alors approximé par l'une de ces directions sur l'ensemble où le gradient est constant. Partant de ce constat, les articles [26, 39] proposent des approches probabilistes permettant une meilleure approximation du gradient. En effet, le coeur des algorithmes qui construisent les paires de cellules décrivant le gradient est la fonction permettant de choisir qu'elle est la paire la plus susceptible de représenter correctement le gradient. Dans ces deux articles, ces fonctions intègrent aussi une probabilité aux paires d'être celle qui sera choisie pour être dans le champ de gradient. Ensuite, un tirage aléatoire est effectué utilisant les probabilités des paires. Cela permet d'être plus proche de la géométrie de la fonction initiale.

### 3.2.2 Reconstruction par lissages Laplacien itératifs

La première méthode de reconstruction de fonctions à partir de complexes de Morse-Smale simplifiés a été introduite par Bremer *et al.* [7] en 2004. Elle tire partie du *principe du maximum faible*. Ce principe énonce des propriétés pour les fonctions solutions d'équations différentielles elliptiques : le maximum d'une telle fonction $f$ sur l'adhérence de son domaine de définition $\overline{\Omega}$ est inférieur au maximum de la fonction $max(f, 0)$ sur la frontière de son domaine de définition $\partial\Omega$. Les fonctions harmoniques sont un cas particulier de ce type de fonctions, ce sont les fonctions $u$ qui sont solutions de l'équation laplacienne $\triangle u = 0$. Elles vérifient donc le principe du maximum faible, mais elles possèdent une caractérisation plus précise sur l'emplacement de leur maximum. En effet, le principe du maximum faible indique que le maximum d'une fonction harmonique se trouve nécessairement sur la frontière de son domaine de définition. Le même résultat pour les fonctions harmoniques reste valable en remplaçant maximum par minimum.

Cette méthode va appliquer une série de lissages sur le maillage définissant la fonction $f$ après suppression d'une paire de points critiques jusqu'à ce que le complexe de Morse-Smale soit simplifié au niveau souhaité par l'utilisateur. Soit $(p_1, p_2)$ la paire de points critiques supprimée du complexe de Morse-Smale . La persistence de cette paire est $p = |f(p_1) - f(p_2)|$. Les étapes permettant la construction de la fonction après la suppression de cette paire sont :

1. Trouver les 1-cellules affectées par la suppression. C'est-à-dire les 1-cellules ayant l'un des deux points critiques comme face.
2. Lisser le gradient de la fonction le long de ces 1-cellules de telle façon que la différence en valeur absolu entre la fonction avant le lissage et après le lissage soit inférieur à la valeur de persistence $p$.
3. Appliquer une série de lissages Laplacien[47] sur les surfaces définies sur les 2-cellules dont les faces ont été modifiées à l'étape précédente. Ces lissages sont appliqués tant que la surface n'est pas monotone.
4. Calcul du complexe de Morse-Smale pour la fonction modifiée.

Les étapes 2 à 4 sont effectuées à nouveau tant que l'erreur entre la fonction en entrée de l'algorithme et la fonction lissée le long des 1-cellules n'est pas inférieure à $p$.

L'étape 3 consiste en une applications de lissages Laplacien sur le maillage à l'intérieur des 2-cellules concernées sans modifier les 1-cellules les bordant. À chaque application du lissage on modifie les points internes aux 2-cellules en modifiant leurs positions : pour un sommet de la triangulation, sa position devient la position moyenne des sommets liés par une arête de la triangulation à ce sommet. Les 1-cellules bordant une des 2-cellules modifiées possèdent un seul maximum et un seul minimum. Par le principe du maximum faible, l'application de lissages Laplacien sur la surface de la 2-cellule converge vers une surface ne possédant pas de point critique sur l'intérieur de son domaine de définition.

Cette méthode permet d'avoir en résultat une surface globalement $C^0$ et $C^1$ à l'intérieur des 2-cellules. Les auteurs n'ont pas de résultats théoriques sur la convergence de leur algorithme pour l'application d'une étape de reconstruction après la suppression d'une paire de points critiques. Ils écrivent que les expériences qu'ils ont menées ont convergé en un « nombre constant d'itérations ». De plus, ils ont noté que la convergence du lissage Laplacien est lente. Enfin, le résultat obtenu est dépendant du maillage initial. En effet, le lissage Laplacien agit sur les sommets en tenant compte des arêtes du maillage. La connectivité, ainsi que le nombre initial de sommets du maillage auront donc une influence sur le résultat final de cette méthode.

### 3.2.3 Reconstruction par résolution d'une minimisation sur l'ensemble du maillage

Nous avons vu à la partie précédente une méthode qui applique des lissages itératifs sur la fonction initiale pour que la fonction lissée corresponde au complexe de Morse-Smale donné en entrée et que les restrictions de la fonction lissée aux 2-cellules soient monotones. Une deuxième méthode est de faire la minimisation d'une énergie sur l'ensemble du maillage avec des inégalités encodant les contraintes de monotonie le long des arêtes de la grille afin de trouver une fonction globalement $C^1$. Nous allons tout d'abord présenter les travaux initiaux, puis deux extensions.

#### Article initial

Weinkauf *et al.* [50] proposent deux méthodes de reconstruction de fonction à partir de complexes de Morse-Smale simplifiés. Ces deux méthodes prennent en entré une fonction échantillonnée sur une triangulation et un complexe de Morse-Smale simplifié calculé par une méthode utilisant une extension de la fonction en une fonction de Morse discrète[25]. La première méthode permet d'avoir en sortie une reconstruction $C^0$, alors que la deuxième donne en sortie une surface $C^1$. Les deux méthodes ont en sortie une surface qui est elle aussi définie sur une triangulation : la première sur la triangulation initiale et la deuxième sur la triangulation initiale dont certains points auront été déplacés. Elles sont toutes les deux décomposées en deux étapes. D'abord la fonction le long des 1-cellules est reconstruite. Ensuite, la restriction de la fonction aux domaines des 2-cellules est calculée.

***Prévisualisation $C^0$ :***

Cette méthode permet d'avoir une reconstruction $C^0$ de la fonction de façon interactive. De plus, cette fonction sera utilisée par la méthode présentée au paragraphe suivant afin d'obtenir une reconstruction $C^1$. La première étape de cette méthode est de linéariser les valeurs de la fonction reconstruite le long des 1-cellules entre les points critiques. Ensuite, l'équation de

Laplace $\triangle f = 0$ est résolue pour chaque 2-cellule avec comme conditions de bords les valeurs de la fonction sur les 1-cellules et aux points critiques. D'après le principe du maximum, une telle équation ayant des conditions de bords monotones telles que celles que nous avons ici permet d'obtenir comme résultat une surface ne contenant pas de point critique à l'intérieur de son domaine de définition. Cette résolution passe par une discrétisation de l'opérateur Laplacien sur la triangulation de la 2-cellule et consiste alors en une résolution d'un système linéaire creux. Elle est $C^\infty$ sur les 2-cellules et $C^0$ à travers les 1-cellules.

***Reconstruction $C^1$ :***

Cette reconstruction utilise la prévisualisation $C^0$ afin de construire une surface $C^1$ cohérente avec le complexe de Morse-Smale simplifié donné en entrée. Elle est composée de deux étapes. Premièrement, les 1-cellules sont lissées dans le plan $(x, y)$. Pour cela la minimisation d'une somme de trois énergies est effectuée. Celle-ci contient une énergie correspondant à un lissage Laplacien 1D des points de la triangulation le long des 1-cellules, ainsi qu'un terme permettant de ne pas avoir d'inversion de triangle dans la triangulation résultante et un terme imposant aux points de la triangulation de ne pas bouger trop par rapport à la triangulation initiale. Cette minimisation a pour contrainte de ne pas changer la position des points critiques.

Deuxièmement, une autre minimisation est appliquée. Celle-ci consiste en une énergie bi-laplacienne sommée avec un terme pour que la reconstruction soit proche de la fonction initiale. Soit $\tilde{\mathbf{f}} = (\tilde{\mathbf{f_1}}, \ldots, \tilde{\mathbf{f_n}})$ la fonction initiale aux sommets du maillage, l'énergie à minimiser en fonction de $\mathbf{f} = (\mathbf{f_1}, \ldots, \mathbf{f_n})$ est

$$E(f) = \|\mathscr{L}\mathbf{f}\|^2 + \omega_d \|\mathbf{f} - \tilde{\mathbf{f}}\|, \tag{3.2}$$

où $\mathscr{L}$ est l'opérateur Laplacien et $\omega_d > 0$ est le poids de l'approximation des valeurs de la fonction initiale. L'interpolation de la valeur de la fonction aux points critiques est imposée par la condition de Dirichlet $\mathbf{f_i} = \tilde{\mathbf{f}}_\mathbf{i}$ avec $i$ les indices des points critiques. La solution de cette minimisation ne correspond pas à une fonction harmonique, il faut donc ajouter des contraintes de monotonie pour que la solution soit compatibles avec le complexe de Morse-Smale donné. Les auteurs ont encodé ces contraintes par un graphe de sommets les points de la triangulation et d'arcs les arêtes de la triangulation, chaque arc de ce graphe orienté indique la relation « la valeur à la pointe de l'arc est inférieure à la valeur à l'origine de l'arc ». Tout sommet de la triangulation ayant plusieurs arcs pointés vers celui-ci est un point critique de la triangulation. Ce graphe est calculé à partir de la prévisualisation $C^0$ proposée dans le même article. Résoudre cette minimisation avec les contraintes issues du graphe de monotonie (qui sont des inégalités) revient alors à résoudre un problème quadratique. Les auteurs modifient alors les inégalités en égalités non-linéaires, ce qui leur donne un problème d'optimisation non-linéaire. Cette minimisation est tout d'abord appliquée en 1D sur les 1-cellules reliant les minima et les points cols avec comme contraintes supplémentaires d'interpoler les valeurs aux points critiques. Elle est ensuite de nouveaux appliquée sur l'ensemble de la triangulation avec cette fois-ci, les valeurs calculées précédemment ainsi que les valeurs aux points critiques comme contraintes supplémentaires aux graphe de monotonie.

Cette méthode permet d'obtenir une reconstruction $C^1$ du complexe de Morse-Smale simplifié. Mais cette méthode demande un long temps de calcul. De plus, la reconstruction est effectuée sur une triangulation de densité similaire à celle de la triangulation initiale. Cela signifie que la fonction recosntruite ne peu pas être évaluée en tout point du domaine de définition. Les auteurs ont remarqué que le résultat de leur méthode est dépendant du graphe de monotonie utilisé pour contraindre la minimisation de l'énergie bi-laplacienne. La construction de

ce graphe à partir de la prévisualtion $C^0$ est un exemple permettant d'obtenir un graphe valide, un autre graphe peut donc être utilisé tant que celui-ci est topologiquemen cohérent avec le complexe de Morse-Smale donné.

### Extensions

Les deux articles présentés ici sont deux extensions successives de [50]. Ils ont tout les deux fait le choix de relâcher certaines des contraintes données par le complexe de Morse-Smale. En effet, tout les points critiques ne sont pas interpolés, seuls les minima et les maxima le sont. Les points cols peuvent donc être déplacés pendant le processus. La deuxième contrainte n'étant plus prise en compte est la connectivité faite entre les points critiques par les 1-cellules. Ces méthodes font donc de l'interpolation de maxima et minima tout en restant proche de la fonction initiale.

#### *Première extension : relâche de la contrainte des 1-cellules et des points cols*

Cette extension correspond à l'article de Jacobson *et al.* [31]. Celle-ci a pour principale application la déformation de forme pour l'animation. Les auteurs cherchent donc à calculer des fonctions de déformations sur l'intérieur de leurs formes tout en interpolant des valeurs imposées qui sont des extréma pour la fonction. Ils proposent aussi une méthode de lissage de fonctions en lien avec les complexes de Morse-Smale. À partir de la fonction à lisser, son complexe de Morse-Smale est calculé puis simplifié. Les auteurs vont alors chercher une fonction $C^1$ interpolant les minima et maxima du complexe de Morse-Smale simplifié. Ils n'ont alors plus besoin de tenir compte des 1-cellules et des points cols du complexe de Morse-Smale. La fonction en sortie correspond à une minimisation de l'énergie définie dans l'article de Weinkauf *et al.* [50] (3.2). Les auteurs posent deux contraintes sur leur minimisation : une contrainte d'interpolation des extrema et une contrainte de monotonie. Contrairement à l'article de Weinkauf *et al.* [50], ceux-ci ne définissent pas un graphe monotonie, mais il vont aussi utiliser le résultat de la prévisualisation $C^0$ de Weinkauf *et al.* [50] afin de définir les contraintes de monotonie.

Soit $u$ la fonction correspondant à la prévisualisation $C^0$ de Weinkauf *et al.* [50], dite fonction *représentante*. Le problème de minimisation va alors aligner la direction du gradient de la fonction solution $f$ avec la direction de $\bigtriangledown u/\|u\|$. La condition contraignant cette direction est alors : pour deux points $i$ et $j$ liés par une arête de la triangulation, il faut $(f_i - f_j)(u_i - u_j) > 0$. Le problème de minimisation est alors un problème convexe.

Le principal intérêt de cette méthode est d'avoir changé les contraintes de monotonie de l'optimisation de Weinkauf *et al.* [50] afin d'obtenir une optimisation convexe. Cela permet d'avoir un résultat en un temps plus rapide. La fonction représentante n'est pas unique, les auteurs ont donc fait le choix d'utiliser la prévisualisation $C^0$ de Weinkauf *et al.* [50], mais d'autres choix sont possibles. Cette méthode n'a pas toutes les contraintes permettant de reconstruire une fonction à partir d'un complexe de Morse-Smale simplifié. En effet, les points cols ne sont plus interpolés et les 1-cellules du complexe de Morse-Smale ne sont plus prises en compte lors de la reconstruction.

#### *Deuxième extension : Optimisation avec changement de graphe de monotonie*

Günther *et al.* [23] part de la conclusion que la méthode de Jacobson *et al.* [31] présentée précédemment n'est pas efficace pour les fonctions définies sur un grande région de $\mathbb{R}^2$ ou encore pour les fonctions définies sur une partie de $\mathbb{R}^3$. De plus, la solution trouvée par la

méthode précédente ne correspond dans la plupart des cas à un minimum local de l'énergie :

$$E(f) = \|\mathscr{L}\mathbf{f}\|^2 + \omega_d\|\mathbf{f} - \tilde{\mathbf{f}}\|,$$

où l'opérateur $\mathscr{L}$ correspond à l'opérateur laplacien, et $\tilde{f}$ est la fonction initiale. La minimisation de cette énergie se fait sous la contrainte que les minima et maxima de la fonction que l'utilisateur souhaite conserver sont interpolés. Afin de chercher le minimum de ce problème, Günther *et al.* utilise le graphe de monotonie défini par Weinkauf *et al.* [50].

La méthode se décompose en deux étapes. La première est le calcul d'un graphe de monotonie initial. Pour cela, les auteurs calculent le gradient discret de la fonction de Morse ayant une résolution $1/2$ par rapport à la grille où est définie la fonction initiale. Les sommets du maillage initial sont alors en correspondance avec les cellules du complexe de la fonction discrète, les valeurs de la fonction sur les cellules du complexe sont les mêmes que celle du sommet correspondant. Il faut ensuite supprimer les extrema non désirés. Pour cela les 1-cellules du complexe de Morse-Smale de la fonction de Morse discrète sont calculées, ainsi que les points cols de cette fonction. Pour supprimer un des extrema, le gradient le long de la 1-cellule ayant comme face cet extrema et le point col ayant la plus faible persistence est inversé. Après la suppression des extrema, le gradient est projeté sur la grille initiale afin d'obtenir le graphe de monotonie initial.

Ensuite, l'énergie $E(f)$ est minimisée sous les contraintes :

$$\begin{aligned} f(v_i) &> f(v_j) \;\; \forall (v_i, v_j) \text{ arête du graphe de monotonie} \\ f(v_i) &= \tilde{f}(v_i) \;\; \forall v_i \text{ extrema à interpoler} \end{aligned}$$

L'utilisation du graphe de monotonie dans la résolution du problème de minimisation permet de passer d'un problème non-linéaire à un problème convexe. Mais celui ne donne pas la solution du problème initial. Pour se rapprocher de cette solution, Günther *et al.* proposent alors de recommencer la méthode depuis la première étape en utilisant la fonction trouvée à cette deuxième étape pour calculer le nouveau graphe de monotonie. Ces deux étapes sont itérées tant que l'énergie n'a pas convergé. Comme la fonction est décroissante pour chaque optimisation effectuée, celle-ci est garantie de converger.

En plus de ces résultats, les auteurs proposent une méthode de décomposition du domaine de définition de la fonction afin de réduire le temps de calcul et l'utilisation de la mémoire. Ce temps va dépendre de la taille des blocs qui décomposent le domaine et qui est donnée en entrée de l'algorithme. Ceci donne de bon résultats pour des domaines dont la taille est plus importante que des domaines $256 \times 256$, mais surtout pour les domaines 3D.

Cette méthode permet d'avoir un temps de calcul plus rapide que dans les méthodes précédentes pour construire une fonction approximant une fonction initiale contenant seulement les extrema voulus. De plus, cette méthode peut être utilisée sur des fonctions définies sur un domaine de $\mathbb{R}^3$. Une contrainte de cette méthode est que le maillage de la fonction initiale doit être un maillage régulier.

### 3.2.4 Conclusion

Dans cette partie, nous avons vu que les méthodes permettant de reconstruire des fonctions scalaires à partir de complexes de Morse-Smale simplifiés [7, 50] sont dépendantes de la triangulation initiale de la fonction. La triangulation initiale a donc une importance quant au temps de calcul de la reconstruction. De plus, on ne pourra évaluer la fonction qu'en les points où la fonction initiale a été définie. Dans notre méthode nous allons construire une surface

polynomiale par morceaux qui pourra être évaluée en tout point du domaine de définition de la fonction reconstruite. Nous reprenons la même structure de ces algorithmes, c'est-à-dire que notre méthode débute par un lissage des 1-cellules et calcule ensuite une surface définie sur les 2-cellules.

Quant aux deux extensions [31, 23] que nous avons présentées, elles interpolent seulement les minima et maxima. La reconstruction n'est donc pas compatible avec le complexe de Morse-Smale de la fonction. Notre méthode s'éloigne donc de ces méthodes en construisant une fonction compatible avec le complexe de Morse-Smale simplifié de la fonction initiale.

## 3.3 CONCLUSION

Dans cette thèse, nous construisons une fonction compatible avec un complexe de Morse-Smale simplifié calculée à partir d'une fonction initiale. Cela signifie que les points critiques de la fonction reconstruite sont uniquement ceux du complexe de Morse-Smale , que les points critiques sont agencés comme dans le complexe de Morse-Smale et qu'à l'intérieur de chaque cellule du complexe de Morse-Smale la fonction est monotone.

Nous avons vu deux méthodes qui construisent une fonction à partir d'un complexe de Morse-Smale simplifié et d'une fonction initiale [7, 50]. Ces deux méthodes utilisent le *principe du maximum faible* pour construire la fonction reconstruite. De plus, elles commencent toutes les deux par travailler sur les 1-cellules puis reconstruisent la fonction. La méthode de Bremer *et al.* [7] va alors faire la reconstruction 2-cellule par 2-cellule alors que la méthode de Weinkauf *et al.* [50] applique une optimisation sur tout l'ensemble de définition de la fonction. En sortie de la première méthode, la fonction est $C^\infty$ à l'intérieur des 2-cellules et $C^0$ à travers les 1-cellules. Quant à la deuxième méthode, la reconstruction est globalement $C^1$ sur le domaine de définition. La première méthode ne possède pas de garantie théorique sur sa convergence, et demande un long temps de calcul. La méthode de Weinkauf *et al.* [50] résout un problème d'optimisation non-linéaire afin de trouver la fonction reconstruite. Pour cela, un graphe de monotonie est construit en calculant une reconstruction $C^0$ de la fonction. Comme le montre la méthode de Günther *et al.* [23], la solution de cette optimisation est dépendante de ce graphe. De plus, la solution trouvée ne correspond pas à la solution du problème initial puisque le graphe de monotonie ajoute des contraintes supplémentaires. Mais le temps de calcul en utilisant une technique similaire à celle de Günther *et al.* [23] deviendrait trop élevé. De plus, ces deux méthodes sont basées sur le maillage initiale de la fonction. Le temps de calcul est donc corrélé avec la taille du maillage. De plus, la fonction reconstruite ne pourra être évaluée seulement aux points du maillage initial de la fonction. Dans cette thèse, nous enlevons la corrélation au maillage initial en construisant une surface polynomiale par morceaux sur les 2-cellules. Cela nous permet d'évaluer la fonction reconstruite en tout point du domaine de définition ainsi que d'avoir un temps de calcul pour la fonction reconstruite qui ne dépend pas de la taille des 2-cellules.

Dans cette thèse, nous présentons des contributions dans le domaine de l'interpolation monotone de données définies sur une grille. Ces contributions sont :

- un nouvel interpolant de type division en triangles (méthode de *splitting*),
- une méthode d'interpolation monotone selon une direction diagonale :
  - des conditions suffisantes sont données sur les valeurs de fonction et les valeurs de gradients dans les directions $x$ et $y$ afin que la surface correspondant à notre interpolant soit monotone,

- deux algorithmes permettant de calculer ou de modifier les valeurs de gradients aux points de la grille pour que celles-ci vérifient les conditions suffisantes de monotonie sont proposés.

Quant à la reconstruction de fonction à partir de complexes de Morse-Smale simplifiés, nous proposons une méthode décomposée en deux parties. Premièrement, nous approximons les 1-cellules par des courbes de Bézier monotone. Ensuite, à l'intérieur de chaque 2-cellule, nous construisons une surface polynomiale par morceaux ne contenant aucun point critique sur l'intérieur de son domaine de définition.

CHAPITRE

# 4

# INTERPOLATION MONOTONE DE DONNÉES SUR UNE GRILLE

## 4.1 INTRODUCTION

Le sujet de cette thèse est la construction d'une surface en cohérence avec un complexe de Morse-Smale . En contraste avec toutes les méthodes précédentes de reconstruction de fonction à partir de complexes de Morse-Smale simplifiés (voir chapitre 3), nous allons proposer une nouvelle approche issue des méthodes dîtes "Shape-preserving" en CAGD. La méthode que nous allons proposer en chapitre 5 décomposera le problème en calculant une surface polynomiale par morceau pour chaque cellule du complexe de Morse-Smale individuellement. Une des contraintes topologiques qu'impose le complexe de Morse-Smale est l'absence de tout point critique à l'intérieur de chaque cellule. Pour cette raison, nous proposons d'utiliser des surfaces monotones dont le gradient doit être positif en tout point. Notre choix d'interpolant se porte sur une méthode de division de type division de Sibson pour plusieurs raisons. Tout d'abord, cette méthode est locale, ce qui signifie que le changement d'une valeur de fonction ou de gradient en un point de la grille va engendrer un changement de la surface résultante seulement au voisinage de ce point de la grille. De plus, la continuité d'une surface construite par cette interpolant sur une grille est de continuité globalement $C^1$. Ensuite, aucune méthode que nous avons montrée au chapitre 3 ne correspond à notre besoin. En effet, nous voulons une méthode étant monotone selon la direction $(x + y)$. Or, les méthodes que nous avons présentées sont monotones soit dans la direction $x$, soit dans la direction $y$, ou encore dans les deux directions simultanément. Nous ne pouvons donc pas les utiliser pour résoudre notre problème.

Dans ce chapitre, nous allons présenter nos contributions au problème mathématique sous-jacent qui est le calcul d'une surface monotone interpolant les valeurs de fonction à deux variables données sur une grille rectangulaire 2D dans le domaine de définition. Contrairement aux méthodes existantes définissant la monotonie selon les deux axes du plan, nous allons baser notre interpolant sur une autre définition de la monotonie, une monotonie selon une direction diagonale dans le plan, afin de satisfaire les contraintes topologiques imposées par le complexe de Morse-Smale. Nous proposons notamment une modification de l'interpolant de Sibson qui en cohérence avec la monotonie en diagonale permet de développer des conditions suffisantes sur les données à interpoler. Dans un souci d'efficacité en terme de temps de calcul, sachant que notre interpolant devra être utilisé des centaines voir des milliers de fois pour la reconstruction d'un complexe de Morse-Smale , nous développons deux algorithmes très simples, mais rapides, pour le calcul de notre interpolant de Sibson modifié. Le problème d'interpolation que nous allons résoudre dans ce chapitre se pose finalement ainsi :

**Problème d'interpolation :**
Soit donné un ensemble de données $(x_i, y_j; z_{ij})$, où les valeurs d'abscisses se trouvent aux sommets d'une grille rectangulaire régulière $G = \{(x_i, y_j), i \in \{1, \ldots, n_i\} \wedge j \in \{1, \ldots, n_j\}\}$ avec $h^x = x_{i+1} - x_i$ pour tout $i$, $h^y = y_{j+1} - y_j$ pour tout $j$ et $z_{ij}$ sont les valeurs de fonction associées. On cherche une fonction

$$f : D = [x_1, x_{n_i}] \times [y_1, y_{n_j}] \subset \mathbb{R}^2 \to \mathbb{R} \quad \text{telle que}$$

$$f \in C^1(D), \quad \text{f est monotone et} \quad f(x_i, y_j) = z_{ij}.$$

Par monotone nous entendons pour l'instant de façon générale que $f$ ne possède ni maximum, ni minimum, ni point col sur $int(D)$.

En section 4.2.1 seront présentées les notions de base sur les surfaces de Bézier triangulaires ainsi que leur emploi dans la définition de l'interpolation de Sibson. Inspiré de la méthode de l'article [27] nous allons introduire un nouvel interpolant respectant des conditions de monotonie. Après avoir donné deux définitions de monotonie en section 4.5, suivent trois sections

avec nos contributions. Nous présenterons en section 4.4 un nouveau patch de Sibson modifié. Des conditions sur les dérivées partielles aux points de la grille en entrée sont dérivées en section 4.5 et permettent de définir des dérivées admissibles pour lesquelles les patchs de Sibson modifiés résulent en une surface monotone. En section 4.6 nous introduisons deux algorithmes permettant de créer ou de modifier les dérivées partielles aux points de la grille pour qu'elles satisfassent les conditions de monotonie. Enfin, nous présenterons nos résultats en section 4.7.

Ces travaux ont été présentés au workshop « Workshop on the analysis of large-scale, high-dimensional, and multivariate data using topology and statistics » au Barp du 12 au 14 juin 2013. et ont donnés lieu à une publication comme chapitre de livre [5].

## 4.2 Interpolation de Sibson

Cette section rappelle d'abord les notions de base sur les surfaces de Bézier triangulaires et la continuité $C^1$ qui seront utilisées au cours de ce chapitre. Ensuite, la méthode originale de l'interpolation par découpage de Sibson du domaine de définition sera présentée. Plus de détails peuvent être trouvés dans [15].

### 4.2.1 Surfaces de Bézier triangulaires

Cette section rappelle les résultats classiques concernant les surfaces de Bézier triangulaires Soit $T \subset \mathbb{R}^2$ un triangle non dégénéré du plan. Celui-ci est défini par trois points non-colinéaires $\mathbf{p}_0$, $\mathbf{p}_1$ et $\mathbf{p}_2$. Un point $\tau$ de $T$ peut alors être défini par ses coordonnées barycentriques par rapport aux sommets du triangle $\tau = (\tau_0, \tau_1, \tau_2)$. Une surface de Bézier (aussi appellée patch de Bézier) triangulaire de degré 3 est une fonction polynomiale définie par la fonction $f : T \to \mathbb{R}$ :

$$f(\tau) = \sum_{\substack{i+j+k=3 \\ i,j,k \geqslant 0}} b_{ijk}\, B^3_{ijk}(\tau), \qquad \tau \in T \tag{4.1}$$

où $b_{ijk} \in \mathbb{R}$ sont les coefficients (ordonnées) de Bézier correspondent aux abscisses ayant $(\frac{i}{3}, \frac{j}{3}, \frac{k}{3})$ comme coordonnées barycentriques. $B^3_{ijk}(\tau) = \frac{3!}{i!j!k!}\tau_0^i\tau_1^j\tau_2^k$ sont les polynômes de Bernstein généralisés de degré 3, voir figure 4.1.
Les **dérivées partielles d'ordre 1** des polynômes de Bernstein sont données par :

$$\begin{aligned}
\frac{\partial B^3_{ijk}}{\partial \tau_0}(\tau) &= 3\left(B^2_{i-1jk}(\tau) - B^2_{ijk-1}(\tau)\right) \\
\frac{\partial B^3_{ijk}}{\partial \tau_1}(\tau) &= 3\left(B^2_{ij-1k}(\tau) - B^2_{i-1jk}(\tau)\right) \\
\frac{\partial B^3_{ijk}}{\partial \tau_2}(\tau) &= 3\left(B^2_{ijk-1}(\tau) - B^2_{ij-1k}(\tau)\right)
\end{aligned}$$

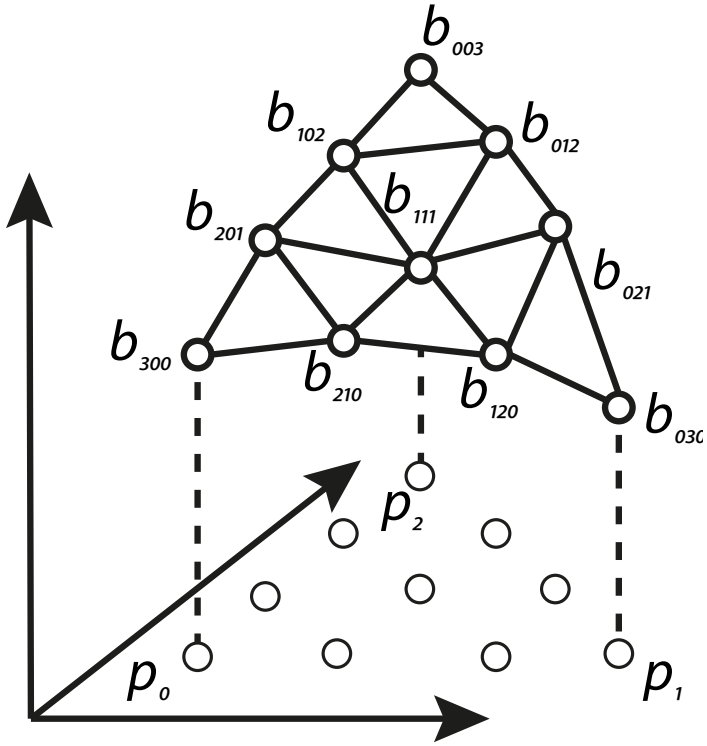



**FIGURE 4.1 –** *Points de contrôles (abscisses, ordonnées) d'une surface de Bézier triangulaire.*

avec $B^2_{ijk}(\tau) = \frac{2!}{i!j!k!}\tau_0^i\tau_1^j\tau_2^k$. Ainsi, les dérivées partielles d'ordre 1 de $f$ par rapport aux coordonnées barycentriques de $T$ sont données par :

$$\begin{aligned}
\frac{\partial f}{\partial \tau_0}(\tau) &= 3 \sum_{\substack{i+j+k=2 \\ i,j,k\geqslant 0}} (b_{i+1jk} - b_{ijk+1})\, B^2_{ijk}(\tau) \\
\frac{\partial f}{\partial \tau_1}(\tau) &= 3 \sum_{\substack{i+j+k=2 \\ i,j,k\geqslant 0}} (b_{ij+1k} - b_{i+1jk})\, B^2_{ijk}(\tau) \\
\frac{\partial f}{\partial \tau_2}(\tau) &= 3 \sum_{\substack{i+j+k=2 \\ i,j,k\geqslant 0}} (b_{ijk+1} - b_{ij+1k})\, B^2_{ijk}(\tau)
\end{aligned} \tag{4.2}$$

Les dérivées partielles d'ordre 1 des surfaces de Bézier triangulaires sont donc des combinaisons linéaires de différences entre coefficients de Bézier.

Ces formules permetttent de caractériser la **continuité entre deux patchs de Bézier**. Soient deux surfaces triangulaires de Bézier (voir fig. 4.2) : $f_1$ de points de coefficients $(b^1_{ijk})_{\substack{i+j+k=3 \\ i,j,k\geqslant 0}}$ définie sur le triangle de points $(T_0, T_1, T_2)$ et $f_2$ de coefficients $(b^2_{ijk})_{\substack{i+j+k=3 \\ i,j,k\geqslant 0}}$ définie sur le triangle $(\hat{T}_0, T_2, T_1)$. $f_1$ et $f_2$ se raccordent avec continuité $C^0$ le long de l'arête $\tau_0 = 0$ si les courbes frontières sont identiques, i.e.

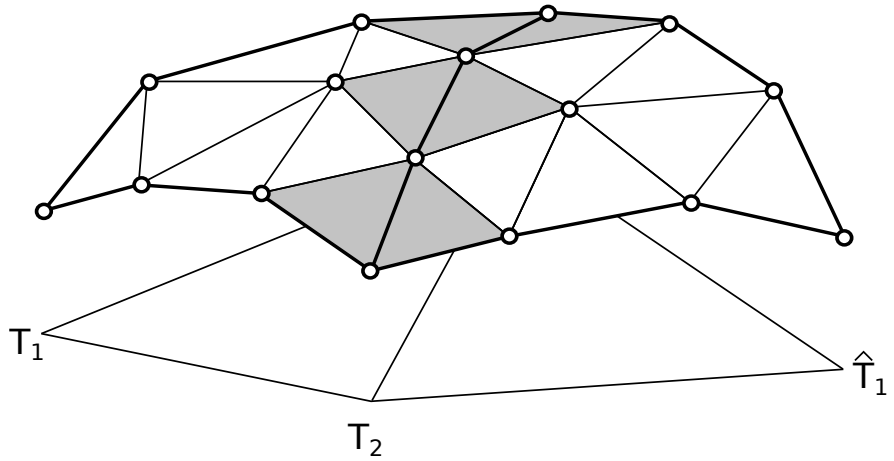



**FIGURE 4.2 –** *Les triangles grisés sont coplanaires deux à deux. Cela permet aux deux surfaces de Bézier triangulaires d'avoir une continuité $C^1$ le long de leur arête commune.*

$$f_1(0,\tau_1,\tau_2) = f_2(0,\hat{\tau}_2,\hat{\tau}_1) \iff \begin{cases} b^1_{003} = b^2_{003} \\ b^1_{012} = b^2_{012} \\ b^1_{021} = b^2_{021} \\ b^1_{030} = b^2_{030} \end{cases} \tag{4.3}$$

De plus, $f_1$ et $f_2$ se raccordent avec continuité $C^1$ lelong de l'arête $\tau_0 = 0$ si

$$Df_1(0,\tau_1,\tau_2) = Df_2(0,\hat{\tau}_2,\hat{\tau}_1) \iff \begin{cases} b^2_{120} = ub^1_{120} + vb^1_{030} + wb^1_{021} \\ b^2_{111} = ub^1_{111} + vb^1_{021} + wb^1_{012} \\ b^2_{102} = ub^1_{102} + vb^1_{012} + wb^1_{003} \end{cases} \tag{4.4}$$

où $(u,v,w)$ sont les coordonnées barycentriques du sommet $\hat{T}_0$ par rapport au triangle composé des points $(T_0,T_1,T_2)$ et $D$ l'opérateur de dérivée directionnelle transverse au bord. Cela signifie que les couples de triangles hachurés en figure 4.2 doivent être coplanaires. La figure 4.2 illustre cette propriété pour deux patchs triangulaires de Bézier adjacents.
Ces résultats seront utilisés dans la suite de ce chapitre pour construire une surface $C^1$ monotone composée de plusieurs patchs triangulaires de Bézier.

### 4.2.2 Interpolant de Sibson

L'idée de la méthode de Sibson est de mettre quatre surfaces de Bézier triangulaires entre quatre points de la grille. En section 4.2.1, nous allons donner les définitions des surfaces de Bézier. La section 4.2.2 est un récapitulatif sur l'interpolant de Sibson. Tout d'abord, nous allons modéliser ? ? notre problème.

Étant données des valeurs de fonction $z_{ij} \in \mathbb{R}$ et les dérivées partielles $z^x_{ij}, z^y_{ij}$ aux sommets d'une grille rectangulaire (comme illustrée en figure 4.3)

$$G = \{(x_i,y_j),\ i \in \{1,\dots,n_i\} \wedge j \in \{1,\dots,n_j\}\}.$$

L'interpolant de Sibson [46] permet d'interpoler ces données par une surface polynomiale par morceaux de continuité $C^1$. Il est basé sur un découpage des rectangles de la grille selon les diagonales, quatre triangles sont alors décrits sur lesquels sont définies quatre surfaces triangulaires de Bézier de degré 3. La figure 4.3 montre la grille $G$ des points du domaine de la

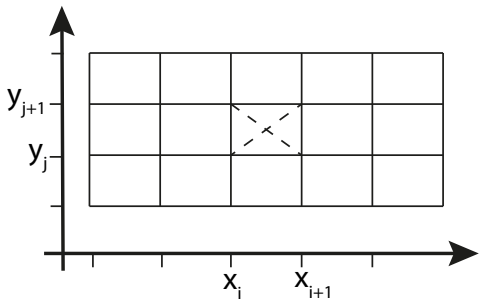



**FIGURE 4.3 –** *Grille $G$ sur le domaine de définition de la fonction $f$. L'un des rectangles est découpé selon ses diagonales, ce qui créé quatre triangles sur lesquels seront définis des surfaces de Bézier triangulaires.*

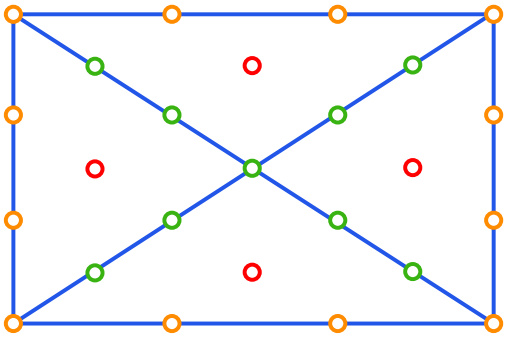


**FIGURE 4.4 –** *Découpage de Sibson (Sibson-Split) d'un rectangle. L'emplacement des coefficients de Bézier est indiqué par des cercles.*

fonction. L'un des rectangles est découpé comme expliqué ci-dessus. Un zoom sur ce rectangle découpé en 4 patchs de Bézier est montré à la figure 4.4 dans laquelle on schématise par des cercles la position dans le plan $(x, y)$ des coefficients (ordonnées) de Bézier. Comme les quatre patchs se recoupent selon des arêtes communes, 25 coefficients suffisent à définir l'interpolant de Sibson pour un rectangle de la grille du domaine. Notons les 25 coefficients $(c_i)_{i=1,\dots,25}$ comme indiqué en figure 4.5.

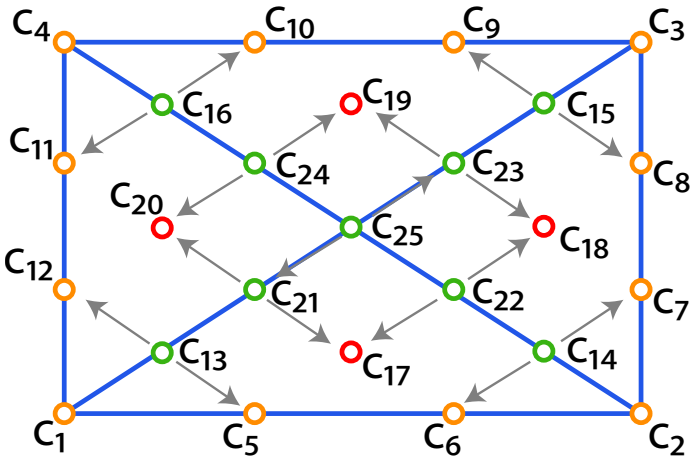



**FIGURE 4.5 –** *Interpolant cubique de Sibson. Les points oranges sont évalués grâce aux données d'entrées, les valeurs aux points rouges sont construits pour avoir les dérivées transverses linéaires. Les points verts sont fixés par les conditions de continuité $C^1$ : chaque valeur aux points verts doit être la moyenne des valeurs des deux points pointés par les flèches partant de celui-ci.*

Nous allons montrer maintenant comment les 25 coefficients de l'interpolant de Sibson sont calculés afin d'obtenir une surface globalement $C^1$ sur le domaine de définition (i.e. la grille en figure 4.3). Ils sont déterminés de manière unique si on impose aux dérivées transverses

au bord des patchs de Sibson d'être linéaires. Les 25 coefficients par rectangle doivent alors satisfaire les conditions d'interpolation et de continuité de façon suivante.

Considérons le patch définit sur le domaine $D_{ij} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$. Les valeurs en entrée sont les valeurs de fonction aux coins du rectangle, notées $z_{i+k,j+l}$ avec $k, l \in \{0, 1\}$ et les dérivées partielles selon $x$ et $y$, notées $z^x_{i+k,j+l}$ et $z^y_{i+k,j+l}$. Notons aussi $h^x_i = x_{i+1} - x_i$ et $h^y_j = y_{j+1} - y_j$ les pas de la grille en $x$ et $y$.
Les coefficients du patch de Sibson doivent alors satisfaire les conditions d'interpolation et de continuité suivantes :
**(1) Interpolation des valeurs aux sommets :**

$$f(x_{i+k}, y_{j+l}) = z_{i+k,j+l}\,, \quad k, l \in \{0, 1\}$$

Les coefficients $c_1$ à $c_4$ sont obtenus grâce à l'interpolation des valeurs de fonction données aux sommets du rectangle.

**(2) Interpolation des valeurs de dérivées partielles aux sommets :**

$$\frac{\partial f}{\partial x}(x_{i+k}, y_{j+l}) = z^x_{i+k,j+l}\,, \quad \frac{\partial f}{\partial y}(x_{i+k}, y_{j+l}) = z^y_{i+k,j+l}\,, \quad k, l \in \{0, 1\}$$

Les coefficients $c_6$ à $c_{12}$ sont fixés par les valeurs de dérivées partielles aux sommets. Les coefficients $c_5,\ c_6,\ c_9$ et $c_{10}$ par les dérivées partielles dans la direction $x$ et les coefficients $c_7,\ c_8,\ c_{11}$ et $c_{12}$ par celles dans la direction $y$.

**(3) Continuité $C^1$ entre les patchs triangulaires de Bézier :** Les coefficients $c_{13}$ à $c_{16}$ et $c_{21}$ à $c_{25}$ sont calculés de façon à avoir la continuité $C^1$ entre les quatre patchs à l'intérieur de l'interpolant de Sibson. Pour satisfaire les conditions de continuité $C^1$ entre les différents patchs, il faut que les points de contrôle proche des bords intérieurs soient coplanaires en vérifiant les relations (4.4).

Par exemple, il faut que les points de contrôles correspondants aux coefficients $c_1,\ c_5,\ c_{12}$ et $c_{13}$ donnés par $(x_i, y_j, c_1),\ (x_i + \frac{1}{3}h^x_i, y_j, c_5),\ (x_i, y_j + \frac{1}{3}h^y_j, c_{12})$ et $(x_i + \frac{1}{6}h^x_i, y_j + \frac{1}{6}h^y_j, c_{13})$ soient coplanaires. Le seul coefficient inconnu à ce stade est $c_{13}$, puisque $c_5$ et $c_{12}$ ont été fixés en (1) et (2). En utilisant la condition de continuité $C^1$ (4.4) on obtient alors $c_{13} = \frac{1}{2}(c_5 + c_{12})$. Tous les autres coefficients intervenant dans les conditions $C^1$ (coefficients en vert en figure 4.5 seront déterminés de même façon, i.e. en prenant la moyenne des coefficients pointés par les flèches qui partent des coefficients verts.

**(4) Continuité $C^1$ entre patchs de Sibson :** Les coefficients $c_{17}$, $c_{18}$, $c_{19}$ et $c_{20}$ permettent d'avoir la continuité $C^1$ entre les patchs de Sibson adjacents (ayant un bord en commun). Pour avoir la continuité $C^1$ il suffit que les dérivées transverses au bord des deux patchs soient égales le long du bord commun. Notons, que la dérivée transverse au bord pour des patchs cubiques est un polynôme de degré 2. Comme les valeurs de dérivée sont déjà connues aux extrémités il suffit alors d'exiger que la dérivée transverse au bord soit linéaire, ce qui détermine de façon unique les coefficients $c_{17}$, $c_{18}$, $c_{19}$ et $c_{20}$.

Les 21 coefficients qui satisfont les conditions (1), (2) et (3) peuvent alors être déterminés de façon explicite par

$$
\begin{array}{lll}
c_1 = z_{ij}, & c_2 = z_{i+1,j}, & c_3 = z_{i+1,j+1}, \\
c_4 = z_{i,j+1}, & c_5 = c_1 + \frac{h^x}{3} z^x_{ij}, & c_6 = c_2 - \frac{h^x}{3} z^x_{i+1,j}, \\
c_9 = c_3 - \frac{h^x}{3} z^x_{i+1,j+1}, & c_{10} = c_4 + \frac{h^x}{3} z^x_{i,j+1}, & c_{11} = c_4 - \frac{h^y}{3} z^x_{i,j+1}, \\
c_7 = c_2 + \frac{h^y}{3} z^y_{i+1,j}, & c_8 = c_3 - \frac{h^y}{3} z^y_{i+1,j+1}, & c_{12} = c_1 + \frac{h^y}{3} z^y_{ij}, \\
c_{13} = \frac{1}{2}(c_5 + c_{12}), & c_{14} = \frac{1}{2}(c_6 + c_7), & c_{15} = \frac{1}{2}(c_8 + c_9), \\
c_{16} = \frac{1}{2}(c_{11} + c_{10}), & c_{21} = \frac{1}{2}(c_{20} + c_{17}), & c_{22} = \frac{1}{2}(c_{17} + c_{18}), \\
c_{23} = \frac{1}{2}(c_{18} + c_{19}), & c_{24} = \frac{1}{2}(c_{19} + c_{20}), & c_{25} = \frac{1}{2}(c_{21} + c_{23}) = \frac{1}{2}(c_{22} + c_{24}),
\end{array}
\tag{4.5}
$$

et les 4 coefficients qui assurent la continuité entre patchs de Sibson adjacents sont donnés par [15]

$$
\begin{array}{ll}
c_{17} & = (2c_{13} + 2c_{14} + c_5 + c_6 - c_1 - c_2)/4, \\
c_{18} & = (2c_{14} + 2c_{15} + c_7 + c_8 - c_2 - c_3)/4, \\
c_{19} & = (2c_{15} + 2c_{16} + c_9 + c_{10} - c_3 - c_4)/4, \\
c_{20} & = (2c_{16} + 2c_{13} + c_{11} + c_{12} - c_4 - c_1)/4.
\end{array}
\tag{4.6}
$$

## 4.3 MONOTONIE AXIALE ET DIAGONALE

Comme nous avons vu dans l'état de l'art, l'interpolation monotone de données est un problème complexe. Les peu de travaux existant se basent sur une définition de la monotonie que nous appelons « axiale ». Or, nous allons, en section 5.4.2, montrer pourquoi nous avons besoin d'une définition moins restrictive de monotonie dans les applications que nous visons dans cette thèse. Nous allons proposer une autre définition plus générale de la monotonie que nous appelons « diagonale ».

### 4.3.1 Monotonie axiale

Les méthodes d'interpolation dites "shape preserving" [10, 9, 27, 51, 19] (voir Etat de l'art en section 3) se basent généralement sur la définition de monotonie [27] de fonction suivante.

**Définition 4.1 (fonction monotone).** Soit $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ une fonction continue. $f$ est *monotone* sur $\Omega$ si pour tout couples $(x_1, y_1),\ (x_2, y_2) \in \Omega$ tels que $x_1 \leqslant x_2$ et $y_1 \leqslant y_2$ on ait $f(x_1, y_1) \leqslant f(x_2, y_2)$.

Cette fonction est dite *strictement monotone* lorsque les inégalités sont des inégalités strictes.

On en déduit que pour une fonction $f$ et pour tout $(x, y) \in \Omega$, on a $\frac{\partial f}{\partial x}(x, y) \geqslant 0$ et $\frac{\partial f}{\partial y}(x, y) \geqslant 0$. Cette définition de monotonie stricte est en accord avec celle que nous avons énoncée , bien qu'elle soit plus restrictive. En effet, une fonction peut ne posséder aucun point critique dans l'intérieur de son domaine de définition mais ne pas être monotone au sens de la définition 4.1. Par exemple, la fonction $f : [0, 1]^2 \to \mathbb{R}$ définie par $f(x, y) = (x - y)^2 + x$ est monotone puisqu'elle ne possède ni maximum, ni minimum, ni point col dans l'intérieur

de son domaine de définition. Pourtant, elle ne satisfait pas les conditions de monotonie de la définition 4.1. En effet, si nous prenons $(x_1, y_1) = (\frac{1}{2}, 0)$ et $(x_2, y_2) = (\frac{1}{2}, \frac{1}{2})$, alors nous avons bien que $x_1 \leqslant x_2$ et $y_1 \leqslant y_2$, mais nous avons aussi $f(x_1, y_1) = \frac{3}{4} > f(x_2, y_2) = \frac{1}{2}$.

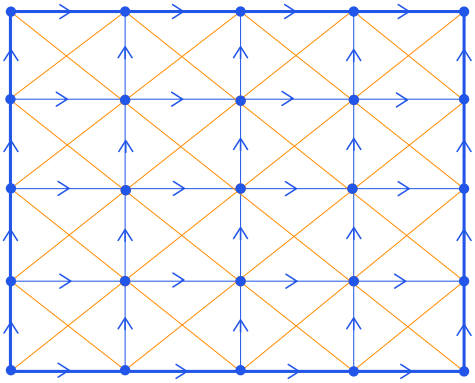


**FIGURE 4.6 –** *Ensemble de données monotones selon les directions $x$ et $y$ (définition 4.1). Les valeurs aux sommets de la grille doivent être croissantes en suivant les flêches bleues.*

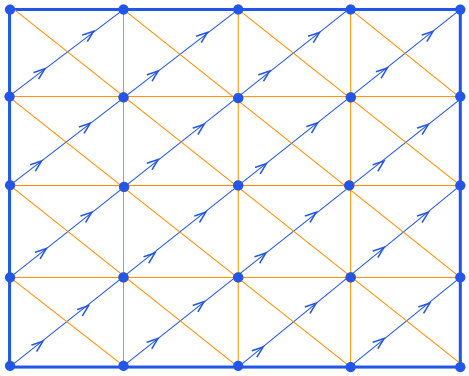


**FIGURE 4.7 –** *Ensemble de données monotones selon les directions $x + y$ (définition 4.3). Les valeurs aux sommets de la grille doivent être croissantes en suivant les flêches bleues.*

Afin de pouvoir reconstuire une fonction monotone à partir de données sur une grille, il faut que les données en entrée soient cohérentes avec la définition 4.1, ce qui mène à la définition de monotonie des données en entrée à la méthode d'interpolation suivante :

**Définition 4.2 (Monotonie axiale).** Soit $G = \{(x_i, y_j),\ i \in \{1, \ldots, n_i\} \wedge j \in \{1, \ldots, n_j\}\}$ une grille rectangulaire de valeurs d'abcisses et $z_{ij}$ les valeurs de fonction associées. L'ensemble de données $\{(x_i, y_j, z_{ij})\}_{\substack{1 \leqslant i \leqslant n_i \\ 1 \leqslant j \leqslant n_j}}$ est dit *monotone* si les valeurs de fonction sont croissant monotone selon les axes $x$ et $y$, i.e. si pour tout couple $(i_1, j_1)$ et $(i_2, j_2)$ tels que $i_1 \leqslant i_2$ et $j_1 \leqslant j_2$ on ait $z_{i_1 j_1} \leqslant z_{i_2 j_2}$.

La figure 4.6 montre de façon schématique comment doivent se comporter les valeurs aux sommets de la grille : elles doivent être croissantes le long des lignes et des colonnes.

### 4.3.2 Monotonie relâchée

L'interpolant de Sibson modifié que nous allons introduire plus tard répond à une définition de la monotonie différente.

**Définition 4.3 (Monotonie diagonale).** Supposons que nous ayons un domaine rectangulaire $D \subset \mathbb{R}^2$ subdivisé en rectangles $D_{ij} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$ avec $1 \leqslant i < n_i$ et $1 \leqslant j < n_j$. De plus, soit donné en chaque point $(x_i, y_j)$ une valeur de fonction $z_{ij} \in \mathbb{R}$. L'ensemble de données $\{(x_i, y_j, z_{ij})\}_{\substack{1 \leqslant i \leqslant n_i \\ 1 \leqslant j \leqslant n_j}}$ est dit *croissant monotone selon la diagonale* si

$$z_{ij} \leqslant z_{i+1,j+1} \tag{4.7}$$

pour tout $1 \leqslant i < n_i$ et $1 \leqslant j < n_j$.

Figure 4.7 montre schématiquement de quelle façon les valeurs $z_{ij}$ aux sommets de la grille doivent se comporter : elles doivent croître selon des diagonales. Cette définition de la monotonie est plus générale que celle donnée en section 4.3.1. En effet, la monotonie diagonale impose seulement que les valeurs données sur la grille soient croissantes le long des diagonales, alors que pour la monotonie axiale, les valeurs doivent être croissantes le long des lignes et des colonnes. La monotonie axiale est inclus dans la monotonie diagonale, car si les valeurs sont croissantes le long des lignes et des colonnes, alors les valeurs sont aussi croissantes le long des diagonales. L'inverse n'est pas vrai puisque l'on peut avoir des valeurs croissantes selon la diagonale qui ne seront pas croissantes selon les lignes et les colonnes.

Une méthode d'interpolation satisfaisant la monotonie diagonale serait donc plus générale qu'une méthode ne gérant uniquement la monotonie axiale. Le reste de ce chapitre est dédié au développement d'une méthode d'interpolation monotone se basant sur la monotonie diagonale. Ce travail a été inspiré par la méthode d'interpolation axiale de Han et Schumaker [27].

### 4.3.3 Problème à résoudre

Avec $n_i \cdot n_j$ données en entrée $(x_i, y_j; z_{ij})$, $i = 1, \dots n_i,\ j = 1, \dots n_j$ comme présenté dans l'introduction de ce chapitre, nous cherchons à calculer une fonction $C^1$-continue $f : D \to \mathbb{R}$ telle que $f$ soit monotone selon la direction $(x + y)$, i.e.

$$\frac{\partial f}{\partial (x+y)}(x, y) \geqslant 0 \quad \forall (x, y) \in D. \tag{4.8}$$

et telle que $f$ interpole un ensemble de données croissant monotone selon la diagonale, i.e.

$$f(x_i, y_j) = z_{ij}, \quad i = 1, \dots n_i,\ j = 1, \dots n_j \tag{4.9}$$

Dans les sections suivantes nous allons d'abord introduire un nouvel interpolant de Sibson (section 4.4). Ensuite nous développons des conditions suffisantes de monotonie pour ce nouvel interpolant (section 4.5). Et finalement, nous proposons deux algorithmes pour calculer des surfaces monotones (section 4.6).

## 4.4 Interpolation de Sibson modifiée

Dans cette section, nous allons présenter un nouvel interpolant de Sibson. Il s'agit en fait d'une modification de l'interpolant original pour qu'il soit apte à être utilisé pour de l'interpolation monotone suivant la définition de la monotonie diagonale introduite dans la section précédente.

Rappelons que dans la méthode d'interpolation originale de Sibson (voir section 4.2.2) les quatre coefficients $c_{17}$, $c_{18}$,$c_{19}$,$c_{20}$ en (4.6), qui assurent la continuité entre patchs voisins, sont déterminé par le choix de dérivées transverses au bord *orthonales et linéaires* aux bords.

Notre méthode se base sur un autre choix de dérivées transverses aux bords *en direction* $(x + y)$ *et linéaires* aux bords. Ce choix nous amène alors aux valeurs données dans le lemme qui suit pour les quatre coefficients libres.

**Lemme.** *Soient* $c_i$, $i = 1, \dots, 25$ *les coefficients d'un morceau de surface composé de 4 patchs de Bézier triangulaires de degré 3 comme illustrée en figure 4.5. Soient les coefficients* $c_1, \dots, c_{16}$ *et* $c_{21}, \dots, c_{25}$ *donnés par les formules* (4.5)*. Si les dérivées transverses aux bords*

*du rectangle dans la direction $(x+y)$ sont linéaires, alors on a*

$$c_{17} = \frac{(-(h^x+h^y)c_1 + (h^x+3h^y)c_5 + (h^x-3h^y)c_6 + 2h^x c_{13} + 2h^x c_{14} + (h^y-h^x)c_2)}{(4h^x)}$$
$$c_{18} = \frac{(-(h^x+h^y)c_3 + (3h^x+h^y)c_8 + (h^y-3h^x)c_7 + 2h^y c_{14} + 2h^y c_{15} + (h^x-h^y)c_2)}{(4h^y)}$$
$$c_{19} = \frac{(-(h^x+h^y)c_3 + (h^x+3h^y)c_9 + (h^x-3h^y)c_{10} + 2h^x c_{15} + 2h^x c_{16} + (h^y-h^x)c_4)}{(4h^x)}$$
$$c_{20} = \frac{(-(h^x+h^y)c_1 + (3h^x+h^y)c_{12} + (h^y-3h^x)c_{11} + 2h^y c_{13} + 2h^y c_{16} + (h^x-h^y)c_4)}{(4h^y)}$$

*Démonstration.* Nous allons montrer que la valeur que nous avons donné pour le point $c_{17}$ implique bien une dérivée transverse linéaire dans la direction $(x+y)$. Ce coefficient permet d'avoir la continuité $C^1$ entre le patch de Sibson considéré et celui qui se situe en dessous. Le bord à considérer est donc celui du bas dans la figure 4.5, il faut donc calculer la dérivée transverse le long de ce bord. Celle-ci est définie dans le triangle de Bézier du bas. Nous allons donc calculer la dérivée dans la direction $(x+y)$ pour ce bord dans le triangle du bas. Soit $P$ un point défini par ses coordonnées barycentriques $(r,s,t)$ dans le triangle défini par les points $(x_i, y_j, c_1)$, $(x_{i+1}, y_j, c_2)$ et $(\frac{x_i+x_{i+1}}{2}, \frac{y_j+y_{j+1}}{2}, c_{25})$. Nous voulons donc évaluer $\frac{\partial f(P)}{\partial(x+y)}\Big|_{y=0}$.

$P$ est définie par ses coordonnées barycentriques alors que nous voulons une expression en fonction de $x$ et $y$. Il faut donc réécrire cette expression en une autre qui prend en compte les coordonnées barycentriques.

$$\frac{\partial f(P)}{\partial(x+y)}\Big|_{y=0} = \left(\frac{\partial f(P)}{\partial x} + \frac{\partial f(P)}{\partial y}\right)\Big|_{y=0}, \tag{4.10}$$
$$= \left(\frac{\partial f(P)}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial f(P)}{\partial s}\frac{\partial s}{\partial x} + \frac{\partial f(P)}{\partial t}\frac{\partial t}{\partial x} + \frac{\partial f(P)}{\partial r}\frac{\partial r}{\partial y} + \frac{\partial f(P)}{\partial s}\frac{\partial s}{\partial y} + \frac{\partial f(P)}{\partial t}\frac{\partial t}{\partial y}\right)\Big|_{y=0}. \tag{4.11}$$

Supposons que les coordonnées barycentriques de $P$ soient données par rapport aux points $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ et $P_3 = (x_3, y_3)$ et que l'aire du triangle soit $A$, alors nous avons les égalités suivantes :

$$\frac{\partial r}{\partial x} = \frac{y_2 - y_3}{2A}, \qquad \frac{\partial r}{\partial y} = \frac{x_3 - x_2}{2A},$$
$$\frac{\partial s}{\partial x} = \frac{y_3 - y_1}{2A}, \qquad \frac{\partial s}{\partial y} = \frac{x_1 - x_3}{2A},$$
$$\frac{\partial t}{\partial x} = \frac{y_1 - y_2}{2A}, \qquad \frac{\partial t}{\partial y} = \frac{x_2 - x_1}{2A}.$$

Se référer à l'annexe B pour plus de détails.

Dans le cas du triangle du bas, nous pouvons évaluer ces valeurs. En reprenant l'équation 4.11, nous obtenons :

$$\begin{aligned}\left.\frac{\partial f(P)}{\partial(x+y)}\right|_{y=0} &= \frac{1}{2A}\left((y_2-y_3+x_3-x_2)\frac{\partial f(P)}{\partial r}\right.\\ &\left.+\ (y_3-y_1+x_1-x_3)\frac{\partial f(P)}{\partial s}+(y_1-y_2+x_2-x_1)\frac{\partial f(P)}{\partial t}\right)\Bigg|_{y=0}\\ &= \frac{2}{h^x h^y}\left(-\frac{h^x+h^y}{2}\frac{\partial f(P)}{\partial r}+\frac{h^y-h^x}{2}\frac{\partial f(P)}{\partial s}+h^x\frac{\partial f(P)}{\partial t}\right)\Bigg|_{y=0}. \quad (4.12)\end{aligned}$$

Or la valeur de la fonction $f$ définie sur le triangle est connue

$$f(P) = r^3c_1+3r^2sc_5+3r^2tc_{13}+3rs^2c_6+6rstc_{17}+3rt^2c_{21}+s^3c_2+3s^2tc_{14}+3st^2c_{22}+t^3c_{25}.$$

Toutes les dérivées de l'équation (4.12) peuvent donc être calculées. De plus, pour $y = 0$, nous avons que $t = 0$ et $r = 1 - s$. Après tout ces remplacement, nous trouvons :

$$\begin{aligned}\left.\frac{\partial f(P)}{\partial(x+y)}\right|_{y=0} = \frac{1}{h^x h^y}\quad &\big(-3(h^x+h^y)(1-s)^2c_1+3((h^y-h^x)(1-s)^2-2(h^x+h^y)s(1-s))c_5\\ &+3(2(h^y-h^x)s(1-s)-(h^x+h^y)s^2)c_6+3(h^y-h^x)s^2c_2\\ &+6h^x(1-s)^2c_{13}+12h^xs(1-s)c_{17}+6h^xs^2c_{14}\big)\end{aligned} \quad (4.13)$$

Nous voulons que ce polynôme soit linéaire en $s$. C'est équivalent à ce que le coefficient $C_2$ devant $s^2$ qui correspond au terme de degré 2 du polynôme soit nul.

$$\begin{aligned}C_2 = \frac{3}{h^x h^y}&(-(h^x+h^y)c_1+(h^x+3h^y)c_5+(h-3h^y)c_6\\ &+(h^y-h^x)c_2+2h^xc_{13}-4h^xc_{17}+2h^xc_{14})\\ &=0.\end{aligned} \quad (4.14)$$

Dans l'équation précédente, l'inconnue est le coefficient $c_{17}$, la résolution de cette équation nous donne alors :

$$c_{17} = (-(h^x+h^y)c_1+(h^x+3h^y)c_5+(h^x-3h^y)c_6+2h^xc_{13}+2h^xc_{14}+(h^y-h^x)c_2)/(4h^x),$$

ce qui est bien la formule voulue.

Pour trouver les trois autres expressions, on applique le même raisonnement aux différents triangles de Bézier.

□

Ce choix de valeur pour $c_{17}$, $c_{18}$, $c_{19}$ et $c_{20}$ permet, de plus, d'avoir une continuité $C^1$ entre les quatre patchs triangulaires de Bézier à l'intérieur du patch de Sibson, mais aussi d'avoir la continuité $C^1$ entre deux patchs de Sibson adjacents. La continuité entre deux patchs adjacents est au moins $C^0$ par définition puisque les points de contrôles sur une arête commune ont les mêmes valeurs. Le lemme suivant montre que la continuité entre deux patchs est au minimum $C^1$.

**Lemme.** *Soient deux patchs de Sibson modifiés adjacents. Si leurs coefficients $c_1, \ldots, c_{25}$, voir figure 4.5, sont calculés tels que définis à l'équation (4.5) et au lemme 4.4, alors la continuité entre les patchs est $C^1$.*

*Démonstration.* Supposons que nous ayons deux patchs l'un au-dessus de l'autre. Le premier contenu dans $[x_i, x_{i+1}] \times [y_i, y_{i+1}]$ ayant les points de contrôles $(c_i)_{i=1,\dots,25}$. Le deuxième étant défini sur $[x_i, x_{i+1}] \times [y_{i-1}, y_i]$ ayant les points de contrôles $(c'_i)_{i=1,\dots,25}$. Vérifions alors que les conditions de continuité $C^1$ concernant la coplanarité des points de contrôles définies à la section 4.2.1 sont respectées.

$$\begin{aligned}
c_1 - c_{13} &= -\frac{h^x}{6} z^x_{ij} - \frac{h^y}{6} z^y_{ij} \\
&= c'_{16} - c_5 \\
c_5 - c_{17} &= \frac{h^y}{4h^x} c_1 - \frac{h^y}{4h^x} c_2 + \frac{2h^x - 3h^y}{4h^x} c_5 + \frac{3h^y - 2h^x}{4h^x} c_6 - \frac{h^y}{12}(z^y_{ij} + z^y_{i+1j}) \\
&= c'_{19} - c_6 \\
c_6 - c_{14} &= -\frac{h^x}{6} z^x_{i+1j} - \frac{h^y}{6} z^y_{i+1j} \\
&= c'_{16} - c_5
\end{aligned}$$

Ces égalités montrent bien la coplanarité des points se trouvant dans la même égalité. De même, on prouve que pour deux patchs côte à côte, les conditions de continuité $C^1$ sont toujours respectées. Il y a donc bien continuité $C^1$ entre les patchs.

Regardons aussi la continuité entre deux patchs triangulaires de Bézier à l'intérieur du Sibson-Split. Comme auparavant, les conditions de la continuité $C^0$ sont respectées puisque les points sur les arêtes de bords entre deux patchs à l'intérieur du Sibson-Split sont définis de façon unique. Examinons les conditions de continuité $C^1$ le long de l'arête commune entre le patch du bas et le patch de gauche :

- $c_1$, $c_5$, $c_{12}$ et $c_{13}$ sont coplanaires : en effet, $c_1$, $c_5$ et $c_{12}$ sont coplanaires. De plus, nous avons que $c_{13} = \frac{1}{2}(c_{12} + c_5)$, et donc que le point $c_{13}$ se trouve sur la droite $(c_{12}c_5)$. Ces quatre points sont donc coplanaires.
- $c_{13}$, $c_{17}$, $c_{20}$ et $c_{21}$ sont coplanaire. Même argument que précédemment en utilisant que $c_{21} = \frac{1}{2}(c_{17} + c_{20})$.
- $c_{21}$, $c_{22}$, $c_{24}$ et $c_{25}$ sont coplanaire. Même argument que précédemment en utilisant que $c_{21} = \frac{1}{2}(c_{22} + c_{24})$.

La continuité est donc bien $C^1$ le long de l'arête commune entre ces deux patchs. On peut montrer de même qu'à l'intérieur d'un interpolant de Sibson-Split, on atteint bien la continuité $C^1$. □

Le lemme 4.4 montre qu'une surface construite à partir d'un ensemble de patchs de Sibson modifié interpolant des valeurs sur une grille régulière est de continuité $C^1$. Nous avons ainsi démontré le théorème suivant :

**Théorème 4.1 (Interpolant de Sibson modifié).** *Soit $n_i, n_j$ deux entiers et $G = \{(x_i, y_j),\ i \in \{1, \dots, n_i\} \wedge j \in \{1, \dots, n_j\}\}$ une grille régulier dans le plan $(x, y)$. Soient données en chaque point de cette grille les valeurs de fonction $z_{ij}$ et les dérivées partielles d'ordre 1 $z^x_{ij}$ et $z^y_{ij}$ selon les axes $x$ et $y$. Soit $h^x_i = x_{i+1} - x_i$ et $h^y_j = y_{j+1} - y_j$.*
*L'interpolant de Sibson modifié construit une surface polynomiale $C^1$ par morceaux qui interpole les valeurs de fonction et dérivées données aux sommets de la grille $G$. Les 25 coefficients de Bézier, numérotés comme à la figure 4.5, définissant un patch de Sibson modifié composée de quatre surfaces de Bézier triangulaires sur chacun des rectangles $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$,*

$1 \leq i \leq n_i - 1$ *et* $1 \leq j \leq n_j - 1$ *sont donnés par :*

$$
\begin{aligned}
&c_1 = z_{ij}, && c_2 = z_{i+1,j}, && c_3 = z_{i+1,j+1},\\
&c_4 = z_{i,j+1}, && c_5 = c_1 + \frac{h^x}{3} z^x_{ij}, && c_6 = c_2 - \frac{h^x}{3} z^x_{i+1,j},\\
&c_9 = c_3 - \frac{h^x}{3} z^x_{i+1,j+1}, && c_{10} = c_4 + \frac{h^x}{3} z^x_{i,j+1}, && c_{11} = c_4 - \frac{h^y}{3} z^x_{i,j+1},\\
&c_7 = c_2 + \frac{h^y}{3} z^y_{i+1,j}, && c_8 = c_3 - \frac{h^y}{3} z^y_{i+1,j+1}, && c_{12} = c_1 + \frac{h^y}{3} z^y_{ij},\\
&c_{13} = \frac{1}{2}(c_5 + c_{12}), && c_{14} = \frac{1}{2}(c_6 + c_7), && c_{15} = \frac{1}{2}(c_8 + c_9),\\
&c_{16} = \frac{1}{2}(c_{11} + c_{10}), && c_{21} = \frac{1}{2}(c_{20} + c_{17}), && c_{22} = \frac{1}{2}(c_{17} + c_{18}),\\
&c_{23} = \frac{1}{2}(c_{18} + c_{19}), && c_{24} = \frac{1}{2}(c_{19} + c_{20}), && c_{25} = \frac{1}{2}(c_{21} + c_{23}) = \frac{1}{2}(c_{22} + c_{24}),
\end{aligned}
$$

*et*

$$
\begin{aligned}
c_{17} &= \frac{(-(h^x + h^y)c_1 + (h^x + 3h^y)c_5 + (h^x - 3h^y)c_6 + 2h^x c_{13} + 2h^x c_{14} + (h^y - h^x)c_2)}{(4h^x)},\\
c_{18} &= \frac{(-(h^x + h^y)c_3 + (3h^x + h^y)c_8 + (h^y - 3h^x)c_7 + 2h^y c_{14} + 2h^y c_{15} + (h^x - h^y)c_2)}{(4h^y)},\\
c_{19} &= \frac{(-(h^x + h^y)c_3 + (h^x + 3h^y)c_9 + (h^x - 3h^y)c_{10} + 2h^x c_{15} + 2h^x c_{16} + (h^y - h^x)c_4)}{(4h^x)},\\
c_{20} &= \frac{(-(h^x + h^y)c_1 + (3h^x + h^y)c_{12} + (h^y - 3h^x)c_{11} + 2h^y c_{13} + 2h^y c_{16} + (h^x - h^y)c_4)}{(4h^y)}.
\end{aligned}
$$

*Remarque :* Dans la plupart des cas, nous aurons une grille régulière où le pas de longueur $h^x$ et le pas de hauteur $h^y$ seront égaux, c'est-à-dire que $h^x = h^y = h$. Cela donne alors les formules suivantes :

$$
\begin{cases}
c_{17} = (-c_1 + 2c_5 - c_6 + c_{13} + c_{14})/2\\
c_{18} = (-c_3 + 2c_8 - c_7 + c_{14} + c_{15})/2\\
c_{19} = (-c_3 + 2c_9 - c_{10} + c_{15} + c_{16})/2\\
c_{20} = (-c_1 + 2c_{12} - c_{11} + c_{13} + c_{16})/2.
\end{cases}
\tag{4.15}
$$

## 4.5 CONDITIONS SUFFISANTES DE MONOTONIE

Nous allons maintenant développer des conditions suffisantes sur les dérivées partielles en chaque point de la grille dans les directions $x$ et $y$ pour que notre surface de Sibson modifiée qui interpole ces valeurs soit croissante monotone selon la direction $(x + y)$. Il est supposé ici que les pas verticaux $h^y_j$ et horizontaux $h^x_i$ sont tous égaux, c'est-à-dire que pour tout $i \in \{1, \ldots, n_i - 1\}$ et pour tout $j \in \{1, \ldots, n_j - 1\}$, $h^x_i = h^y_j = h$.

**Théorème 4.2 (Conditions suffisantes de monotonie).** *L'interpolant de Sibson modifié qui interpole les valeurs $z_{i+k,j+l}$ et les dérivées partielles $z^x_{i+k,j+l}$ et $z^y_{i+k,j+l}$ $(k, l \in \{0, 1\})$ sur $D_{ij}$ est monotone selon la définition 4.8 si les conditions suivantes sont respectées :*

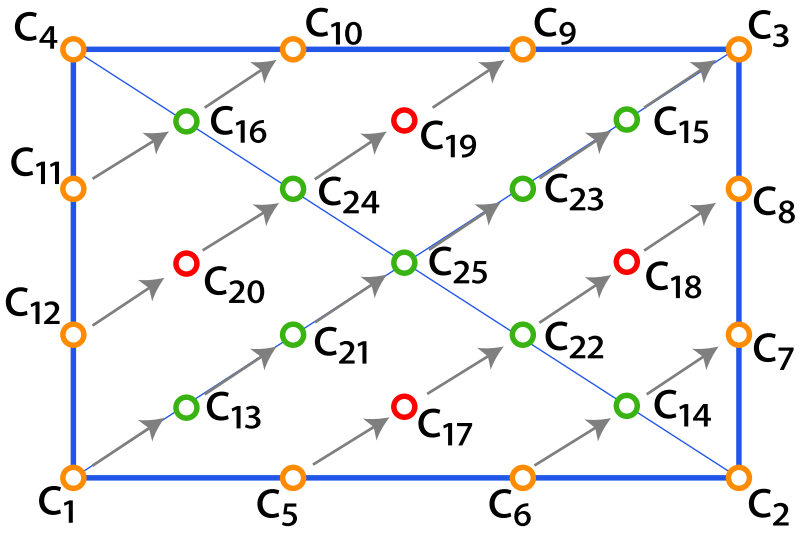



**FIGURE 4.8 –** *Les quatre patchs triangulaires de Bézier respectent la condition de monotonie selon la direction $(x + y)$ si pour toutes flèches dans cette image, la valeur du coefficient à la base de cette flèche est plus petite que la valeur à la pointe.*

$$z_{i+1j+1} \geqslant z_{ij} \ , \tag{4.16}$$
$$z^x_{ij} + z^y_{ij} \geqslant 0 \ , \tag{4.17}$$
$$z^x_{i+1j} + z^y_{i+1j} \geqslant 0 \ , \tag{4.18}$$
$$z^x_{ij+1} + z^y_{ij+1} \geqslant 0 \ , \tag{4.19}$$
$$z^x_{i+1j+1} + z^y_{i+1j+1} \geqslant 0, \tag{4.20}$$
$$\frac{12}{h}(z_{i+1j+1} - z_{ij}) \geqslant 5z^x_{ij} + z^y_{ij} + 2z^x_{i+1j} + 2z^y_{i+1j} + z^x_{i+1j+1} + 5z^y_{i+1j+1} \ , \tag{4.21}$$
$$\frac{12}{h}(z_{i+1j+1} - z_{ij}) \geqslant z^x_{ij} + 5z^y_{ij} + 2z^x_{ij+1} + 2z^y_{ij+1} + 5z^x_{i+1j+1} + z^y_{i+1j+1} \ . \tag{4.22}$$

*Démonstration.* Les conditions ci-dessus peuvent être classées en trois groupes :

1. (4.16) : Les valeurs $\{(z_{ij})\}$ sont les valeurs que nous souhaitons interpoler, il faut donc qu'elles vérifient la condition de croissance monotone selon les diagonales. D'où cette condition.
2. (4.17) à (4.20) : Ces conditions permettent de satisfaire (4.8) aux quatre coins de l'interpolant.
3. (4.21) et (4.22) : Elles permettent de propager les contraintes de monotonie à l'intérieur du patch de Sibson modifié.

D après la section 4.2.1, les dérivées partielles d'une surface triangulaire de Bézier peuvent être évaluées à partir de différences entre points de contrôle de la surface. En effet, les formules (4.2) sont des combinaisons linéaires de différences entre deux coefficients de Bézier. Les coefficients de ces combinaisons linéaires sont des polynômes de Bernstein $B^2_{ijk}$ qui sont positifs car lorsqu'un point est à l'intérieur d'un triangle, alors ses coordonnées barycentriques

par rapport aux sommets de ce triangle sont positives. Donc si toutes les différences entre coefficients de Bézier sont positives, alors la dérivée partielle considérée est elle aussi positive. Il faut alors que pour tout couple $(c_j \to c_k)$ tel qu'indiqué à la figure 4.8, la relation $c_j \leqslant c_k$ soit vérifiée. Nous allons maintenant passer en revue ces couples de coefficients, les remplacer par leur valeur définie à la définition 4.1 et montrer que les conditions que nous avons données dans le théorème impliquent que la relation d'inégalité est bien respectée.

Pour le couple $(c_1 \to c_{13})$, nous avons $c_{13} - c_1 = z^x_{ij} + z^y_{ij}$. La condition (4.17) donne alors $c_{13} - c_1 \geqslant 0$. De même, la condition (4.18) implique que la relation donnée par les couples $(c_6 \to c_{14})$ et $(c_{14} \to c_7)$ est vérifiée, ainsi que la condition (4.19) vérifie la condition du couple $(c_{15} \to c_3)$, la condition (4.20) donne les relations des couples $(c_{11} \to c_{16})$ et $(c_{16} \to c_{10})$.

À partir de la première équation de (4.15), on déduit que

$$\begin{aligned} 2(c_{17} - c_5) &= (c_{13} - c_1) + (c_{14} - c_6) \\ &= (z^x_{ij} + z^y_{ij}) + (z^x_{i+1j} + z^y_{i+1j}). \end{aligned}$$

Si les conditions (4.17) et (4.18) sont respectées, alors nous avons que $c_{17} - c_5 \geqslant 0$. De même, on montre que si les conditions (4.17) à (4.20) sont satisfaites, alors on a $c_8 - c_{18} \geqslant 0$, $c_9 - c_{19} \geqslant 0$ et $c_{20} - c_{12} \geqslant 0$.

De plus, si les conditions (4.17) à (4.20) sont satisfaites, alors :

$$2(c_{21} - c_{13}) = (c_{20} - c_{12}) + (c_{17} - c_5) \geqslant 0.$$

On trouve le même résultat pour le couple $(c_{23} \to c_{15})$.

Grâce aux coefficients donnés aux équations (4.5) et (4.15), on trouve comme valeur pour $c_{22} - c_{17}$ :

$$c_{22} - c_{17} = \frac{1}{2}\left[(z_{i+1j+1} - z_{ij}) - \frac{h}{12}(5z^x_{ij} + z^y_{ij} + 2z^x_{i+1j} + 2z^y_{i+1j} + z^x_{i+1j+1} + 5z^y_{i+1j+1})\right]$$

La condition (4.21) implique donc que cette différence est positive. De plus, comme $c_{22} = \frac{1}{2}(c_{17} + c_{18})$, la condition (4.21) donne que $c_{18} - c_{22} \geqslant 0$.

De même, on trouve que la condition (4.22) implique que $c_{24} - c_{20} = c_{19} - c_{24} \geqslant 0$.

Finalement, on a l'équation suivante :

$$2(c_{25} - c_{21}) = 2(c_{23} - c_{25}) = (c_{22} - c_{17}) + (c_{24} - c_{20}).$$

Si les conditions (4.21) et (4.22) sont satisfaites, alors $c_{25} - c_{21} = c_{23} - c_{25} \geqslant 0$.

Nous avons donc montré que si les conditions du théorème précédent sont satisfaites, alors la surface résultante est monotone dans la direction $(x + y)$. □

Ce théorème donne les conditions suffisantes pour construire un patch de Sibson modifié monotone. Si les données en entrée sont composées de valeurs de fonctions sur une grille et de valeurs de dérivées partielles dans les directions $x$ et $y$, il suffit alors de vérifier si les conditions sont respectées pour chaque carré de la grille pour savoir si l'union des patchs de Sibson définit une surface monotone dans la direction $(x + y)$.

*Remarque :* Dans le théorème précédent, la condition (4.16) est redondante avec au moins l'une des conditions (4.21) ou (4.22). En effet, les parties droites des deux inégalités ne peuvent pas être négatives en même temps : la somme des deux membres de droite donne

$$6(z^x_{ij} + z^y_{ij}) + 2(z^x_{i+1j} + z^y_{i+1j}) + 2(z^x_{ij+1} + z^y_{ij+1}) + 6(z^x_{i+1j+1} + z^y_{i+1j+1}) \geqslant 0,$$

la positivité venant des conditions (4.17) à (4.20). Donc, si l'un des deux membres est négatif, alors le deuxième est obligatoirement positif et possède une valeur plus grande que la valeur absolu du premier car la somme des deux est positive. La condition (4.16) sera donc satisfaite si les conditions (4.21) et (4.22) le sont aussi. Autrement dit, si les données en entrée sont monotones et si les derivées partielles satisfont les conditions (4.17) à (4.22), alors la surface sera montone.

Nous allons dans la suite présenter deux algorithmes permettant de trouver des dérivées partielles qui satisfont les conditions du théorème 4.2.

## 4.6 Calcul de dérivées partielles admissibles

Dans la section précédente, nous avons développé des conditions suffisantes sur les données en entrées, i.e. les valeurs à interpoler et les valeurs de dérivées partielles aux sommets de la grille pour pouvoir construire une fonction monotone selon la direction $(x+y)$ en utilisant notre méthode d'interpolation de Sibson modifié. Sachant que dans la plupart des cas, un utilisateur aura seulement une valeur de fonction aux sommets et ne connaîtra pas forcément les dérivées partielles dans les directions $x$ et $y$ en ces points, il n'est alors pas facile de trouver ou de modifier ces valeurs de façon à ce qu'elles respectent les conditions du théorème 4.2.

Dans cette partie, nous allons décrire deux algorithmes permettant de trouver des valeurs de dérivées partielles qui respectent les conditions du théorème 4.2. Nous les appelons *dérivées admissibles*. Le premier algorithme permet de modifier les dérivées partielles données en entrées si celles-ci ne sont pas admissibles. Le deuxième algorithme prend en entrée seulement les valeurs de fonction à interpoler et va construire des dérivées partielles admissibles.

### 4.6.1 Algorithme 1

Pour ce premier algorithme, nous supposons que les données en entrées à interpoler sont les valeurs de fonction $z_{ij}$ et les valeurs de dérivées partielles $z^x_{ij}, z^y_{ij}$ en chaque point de la grille $G$. On suppose en plus que les valeurs de fonction vérifient la condition (4.16) du théorème 4.2. Par contre, les valeurs de dérivées partielles ne vérifient pas forcément les conditions du théorème 4.2. L'algorithme 1 a pour but de modifier ces valeurs de telle sorte qu'en sortie, les dérivées respectent les conditions du théorème 4.2. L'algorithme est écrit pour un patch de Sibson. Si l'on veut corriger les dérivées partielles pour une grille entière, il faudra alors appliquer l'algorithme 1 sur tous les patchs composant la grille. Il se compose de deux parties différentes. La première permet de vérifier les conditions (4.17) à (4.20). La deuxième s'intéresse aux conditions (4.21) et (4.22).

#### *(a) Fonctionnement sur le carré $D_{ij}$.*

Considérons un patch de Sibson modifié construit sur le domaine $D_{ij}$. Les valeurs de fonction à interpoler sont supposées vérifier $z_{ij} \leqslant z_{i+1j+1}$. L'algorithme ne modifiera pas ces valeurs. La première partie de l'algorithme va vérifier que la somme des deux dérivées partielles en chaque sommets du carré soit bien positive et donc, que les conditions (4.17) à (4.20) sont bien satisfaites. Si ce n'est pas le cas, il va alors modifier ces valeurs. Supposons que les dérivées $z^x_{ij}$ et $z^y_{ij}$ ne vérifient pas la condition (4.17). Cela signifie que $z^x_{ij} + z^y_{ij} < 0$. Interprétons le point $(z^x_{ij}, z^y_{ij})$ comme un point de $\mathbb{R}^2$. L'espace vérifiant la condition (4.17) correspond alors à $C_1 = \{(x, y) \in \mathbb{R}^2 | x + y \geqslant 0\}$. En notant $(\tilde{z}^x_{ij}, \tilde{z}^y_{ij}) \in \mathbb{R}^2$ le couple de dérivées partielles

**Algorithm 1** Modification des dérivées partielles en entrée pour un patch de Sibson. À la fin de l'algorithme, celles-ci vérifient les conditions données par le théorème 4.2

**Require:** $z_{kl}$ avec $z_{ij} \leqslant z_{i+1j+1}$, et les dérivées partielles $z^x_{kl}, z^y_{kl}$ $(k = i, i+1, l = j, j+1)$
{Première partie de l'algorithme}
**for** $k \in \{i, i+1\}$ **do**
**for** $l \in \{j, j+1\}$ **do**
{Vérification des quatre conditions (4.17)-((4.20)}
**if** $z^x_{kl} + z^y_{kl} < 0$ **then**
$old_x \leftarrow z^x_{kl}$
$old_y \leftarrow z^y_{kl}$
{Projection orthogonale du point $(z^x_{kl}, z^y_{kl})$ sur la droite $z^x_{kl} + z^y_{kl} = 0$}
$z^x_{kl} \leftarrow \frac{old_x - old_y}{2}$
$z^y_{kl} \leftarrow \frac{old_y - old_x}{2}$
**end if**
**end for**
**end for**
{Deuxième partie de l'algorithme}
$\Delta z \leftarrow \frac{12}{h}(z_{i+1j+1} - z_{ij})$
$S_2 \leftarrow (5z^x_{ij} + z^y_{ij})^+ + 2z^x_{i+1j} + 2z^y_{i+1j} + (z^x_{i+1j+1} + 5z^y_{i+1j+1})^+$
$S_3 \leftarrow (z^x_{ij} + 5z^y_{ij})^+ + 2z^x_{ij+1} + 2z^y_{ij+1} + (5z^x_{i+1j+1} + z^y_{i+1j+1})^+$
$S_{max} \leftarrow Maximum(S_2, S_3)$
{Vérification des deux dernières conditions (4.21) et (4.22)}
**if** $\Delta z < S_{max}$ **then**
$c \leftarrow \frac{\Delta z}{S_{max}}$
$z^x_{kl} \leftarrow c\, z^x_{kl}$ for $k = i, i+1, l = j, j+1$
$z^y_{kl} \leftarrow c\, z^y_{kl}$ for $k = i, i+1, l = j, j+1$
**end if**

après modification, nous voulons modifier les valeurs de départ le moins possible, c'est-à-dire

$$(\tilde{z}^x_{ij}, \tilde{z}^y_{ij}) \in C_1 \;\text{ et }\; \| \begin{pmatrix} \tilde{z}^x_{ij} \\ \tilde{z}^y_{ij} \end{pmatrix} - \begin{pmatrix} z^x_{ij} \\ z^y_{ij} \end{pmatrix} \| \to \min.$$

Cela est vérifié si le couple de solution correspond à la projection orthogonale de $(z^x_{ij}, z^y_{ij})$ sur l'espace $C_1$ et donc sur la droite $d : x + y = 0$. L'évaluation de ce projeté est donné par la formule

$$\begin{pmatrix} \tilde{z}^x_{ij} \\ \tilde{z}^y_{ij} \end{pmatrix} = \begin{pmatrix} \frac{z^x_{ij} - z^y_{ij}}{2} \\ \frac{z^y_{ij} - z^x_{ij}}{2} \end{pmatrix}. \tag{4.23}$$

Une illustration de cette transformation est montrée à la figure 4.9. La première partie de l'algorithme 1 utilise donc l'équation (4.23) pour modifier les valeurs des dérivées partielles lorsque leur somme est négative.

La deuxième partie permet de s'assurer que les conditions (4.21) et (4.22) sont bien satisfaites. La fonction

$$(.)^+ : \left| \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R}^+ \\ x & \longmapsto & \begin{cases} x & \text{si } x \geqslant 0 \\ 0 & \text{sinon} \end{cases} \end{array} \right.$$

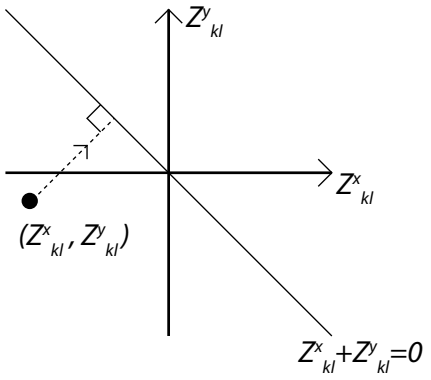



**FIGURE 4.9 –** *Projection orthogonale des dérivées partielles* $(z^x_{kl}, z^y_{kl})$ *sur la droite* $d : z^x_{kl} + z^y_{kl} = 0$ *quand la somme de celles-ci est négative.*

nous permet de prendre en compte les combinaisons linéaires de dérivées partielles en un point introduites aux deux conditions si celles-ci augmentent la valeur de la partie droite de ces conditions. La variable $\Delta z$ correspond à la partie gauche des inégalités et les variables $S_2$ et $S_3$ correspondent respectivement à la partie droite des conditions (4.21) et (4.22). Nous vérifions alors que le maximum $S_{max} = Max(\{S_2, S_3\})$ est bien inférieur à $\Delta z$. Si c'est le cas, alors il n'y a rien à faire puisque les conditions sont satisfaites. En effet, pour tout $x \in \mathbb{R}$, nous avons $(x)^+ \geqslant x$. Nous obtenons alors :

$$\begin{aligned} S_2 &\geqslant 5z^x_{ij} + z^y_{ij} + 2z^x_{i+1j} + 2z^y_{i+1j} + z^x_{i+1j+1} + 5z^y_{i+1j+1} \\ S_3 &\geqslant z^x_{ij} + 5z^y_{ij} + 2z^x_{ij+1} + 2z^y_{ij+1} + 5z^x_{i+1j+1} + z^y_{i+1j+1} \end{aligned}$$

Et donc, si $\Delta z \geqslant S_{max}$ alors les deux conditions sont satisfaites.

Sinon, les dérivées partielles données aux sommets du carré $D_{ij}$ sont multipliées par $c = \frac{\Delta z}{S_{max}}$. En utilisant le fait que pour tout $x \in \mathbb{R}$ et pour tout $\lambda \geqslant 0$ : $(\lambda x)^+ = \lambda(x)^+$, la valeur $\tilde{S}_2$ qui est la valeur $S_2$ avec les modifications de dérivées partielles peut alors être calculée par

$$\begin{aligned} \tilde{S}_2 &= (5cz^x_{ij} + cz^y_{ij})^+ + 2cz^x_{i+1j} + 2cz^y_{i+1j} + (cz^x_{i+1j+1} + 5cz^y_{i+1j+1})^+, \\ &= cS_2. \end{aligned}$$

De même, $\tilde{S}_3 = cS_3$.

Supposons alors que $S_{max} = S_2$, alors

$$\begin{aligned} \tilde{S}_{max} &= \tilde{S}_2, \\ &= \frac{\Delta z}{S_{max}} S_2, \\ &= \Delta z. \end{aligned}$$

De plus, si nous sommes dans le cas où il faut modifier la valeur des dérivées partielles, cela signifie que $0 \leqslant \Delta z < S_{max}$ et donc que $0 \leqslant c < 1$. Or, nous avons que $0 \leqslant S_3 \leqslant S_2$ (puisque $S_{max} = S_2 \geqslant S_3$), d'où $\tilde{S}_3 = cS_3 \leqslant \tilde{S}_2 = \Delta_z$. De même, si $S_{max} = S_3$, alors $\tilde{S}_2 \leqslant \tilde{S}_3 = \Delta_z$.

Les deux conditions sont donc satisfaites après la modification des valeurs des dérivées partielles. De plus, la multiplication par un coefficient $0 \leqslant c < 1$ des valeurs de dérivées en un sommet permet de conserver les conditions (4.17) à (4.20) puisque la somme sera alors $c(z^x_{i+k,j+l} + z^y_{i+k,j+l}) \geqslant 0$.

***(b) Correction des dérivées sur toute la grille.***

Nous venons de montrer comment calculer des dérivées partielles admissibles pour un carré $D_{ij}$ de la grille. Afin de rendre les dérivées partielles admissibles en tout point de la grille, nous proposons d'appliquer l'algorithme 1 de façon séquentiel sur chaque carré de la grille. Les dérivées partielles étant partagées par plusieurs carrés de la grille, la modification de celles-ci sur l'un des carrés a pour conséquences des modifications des dérivées partielles pour les carrés adjacents. Nous allons montrer ici que malgré les modifications, les conditions énoncées au théorème 4.2 seront tout de même satisfaites.

Supposons que nous ayons construit les valeurs de dérivées partielles pour le carré $D_{ij}$, alors en appliquant l'algorithme à l'ensemble des 8 carrés adjacents $\{D_{i\pm1,j\pm1}, D_{i\pm1,j}, D_{i,j\pm1}\}$ il se peut que l'algorithme modifie à nouveau les valeurs des dérivées partielles. Il faut donc prouver que ces changements respectent les conditions dans le carré $D_{ij}$.

La première partie de l'algorithme appliquée aux carrés adjacents ne modifiera pas à nouveau les valeurs des dérivées partielles de $D_{ij}$ puisque les conditions de positivité ((4.17) à (4.20)) de la somme des deux dérivées partielles en un sommet est satisfaite après avoir utilisé l'algorithme sur $D_{ij}$.

Dans la deuxième partie de l'algorithme, lorsque l'algorithme passe dans le carré $D_{k,l}$ (avec $k \in \{i-1, i, i+1\}$ et $l \in \{j-1, j, j+1\}$ et $(k,l) \neq (i,j)$), nous pouvons définir un coefficient $c_{k,l}$ égale au coefficient $c$ de l'algorithme lorsque celui-ci est passé dans la condition *If*. Sinon, on pose $c_{k,l} = 1$, ce qui signifie que la valeur des dérivées n'a pas été modifié. Donc la valeur finale des dérivées partielle est égale à un produit de coefficients $c_{k,l}$ multiplié par la valeur initiale de la dérivée. Par exemple, la valeur de dérivée partielles dans la direction $x$ pour le sommet de la grille $(i,j)$ après le passage de l'algorithme dans le carré $D_{i,j}$ est

$$\hat{z}^x_{i,j} = c_{i,j} z^x_{i,j}$$

Après le passage de l'algorithme dans les carrés $D_{i-1,j}$, $D_{i-1,j-1}$ et $D_{i,j-1}$, la dérivée partielle sera multipliée par le produit des coefficients de chacun de ces carrés. À la fin de l'algorithme nous avons donc une dérivée partielles au point $(i,j)$ dans la direction $x$ :

$$\tilde{z}^x_{i,j} = (c_{i-1,j-1} c_{i,j-1} c_{i-1,j}) \hat{z}^x_{i,j}$$

Or, comme $0 \leqslant c_{k,l} \leqslant 1$, on obtient :

$$\tilde{z}^x_{i,j} \leqslant c_{i,j} z^x_{i,j}$$

De même, nous pouvons montrer que :

$$\begin{aligned} \tilde{z}^x_{i+u,j+v} &\leqslant \hat{z}^x_{i+u,j+v} \\ \tilde{z}^y_{i+u,j+v} &\leqslant \hat{z}^y_{i+u,j+v} \end{aligned}$$

avec $k, j = 0, 1$. Au final, nous avons donc que :

$$\begin{aligned} \tilde{S}_2 &= (5\tilde{z}^x_{ij} + \tilde{z}^y_{ij})^+ + 2\tilde{z}^x_{i+1j} + 2\tilde{z}^y_{i+1j} + (\tilde{z}^x_{i+1j+1} + 5\tilde{z}^y_{i+1j+1})^+ \\ &\leqslant (5\hat{z}^x_{ij} + \hat{z}^y_{ij})^+ + 2\hat{z}^x_{i+1j} + 2\hat{z}^y_{i+1j} + (\hat{z}^x_{i+1j+1} + 5\hat{z}^y_{i+1j+1})^+ \end{aligned}$$

De même :

$$\tilde{S}_3 \leqslant (\hat{z}^x_{ij} + 5\hat{z}^y_{ij})^+ + 2\hat{z}^x_{ij+1} + 2\hat{z}^y_{ij+1} + (5\hat{z}^x_{i+1j+1} + \hat{z}^y_{i+1j+1})^+$$

Or, comme les $\hat{z}^x_{i+u,j+v}$ et $\hat{z}^y_{i+u,j+v}$ (avec $u, v \in \{0, 1\}$) sont les valeurs de dérivées en sortie de l'algorithme appliqué au carré $D_{i,j}$, nous avons en posant $\Delta_z = \frac{12}{h}(z_{i+1j+1} - z_{ij})$ :

$$\begin{aligned}(5\hat{z}^x_{ij} + \hat{z}^y_{ij})^+ + 2\hat{z}^x_{i+1j} + 2\hat{z}^y_{i+1j} + (\hat{z}^x_{i+1j+1} + 5\hat{z}^y_{i+1j+1})^+ &\leqslant \Delta_z,\\ \tilde{S}_3 \leqslant (\hat{z}^x_{ij} + 5\hat{z}^y_{ij})^+ + 2\hat{z}^x_{ij+1} + 2\hat{z}^y_{ij+1} + (5\hat{z}^x_{i+1j+1} + \hat{z}^y_{i+1j+1})^+ &\leqslant \Delta_z.\end{aligned}$$

D'où

$$\begin{aligned}\tilde{S}_2 &\leqslant \Delta_z,\\ \tilde{S}_3 &\leqslant \Delta_z.\end{aligned}$$

Cela signifie que les conditions (4.21) et (4.22) sont toujours satisfaites après avoir appliqué l'algorithme sur tout les carrés de la grille.

L'avantage de cet algorithme est de pouvoir travailler dans les carrés de la grille un par un. Cela signifie que le calcul des dérivées est local, il ne dépend que des valeurs de fonction aux points de la grille dans la 8-connectivité du point où l'on cherche à évaluer la dérivée. Mais il n'est ainsi pas possible de travailler en parallèle sur les carrés de la grille, car à chaque passage dans un carré, les valeurs de dérivées qui sont les valeurs en entrée pour l'interpolation d'un carré adjacent, peuvent être modifiées. Un autre inconvénient est que la valeur des dérivées en un sommet de la grille peut être modifiées jusqu'à quatre fois en le multipliant par un coefficient compris entre 0 et 1 à chaque fois. Cela a pour conséquence de diminuer fortement la valeur des dérivées partielles (et donc de rapprocher ces valeurs de 0 sans l'atteindre) pour les points à l'intérieur de la grille lorsque l'algorithme doit souvent être appliqué. Or, comme nous souhaitons une surface monotone, il est préférable que les valeurs de dérivées partielles soit éloignées de la valeur 0.

### *(c) Exemples numériques*

Le tableau 4.1 correspond à des valeurs de fonction à interpoler définies sur une grille $4 \times 4$. L'algorithme 1 est appliqué sur chaque carré de la grille. Dans les exemples qui suivent, les dérivées partielles en entrée seront égales dans les deux directions $x$ et $y$.

**Exemple 1 :**
On prend en entrée les valeurs de fonction du tableau 4.1 et les dérivées partielles $z^x_{ij} = z^y_{ij} = 5$ pour tout $i, j$ qui ne sont pas admissibles. La grille est parcourue ligne par ligne pour l'application de l'algorithme. Le résultat est montré dans le tableau 4.2. On remarque que les valeurs absolues des valeurs de dérivées partielles sont plus petites pour les points adjacents à quatre carrés de la grille. En effet, comme nous l'avons vu en section 4.6(b), les valeurs de dérivées partielles aux sommets adjacents à quatre carrés sont susceptibles d'être modifiées jusqu'à 4 fois. Cette modification correspond à une multiplication par un coefficient inférieur ou égal à 1, et donc, à une diminution de la valeur absolue de cette valeur.
**Exemple 2 :**
On prend les mêmes valeurs en entrée qu'au précédent exemple, mais le parcours de la grille est fait dans un ordre différent. Celui-ci est fait selon une spirale et correspond à la séquence : (1,1), (1,2), (1,3), (1,4), (2,4), (3,4), (4,4), (4,3), (4,2), (4,1), (3,1), (2,1), (2,2), (2,3), (3,3),

(3,2). Le tableau 4.3 montre les résultats de l'algorithme 1. On observe que le changement de parcours a pour conséquence une différence de résultat par rapport au parcours ligne par ligne. Cela montre que le résultat final de l'application de l'algorithme dépend de l'ordre de parcours de la grille.

**Exemple 3 :**

On prend en entrée les valeurs de fonction du tableau 4.1 et les dérivées partielles non-admissibles $z^x_{ij} = z^y_{ij} = 10$ pour tout $i, j$ et on parcours la grille ligne par ligne. Les valeurs de dérivées partielles en sortie sont montrées dans le tableau 4.4. Elles peuvent alors être plus élevées puisque l'algorithme ne peut pas donner en sortie des valeurs plus grandes que les valeurs d'entrée. Par exemple, dans le coin en bas à droite, nous avons en sortie une valeur d'environ 7.949, alors que dans le tableau 4.2, nous avions la valeur de 4.500. Dans le cas du tableau 4.2, la valeur en sortie ne pouvait pas être supérieure à 5.000 (puisque c'était la valeur en entrée), alors que dans le cas présent, la valeur ne peut pas dépasser 10.000. On observe que la moyenne des valeurs absolues des dérivées admissibles est à peut près identique : 2.364 pour cet exemple et 2.422 pour l'exemple 1. Or, cet exemple contient aussi les plus hautes valeurs. Cela signifie qu'en prenant des valeurs hautes, cela va donner en sortie de l'algorithme des valeurs hautes, mais aussi des valeurs proche de 0.

$\{z_{ij}\}^4_{i,j=1} =$

| | | | |
|---|---|---|---|
| 3 | 7 | 6 | 15 |
| 0 | 2 | 10 | 9 |
| 1 | 6 | 4 | 6 |
| 0 | 1 | 4 | 2 |

**TABLE 4.1 –** *Exemple des valeurs de fonction à interpoler échantillonnée sur une grille* $4 \times 4$*. Ces valeurs sont croissantes le long des diagonales.*

$\{z^x_{ij}\}^4_{i,j=1} = \{z^y_{ij}\}^4_{i,j=1} =$

| | | | |
|---|---|---|---|
| 5.000 | 4.390 | 3.450 | 3.930 |
| 0.779 | 0.684 | 3.450 | 3.930 |
| 0.701 | 0.328 | 0.876 | 1.874 |
| 4.500 | 2.104 | 0.876 | 1.874 |

**TABLE 4.2 –** *Résultats de l'exemple 1 : les dérivées partielles dans les directions $x$ et $y$ (identiques) évaluées avec l'algorithme 1 en partant des valeurs de dérivée toutes égales à 5. L'algorithme 1 est appliqué sur la grille lignes par lignes.*

$\{z^x_{ij}\}^4_{i,j=1} = \{z^y_{ij}\}^4_{i,j=1} =$

| | | | |
|---|---|---|---|
| 5.000 | 3.310 | 2.483 | 3.750 |
| 0.897 | 0.594 | 2.483 | 3.750 |
| 0.808 | 0.378 | 0.876 | 1.874 |
| 4.500 | 2.105 | 0.876 | 1.874 |

**TABLE 4.3 –** *Résultats de l'exemple 2 : les dérivées partielles dans les directions $x$ et $y$ (identiques) évaluées avec l'algorithme 1 en partant des valeurs de dérivée toutes égales à 5. L'algorithme 1 est appliqué sur la grille avec un parcours en spirale.*

Ces exemples montrent donc que les résultats de cet algorithme dépendent du parcours de la grille et des données en entrée.

$$\{z^x_{ij}\}^4_{i,j=1} = \{z^y_{ij}\}^4_{i,j=1} =$$

| | | | |
|---|---|---|---|
| 4.500 | 1.276 | 0.581 | 2.051 |
| 0.425 | 0.090 | 0.281 | 1.332 |
| 0.751 | 0.282 | 1.454 | 3.874 |
| 7.949 | 4.010 | 3.008 | 5.963 |

**TABLE 4.4 –** *Résultats de l'exemple 3 : les dérivées partielles dans les directions $x$ et $y$ (identiques) évaluées avec l'algorithme 1 en partant des valeurs de dérivée toutes égales à 10. L'algorithme 1 est appliqué sur la grille lignes par lignes.*

### 4.6.2 Algorithme 2

Avec ce deuxième algorithme, nous cherchons à résoudre les inconvénients de l'algorithme 1, notamment la nécessité d'avoir des valeurs de dérivées en entrées de l'algorithme ainsi que les dépendances du résultat en fonction des valeurs des dérivées en entrée et du sens de parcours de l'algorithme sur la grille. Nous cherchons donc un algorithme qui, pour une grille ayant des valeurs de fonction en chaque point, nous donne un unique résultat quant aux dérivées partielles dans les directions $x$ et $y$. L'algorithme 2 permettra de calculer des valeurs de dérivées partielles admissibles sans partir d'une estimation préalable de celles-ci. Il permet de vérifier les conditions du théorème 4.2. Le principe de cet algorithme est de calculer, pour chaque carré de la grille, la valeur maximale que pourrait prendre les dérivées partielles pour que toutes les conditions du théorème 4.2 soient respectées, en supposant que dans un carré, toutes les dérivées partielles sont égales. Ensuite, pour un point de la grille, les dérivées partielles sont choisies comme le minimum des dérivées partielles calculées précédemment dans les carrés de la grille qui possèdent ce point comme sommet.

Les conditions (4.17) à (4.20) sont satisfaites car les dérivées partielles dans les directions $x$ et $y$ sont positives. Le reste des conditions est aussi vérifié car l'algorithme fait en sorte que les parties droites des conditions (4.21) et (4.22) soient inférieures ou égales à $\lambda\frac{12}{h}(z_{i+1,j+1} - z_{i,j})$ avec $\lambda \in [0, 1]$. Nous allons donc montrer qu'à la fin de l'algorithme, les conditions sont vérifiées pour tous les patchs de Sibson modifiés

*Démonstration.* (Algorithme 2)

Nous allons vérifier que les conditions énoncées au théorème 4.2 sont bien satisfaites à la fin de l'algorithme.

Tout d'abord, les valeurs d'entrées sont telles que $z_{i,j} \leqslant z_{i+1,j+1}$. Il en résulte que tout les $K_{i,j}$ et donc aussi les $KMin_{i,j}$ sont positifs. Les valeurs des dérivées partielles en sortie sont donc, elles aussi, positives. Les conditions (4.17) à (4.20) sont donc vérifiées. Il nous reste à regarder les conditions (4.21) et (4.22).

Posons $S$ la partie droite de l'inégalité de la condition (4.21). Nous avons alors :

$$\begin{aligned} S &= 5z^x_{i,j} + z^y_{i,j} + 2z^x_{i+1,j} + 2z^y_{i+1,j} + z^x_{i+1,j+1} + 5z^y_{i+1,j+1}, \\ &= 6\frac{3\lambda}{4h}KMin_{i,j} + 4\frac{3\lambda}{4h}KMin_{i+1,j} + 6\frac{3\lambda}{4h}KMin_{i+1,j+1}. \end{aligned}$$

Or, l'algorithme calcule les valeurs $KMin$ de telle sorte que :

$$KMin_{i,j},\ KMin_{i+1,j},\ KMin_{i+1,j+1} \leqslant z_{i+1,j+1} - z_{i,j}.$$

**Algorithm 2** Calcul des dérivées partielles pour tout les patchs de Sibson-Split. En sortie, les conditions du théorème 4.2 sont satisfaites.

**Require:** $z_{i,j}$ pour $1 \leq i \leq n_x$ et $1 \leq j \leq n_y$, telles que $z_{i,j} \leqslant z_{i+1,j+1}$
**Require:** Constante $\lambda$ telle que $0 \leqslant \lambda \leqslant 1$
  **for** $i = 1 \cdots n_x - 1$ **do**
    **for** $j = 1 \cdots n_y - 1$ **do**
      $K_{i,j} \leftarrow (z_{i+1,j+1} - z_{i,j})$
    **end for**
  **end for**
  $KMin_{1,1} \leftarrow K_{1,1}$
  $KMin_{n_x,n_y} \leftarrow K_{n_x-1,n_y-1}$
  $KMin_{n_x,1} \leftarrow K_{n_x-1,1}$
  $KMin_{1,n_y} \leftarrow K_{1,n_y-1}$
  {Parcours de l'intérieur du domaine.}
  **for** $i = 1 \cdots n_x - 2$ **do**
    **for** $j = 1 \cdots n_y - 2$ **do**
      $KMin_{i+1,j+1} \leftarrow min\left(K_{i,j}, K_{i+1,j}, K_{i,j+1}, K_{i+1,j+1}\right)$
    **end for**
  **end for**
  {Parcours des frontières gauche et droite du domaine.}
  **for** $j = 1 \cdots n_y - 2$ **do**
    $KMin_{1,j+1} \leftarrow min\left(K_{1,j}, K_{1,j+1}\right)$
    $KMin_{n_x,j+1} \leftarrow min\left(K_{n_x-1,j}, K_{n_x-1,j+1}\right)$
  **end for**
  {Parcours des frontières basse et haute du domaine.}
  **for** $i = 1 \cdots n_x - 2$ **do**
    $KMin_{i+1,1} \leftarrow min\left(K_{i,1}, K_{i+1,1}\right)$
    $KMin_{i+1,n_y} \leftarrow min\left(K_{i,n_y-1}, K_{i+1,n_y-1}\right)$
  **end for**
  {Calcul des dérivées partielles à partir du tableau contenant les $KMin$.}
  **for** $i = 1 \cdots n_x$ **do**
    **for** $j = 1 \cdots n_y$ **do**
      $z^x_{i,j} \leftarrow \lambda \frac{3}{4h} KMin_{i,j}$
      $z^y_{i,j} \leftarrow z^x_{i,j}$
    **end for**
  **end for**

D'où :

$$\begin{aligned} S &\leqslant \lambda \frac{12}{h} \left(z_{i+1,j+1} - z_{i,j}\right), \\ &\leqslant \frac{12}{h} \left(z_{i+1,j+1} - z_{i,j}\right). \end{aligned}$$

La condition (4.21) est donc satisfaite en sortie de l'algorithme.

De même, on prouve que la condition (4.22) est elle aussi satisfaite en sortie d'algorithme.

Nous avons donc vu que les valeurs des dérivées partielles en sortie de l'algorithme 2 vérifient les conditions du théorème 4.2. □

***Exemple numérique***

En reprenant l'exemple du tableau 4.1 contenant les valeurs de hauteur d'une fonction échantillonnée sur une grille $4 \times 4$, nous allons cette fois-ci utiliser l'algorithme 2 pour calculer des dérivées admissibles sans avoir besoin d'estimation initiales. Le résultat est montré au tableau 4.5.

| | | | |
|---|---|---|---|
| 5.25 | 3. | 3. | 3.75 |
| 0.75 | 0.75 | 3. | 3.75 |
| 0.75 | 0.75 | 1.5 | 1.5 |
| 4.5 | 2.25 | 1.5 | 1.5 |

**TABLE 4.5 –** *Les dérivées partielles admissibles calculées avec l'algorithme 2. Les valeurs de fonctions en entrée sont données dans le tableau 4.1.*

## 4.7 RÉSULTATS

Le premier exemple correspond à une grille de $4 \times 2$ valeurs de fonction satisfaisant les contraintes de monotonie relâché (4.7). Les valeurs de fonctions sont bien croissantes selon les lignes, mais pas selon les colonnes. elles ne satisfont donc pas la monotonie axiale. Ceci implique que la méthode précédente comme en [27] ne peut pas être appliquée dans ce cas. Quant à notre méthode, elle permet de construire une surface monotone, i.e. une surface qui ne contient pas de point critique à partir de ces données. Figure 4.10 montre le résultat de notre méthode : une surface de degré 3 et $C^1$ par morceaux, composée de 3 patchs de Sibson modifiés (12 patchs de Bézier triangulaires). A gauche, la surface est visualisée avec des lignes isovaleur. Comme il n'y a pas de lignes isovaleur fermées, la surface est bien monotone sans aucun maximum, ni minimum, ni point col dans l'intérieur de son domaine de définition. La nature lisse des ces lignes indique en plus que la surface est $C^1$, ce que nous avons demontré en section 4.4. A droite en figure 4.10 est montré la même surface avec, en bleu, les lignes isoparamétriques. Comme nous travaillons avec des patchs triangulaires, il y a trois directions pour les lignes isoparamétriques pour chaque triangle de Bézier.

Le deuxième exemple en figure 4.11 illustre l'estimation des dérivées partielles grâce à l'algorithme 1 présenté à la section 4.6. Dans cet exemple, la fonction $(x, y) \rightarrow (x + y)^3$ définie sur le domaine $[-0.4, +0.4]^2$ est évaluée sur une grille $5 \times 5$ échantillonnant le domaine de définition. Les points de l'ensemble $y = -x$ correspondent aux points critiques dégénérés de la fonction. Mais cette fonction est monotone croissante selon la direction diagonale $(x + y)$. Les valeurs des dérivées partielles données en entrées à l'algorithme 1 sont les valeurs exactes des dérivées partielles de la fonction évaluées aux points de la grille. Les deux surfaces à gauche en figure 4.11 montrent notre interpolant de Sibson modifié avec les valeurs de dérivées partielles non modifiées. Les deux surfaces à droite montrent notre interpolant de Sibson modifié après application de l'algorithme 1 qui a corrigé les dérivées partielles pour qu'elles deviennent admissibles. Les lignes isovaleurs qui sont tracées sur les surfaces en bas de figure 4.11 montrent que l'utilisation des dérivées partielles exactes de la fonction conduit à des points critiques dans la surface reconstruite, alors que ces derniers sont supprimés par la correction des dérivées partielles. Notre interpolant de Sibson modifié ne préserve pas le caractère monotone de la fonction selon la direction $(x + y)$ quand les données proviennent d'une surface monotone. Il faut donc utiliser les algorithmes que nous avons proposés en section 4.6 afin de

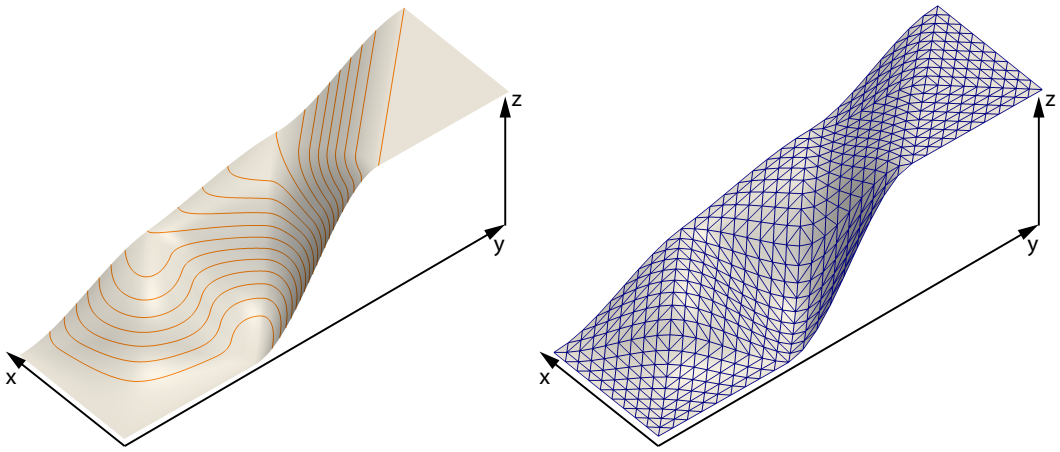



**FIGURE 4.10 –** *Une grille de $4 \times 2$ valeurs de fonction est interpolée par notre interpolant monotone $C^1$. Image de gauche : On peut voir que notre méthode ne crée pas de points critiques car les lignes isovaleurs ne sont pas fermées. Elle est donc bien monotone. Image de droite : Les lignes isoparamétriques des douze patchs de Bézier cubiques (3 patchs de Sibson modifiés).*

construire une surface monotone.

L'exemple suivant en figure 4.12 montre l'interpolation nomotone d'une grille de $10 \times 10$ valeurs de fonction. Les valeurs aux sommets de la grille ont été prises aléatoirement , mais celles-ci devaient respecter la condition (4.7) de croissance monotone le long des diagonales de la grille. Pour cela, on applique un traitement à chaque diagonale. Pour la diagonale comprise entre les points $(1, j)$ et $(n - j + 1, n)$ avec $1 \leqslant j \leqslant n$, nous faisons :

- une valeur aléatoire entre 0.0 et 1.0 est tirée. Cette valeur est assignée à la valeur de fonction au point $(1, j)$.
- Pour $k$ allant de 1 à $j$, on tire une valeur aléatoire $a$ entre 0.0 et 1.0. On peut alors donner une valeur à la fonction au point $(1+k, j+k)$ par $f(1+k, j+k) = a + f(k, j+k-1)$.

Dans cet exemple, nous avons utilisé l'algorithme 2, décrit à la section 4.6 pour produire des valeurs de dérivées partielles admissibles aux sommets de la grille. Notons que ces données ne peuvent pas être gérées par les méthodes précédentes telle que [27] car les valeurs aux sommets de la grille ne sont monotones ni le long des lignes, ni le long des colonnes. Notre méthode par contre est capable de générer une surface $C^1$ monotone qui ne contient pas de point critique, comme cela peut être vu grâce aux lignes isovaleurs. L'image de gauche montre un zoom où l'on peut voir les lignes isoparamétriques des patchs de Bézier.

Concernant les questions de temps de calculs, tous les exemples présentés ont été calculé en moins de 1 ms. Les exemples que nous avons montrés ici ont été calculés sur de petites tailles de grilles pour pouvoir mieux voir les propriétés topologiques et géométriques des résultats. Cependant, nous avons généré des exemples sur des grilles de taille $10^3 \times 10^3$ où nous avons appliqué nos algorithmes d'évaluation des dérivées partielles. Les résultats ont été obtenus en 3 ms, ce qui comprend le temps de calculs des dérivées partielles admissibles et l'évaluation de la surface. Il est à noter que dès lors que les dérivées partielles ont été calculées, l'évaluation des points de contrôles des patchs de Sibson modifié et l'évaluation des surfaces pour les patchs de

**FIGURE 4.11 –** *Interpolation d'une grille échantillonnée à partie d'une fonction analytique. Les images de la partie gauche montrent le résultat de l'interpolation lorsque les dérivées partielles exactes de la fonction sont utilisées pour calculer l'interpolant. Les lignes isovaleurs indiquées sur l'image du bas montrent la présence de points critiques sur la surface. La partie droite montre que les points critiques sont supprimés après la modification des dérivées partielles par notre algorithme 1 (section 4.6).*

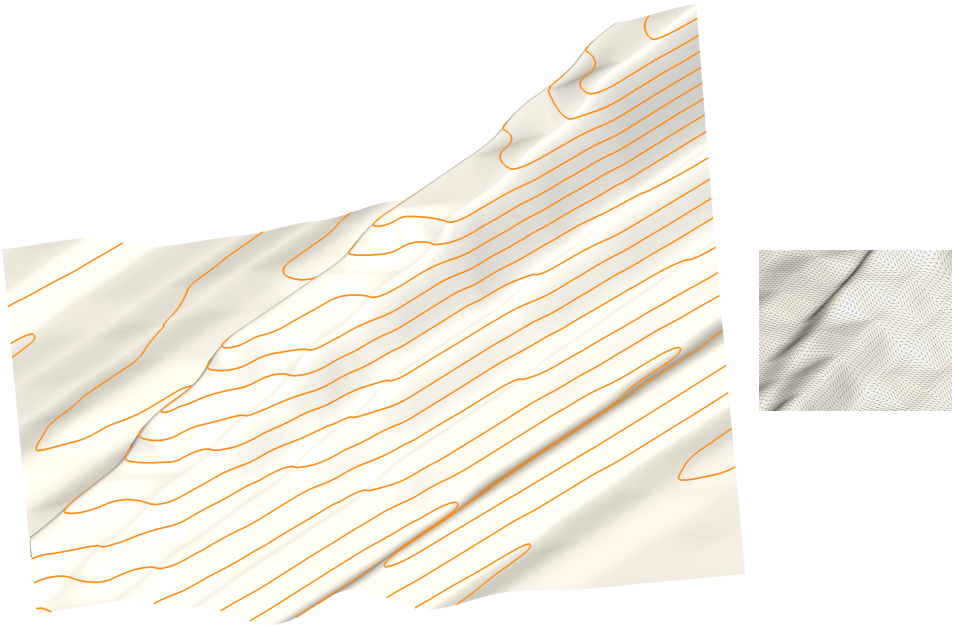


**FIGURE 4.12 –** *Interpolation de valeurs aux sommets d'une grille régulière* $10 \times 10$*. Les valeurs aux sommets ont été calculées aléatoirement sous conditions de monotonie le long des diagonales* (4.7)*. Comme ni les valeurs le long des lignes, ni les valeurs le long des colonnes ne sont croissantes, nous ne pouvons pas appliquer les méthodes précédentes à ce problème d'interpolation monotone. Notre méthode permet de calculer une surface* $C^1$ *ne contenant pas de point critique. Les dérivées partielles sont calculées avec l'algorithme 2 défini à la section 4.6. L'image de gauche montre un zoom avec les lignes isoparamétriques.*

Bézier peuvent être fait en parallèle puisque pour faire ces calculs pour une cellule de la grille, nous avons seulement besoin de connaître la valeur à interpoler aux quatre sommets ainsi que les dérivées partielles en ces points.

## 4.8 CONCLUSION

Dans ce chapitre, nous avons présenté une méthode permettant d'interpoler de manière monotone des valeurs sur une grille 2D qui sont monotones selon la direction $x + y$. La surface interpolante est $C^1$ et est créée grâce à l'utilisation des divisions de Sibson qui correspondent à un assemblage de quatre surfaces de Bézier triangulaires cubiques. Nous avons dérivé les conditions sur les dérivées partielles dans les directions $x$ et $y$ afin que la surface interpolante soit monotone. De plus, nous avons proposé un algorithme permettant de construire de telles valeurs de dérivées et un deuxième qui modifie les valeurs des dérivées partielles lorsque celles-ci ne satisfont pas les conditions énoncées.

Cette méthode va nous permettre dans la suite de construire des surfaces monotones que nous utiliserons afin de calculer des surfaces monotones sur les 2-cellules des complexes de Morse-Smale simplifiés. Notre aimerions travailler sur la généralisation de ces résultats à des valeurs données sur un maillage triangulaire et non plus un maillage régulier. Nous voudrions aussi porter notre attention sur le fait que la surface que nous trouvons n'est pas toujours visuellement plaisante. Nous pensons qu'il serait intéressant de trouver comment choisir

les dérivées partielles aux points de la grille afin que notre reconstruction soit visuellement plaisante dans tout les cas de figure.

CHAPITRE

# 5

# RECONSTRUCTION POLYNOMIALE PAR MORCEAUX DE CHAMPS SCALAIRES À PARTIR DE COMPLEXES DE MORSE-SMALE SIMPLIFIÉS

## 5.1 INTRODUCTION

Dans cette partie, nous allons nous intéresser à la reconstruction de champs scalaires à partir de complexes de Morse-Smale simplifiés. À l'instar des méthodes de reconstruction par minimisation d'une énergie sur un maillage (section 3.2.3), notre méthode comporte deux étapes. La première est un lissage des 1-cellules tout d'abord dans le plan $(x, y)$, mais aussi les valeurs de hauteurs du champs scalaires le long de celles-ci. Ce dernier lissage comporte des contraintes de monotonie pour ne pas avoir de points critiques le long des 1-cellules. Ensuite, pour chaque 2-cellule, nous construisons une surface polynomiale par morceaux monotone interpolant les courbes définies sur les 1-cellules qui sont les faces de la 2-cellules. Pour cela, nous paramétrisons la 2-cellule sur le carré unité. La surface est alors calculée grâce aux résultats du chapitre précédent.

La section 5.2 décrit les données que nous avons en entré de notre algorithme, c'est-à-dire comment nous les obtenons et comment nous les transformons avant de les utiliser. Ensuite, la section 5.3 montre le traitement que nous appliquons aux 1-cellules. La section 5.4 définit la reconstruction de la surface sur chaque 2-cellule. En section 5.6, nous montrons des résultats de notre algorithme.

Les travaux présentés dans ce chapitre ont fait l'objet d'une présentation d'un poster à la conférence IEEE Vis 2014[4] ainsi que d'une présentation à la conférence AFIG 2014[3]. Ils ont de plus été présentés à la conférence internationale « Topology-Based Methods in Visualization 2015 » dont l'article correspondant est en cours de parution.

## 5.2 DONNÉES EN ENTRÉE

***Algorithme de calcul d'un complexe de Morse-Smale simplifié :***

Notre méthode prend en entrée un champ scalaire discret $f : D \subset \mathbb{R}^2 \to \mathbb{R}$ avec $D$ une triangulation planaire. Les valeurs de $f$ sont données en chaque sommet de la triangulation $D$. Nous avons, de plus, un complexe de Morse-Smale simplifié calculé à partir d'une valeur de seuil de persistence. Cette valeur est donnée sous la forme d'un pourcentage $p$. Toutes les paires de points critiques ayant une valeur de persistence inférieure à $p$ multiplié par la persistence maximale (c'est-à-dire la valeur maximale de la fonction moins la valeur minimale) sont supprimées du complexe de Morse-Smale par ordre croissant de persistence. Une valeur de $p$ à $100\%$ signifie donc une simplification maximale, alors que si $p$ est à $0\%$, il n'y a pas de simplification.

Pour calculer le complexe de Morse-Smale simplifié, nous utilisons l'algorithme de Shivashankar *et al.* [44]. Cet algorithme est basé sur l'approche discrète (voir la section 2.4 du chapitre 2) pour calculer les complexes de Morse-Smale et les simplifier. Il permet à l'utilisateur de donner une fonction définie sur une grille et va lui-même calculer l'extension de cette fonction sur le complexe sous-jacent à la grille.

***Données en sortie de l'algorithme de calcul :***

En sortie de l'algorithme de calcul d'un complexe de Morse-Smale, nous avons des fichiers encodant les cellules du complexe de Morse-Smale simplifié. Les cellules sont données par des listes d'indices de la triangulation pouvant correspondre à des sommets, à des arêtes ou encore à des triangles :

- les 0-cellules du complexe de Morse-Smale (donc les points critiques de la fonction) sont encodées par la composante de même dimension que l'indice du point critique qui doit être décrit, cela signifie qu'un minimum sera un sommet, un point col une arête et un maximum sera donné par un indice de face ;
- il y a deux cas pour l'encodage des 1-cellules : les 1-cellules liant les minima aux points col sont données sous la forme d'une liste d'arêtes et celles reliant les maxima aux points col correspondent à une liste de faces ;
- les 2-cellules sont données par des listes de faces de la triangulation.

***Transformation des données des 1-cellules :***

Il est préférable de travailler sur le même type de composante pour un même type de cellule du complexe de Morse-Smale. Les 1-cellules vont donc être transformées pour être définies sous la forme de liste d'arêtes de la triangulation. Pour cela nous allons devoir insérer des triangles à l'intérieur de la triangulation initiale. De plus, nous allons aussi transformer les 0-cellules en sommets de la triangulation. Cette transformation va découler de celle donnée par les 1-cellules.

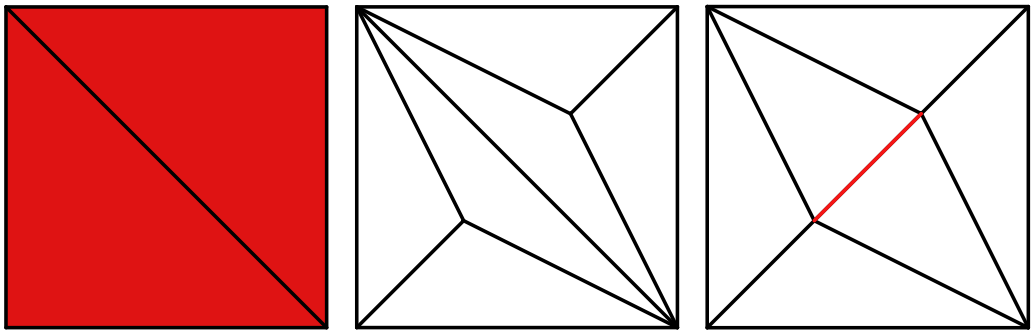
Supposons que nous voulions transformer une 1-cellule reliant un maximum à un point col. Cette cellule nous est donnée sous la forme d'une liste de triangles que nous voulons transformer en une liste d'arêtes. Cette liste d'arêtes correspond à une polyline passant par les barycentres des triangles de la ligne initiale. Nous commençons donc par scinder les triangles de la liste en trois triangles en ajoutant comme point interne le barycentre du triangle. Ensuite, il faut créer une arête entre deux barycentres de triangles se succédant dans la polyline. Pour cela, nous cherchons les deux triangles qui ont une arête commune et qui ont chacun pour sommet l'un des deux barycentres. Nous échangeons alors l'arête commune par une arête reliant ces deux barycentres. La transformation est illustrée sur deux triangles adjacents d'une 1-cellule à la figure 5.1. Nous voyons alors que le maximum sera créé au barycentre d'un triangle. Pour le point col, défini comme une arête dans la théorie de Morse discrète, nous allons créer un point au milieu de l'arête. Nous ajoutons une arête entre ce point et les barycentres des triangles adjacents à l'arête de point col.

À la figure 5.2, nous pouvons voir le résultat de la transformation des 1-cellules. Après cette transformation, nous avons donc bien que toutes les 0-cellules sont des sommets de la triangulation et les 1-cellules des suites d'arêtes. Il est à noter que des algorithmes tels que ceux décrits dans les articles [7] et [14] permettent d'avoir directement les cellules du complexe de Morse-Smale sous la forme désirée. Nous avons choisi d'utiliser cet algorithme car son code source peut être trouvé sur internet [1].

## 5.3 Lissage monotone des 1-cellules

Dans un premier temps, notre méthode reconstruit les 1-cellules. Deux lissages sont à appliquer sur les 1-cellules. La projection des 1-cellules dans le plan $(x, y)$ est seulement $C^0$. Une reconstruction lisse à partir de ces courbes n'est alors pas possible [7, 50]. Il nous faut donc d'abord lisser les 1-cellules dans le plan $(x, y)$. Ce lissage est présenté en section 5.3.2. Ensuite, le lissage des valeurs de hauteur le long des 1-cellules permet d'avoir une fonction monotone entre deux points critiques, alors qu'après les étapes de simplification du complexe de Morse-Smale, celles-ci ne sont pas garanties de l'être. Cette étape est présentée en section 5.3.3. Mais

1. À l'adresse http://vgl.serc.iisc.ernet.in/mscomplex/index.html



**(a)** *Deux triangles adjacents contenus dans la liste définissant une 1-cellule reliant un maximum à un point col.*

**(b)** *Les triangles sont subdivisés en trois triangles. Le point interne ajouté est le barycentre des triangles.*

**(c)** *Les deux barycentres devant être reliés par une arête, on applique un échange d'arêtes pour avoir cette arête.*

**FIGURE 5.1 –** *Étapes pour la transformation des 1-cellules reliant les maxima aux points col. Les étapes sont ici illustrées sur deux triangles adjacents appartenant à la liste des triangles définissant une de ces 1-cellules.*

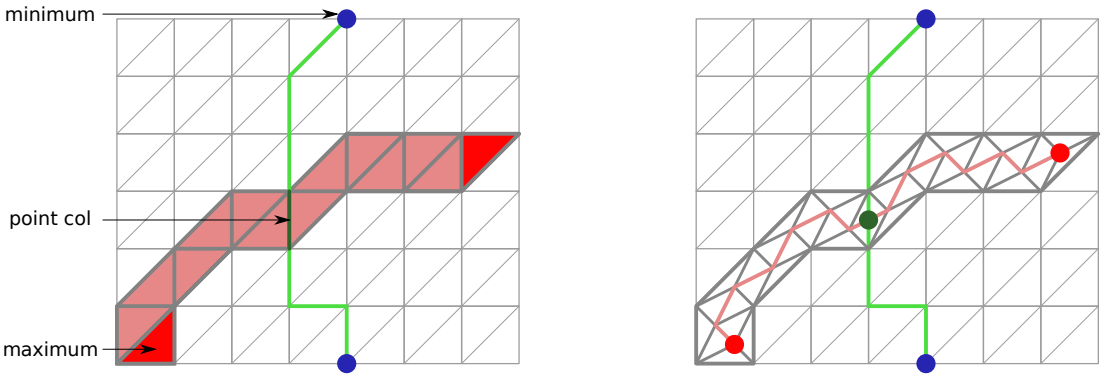



**FIGURE 5.2 –** *Les 1-cellules reliant les maxima aux points col nous sont données sous la forme de listes de triangles. Nous les transformons en une suite d'arêtes en ajoutant des triangles à l'intérieur des triangles contenus dans la liste des faces définissant la 1-cellule. Ces triangles ont pour sommets les anciens sommets de la triangulation et les barycentres des triangles de la liste des faces. Les points critiques sont eux aussi convertis. Ils deviennent des sommets de la triangulation.*

tout d'abord en section 5.3.1, nous allons présenter les points de jonctions : leur définition, la raison de leur présence dans nos données et le traitement que nous leur appliquons.

### 5.3.1 Points de jonctions : définition et traitement

La définition 2.21 présentée en section 2.2.2 présente les lignes intégrales comme des courbes qui suivent la direction du gradient de la fonction. Le théorème 2.3 nous indique que par un point passe une unique ligne intégrale. Ceci est vrai quand la fonction $f$ est définie de façon analytique. Dans notre cas, la fonction est définie de façon discrète, c'est-à-dire aux sommets d'une triangulation. La propriété de l'unicité de la ligne intégrale en un point du domaine n'est alors plus vérifiée. Il est donc possible que deux $V$-chemins (les équivalents des lignes intégrales dans le cas discret) soient égaux sur une partie de leur chemins.

Les 1-cellules sont des $V$-chemins et donc, deux 1-cellules peuvent être fusionnées sur un ensemble de vecteurs discrets les définissant. Par définition, les 1-cellules ascendantes (reliant les maxima aux points cols) correspondent à des ensembles de triangles alors que les 1-cellules descendantes (reliant les minima aux points cols) sont des ensembles d'arêtes de la triangulation. Pour que deux 1-cellules soient fusionnées sur une partie de leur trajet, il faut donc qu'elles soient soit toutes les deux ascendantes, soit toutes les deux descendantes. De plus, si deux 1-cellules ont une partie de leur parcours en commun, alors elles possèdent un point critique en commun comme face qui est soit un minimum dans le cas de 1-cellules descendantes, soit un maximum dans le cas des 1-cellules ascendantes. Prenons deux cellules ascendantes ayant une partie de leur géométrie commune. Celle-ci sera commune entre la maximum et un point que nous appellerons *point de jonction*. En ce point de jonction, les deux 1-cellules vont se séparer et ne se joindront plus.

Nous avons regardé le cas de deux 1-cellules fusionnées, mais il peut y avoir un nombre quelconque de 1-cellules fusionnées. Dans ce cas, il va exister un point de jonction par paire de 1-cellules, en sachant que deux ou plusieurs points de jonctions peuvent se retrouver au même endroit. Dans ce dernier cas, ils seront fusionnés et seront considérés comme un seul et unique point de jonction. La figure 5.3 montre des exemples de points de jonction.

Il existe deux types de points de jonction qui correspondent à la façon dont ils ont été créés :

**Le premier type** existe dans le complexe de Morse-Smale qui n'a pas subi de simplification. Ils apparaissent car le nombre de directions pour créer une 1-cellule à partir d'un point critique est fini. Par exemple, pour les 1-cellules ascendantes, à partir d'un maximum, il n'existe que trois direction. S'il existe plus de trois 1-cellules partant de ce maximum, il est alors obligatoire d'avoir des 1-cellules fusionnées et donc des points de jonction. Mais il n'est pas obligatoire que toutes les directions soient utilisées, et donc, s'il y a plus de 1-cellules que de nombre de direction sélectionnées par l'algorithme, alors des points de jonction apparaîtront.

**Le deuxième type** apparaît lors des étapes de simplification. Nous avons vu en section 2.4.3 que lors de la simplification explicite (le procédé n'est pas le même pour la simplification implicite, mais engendre des résultats similaires), des arcs ont été créés sur les chemins des arcs supprimés. Or ces arcs passaient par les points critiques supprimés. Ces points critiques supprimés peuvent alors devenir des points de jonction.

Pour le premier type, nous avions des points de jonction qui sont des points ordinaires de la fonction initiale, alors que les points de jonction du deuxième type correspondent à des points critiques supprimés dans le complexe de Morse-Smale simplifié. En pratique, nous ne faisons pas la distinction entre les deux types car il n'est pas possible de connaître le type d'un point critique en regardant seulement le complexe de Morse-Smale simplifié. Dans la suite, nous allons donc appliquer un traitement équivalent à tout les points de jonction.

Pour lisser de façon monotone les 1-cellules, il va nous falloir faire attention au fait que celles-ci peuvent être partiellement fusionnées. Nous allons donc découper les 1-cellules aux points de jonctions, puis lisser les différentes parties indépendamment. Il nous faut donc que les valeurs aux extrémités des morceaux de 1-cellules soient monotones le long des 1-cellules et bornées par les valeurs des deux points critiques de la 1-cellule, sinon il ne sera pas possible de reconstruire des 1-cellules globalement monotones. Or, comme les points de jonction du deuxième type correspondent à des points critiques de la fonction initiale, la suite comprenant la valeur des points critiques et des points de jonction le long d'une 1-cellule n'est pas assurée d'être monotone. Il faut donc calculer des valeurs pour la fonction reconstruite aux points de jonction respectant la règle de monotonie le long d'une 1-cellule.

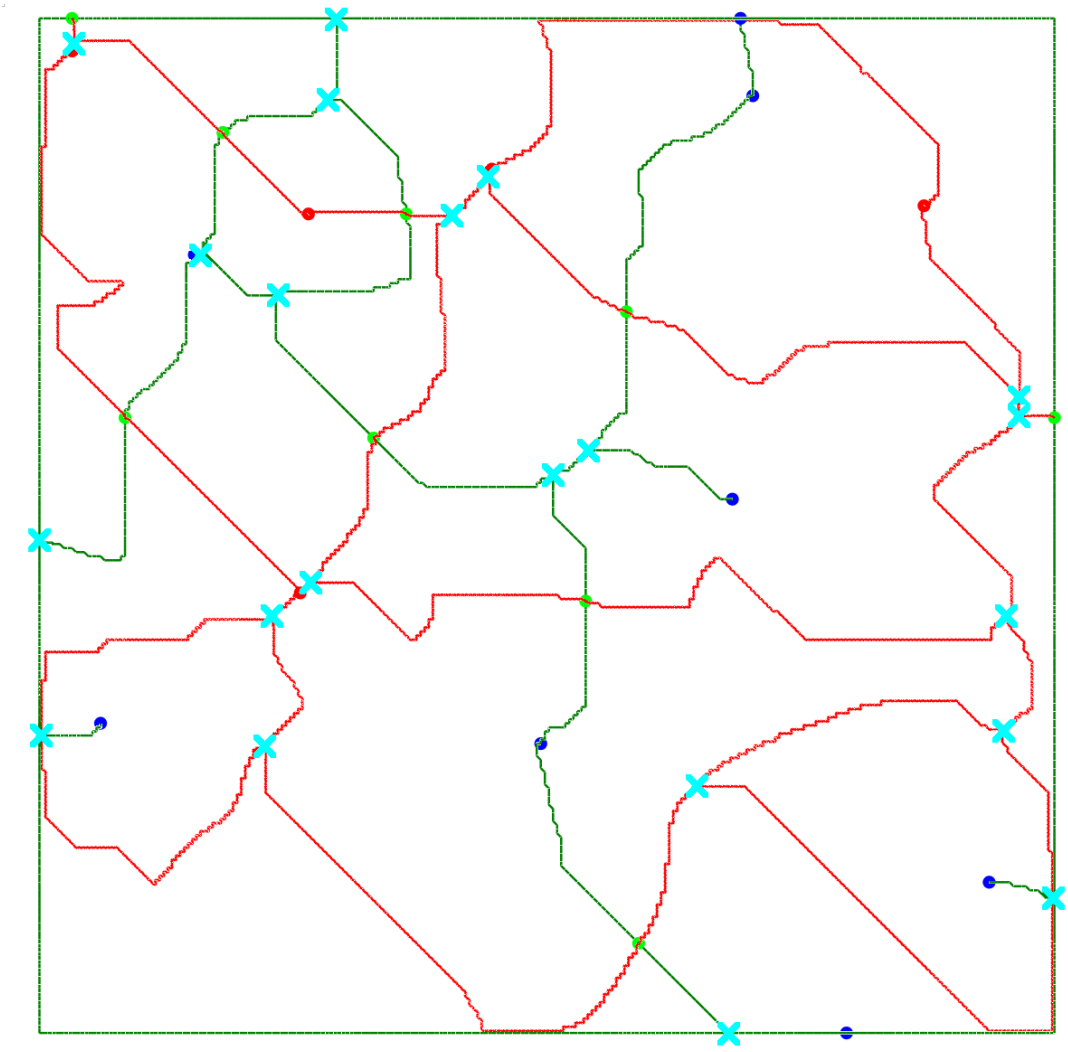


**FIGURE 5.3 –** *Exemple de complexe de Morse-Smale contenant des points de jonctions. Les croix en bleus claires indiquent des points de jonctions. Les points en rouge sont des maxima, les verts des points cols et les bleus foncés des minima. Les lignes vertes sont les 1-cellules descendantes alors que les rouges sont les 1-cellules ascendantes. Les points de jonction indiquent que des 1-cellules sont fusionnées.*

**Choix des valeurs de fonction aux points de jonction :** La méthode que nous proposons pour choisir la valeur de hauteur aux points de jonction est de résoudre l'optimisation suivante :

$$\sum |\tilde{f}(\mathbf{p_j}) - f(\mathbf{p_j})| \longrightarrow min$$

sous les contraintes :

$$max\_lower(\mathbf{p_j}) \leqslant \tilde{f}(\mathbf{p_j})$$
$$\tilde{f}(\mathbf{p_j}) \leqslant min\_larger(\mathbf{p_j})$$

avec $f$ la fonction initiale, $\tilde{f}$ la fonction que l'on veut évaluer et $\mathbf{p_j}$ les points de jonction du complexe de Morse-Smale. La fonction $max\_lower$ (respectivement $min\_larger$) correspond au maximum (respectivement minimum) des valeurs de hauteurs des points de jonctions et des

points critiques liés au point de jonction $\mathbf{p_j}$ par un morceau de 1-cellules qui devraient avoir une plus petite (respectivement plus grande) valeur de fonction que lui.

L'optimisation ci-dessus n'est pas linéaire à cause de la valeur absolue dans la fonction-coût. Il est possible de linéariser ce problème en utilisant le résultat de l'annexe A. L'optimisation linéaire obtenue est alors :

$$\sum (r_i^+ + r_i^-) \longrightarrow min$$

sous les contraintes :

$$\begin{aligned}
& r_i^+ \geqslant 0 \\
& r_i^- \geqslant 0 \\
& r_i^+ - r_i^- = \tilde{f}(\mathbf{p_j}) - f(\mathbf{p_j}) \\
& max_lower(\mathbf{p_j}) \leqslant \tilde{f}(\mathbf{p_j}) \\
& \tilde{f}(\mathbf{p_j}) \leqslant min_larger(\mathbf{p_j}).
\end{aligned}$$

Pour résoudre ce problème linéaire, nous utilisons la bibliothèque GLPK [2]. Cette bibliothèque propose deux algorithmes différents pour résoudre les problèmes d'optimisation linéaire. Le premier est l'algorithme du simplexe et le deuxième est l'algorithme des points intérieurs.

Le principe de la *méthode du simplexe* est de construire le polyèdre convexe correspondant à l'ensemble des points qui respectent les contraintes. En partant d'un sommet du polyèdre, il va se déplacer de sommet en sommet en diminuant à chaque fois la valeur de la fonction-objectif. Cet algorithme peut soit donner une solution au problème, soit dire que le problème est non borné (ce qui signifie que la valeur optimale de la minimisation est $-\infty$), soit dire que le problème n'est pas réalisable, c'est-à-dire que les contraintes ne sont pas compatibles et qu'il n'existe donc pas de point dans l'espace des solutions admissibles. La solution est donnée en un nombre finie d'opérations. La complexité de cette algorithme est non polynomiale, mais la complexité en moyenne est polynomiale.

Le deuxième type d'algorithme correspond aux *méthodes des points intérieurs+*. Ces algorithmes partent d'un point à l'intérieur de l'espace des solutions admissibles et donnent différentes méthodes pour se rapprocher d'une solution du problème. La solution donnée par ces algorithmes est une approximation d'une des solutions au problèmes avec une erreur $\epsilon$ donnée. On peut ensuite appliquer d'autres méthodes pour donner une des solutions à partir de cette approximation. La complexité est polynomiale, mais est, pour les meilleurs méthodes, équivalente à la complexité en moyenne de l'algorithme du simplexe.

Nous avons choisi d'utiliser la méthode du simplexe implémentée dans la bibliothèque GLPK pour sa simplicité d'utilisation et car son utilisation ne comporte pas de cas particulier qui risquerait de ne pas fonctionner correctement dans nos cas, contrairement à la méthode des points intérieurs implémentée dans GLPK.

Cette optimisation linéaire permet donc de calculer une fonction $\tilde{f}$ définie pour les points de jonctions ayant ses valeurs le plus proche possible des valeurs de $f$ et qui vérifie la monotonie le long des 1-cellules. Après cette optimisation, nous pouvons alors procéder au lissage des 1-cellules entre deux points (de jonction ou critiques) en interpolant les valeurs aux deux extrémités.

### 5.3.2 Lissage des 1-cellules dans le plan $(x, y)$

Dans cette section, nous cherchons à faire un lissage des morceaux des 1-cellules. Dans la partie précédente, les 1-cellules ont été découpées aux points de jonction. Nous allons appliquer

le lissage sur les différents morceaux de 1-cellules après la découpe. Cela signifie que nous allons travailler sur des parties des 1-cellules qui ne contiennent pas de points de jonction dans leurs intérieurs. Cela veut aussi dire que si deux 1-cellules sont partiellement fusionnées, alors le lissage sur le morceau commun des deux 1-cellules ne sera faît qu'une seule fois et les deux 1-cellules vont récupérer les valeurs à la fin du lissage.

Nous commençons par lisser les 1-cellules dans le plan $(x, y)$. Cela consiste en une approximation dans le plan des points le long des morceaux de 1-cellules par des courbes de degré 3 par morceaux. Ces courbes seront des courbes de Bézier afin de pouvoir utiliser les résultats du chapitre 4. L'ensemble des courbes de Bézier au bord d'une 2-cellule donnera alors des conditions pour construire une grille où seront définis des interpolants de Sibson.

Cette étape de lissage est composée de trois étapes :

- approximation au sens des moindres carrés des points composant le morceau de 1-cellule par une courbe B-spline quadratique ;
- transformation de la B-spline quadratique en ensembles de courbes de Bézier quadratiques par insertion de noeuds ;
- élévation de degré des courbes de Bézier, le degré des courbes passant de 2 à 3.

***Étape 1 : Approximation au sens des moindres carrés par une courbe B-spline quadratique***

Soit $(\mathbf{x}_i)_{0\leqslant i\leqslant n}$ les $n+1$ points le long du morceaux de 1-cellules. Les points $\mathbf{x}_0$ et $\mathbf{x}_n$ correspondent à des points critiques ou à des points de jonction. Nous cherchons à approximer cet ensemble de points au sens des moindres carrés par une courbe B-spline quadratique et uniforme $\mathbf{c}(t) = \sum_{i=0}^{k+1} \mathbf{c}_i N_i^2(t) \subset \mathbb{R}^2$, avec $t \in [0,1]$. Les valeurs $(\mathbf{c}_i)_{0\leqslant i\leqslant k+1}$ sont les points de contrôle de la courbe $\mathbf{c}$ avec $k \in \mathbb{N}^*$ le nombre de points de contrôle. De plus, l'ensemble des fonctions $(N_i^2)_{i=0,\dots,k+1}$ correspond à la base des fonctions B-splines de degré 2 définies sur les noeuds $0 = u_{-2} = u_{-1} = u_0 < \cdots < u_k = u_{k+1} = u_{k+2} = 1$ pris uniformément sur $[0,1]$. Les fonctions $N_i^p : [0,1] \to \mathbb{R}$, $p \in \mathbb{N}$, sont définies récursivement par :

$$N_i^0(t) = \begin{cases} 1 & \text{si } u_i \leqslant t < u_{i+1} \\ 0 & \text{sinon} \end{cases}$$

$$N_i^p(t) = \frac{t - u_i}{u_{i+p} - u_i} N_i^{p-1}(t) + \frac{u_{i+p+1} - t}{u_{i+p+1} - u_{i+1}} N_{i+1}^{p-1}(t).$$

La dérivée de la fonction B-spline quadratique est donnée par

$$\frac{d}{dt}\mathbf{c}(t) = \sum_{i=0}^{k+1} \mathbf{c}_i \frac{d}{dt} N_i^2(t), \tag{5.1}$$

où

$$\frac{d}{dt} N_i^2(t) = \frac{p}{u_{i+p} - u_i} N_i^1(t) + \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1}^1(t). \tag{5.2}$$

correspond à la dérivée des fonctions $N_i^2$.

**Remarque :** Dans le cas général, la dérivée d'une fonction B-spline $h_\alpha = \sum_{i=0}^{k+1} \alpha_i N_i^p(t)$ de degré $p$ est donnée par

$$\begin{aligned}\frac{d}{dt}h_\alpha(t) &= \sum_{i=0}^{k+1} \alpha_i \frac{d}{dt} N_i^p(t) \\ &= \sum_{i=0}^{k} \gamma_i N_{i+1}^{p-1}(t)\end{aligned}$$

avec $\gamma_i = \frac{p}{u_{i+p+1}-u_{i+1}}(\alpha_{i+1} - \alpha_i)$.

Soit les $n+1$ paramètres $0 = t_0 < t_1 < \cdots < t_n = 1$ pris uniformément sur le segment $[0,1]$. Pour tout $i = 0, \ldots, n$, nous allons chercher à approximer le point $x_i$ par le point $\mathbf{c}(t_i)$. Nous avons deux contraintes : nous voulons que les points $\mathbf{x}_0 = \mathbf{c}(0)$ et $\mathbf{x}_n = \mathbf{c}(1)$, ce qui signifie que nous voulons interpoler les positions des points critiques et des points de jonction en début et fin de courbe. Ensuite, nous allons chercher à résoudre le problème d'approximation au sens des moindres carrés suivant :

$$\left\| \begin{pmatrix} \mathbf{x}_1 - \sum_{i=0}^{k+1} \mathbf{c}_i N_i^2(t_1) \\ \vdots \\ \mathbf{x}_{n-1} - \sum_{i=0}^{k+1} \mathbf{c}_i N_i^2(t_{n-1}) \end{pmatrix} \right\|_2 \longrightarrow \min_{(\mathbf{c}_1,\ldots,\mathbf{c}_k)}$$

En posant $A = \begin{pmatrix} N_1^2(t_1) & \cdots & N_k^2(t_1) \\ \vdots & \ddots & \vdots \\ N_1^2(t_{n-1}) & \cdots & N_k^2(t_{n-1}) \end{pmatrix}$, résoudre ce problème de minimisation sur les $(\mathbf{c}_1, \ldots, \mathbf{c}_k)$ revient à résoudre l'équation matricielle suivante :

$$A^t A \begin{pmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_k \end{pmatrix} = A^t \begin{pmatrix} \mathbf{x}_1 - N_0^2(t_1)\mathbf{c}_0 - N_{k+1}^2(t_1)\mathbf{c}_{k+1} \\ \vdots \\ \mathbf{x}_n - N_0^2(t_{n-1})\mathbf{c}_0 - N_{k+1}^2(t_{n-1})\mathbf{c}_{k+1} \end{pmatrix}$$

Ces valeurs $(\mathbf{c}_i)_{i=1,\ldots,k+1}$ définissent alors la courbe approximant la courbe passant par la suite des points $(\mathbf{x}_i)$. À la section 5.4, nous verrons que les surfaces que nous construisons sont composées de patchs triangulaires de Bézier cubiques. Comme les courbes que nous venons de lisser sont les courbes que nous souhaiterons interpoler dans la suite, il nous faut avoir la même représentation pour les courbes et pour les patchs. Nous devons donc transformer les courbes B-splines quadratiques en courbes de Bézier cubiques. Cette conversion est composée de deux étapes : tout d'abord, il faut insérer des noeuds pour transformer les courbes B-splines en courbes de Bézier quadratiques, ensuite, nous appliquons une élévation de degré sur les courbes de Bézier pour avoir des courbes de Bézier cubiques.

#### *Étape 2 : Insertion de noeuds*

Cette étape a pour but de transformer les courbes B-splines quadratiques en un ensemble de courbes de Bézier quadratiques. Pour cela, il faut saturer les noeuds de la B-spline, c'est-à-dire insérer des noeuds jusqu'à ce que ceux-ci soient de multiplicité égale au degré de la courbe B-spline. L'insertion d'un noeud dans une courbe B-spline a pour effet l'ajout d'un point de contrôle pour la courbe.

Soit une B-spline de degré 2 $\mathbf{c}(t) = \sum_{i=0}^{k+1} \mathbf{d}_i N_i^2(t)$. Comme à l'étape précédente, l'ensemble des noeuds de la B-spline est uniformes sur le segment $[0,1]$. Soient $0 = t_0 = t_1 = t_2 < t_3 < \ldots < t_{k+1} < t_{k+2} = t_{k+3} = t_{k+4} = 1$ avec $t_j = \frac{j-2}{k}$ pour $j = 2, \ldots, k+2$ les noeuds de la B-spline $\mathbf{c}$ et ses points de contrôles $(d_0, \ldots, d_{k+1})$. Il faut donc ajouter chaque noeud $t_j$ $(j = 3, \ldots, k+1)$ afin de saturer les noeuds de la B-spline. Après l'insertion de tout ces noeuds, la B-spline s'écrit $\mathbf{c}(t) = \sum_{i=0}^{2k} \hat{\mathbf{d}}_i N_i^2(t)$ avec

$$\hat{\mathbf{d}}_i = \begin{cases} \mathbf{d}_i & \text{si } i \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{i/2} + \mathbf{d}_{i/2+1}\right) & \text{si } i \text{ pair et } 1 < i < 2k-1 \\ \mathbf{d}_{(i+1)/2} & \text{si } i \text{ impair et } 1 < i < 2k-1 \\ \mathbf{d}_{i-(k-1)} & \text{si } i \geqslant 2k-1 \end{cases}. \tag{5.3}$$

*Démonstration.* Posons la propriété pour $i = 0, \ldots, k+1$ :

$P_i$ : "Après l'insertion des noeuds $t_3$ à $t_{3+i-1}$, les $k+2+i$ points de contrôle sont pour $j = 0, \ldots, k+1+i$ :

$$\hat{\mathbf{d}}_j = \begin{cases} \mathbf{d}_j & \text{si } j \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{j/2} + \mathbf{d}_{j/2+1}\right) & \text{si } j \text{ pair et } 1 < j < 2i+1 \\ \mathbf{d}_{(j+1)/2} & \text{si } j \text{ impair et } 1 < j < 2i+1 \\ \mathbf{d}_{j-i} & \text{si } j \geqslant 2i+1 \end{cases} \tag{5.4}$$

"

Montrons que celle-ci est vraie par un raisonnement par récurrence.

***Initialisation :***

Après 0 insertion, la B-spline est définie par $k+2$ points de contrôle donnés par :

$$\begin{aligned} \hat{\mathbf{d}}_j &= \begin{cases} \mathbf{d}_j & \text{si } j \leqslant 1 \\ \mathbf{d}_{j-0} & \text{si } j \geqslant 2*0+1 \end{cases} \\ &= \mathbf{d}_j \end{aligned}$$

Donc $P_0$ est vraie.

***Hérédité ($P_i \Rightarrow P_{i+1}$) :***

Supposons que nous avons inséré les $i$ premiers noeuds $t_3, \ldots, t_{3+i-1}$. En utilisant la propriété de récurrence pour $i$, on obtient que les points de contrôle de la B-spline après les $i$ insertions sont pour $j = 0, \ldots, k+1+i$ :

$$\hat{\mathbf{d}}_j = \begin{cases} \mathbf{d}_j & \text{si } j \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{j/2} + \mathbf{d}_{j/2+1}\right) & \text{si } j \text{ pair et } 1 < j < 2i+1 \\ \mathbf{d}_{(j+1)/2} & \text{si } j \text{ impair et } 1 < j < 2i+1 \\ \mathbf{d}_{j-i} & \text{si } j \geqslant 2i+1 \end{cases}$$

Comme nous avons inséré $i$ noeuds, la nouvelle liste de noeuds est $0 = \hat{t}_0 = \hat{t}_1 = \hat{t}_2 < \hat{t}_3 = \hat{t}_4 < \ldots < \hat{t}_{2i+1} = \hat{t}_{2i+2} < \hat{t}_{2i+3} < \ldots < \hat{t}_{k+1+i} < \hat{t}_{k+2+i} = \hat{t}_{k+3+i} = \hat{t}_{k+4+i} = 1$. L'insertion suivante sera celle du noeud $\tilde{t} = t_{3+i} = \hat{t}_{2i+3}$, avec $\tilde{t} \in [\hat{t}_{2i+3}, \hat{t}_{2i+4}]$. En

utilisant la formule de l'insertion de noeud , nous obtenons que les points de contrôle pour $l = 0, \dots k+i+2$ sont :

$$\tilde{\mathbf{d}}_l = \begin{cases} \hat{\mathbf{d}}_l & \text{si } 0 \leqslant l \leqslant 2i+1 \\ \omega_{l,2}(\tilde{t})\hat{\mathbf{d}}_l + (1-\omega_{l,2}(\tilde{t}))\hat{\mathbf{d}}_{l-1} & \text{si } l = 2i+2,\ 2i+3 \\ \hat{\mathbf{d}}_{l-1} & \text{si } 2i+4 \leqslant l \leqslant k+i+2 \end{cases}$$

avec

$$\omega_{l,n}(t) = \begin{cases} \frac{t-t_l}{t_{l+n}-t_l} & \text{si } t_l < t_{l+n} \\ 0 & \text{sinon} \end{cases}$$

Calculons alors la valeur de $\tilde{\mathbf{d}}_l$ pour $l = 2i+2,\ 2i+3$.

– Cas $l = 2i+2$ : Comme les noeuds initiaux ont été pris uniformément sur $[0,1]$, on a $\omega_{2i+2,2}(\tilde{t}) = \frac{\hat{t}_{2i+3}-\hat{t}_{2i+2}}{\hat{t}_{2i+4}-\hat{t}_{2i+2}} = \frac{1}{2}$. Nous obtenons donc que $\tilde{\mathbf{d}}_{2i+2} = \frac{1}{2}\left(\hat{\mathbf{d}}_{2i+1} + \hat{\mathbf{d}}_{2i+2}\right)$.

– Cas $l = 2i+3$ : Dans ce cas, on a que $\omega_{2i+3,2}(\tilde{t}) = \frac{\hat{t}_{2i+3}-\hat{t}_{2i+3}}{\hat{t}_{2i+5}-\hat{t}_{2i+3}} = 0$. D'où $\tilde{\mathbf{d}}_{2i+3} = \hat{\mathbf{d}}_{2i+2}$.

Nous avons donc pour $l = 0, \dots k+i+2$ :

$$\begin{aligned} \tilde{\mathbf{d}}_l &= \begin{cases} \hat{\mathbf{d}}_l & \text{si } 0 \leqslant l \leqslant 2i+1 \\ \frac{1}{2}\left(\hat{\mathbf{d}}_{2i+1} + \hat{\mathbf{d}}_{2i+2}\right) & \text{si } l = 2i+2 \\ \hat{\mathbf{d}}_{l-1} & \text{si } 2i+3 \leqslant l \leqslant k+i+2 \end{cases} \\ &= \begin{cases} \mathbf{d}_l & \text{si } l \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{l/2} + \mathbf{d}_{l/2+1}\right) & \text{si } l \text{ pair et } 1 < l < 2i+1 \\ \mathbf{d}_{(l+1)/2} & \text{si } l \text{ impair et } 1 < l < 2i+1 \\ \mathbf{d}_{2i+1-i} & \text{si } l = 2i+1 \\ \frac{1}{2}\left(\hat{\mathbf{d}}_{2i+1} + \hat{\mathbf{d}}_{2i+2}\right) & \text{si } l = 2i+2 \\ \mathbf{d}_{l-(i+1)} & \text{si } l \geqslant 2i+3 \end{cases} \end{aligned}$$

Or, nous avons que $\mathbf{d}_{2i+1-i} = \mathbf{d}_{i+1} = \mathbf{d}_{(l+1)/2}$ pour $l = 2i+1$. De plus :

$$\begin{aligned} \tilde{\mathbf{d}}_{2i+2} &= \frac{1}{2}\left(\hat{\mathbf{d}}_{2i+1} + \hat{\mathbf{d}}_{2i+2}\right) \\ &= \frac{1}{2}\left(\mathbf{d}_{2i+1-i} + \mathbf{d}_{2i+2-i}\right) \\ &= \frac{1}{2}\left(\mathbf{d}_{l/2} + \mathbf{d}_{l/2+1}\right) \end{aligned}$$

avec $l = 2i+2$. La valeur des points de contrôle est donc :

$$\tilde{\mathbf{d}}_l = \begin{cases} \mathbf{d}_l & \text{si } l \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{l/2} + \mathbf{d}_{l/2+1}\right) & \text{si } l \text{ pair et } 1 < l < 2(i+1)+1 \\ \mathbf{d}_{(l+1)/2} & \text{si } l \text{ impair et } 1 < l < 2(i+1)+1 \\ \mathbf{d}_{l-i} & \text{si } l \geqslant 2(i+1)+1 \end{cases}$$

Donc $P_{i+1}$ est vraie.

Nous avons donc démontré que la propriété $P_i$ est vraie pour tout $i = 0, \dots, k+1$. Notamment, pour $i = k+1$, donc après l'insertion de tout les noeuds, nous obtenons la formule (5.3) voulue :

$$\hat{\mathbf{d}}_i = \begin{cases} \mathbf{d}_i & \text{si } i \leqslant 1 \\ \frac{1}{2}\left(\mathbf{d}_{i/2} + \mathbf{d}_{i/2+1}\right) & \text{si } i \text{ pair et } 1 < i < 2k-1 \\ \mathbf{d}_{(i+1)/2} & \text{si } i \text{ impair et } 1 < i < 2k-1 \\ \mathbf{d}_{i-(k-1)} & \text{si } i \geqslant 2k-1 \end{cases}$$

□

Après avoir utilisé la formule démontrée précédemment pour calculer les points de contrôles de la B-spline, il nous faut récupérer les points de contrôles par groupe de trois pour avoir les courbes sous la forme de courbes de Bézier de degré 2. Le résultat de cette étape est donc l'obtention de $k$ courbes de Bézier définies par les points de contrôles $(\hat{\mathbf{d}}_{2i}, \hat{\mathbf{d}}_{2i+1}, \hat{\mathbf{d}}_{2(i+1)})_{i=0,\dots,k-1}$.

***Étape 3 : Élévation de degré***

Il faut ensuite appliquer une élévation de degré à ces courbes de Bézier pour passer à des courbes de degré 3. Une élévation de degré permet d'avoir une courbe de degré supérieur et donc qui est définie par un point de contrôle de plus que la courbe initiale. Mais la courbe définie après l'application de l'élévation de degré va être exactement la même que la courbe initiale. La formule générale d'élévation de degré pour une courbe de Bézier de degré $n$ avec les points $(\mathbf{P}_i)_{i=0,\dots,n}$ comme points de contrôles ayant pour résultat une courbe de Bézier de degré $n+1$ de points de contrôles $(\mathbf{Q}_i)_{i=0,\dots,n+1}$ est :

$$\begin{cases} \mathbf{Q}_0 & = \mathbf{P}_0 \\ \mathbf{Q}_i & = \frac{i}{n+1}\mathbf{P}_{i-1} + (1 - \frac{i}{n+1})\mathbf{P}_i \text{ pour } 1 \leqslant i \leqslant n \\ \mathbf{Q}_{n+1} & = \mathbf{P}_n \end{cases}$$

Les courbes définies par les $(\mathbf{P}_i)_{i=0,\dots,n}$ et par les $(\mathbf{Q}_i)_{i=0,\dots,n+1}$ décrivent exactement la même courbe.

Dans notre cas, nous avons $n = 2$. Les points $(\mathbf{Q}_i)_{i=0,\dots,3}$ sont alors :

$$\begin{cases} \mathbf{Q}_0 & = \mathbf{P}_0 \\ \mathbf{Q}_1 & = \frac{1}{3}\mathbf{P}_0 + \frac{2}{3}\mathbf{P}_1 \\ \mathbf{Q}_2 & = \frac{2}{3}\mathbf{P}_1 + \frac{1}{3}\mathbf{P}_2 \\ \mathbf{Q}_3 & = \mathbf{P}_2 \end{cases}$$

Un exemple de l'application de cette méthode à une courbe de Bézier de degré 2 se trouve à la Figure 5.4.

À ce moment de la méthode, nous avons un ensemble de courbes de Bézier cubiques reliant les différents points critiques ou de jonction dans le plan $(x, y)$. Le résultat est un ensemble de courbes lisses. Cependant, comme chaque morceaux de courbe est approximé indépendamment des autres, il faut vérifier qu'il n'y ait pas d'intersections entre les morceaux de courbes. En effet, la théorie sur les lignes intégrales interdit d'avoir des 1-cellules qui s'intersectent. Pour chaque 2-cellule, nous vérifions qu'il n'y a pas d'intersections entre les courbes composant les

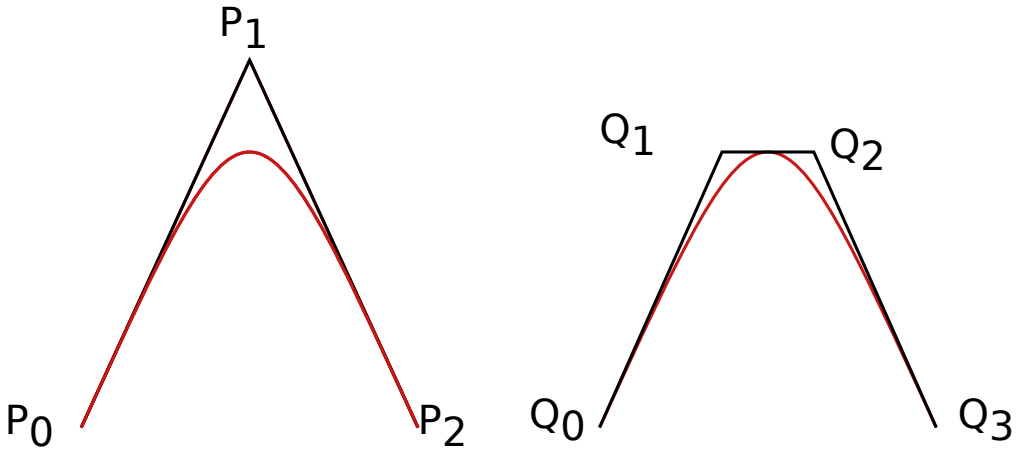



**FIGURE 5.4 –** *Exemple de l'élévation de degré à partir d'une courbe de Bézier de degré 2 (Figure 5.4a). Le résultat de l'élévation de degré est une courbe de Bézier de degré 3 qui correspond exactement à la courbe initiale.*

différentes 1-cellules qui sont ses faces. Pour cela, nous utilisons la méthode d'impliticisation des courbes de Bézier cubiques [42]. Nous décrivons comment celle-ci est définie dans l'annexe C. Cette méthode est celle détectant les intersections le plus rapidement entre deux courbes cubiques de Bézier[41]. Dans tout nos exemples, il n'y a jamais eu d'intersections pour la valeur de $k$ (nombre de morceaux de courbes) égale à 4 ou 7. Dans le cas où une intersection apparaitrait, nous proposons d'augmenter la valeur de $k$ pour les deux courbes s'intersectant. La figure 5.5 montre l'erreur de l'approximation aux moindres carrés en fonction du paramètre $k$ selon la direction $x$ et la direction $y$. En augmentant ce paramètre, l'erreur diminue donc et on peut donc supposer qu'à partir d'une certaine valeur de $k$ les courbes ne s'intersecteront pas. Dans le cas où l'augmentation du paramètre ne permettrait pas d'enlever l'intersection, nous proposons de mettre une courbe de Bézier cubique par segment des deux polylines que nous souhaitons approximer. Comme ces deux polylines ne s'intersectent pas, les courbes de Bézier cubiques ne s'intersecteront pas non plus. Par contre, l'approximation sera seulement $C^0$ dans le plan $(x, y)$ pour ces deux courbes.

L'image 5.6 montre le résultat de ce lissage sur l'exemple du complexe de Morse-Smale de la figure 5.3. Le paramètre $k$ a été fixé à 7. Les 1-cellules sont maintenant lisses et ne s'intersectent pas. Nous avons donc, à la fin de cette partie de l'algorithme, un ensemble de courbes de Bézier qui approximent les 1-cellules du complexe de Morse-Smale dans le plan $(x, y)$. Dans la partie suivante, nous allons faire une approximation monotone des valeurs de hauteur le long des 1-cellules.

### 5.3.3 Lissage monotone du champs scalaire le long des 1-cellules

Les 1-cellules ont été lissées dans le plan $(x, y)$, nous voulons maintenant lisser les valeurs de hauteur le long de celles-ci sous contraintes de monotonie. Le but est de calculer une approximation polynomiale par morceaux des parties de 1-cellules se trouvant entre deux points critiques ou de jonction. Nous cherchons, de plus, à ce que l'approximation soit monotone. En effet, comme nous voulons construire des valeurs de hauteur le long des 1-cellules, il faut qu'en sortie, nous n'ayons pas de points critiques le long des courbes. Dans le cas présent, les points

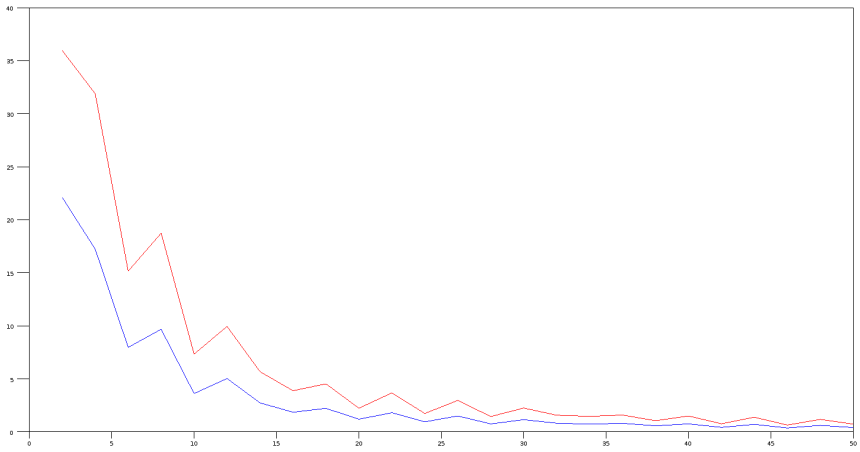



**FIGURE 5.5 –** *Erreur de l'approximation aux moindres carrés en fonction du paramètre $k$ appliquée sur une 1-cellule d'un complexe de Morse-Smale. La courbe en bleue correspond à l'erreur sur l'axe des $x$ et en rouge sur l'axe des $y$.*

critiques sont des points critiques de fonctions définies sur $\mathbb{R}$, il n'y a donc que des minima et des maxima. Or, une fonction $f : D \subset \mathbb{R} \to \mathbb{R}$ continue sur $D$ un ouvert de $\mathbb{R}$ est monotone si et seulement si sa dérivée ne change pas de signe sur $D$. Il nous faut donc trouver une fonction dont la dérivée ne change pas de signe sur l'intérieur de son domaine de définition.

Pour cela, nous avons choisi d'utiliser la méthode de He et Shi [30] qui propose un lissage monotone de données discrètes en utilisant des B-splines. Cette méthode utilise la programmation linéaire pour pouvoir calculer une B-spline monotone qui approxime des données. Nous allons avoir besoin dans la suite d'une représentation des 1-cellules sous la forme de courbes de Bézier cubiques. La représentation des courbes en B-splines étant compatible avec la représentation sous la forme Bézier, cette méthode nous permet de calculer de façon efficace un lissage des valeurs de hauteur des 1-cellules tout en nous donnant un type de courbe compatible avec les courbes de Bézier cubiques.

Soit $(\mathbf{x}_i)_{0 \leqslant i \leqslant n}$ les $n+1$ points le long du morceaux de 1-cellules. Les points $\mathbf{x}_0$ et $\mathbf{x}_n$ sont des points critiques ou des points de jonction. De plus, les points sont ordonnés dans le sens de parcours de la courbe. Les valeurs de la fonction initiale $f$ aux points $\mathbf{x}_0$ et $\mathbf{x}_n$ sont connues. En effet, soit ce sont des points critiques et leurs valeurs sont imposées par $f$, soit ce sont des points de jonction et leurs valeurs ont été calculées dans la partie précédente 5.3.1. Soit $\tilde{f}$ la fonction qui vaut la valeur de la fonction initiale $f$ en tout point sauf aux points de jonctions où elle vaut la valeur trouvée dans la partie précédente. Supposons que l'on ait $\tilde{f}(\mathbf{x}_0) < \tilde{f}(\mathbf{x}_n)$. Sinon, il suffit d'inverser la numérotation des points, i.e. utiliser l'ensemble $(\mathbf{x}'_i)_{0 \leqslant i \leqslant n}$ tels que pour tout $i = 0, \ldots, n$, $\mathbf{x}'_i = \mathbf{x}_{n-i}$. Posons $(t_i)_{0 \leqslant i \leqslant n}$ les paramètres associés aux $(\mathbf{x}_i)$ correspondant à la longueur d'arc normalisée, avec $t_0 = 0 < t_1 < \cdots < t_{n-1} < t_n = 1$.

Soit $g_\alpha : [0,1] \to \mathbb{R}$ :

$$g_\alpha(t) = \sum_{i=0}^{k+1} \alpha_i N_i^2(t)$$

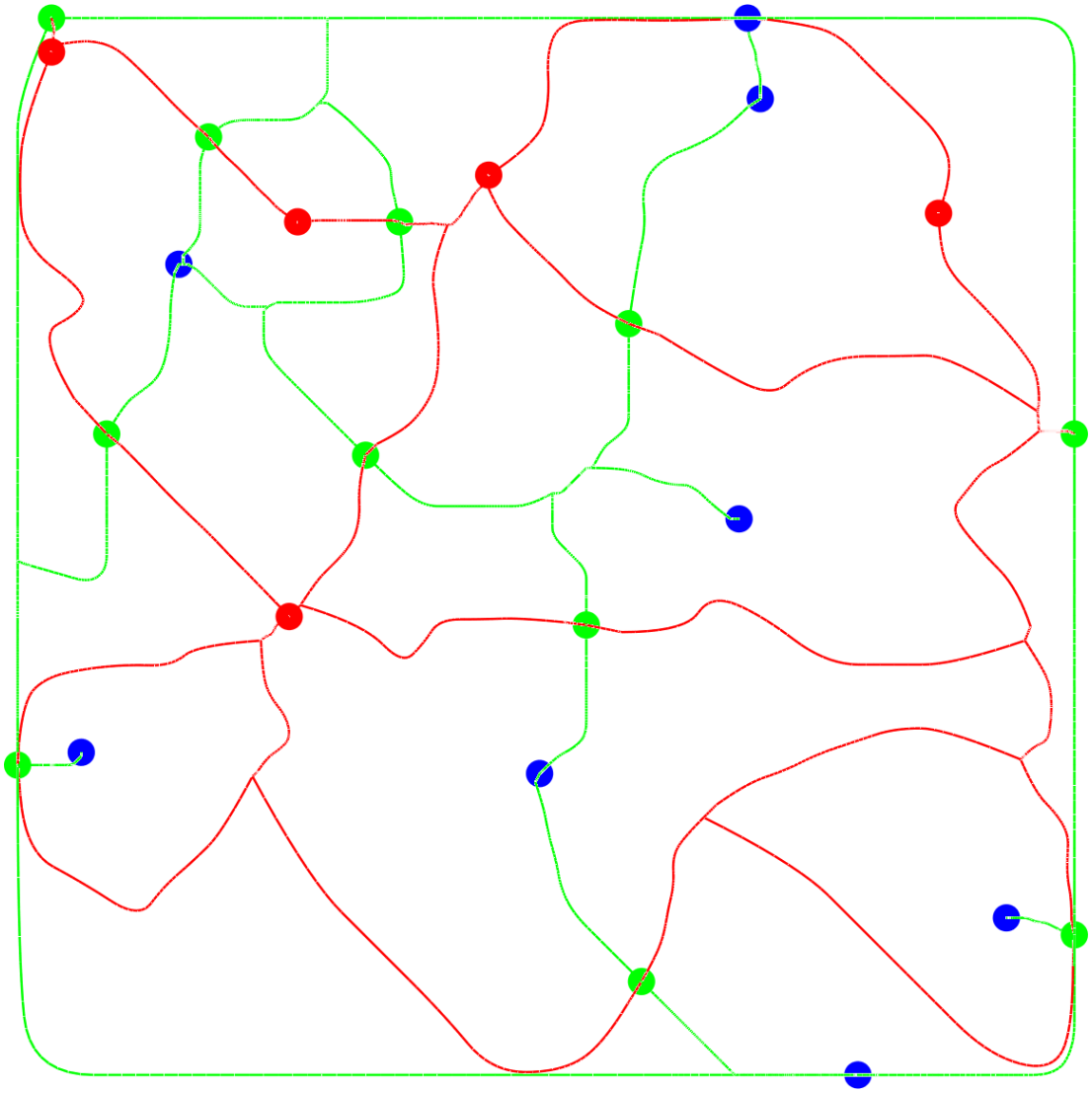


**FIGURE 5.6 –** *Les 1-cellules du complexe de Morse-Smale de l'image 5.3 ont été lissées dans le plan* $(x, y)$.

une fonction B-spline uniforme quadratique sur les noeuds $0 = u_{-2} = u_{-1} = u_0 < \ldots < u_k = u_{k+1} = u_{k+2} = 1$. Soit $N(t) = (N_0^2(t) \cdots N_{k+1}^2(t))^t$, $N'(t) = (\frac{d}{dt}N_0^2(t) \cdots \frac{d}{dt}N_{k+1}^2(t))^t$ et $\alpha = (\alpha_0 \cdots \alpha_{k+1})^t$, on a alors $g_\alpha(t) = N(t)^t\alpha$ et $\frac{d}{dt}g_\alpha(t) = N'(t)^t\alpha$. Nous allons calculer les valeurs $(\alpha_i)_{i=0,\ldots,k+1}$ de sorte que $g_\alpha$ soit croissante, qu'elle approxime les valeurs $(f(\mathbf{x}_i))_{1 \leqslant i \leqslant n-1}$ et qu'elle interpole les valeurs de $f(x_0)$ et de $f(x_n)$.

Les contraintes sur la fonction B-spline que nous souhaitons obtenir sont :

- **Interpolation du premier et du dernier point de la courbe :** cette contrainte nous permet de fixer deux valeurs : $\alpha_0 = \tilde{f}(\mathbf{x}_0)$ et $\alpha_{k+1} = \tilde{f}(\mathbf{x}_n)$.
- **Valeurs de dérivées fixées en début et fin de courbe :** Nous imposons que les valeurs de dérivées soient nulles en début et fin de courbe. Nous posons donc $x'_0 = 0$ et $x'_n = 0$.
- **Positivité de la dérivée :** pour que la fonction soit croissante montone, il suffit que pour tout $j = 1, \ldots, k-1$ on ait $N'(u_j)^t\alpha \geqslant 0$.

Le problème de minimisation est alors le suivant :

$$\sum_{i=1}^{n-1} |\tilde{f}(\mathbf{x_i}) - N(t_i)^t\alpha| \longrightarrow min \tag{5.5}$$

sous les contraintes :

$$\begin{aligned} N'(u_0)^t\alpha &= x'_0 \\ N'(u_k)^t\alpha &= x'_n \\ N'(u_j)^t\alpha &\geqslant 0,\ j = 1, \ldots, k-1 \end{aligned}$$

Ce problème n'étant pas linéaire du fait de la valeur absolue dans la fonction-objectif, nous utilisons le résultat de l'annexe A pour le linéariser. Nous introduisons donc 2 variables supplémentaires $r_i^+$ et $r_i^-$, et on obtient :

$$\sum_{i=1}^{n-1}(r_i^+ + r_i^-) \longrightarrow min \tag{5.6}$$

sous les contraintes :

$$r_j^+ \geqslant 0 \tag{5.7}$$

$$r_j^- \geqslant 0 \tag{5.8}$$

$$r_j^+ - r_j^- = \tilde{f}(\mathbf{x_i}) - N(t_i)^t\alpha \tag{5.9}$$

$$N'(u_0)^t\alpha = x'_0 \tag{5.10}$$

$$N'(u_k)^t\alpha = x'_n \tag{5.11}$$

$$N'(u_j)^t\alpha \geqslant 0,\ j = 1, \ldots, k-1. \tag{5.12}$$

Les conditions (5.7) à (5.9) permettent d'encoder la valeur absolue de la fonction coût qui se trouvait dans le problème d'optimisation (5.5) (voir l'annexe (A)). Les conditions (5.10) et (5.11) permettent de fixer les dérivées à 0 aux extrémités de la courbe. La condition (5.12) force la dérivée à être positive le long de la courbe, c'est-à-dire à avoir une courbe monotone croissante. Pour résoudre cette optimisation linéaire, nous utilisons la même méthode de simplexe de la bibliothèque GLPK [2] que précédemment pour résoudre le problème linéaire de la section 5.3.1.

L'image 5.7 montre le résultat de ce lissage sur un ensemble de données tiré d'une 1-cellule du complexe de Morse-Smale de l'image 5.3. Nous pouvons voir que les résultats sont

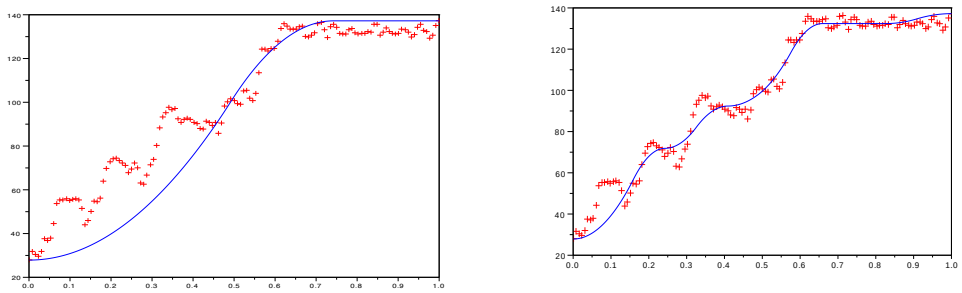


**FIGURE 5.7 –** *Approximation des valeurs de hauteur (points rouges) par une fonction B-spline quadratique (courbe bleue). À gauche avec $k = 4$ et à droite avec $k = 12$.*

bien lisses et monotones. De plus, plus le paramètre $k$ est élevé, plus la courbe est proche des valeurs d'entrées.

Nous nous retrouvons donc avec les valeurs de la fonction restreinte au domaine défini par l'union des 0-cellules et des 1-cellules définies sous la forme de courbes de Béziers cubiques. La fonction a donc été lissée le long des 1-cellules. De plus, la résolution du problème linéaire (5.6) nous assure que les courbes sont monotones le long des 1-cellules.

## 5.4 INTERPOLATION MONOTONE À L'INTÉRIEUR DES 2-CELLULES

Dans la section précédente, nous avons construit un réseau de courbes reliant les différents points critiques et points de jonction entre eux. Entre deux points, ces courbes sont composées de plusieurs morceaux de courbes de Bézier cubiques. Nous allons considérer que la fonction reconstruite restreinte aux 1-cellules correspond aux valeurs calculées dans la section précédente. Nous allons construire une fonction pour chaque 2-cellules qui sera définie sur celle-ci. Pour une 2-cellule, cette fonction devra respecter les conditions suivantes :

- interpolation des 1-cellules sur les bords pour avoir une fonction continue ;
- monotonie, c'est-à-dire qu'elle ne contient pas de point critique à l'intérieur de la 2-cellule car les points critiques d'un complexe de Morse-Smale sont les 0-cellules ;
- gradient nul aux points critiques car les 0-cellules du complexe de Morse-Smale sont les points critiques de la fonction.

Une 2-cellule est bornée par quatre 1-cellules que nous appelerons aussi courbes de bords. Ces courbes peuvent contenir des points de jonctions et sont donc suceptibles d'être composées de plusieurs morceaux de courbes. Dans la suite, nous utiliserons la définition d'une 1-cellule comme une succesion de courbe Bézier cubiques et nous ne prendrons plus en compte les points de jonction.

Pour faire cela, nous allons utiliser les résultats du chapitre 4 qui nous permettent de construire une fonction monotone à partir de valeurs données sur une grille. Les domaines des 2-cellules étant quelconques, mais avec quatre courbes de bords, nous allons dans un premier temps les reparamétriser sur le carré unité en section 5.4.1. Ensuite, nous montrerons comment nous construisons la grille pour avoir une fonction monotone vérifiant les conditions énoncées dans ce paragraphe (section 5.4.2). Puis, en section 5.5, nous verrons les deux cas particuliers de 2-cellules et leur traitement.

### 5.4.1 Reparamétrisation

Pour une 2-cellule du complexe de Morse-Smale simplifié, nous souhaitons construire une surface monotone interpolant les courbes de bords. Ces courbes de bords sont au nombre de quatre et sont délimitées par quatre points critiques : un minimum, deux points col et un maximum. Ces quatres courbes ne s'intersectent pas dans le plan $(x, y)$. Elles correspondent à la frontière de la 2-cellule. La forme de la 2-cellule est quelconque. Par exemple, celle-ci peut être convexe ou non. La construction d'une fonction interpolant un réseau de courbes définissant un domaine quelconque n'étant pas aisée, nous reparamétrisons ce domaine sur le carré unité, i.e. $[0, 1]^2$.Il existe deux types de 2-cellules ne possédant pas quatre courbes de bords distinctes. Ces exceptions seront traitées en section 5.5.

Soit $\Omega$ une 2-cellule du complexe de Morse-Smale simplifié. La fermeture de la 2-cellule, notée $\bar{\Omega}$, contient la 2-cellule, ainsi que les 0-cellules et les 1-cellules la bordant. Nous allons alors construire une paramétrisation de $\bar{\Omega}$ sur le carré unité $[0, 1]^2$, c'est-à-dire un difféomorphisme $\phi : \bar{\Omega} \to [0, 1]^2$. Si l'on définit une fonction $F : [0, 1]^2 \to \mathbb{R}$ telle que sa restriction à $]0, 1[^2$ ne contienne pas de point critique, alors la fonction $\tilde{f} = F \circ \phi : \bar{\Omega} \to \mathbb{R}$ ne contiendra pas de points critiques sur sa restriction à la 2-cellule. Cela signifie que si une fonction monotone est définie sur le carré unité, alors grâce à la paramétrisation, une fonction monotone peut être définie sur la 2-cellule.

Il existe plusieurs méthodes permettant de construire une reparamétrisation d'un domaine sur un autre. On peut classer ces méthodes en deux catégories. La première catégorie est celle des méthodes discrètes et la seconde celle des méthodes continues. Dans les méthodes discrètes, nous pouvons citer comme exemple l'algorithme de Tutte [48] de « mapping » barycentrique. À partir d'une triangulation du domaine à paramétrer, l'algorithme construit une image dans l'espace des paramètres pour chaque sommet de la triangulation. Chaque point de l'espace des paramètres est le barycentre de ces voisins. Cette méthode ne minimise ni la distorsion des angles, ni la distorsion des aires. Une autre méthode est l'algorithme « Mean Value Coordinates » de Floater [17] qui est une approximation d'une reparamétrisation discrète conforme, c'est-à-dire d'une paramétrisation conservant les angles. Cette méthode garantie une bijection entre le domaine initial et le domaine des paramètres lorsque le bord du domaine dans l'espace des paramètres est convexe. D'autres méthodes discrètes sont montrées dans l'état de l'art de Floater et Hormann [18]. Quant aux méthodes continues, elles ne garantissent pas d'avoir une bijection pour tout domaine à reparamétriser (voir par exemple [37]). Nous avons choisi d'utiliser la méthode de Floater [17] « Mean Value Coordinates ».

Nous allons maintenant montrer les quatre étapes nous permettant de construire une telle fonction $\phi$.

#### *Étape 1 : Triangulation* $\bar{\Omega}$

Dans un premier temps, nous allons construire une triangulation de $\bar{\Omega}$. Nous commençons par évaluer les courbes de Bézier correspondantes aux 1-cellules, ce qui équivaut à faire des évaluations de $\partial\bar{\Omega}$. Nous évaluons les courbes uniformément un nombre fixe de fois donné en paramètre. Dans nos tests, nous évaluons les courbes en $c = 5$ points. Ensuite, nous calculons une triangulation de Delaunay dans $\bar{\Omega}$ contrainte par les segments correspondant aux courbes de Bézier calculées précédemment. Pour calculer cette triangulation, nous utilisons la bibliothèque CGAL [1].

***Étape 2 : Mapping uniforme de $\partial\bar{\Omega}$ sur le carré unité***

Nous allons calculer une triangulation correspondante à celle calculée précédemment mais celle-ci sera incluse dans le carré unité. Pour cela, il nous faut associer à chaque point de la frontière une valeur de paramètre $(u, v) \in \partial[0, 1]^2$. Pour commencer, nous allons faire correspondre le point $(0, 0)$ avec le minimum, le point $(1, 1)$ avec le maximum et les deux coins restants avec les deux points cols. Ensuite, nous plaçons les points correspondants aux sommets de la triangulation appartenant à la frontière du domaine uniformément sur la frontière du carré unité.

***Étape 3 : Construction d'une triangulation équivalente sur $[0, 1]^2$***

L'algorithme Mean Value Coordinates de Floater[17] est utilisé pour positionner tous les sommets des triangles. Cet algorithme permet d'exprimer un point d'une triangulation comme une combinaison convexe de ses voisins dans la triangulation. Soit $v_0$ le point d'une triangulation $T$ et $(v_i)_{i=1,\dots,k}$ les $k$ sommets de $\mathscr{N}_1(v_0)$, le 1-voisinage de $v_0$ dans $T$. Cet algorithme définit les poids $\lambda_1, \dots, \lambda_k \geqslant 0$ tels que $v_0$ soit le barycentre pondéré des points $(v_i)_{i=1,\dots,k}$ :

$$\sum_{i=1}^{k} \lambda_i v_i = v_0,$$
$$\sum_{i=1}^{k} \lambda_i = 1.$$

Pour les cas où $k > 3$, les poids ne sont pas définis de façon unique. Posons $\alpha_i$ l'angle $\measuredangle v_{i-1} v_0 v_i$ (avec comme cas particulier $\alpha_1$ l'angle $\measuredangle v_k v_0 v_1$ et $\alpha_k = \measuredangle v_1 v_0 v_k$). Le choix des poids est alors :

$$\lambda_i = \frac{w_i}{\sum_{j=1}^{k} w_j},$$
$$w_i = \frac{\tan(\frac{\alpha_{i-1}}{2}) + \tan(\frac{\alpha_i}{2})}{\|v_i - v_0\|}.$$

Ce choix particulier de poids permet de construire une triangulation équivalente à la triangulation de la 2-cellule dans le carré unité en donnant seulement le bord de la triangulation. En résolvant un système linéaire, tout les points sont définis de façon unique dans le domaine borné par le bord donné en entrée car le carré unité est convexe[17]. Le résultat de ce système va mettre en correspondance les points de la triangulation $T$ avec des points du carré unité. Soit $v_k$ un point de la triangulation $T$, il existe alors un unique point de $[0, 1]^2$ correspondant noté $\phi(v_k)$. La procédure pour construire la triangulation équivalente dans le carré unité consiste à construire une arête entre les points $\phi(v_i)$ et $\phi(v_j)$ pour toute arête $(v_i, v_j)$ de la triangulation $T$.

Pour pouvoir appliquer l'algorithme Mean Value Coordinates, il faut vérifier que les poids $\omega_i$ sont bien définis. Tout d'abord, le numérateur des poids $\omega_i$ est composé d'une somme de fonctions tangentes. Pour que celles-ci soient correctement définies, il faut que les angles $(\alpha_i)_{i=1,\dots,k}$ soient tels que pour tout $i = 1, \dots, k$ on ait $0 < \alpha_i < \pi$. Cela signifie qu'il ne doit pas exister de triangles plats dans la triangulation. Or nous construisons une triangulation de Delaunay contrainte par les segments de bords. Cette méthode nous permet d'avoir une triangulation qui ne contient pas de triangles plats, et donc, le numérateur de $\omega_i$ est bien défini.

Concernant le dénominateur de $\omega_i$, celui-ci sera nul si et seulement s'il existe $v_i$ un point de la triangulation identique à $v_0$, i.e. $v_i = v_0$. La construction de la triangulation par la méthode de Delaunay contrainte nous garantie néanmoins que tous les points sont distincts. De plus, les points sur le bord ont été obtenus par des évaluations uniformes le long des courbes de Bézier ne s'intersectant pas. De tels points identiques n'existent donc pas non plus sur le bord. Dans notre cas, cet algorithme peut donc être appliqué et a pour résultat une triangulation conforme.

***Étape 4 : Changement de paramètre***

Définissons maintenant le difféomorphisme $\phi : \bar{\Omega} \to [0,1]^2$ à l'aide des triangulations $T$ de $\bar{\Omega}$ et $\hat{T}$ de $[0,1]^2$. Chaque triangle $\hat{\triangle} \in \hat{T}$ est l'image affine de son triangle correspondant $\triangle \in T$. Comme les combinaisons affines sont invariantes par transformation affine, on peut définir $\phi$ sur le triangle $\triangle = (t_1, t_2, t_3)$ par

$$\phi(P) = \tau_1\phi(t_1) + \tau_2\phi(t_2) + \tau_3\phi(t_3),$$

où $P = \tau_1 t_1 + \tau_2 t_2 + \tau_3 t_3$ est un point de $\triangle$ et $\tau_1,\ tau_2$ et $\tau_3$ ses coordonnées barycentriques par rapport à $\triangle$. $\phi$ est alors la carte affine par morceaux de $T$ telle que

$$\forall \triangle \in T,\ \phi(\triangle) = \hat{\triangle}.$$

Celle-ci est illustrée à la figure 5.8.

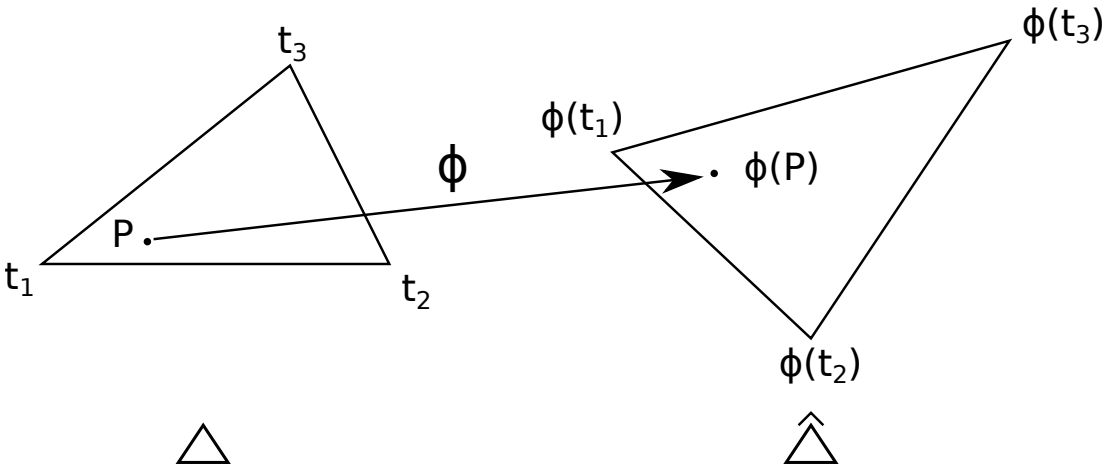
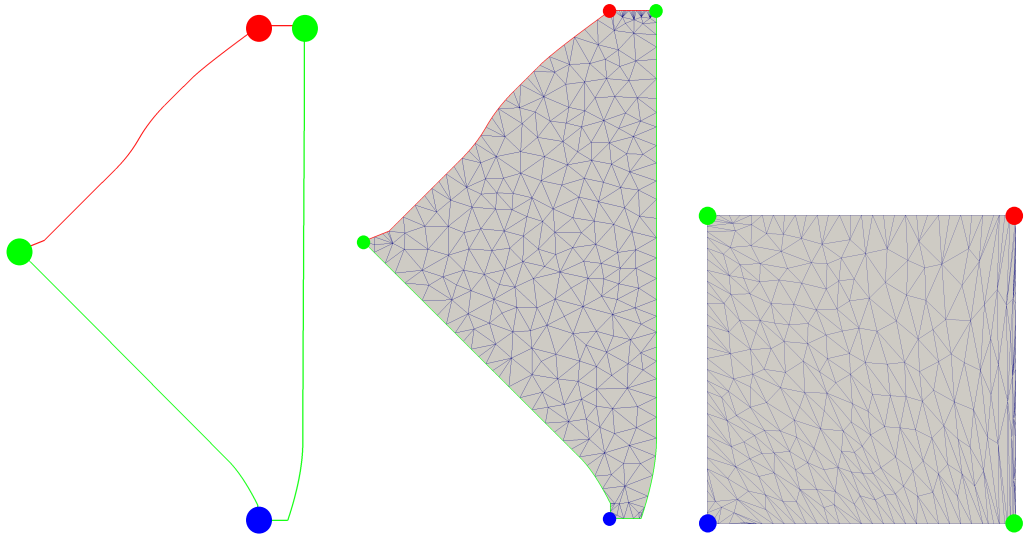



**FIGURE 5.8 –** *Changement de paramètre pour un triangle* $\triangle$

***Résultats***

La figure 5.9 montre un exemple de deux triangulations $T$ et $\hat{T}$ définissant le difféomorphisme $\phi$. En partant de la 2-cellule de la sous-figure 5.9a, nous calculons une triangulation de Delaunay contrainte $T$ de la 2-cellule (sous-figure 5.9b), puis grâce à l'algorithme Mean Value Coordinates de Floater, nous calculons une triangulation $\hat{T}$ dans le carré unité correspondant à la triangulation de la 2-cellule (sous-figure 5.9c).



**(a)** *Exemple d'une 2-cellule.* **(b)** *triangulation de la 2-cellule.* **(c)** *Paramétrisation de la 2-cellule dans le carré unité.*

**FIGURE 5.9 –** *Illustration des étapes de la paramétrisation d'une 2-cellule.*

### 5.4.2 Champs scalaire monotone à l'intérieur du carré unité et à l'intérieur de la 2-cellule

À la section 5.3, nous avons lissé les 1-cellules et nous avons lissé de façon monotone la fonction reconstruite restreinte au domaine composé de l'union des 1-cellules. Nous utilisons alors la paramétrisation calculée à la section précédente (5.4.1) entre la 2-cellule et le carré unité pour avoir des valeurs de fonction sur le bord du carré unité. Dans ce qui suit, nous allons montrer comment nous construisons une fonction monotone à l'intérieur du carré unité qui interpole les valeurs de fonction déjà calculées au bord. Puis, nous utiliserons la paramétrisation pour construire la fonction sur la 2-cellule.

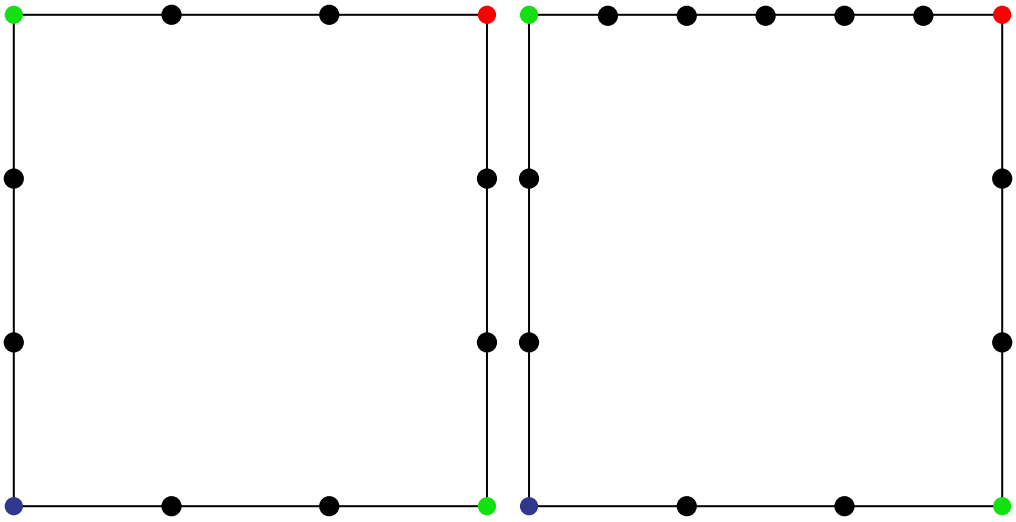
En reprenant les notations de la section 5.4.1, nous cherchons à calculer une fonction $F : [0,1]^2 \to \mathbb{R}$ qui ne contient pas de points critiques sur $]0,1[^2$. À la section 5.3, nous avons construit la fonction $f|_{\partial\bar{\Omega}} : \partial\bar{\Omega} \to \mathbb{R}$ qui est composée de courbes de Bézier cubiques. En utilisant la changement de paramètres $\phi : \bar{\Omega} \to [0,1]^2$ introduit à la section 5.4.1, nous pouvons définir la fonction $F$ sur la frontière de $[0,1]^2$ par

$$F|_{\partial[0,1]^2} = \phi(f|_{\partial\bar{\Omega}}).$$

La construction de la fonction va utiliser les résultats du chapitre 4 où, à partir de valeurs de fonction données sur une grille régulière qui sont croissantes le long des diagonales, nous construisons une fonction monotone interpolant ces valeurs. Il faut donc tout d'abord construire une grille de valeurs sur le carré unité où les valeurs le long des diagonales sont croissantes.

#### *Conditions de bords*

La paramétrisation faîte dans la section précédente (5.4.1) est construite pour répartir de façon uniforme les points séparant les différentes courbes de Bézier le long des arêtes du carré unité. Deux cas vont alors se présenter : le cas où il y a le même nombre de courbes de Bézier sur chaque arête du carré unité, celui-ci est illustré à la figure 5.10a. Le deuxième étant quand,



**(a)** *Cas où le nombre de courbes de Bézier est le même sur toutes les arêtes.*

**(b)** *Cas où l'arête du haut est composée de plus de courbes de Bézier que les autres arêtes.*

**FIGURE 5.10 –** *Découpage des arêtes du carré unité selon les courbes de Bézier. Deux cas sont présentés : (5.10a) Le nombre de courbes de Bézier est le même sur toutes les arêtes. (5.10b) Au moins une des arêtes est composée de plus de courbes de Bézier que les autres.*

au moins l'une des arêtes est composée d'un nombre différent de courbes de Bézier que les autres arêtes. Ce cas apparaît quand les 1-cellules sont découpées par un nombre différent de points de jonction. Un exemple de ce cas est montré à la figure 5.10b.

Pour construire une grille régulière, nous allons avoir besoin qu'il y ait le même nombre de courbes de Bézier sur toutes les arêtes du carré unité. Dans le premier cas, il n'y a donc rien à faire. Dans le deuxième cas, nous allons effectuer des subdivisions des courbes de Bézier jusqu'à ce que toutes les arêtes soient composées du même nombre de courbe de Bézier. La subdivision des courbes de Bézier est une technique permettant de découper une courbe de Bézier en deux courbes de Bézier qui décrivent exactement la courbe initiale. Soit une courbe de Bézier $\mathbf{P}(t) = \sum_{i=0}^{n} \mathbf{b}_i B_i^n(t)$ avec $t \in [0,1]$. L'algorithme de de Casteljau permet alors d'évaluer la valeur $\mathbf{P}(t)$ grâce à un algorithme récursif :

$$\begin{cases} \mathbf{b}_i^0(t) = \mathbf{b}_i & \text{pour } i = 0, \dots, n \\ \mathbf{b}_i^r(t) = (1-t)\mathbf{b}_i^{r-1}(t) + t\mathbf{b}_{i+1}^{r-1}(t) & \text{pour } \begin{cases} r = 1, \dots, n \\ i = 0, \dots, n-r \end{cases} \end{cases}$$

Nous pouvons alors utiliser ces points pour créer deux polygones de contrôles qui à eux deux définissent exactement la même courbe que la courbe de Bézier initiale. Les deux courbes sont alors $\mathbf{P}_1(t) = \sum_{i=0}^{n} \mathbf{b}_0^i(\alpha) B_i^n(t)$ et $\mathbf{P}_2(t) = \sum_{i=0}^{n} \mathbf{b}_i^{n-i}(\alpha) B_i^n(t)$ avec $\alpha \in ]0,1[$ donné. En pratique, nous utilisons $\alpha = \frac{1}{2}$ comme paramètre de subdivision. L'image 5.11 montre l'exemple d'une subdivision d'une courbe de Bézier de degré 2 en deux courbes de Bézier de degré 2.

#### *Construction de la grille régulière sur* $[0,1]^2$

Après application des subdivisions pour avoir le même nombre de courbes de Bézier cubiques pour chaque arête du carré unité, nous pouvons construire une grille régulière sur le carré

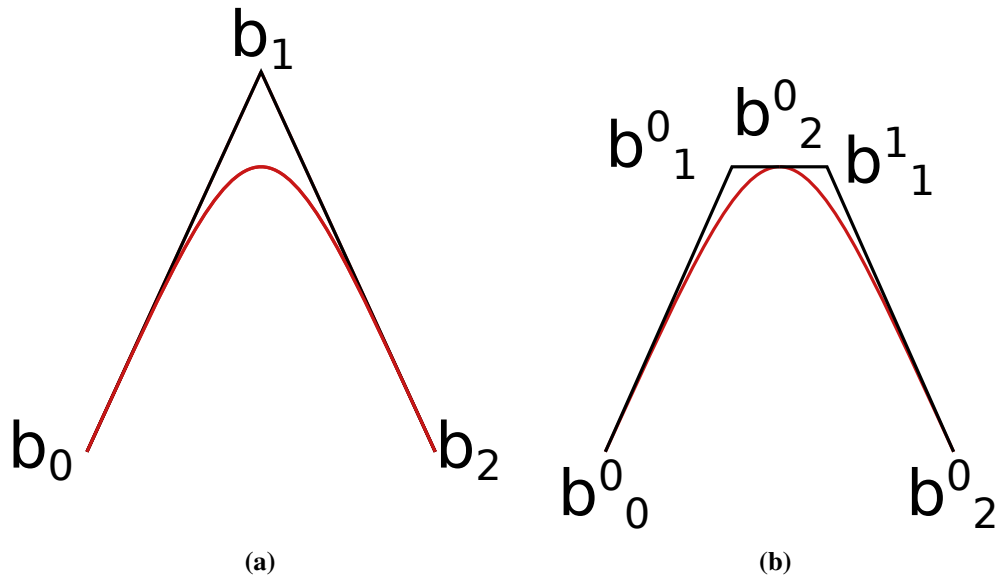



**FIGURE 5.11 –** *Exemple de la subdivision d'une courbe de Bézier de degré 2 avec un paramètre $\alpha = \frac{1}{2}$. La courbe de Bézier est tracée en rouge et le polygone de contrôle est en noir. (a) Courbe de Bézier avant subdivision. (b) Courbe subdivisée en deux courbes de Bézier de degré 2. On remarque que la courbe est bien la même dans les deux cas.*

unité. Pour cela, nous prenons toutes les droites du types $u = c_1$ et $v = c_2$, avec $c_1, c_2 \in ]0, 1[$, telles qu'elles relient les points séparant les courbes de Bézier sur le bord du carré unité entre elles. Deux exemples sont montrés à la figure 5.12.

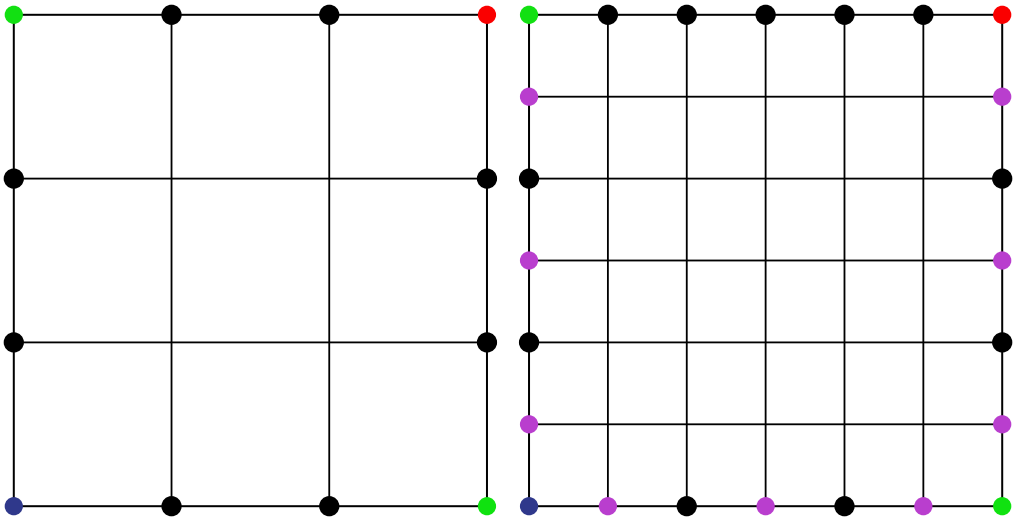
***Calcul de valeurs croissantes le long des diagonales de la grille régulière***

Nous allons utiliser les résultats du chapitre 4 pour construire une fonction monotone $F$ sur le carré unité. En effet, nous avons développé une méthode d'interpolation monotone à partir des valeurs discrètes le long des diagonales de la grille. Vérifions d'abord que les conditions de bords nous permettent de construire une telle grille : comme dans notre paramétrisation nous avons placé le minimum dans le coin inférieur gauche $(0, 0)$, le maximum dans le coin supérieur droit $(1, 1)$ et les deux points cols dans les deux coins restants $(0, 1)$ et $(1, 0)$ et comme les courbes de bords sont croissantes entre deux points critiques, nous pouvons conclure que nous avons quatre conditions sur les bords correspondant à la croissance des courbes les long des bords du carré unité. Considérons que les arêtes du carré sont constituées de $n + 1$ points. Soit $(p_{ij})_{i,j=0,\dots,n}$ les points de la grille et $z_{ij} = F(p_{ij})$, ces quatre conditions s'expriment ainsi :

$$\begin{cases} z_{0,j_1} \leqslant z_{0,j_2} & \text{pour } 0 \leqslant j_1 \leqslant j_2 \leqslant n \\ z_{n,j_1} \leqslant z_{n,j_2} & \text{pour } 0 \leqslant j_1 \leqslant j_2 \leqslant n \\ z_{i_1,0} \leqslant z_{i_2,0} & \text{pour } 0 \leqslant i_1 \leqslant i_2 \leqslant n \\ z_{i_1,n} \leqslant z_{i_2,n} & \text{pour } 0 \leqslant i_1 \leqslant i_2 \leqslant n \end{cases}$$

Nous en déduisons que (voir les flèches rouges à la figure 5.13a) :

$$\begin{cases} z_{0,j} \leqslant z_{0,n} \leqslant z_{i,n} & \text{pour } 0 \leqslant i, j \leqslant n \\ z_{i,0} \leqslant z_{n,0} \leqslant z_{n,j} & \text{pour } 0 \leqslant i, j \leqslant n \end{cases}$$



**(a)** *Construction de la grille dans l'exemple de la figure 5.10a.*

**(b)** *Construction de la grille dans l'exemple de la figure 5.10b. Les points violets sont les points obtenus par subdivision des courbes de Bézier afin d'avoir un nombre de courbes de Bézier composant les arêtes étant le même pour toutes les arêtes.*

**FIGURE 5.12 –** *Construction de la grille à partir des exemples de la figure 5.10. Dans l'exemple (b) des subdivisions ont été effectuées avant la construction de la grille.*

Et en particulier (cf les diagonales bleues à la figure 5.13a) :

$$\begin{cases} z_{0,j} \leqslant z_{n-j,n} & \text{pour } 0 \leqslant j \leqslant n \\ z_{i,0} \leqslant z_{n,n-i} & \text{pour } 0 \leqslant i \leqslant n \end{cases}$$

Nous venons de montrer que les conditions de bords, i.e. les valeurs de fonction $f$ et $F$ au bord du domaine $\bar{\Omega}$ et $[0,1]^2$ satisfont la condition de monotonie le long des diagonales (définition 4.3, section 4.3.2). Il est donc possible de construire des diagonales croissantes à partir de nos données. Les conditions de bords sont illustrées à la figure 5.13a. La figure 5.13b montre les conditions que l'on ajoute sur les points internes avec la condition de montonie le long des diagonales. Nous venons de montrer que cette condition est déjà respectée aux extrémités des diagonales par les conditions aux bords. Il nous faut donc trouver des valeurs croissantes le long des diagonales à l'intérieur du domaine $[0,1]^2$, ce qui nous permettra d'utiliser les résultats du chapitre 4.

Plusieurs méthodes sont possibles afin de construire des valeurs croissantes le long des diagonales sur les points internes de la grille. Ces méthodes peuvent être classées en deux catégories distinctes. La première catégorie est composée des méthodes calculant les valeurs le long de chaque diagonales de la grille. Dans ces méthodes, nous pouvons citer l'interpolation linéaire des valeurs le long de chaque diagonales. La méthode que nous avons utilisée à la section 5.3.3 peut aussi être appliquée le long de chaque diagonale en évaluant la fonction initiale aux points de la grille grâce à la paramétrisation calculée précédemment. Le problème commun à ces méthodes est de calculer des valeurs de façon totalement indépendante des points du voisinage n'étant pas sur la diagonale. La deuxième catégorie consiste à calculer des valeurs globalement sur toutes la grille. Comme nous avons un unique minimum et un unique

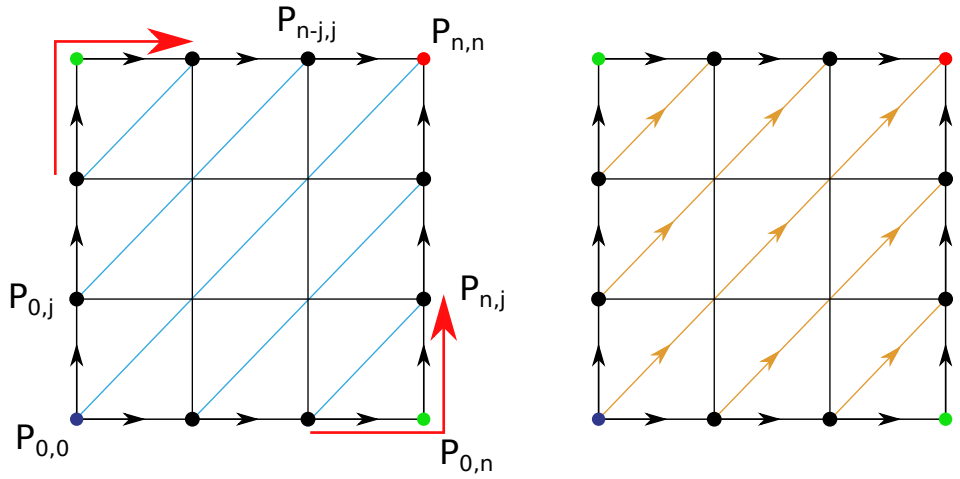



**FIGURE 5.13 –** *Condition de monotonie sur le carré unité. La reladtion $x \to y$ équivaut à $x \leqslant y$. (a) Les conditions de monotonie aux bords du carré unité. (b) Conditions aux bords du carré plus les conditions du chapitre 4. On remarque que les conditions sont consistantes entre elles.*

maximum, nous avons choisi d'exploiter les propriétés des fonctions harmoniques comme le principe du maximum. Nous résolvons donc le système linéaire correspondant à l'équation de Laplace $\Delta f = 0$ aux points internes de la grille, avec comme conditions de bords, les valeurs de la grille sur le bord du carré unité.

Le résultat de l'équation de Laplace nous donne une fonction qui est monotone, au sens où il n'y a pas de point critique discret créé. Mais celui-ci n'est pas assuré d'être croissant le long des diagonales de la grille. Nous vérifions donc si les diagonales sont bien croissantes. Dans la plupart des cas testés, la monotonie était vérifiée. Si ce n'est pas le cas, alors il nous suffit de faire l'interpolation linéaire des valeurs le long de la diagonale qui sera assurée de nous donner des valeurs croissantes. Cette équation est appliqué sur des petits problèmes : dans le cas où il n'y a pas eu de subdivisions, si l'on prend k morceaux de courbes par 1-cellule, il nous faudra alors inverser une matrice $k-2 \times k-2$ par 2-cellule. Dans nos tests, nous avons pris en général $k = 7$, d'où des sytèmes linéaires $25 \times 25$ à résoudre.

La table 5.1 montre un exemple d'application du Laplacien sur un problème où l'on recherche la valeur de la fonction pour 9 points. Les valeurs des premières et dernières lignes et colonnes sont fixées et l'équation de Laplace est appliquée sur les points internes de la grille. Cela correspond à une inversion d'une matrice $9 \times 9$. Dans cet exemple, le résultat est croissant selon les diagonales, il n'y a donc pas à appliquer une correction des valeurs dans cet exemple.

### *Estimation des dérivées partielles*

À partir de ces valeurs, nous calculons les valeurs des dérivées partielles en utilisant les résultats du chapitre 4. Nous allons utiliser l'algorithme 2 qui construit des valeurs de dérivées admissibles à partir de la grille de valeurs. Pour tout les points internes, nous utilisons les valeurs produites par cet algorithme. Les points sur le bords possèdent tous une ou deux valeurs de dérivées imposées par la courbe que nous souhaitons interpoler. Nous remplaçons donc les valeurs que nous donne l'algorithme par ces valeurs imposées et gardons les valeurs trouvées par l'algorithme pour les dérivées non imposées. Nous vérifions alors que les conditions du

| | | | | |
|---|---|---|---|---|
| 4.000 | 5.000 | 6.000 | 7.000 | 8.000 |
| 3.000 | 4.107 | 5.192 | 6.286 | 7.500 |
| 2.000 | 3.237 | 4.375 | 5.451 | 6.500 |
| 1.000 | 2.464 | 3.621 | 4.643 | 6.000 |
| 0.000 | 2.000 | 3.000 | 3.500 | 5.000 |

**TABLE 5.1 –** *Exemple de l'application de l'équation de Laplace. Les conditions de bords correspondent aux premières et dernières lignes et colonnes. Le résultat est croissant selon les diagonales. Les valeurs sont arrondies à* $10^{-3}$ *près.*

théorème 4.2 sont bien satisfaites sur les patchs au bord du carré unité. Afin que ces conditions soient respectées pour tout les patchs aux bord du carré unité, nous prenons une valeur $\lambda$ de paramètre de l'algorithme 2 faible. En pratique, nous posons $\lambda = \frac{1}{4}$. Dans la très grande majorité des cas, ces conditions sont bien respectées et ces valeurs de dérivées nous permettent de construire une surface monotone.

Dans le cas où ces conditions ne sont pas satisfaites, nous devons modifier les dérivées partielles le long des courbes de bords. Nous allons alors utiliser les valeurs trouvées par l'algorithme 2 que nous avons appliqué précédemment. Soient $(a^x_{i,j}, a^y_{i,j})_{0\leqslant i,j\leqslant n}$ les dérivées données par cet algorithme. Les dérivées partielles dans le long du bord sont $(\tilde{z}^y_{0,j})_{0\leqslant j\leqslant n}$, $(\tilde{z}^y_{i,n})_{0\leqslant i\leqslant n}$, $(\tilde{z}^y_{n,j})_{0\leqslant j\leqslant n}$ et $(\tilde{z}^y_{i,0})_{0\leqslant i\leqslant n}$. Nous posons alors pour les dérivées partielles le long du bord du carré unité :

$$
\begin{aligned}
z^y_{0,j} &= \min\{\tilde{z}^y_{0,j}, a^y_{0,j}\} \;\; 0 \leqslant j \leqslant n,\\
z^x_{i,n} &= \min\{\tilde{z}^x_{i,n}, a^x_{i,n}\} \;\; 0 \leqslant i \leqslant n,\\
z^y_{n,j} &= \min\{\tilde{z}^y_{n,j}, a^y_{n,j}\} \;\; 0 \leqslant j \leqslant n,\\
z^x_{i,0} &= \min\{\tilde{z}^x_{i,0}, a^x_{i,0}\} \;\; 0 \leqslant i \leqslant n.
\end{aligned}
$$

Comme les valeurs de dérivées partielles le long du bord et les dérivées partielles données par l'algorithme sont positives, les dérivées partielles telles que nous les posons sont alors elles aussi positives. Les dérivées partielles restantes sont les valeurs données par l'algorithme.

Montrons alors que ces dérivées partielles nous assurent de respecter les conditions du théorème 4.2. Les valeurs de dérivées partielles sont garanties de satisfaire les conditions du théorème dans les patchs ne se trouvant pas au bord du carré unité. Regardons alors un patch au bord. Soit le patch $(0, j)$ avec $0 \leqslant j \leqslant j-1$. Les conditions (4.17) à (4.20) sont satisfaites car toutes les dérivées partielles sont positives. Ensuite :

$$
\begin{aligned}
S_1 &= 5z^x_{0,j} + z^y_{0,j} + 2z^x_{1,j} + 2z^y_{1,j} + z^x_{1,j+1} + 5z^y_{1,j+1}\\
&= 5\min\{\tilde{z}^y_{0,j}, a^y_{0,j}\}^x + a^y_{0,j} + 2\min\{\tilde{z}^y_{1,j}, a^y_{1,j}\}^x + 2a^y_{1,j} + a^x_{1,j+1} + 5a^y_{1,j+1}\\
&\leqslant 5a^x_{0,j} + a^y_{0,j} + 2a^x_{1,j} + 2a^y_{1,j} + a^x_{1,j+1} + 5a^y_{1,j+1}\\
&\leqslant \lambda\frac{12}{h}(z_{1,j+1} - z_{0,j})\\
&\leqslant \frac{12}{h}(z_{1,j+1} - z_{0,j})
\end{aligned}
$$

La condition (4.21) est donc satisfaite. De même, on montre que la condition (4.22) est elle aussi respectée. La surface donnée par l'interpolant de Sibson modifié est donc monotone à l'intérieur du patch $(0, j)$.

De la même manière, la surface donnée par l'interpolant de Sibson modifié est monotone à l'intérieur des patchs au bord du carré unité. La surface à l'intérieur du carré unité est donc monotone croissante dans la direction $(x + y)$.

Comme les 1-cellules sont modifiées lorsque ce cas apparaît, nous devons faire le calcul pour les 2-cellules adjacentes avec les 1-cellules modifiées. Si certaines de ces 2-cellules ont déjà été reconstruites, il faut alors faire à nouveau le calcul de la surface

#### *Évaluation de* $\mathbf{F} : [\mathbf{0}, \mathbf{1}]^{\mathbf{2}} \to \mathbb{R}$

Les valeurs aux points de la grille et les dérivées partielles calculées précédemment permettent de définir une surface $F : [0, 1]^2 \to \mathbb{R}$ telle que :

- $F$ soit monotone sur $]0, 1[^2$, elle ne contient donc pas de point critique sur l'intérieur du carré unité ;
- $F$ interpole les points critiques placés aux quatre coins du carré ;
- $F$ a un gradient nul aux points critiques placés aux coins du carré.

Les points de contrôles des surfaces triangulaires de Bézier sont définis grâce au théorème (4.1).

La fonction est alors définie sur l'ensemble du carré unité. Si l'on souhaite évaluer la fonction $F$ sur un point de $[0, 1]^2$, il suffit de trouver le carré de la grille dans lequel il se trouve, et ensuite de déterminer dans quel triangle de Bézier ce point est positionné. Enfin, il faut évaluer les coordonnées barycentriques de ce point dans ce triangle et utiliser la formule (4.1) qui donne la valeur de la fonction $F$ en ce point.

#### *Calcul de* $\tilde{\mathbf{f}} : \bar{\mathbf{\Omega}} \to \mathbb{R}$

La dernière étape consiste à calculer la fonction sur le domaine de la 2-cellule $\bar{\Omega}$. Nous connaissons la fonction $F : [0, 1]^2 \to \mathbb{R}$ ainsi que la paramétrisation calculée à la section 5.4.1. Celle-ci est représentée par le difféomorphisme $\phi : \bar{\Omega} \to [0, 1]^2$ tel que $\tilde{f} = F \circ \phi$ où $\tilde{f}$ est la fonction à calculer. Ceci nous donne donc la définition de la fonction $\tilde{f}$ sur le domaine $\bar{\Omega}$ de la 2-cellule. Grâce aux propriétés de $\phi$, nous avons :

- $\tilde{f}$ est monotone sur $\Omega$ ;
- $\tilde{f}$ interpole les points critiques au bord de la 2-cellule ;
- $\tilde{f}$ a un gradient nul aux points critiques.

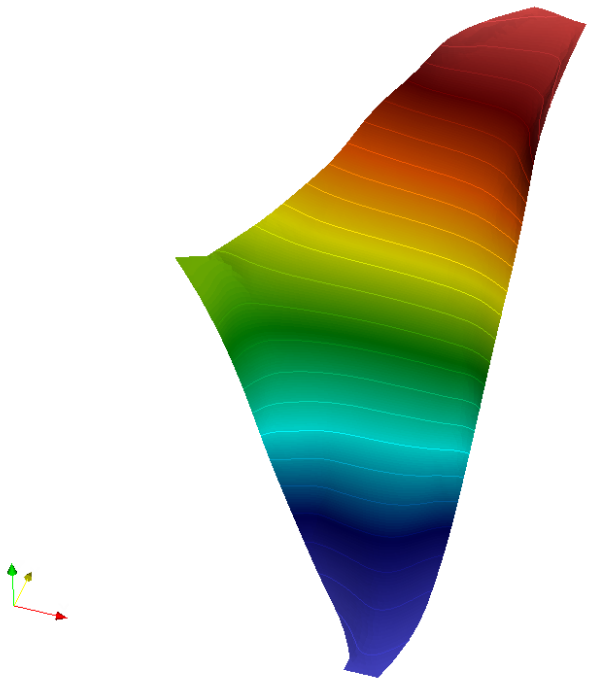
En pratique, nous représentons la surface par un maillage triangulé. Une triangulation de Delaunay contrainte est calculée à partie des courbes de bords de la 2-cellule. Ensuite, pour chaque sommet de la triangulation, ses valeurs de paramètres dans le carré unité sont évaluées, puis la fonction $F$ est calculée en ces paramètres.

#### *Résultats*

À la figure 5.14, nous pouvons voir le résultat de la reconstruction pour une 2-cellule. La figure 5.14a montre le résultat de la reconstruction sur le carré unité. La surface obtenue ne contient pas de point critique en son intérieur, en effet, il n'y a pas de ligne isovaleur fermée. La figure 5.14b montre le résultat de la reconstruction sur la 2-cellule. Cette surface est donc l'application de la paramétrisation $\phi^{-1}$ sur la surface définie sur le carré unité. Cette surface ne contient pas non plus de point critique en son intérieur.

**(a)** *Reconstruction de la hauteur de la fonction sur le carré unité.*

**(b)** *Reconstruction de la hauteur de la fonction sur la 2-cellule.*



**(c)** *Reconstruction de la hauteur de la fonction sur la 2-cellule, vue 3D.*

**FIGURE 5.14 –** *Résultat de la reconstruction tout d'abord à l'intérieur du carré unité, puis sur la 2-cellule.*

## 5.5 CAS PARTICULIERS

La section précédente 5.4 s'intéresse au cas de base des 2-cellules, c'est-à-dire qu'elle traite du cas où les 2-cellules possèdent quatre courbes de bords et quatre points critiques. Il existe deux types de 2-cellules particulières :

**Cellules poches :** C'est un cas que nous avons évoqué à la fin de la partie 2.3. Les deux points cols sont fusionnés, et cela entraîne que soit les 1-cellules ascendantes, soit les 1-cellules descendantes sont aussi fusionnées.

**Quasi-cellules poches :** Ce deuxième cas correspond à une cellule possédant bien quatre points critiques non fusionnés, mais une partie des 1-cellules ascendantes ou descendantes sont fusionnées et cette partie se trouve dans l'intérieur du domaine comprenant l'union entre la 2-cellule et les 1-cellules étant ses faces. L'un des points critiques, soit le minimum soit le maximum, se trouve lui aussi dans l'intérieur de ce domaine. Nous avons appelé ces cellules des quasi-cellules poches car celles-ci sont entre le cas de base et celui des cellules poches.

Le problème qui se pose alors est la paramétrisation du domaine de ces types de 2-cellules. Nous allons montrer le pré traitement que nous appliquons à ces types de cellules avant d'appliquer la reconstruction telle que nous l'avons décrite à la section 5.4.

### 5.5.1 Cellules poches

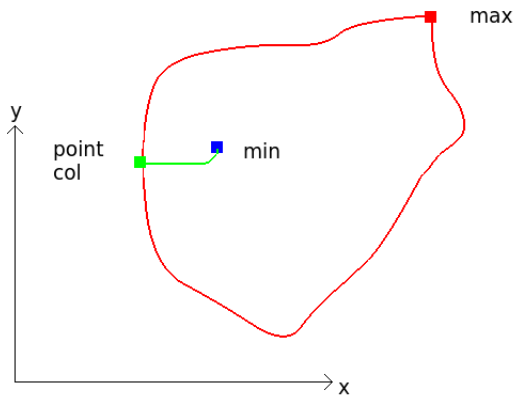


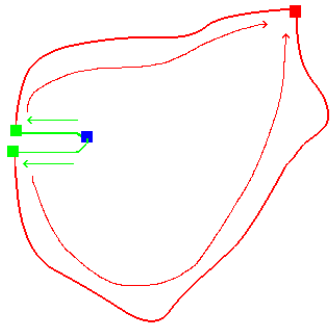

**FIGURE 5.15 –** *Exemple de cellule poche. Les deux points cols sont fusionnés ainsi que les deux 1-cellules descendantes (en vert).*

Le cas des cellules poches correspond à une 2-cellules n'ayant qu'un unique point col sur sa frontière. De plus, deux des 1-cellules sont fusionnées et ne sont pas sur la frontière de l'adhérence de la 2-cellule (voir figure 5.15). Nous ne pouvons donc pas appliquer la méthode de reconstruction telle que décrite précédemment. Il n'est plus possible de faire la mise en correspondance entre $\partial\bar{\Omega}$ et $\partial[0,1]^2$. En effet, le bord de la 2-cellule ne correspond pas aux quatre 1-cellules, mais seulement à deux. Il faut donc bien prendre en compte la 1-cellule qui se trouve à l'intérieur de l'adhérence de la 2-cellule. De plus, celle-ci devra être mise en correspondance avec deux bords du carré unité pour prendre en compte le fait que celle-ci correspond à deux 1-cellules qui ont été fusionnées.

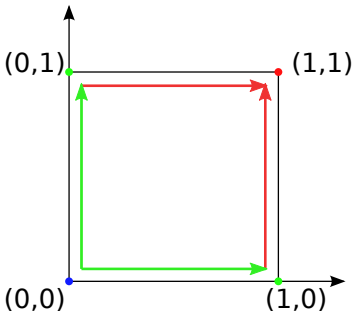
Techniquement, nous construisons une triangulation de la 2-cellule qui va être découpée le long de la 1-cellule interne et dont tout les points le long de cette 1-cellule seront dupliqués. Pour cela, nous commencons par construire une triangulation de Delaunay contrainte par les

arêtes composant les 1-cellules se trouvant sur le bord de la 2-cellule et la 1-cellule interne. Ensuite, nous créons une copie des points se trouvant sur la courbe interne. Puis, nous prenons un des triangles ayant deux points sur la courbe et nous changeons ces deux points par les copies. Enfin, les points se trouvant sur la courbe de tout les triangles se trouvant du même côté de la 1-cellule que le triangle précédent sont changés par les copies. La figure 5.16a schématise le résultat que nous obtenons à la fin de ce pré-traitement. La cellule poche a été découpée le long de la 1-cellule interne. La 2-cellule a alors bien quatre 1-cellules la bordant ainsi que quatre points critiques dont deux points cols. Ensuite, la méthode de reconstruction du cas de base (section 5.4) est appliquée sur la 2-cellule découpée.

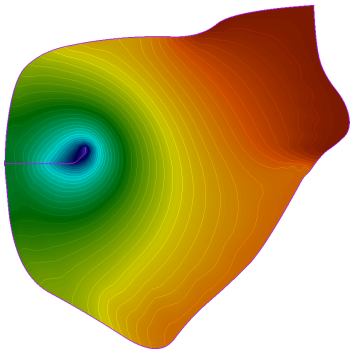
La figure 5.16c montre le résultat de la reconstruction de la fonction pour une cellule poche. La méthode de reconstruction ne créé pas de point critique dans la 2-cellule. Nous pouvons aussi voir que la reconstruction est seulement $C^0$ au travers de la 1-cellule interne.



**(a)** *Schéma du procédé de découpe le long de la 1-cellule interne.*




**(b)** *Les flèches correspondantes aux 1-cellules de la 2-cellule du schéma (a) sont mises en correspondance avec les bords du carré unité.*



**(c)** *Reconstruction de la fonction à l'intérieur d'une cellule poche.*

**FIGURE 5.16 –** *(a) Schéma du découpage d'une cellule poche le long de la 1-cellule interne. (b) Correspondance entre les 1-cellules de la cellule poche et les bords du carré unité. (c) Reconstruction d'une cellule poche.*

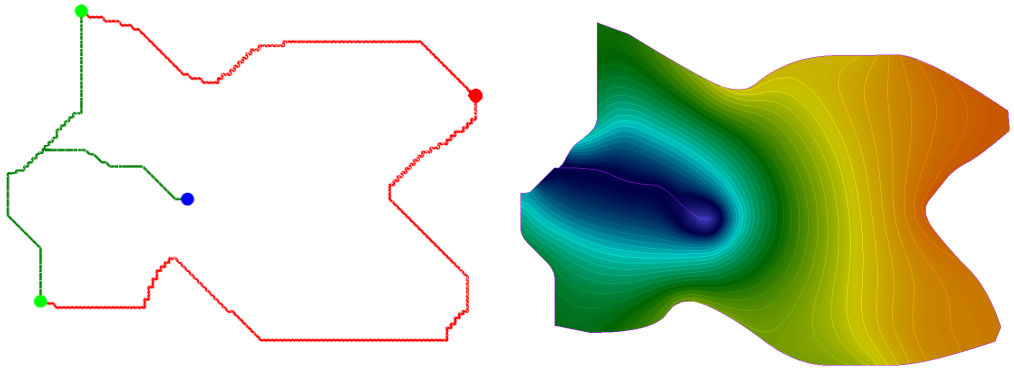
### 5.5.2 Quasi-cellules poches

On parle d'une quasi-cellule poche quand les 1-cellules descendantes ou ascendantes sont fusionnées sur une partie de leur trajet et cette partie fusionnée se trouve à l'intérieur de la 2-cellule (voir figure 5.17a). L'un des extréma se trouve à l'intérieur de la 2-cellule (le minimum à la figure 5.17a). Par contre, il y a bien deux points cols différents contrairement aux cas des

cellules poches. Comme dans le cas de la cellule poche, le problème survient au moment de la paramétrisation : une partie de deux 1-cellules est fusionnée et se trouve à l'intérieur de la 2-cellule. Pour pouvoir mettre cette partie de 1-cellule en correspondance avec les bords du carré unité, nous allons devoir donner deux valeurs de paramètres différents pour chaque point de ce morceau de 1-cellule.

Nous commençons par construire la triangulation de Delaunay ayant pour constraintes les arêtes composant les 1-cellules se trouvant sur le bord de la 2-cellule et la partie fusionnée des deux 1-cellules se trouvant à l'intérieur de la 2-cellules. Nous découpons alors la triangulation le long de la partie des 1-cellules à l'intérieur du domaine de la 2-cellule. Pour cela, nous dupliquons les sommets de la triangulation se trouvant sur cette polyline, puis nous changeons les sommets de tout les triangles se trouvant d'un côté de la courbe par les copies correspondantes. Nous appliquons ensuite la méthode de la section 5.4.

L'image 5.17b montre le résultat de la reconstruction dans ce cas de cellule. Nous pouvons voir que notre méthode ne créé pas de points critiques indésirables. La reconstruction est $C^0$ au travers de la partie de 1-cellule interne.



**(a)** *Exemple de quasi-cellule poche.*

**(b)** *Reconstruction de la fonction à l'intérieur d'une quasi-cellule poche.*

**FIGURE 5.17 –** *(a) Exemple d'une quasi-cellule poche. (b) Reconstruction d'une quasi-cellule poche.*

## 5.6 RÉSULTATS

### 5.6.1 Moyens informatiques

Dans cette section, nous allons présenter les résultats de notre algorithme que nous avons implémenté. Pour le calcul et la simplification des complexes de Morse-Smale, nous utilisons l'implémentation des algorithmes de Shivashankar *et al.* [44] que l'on peut trouver à l'adresse `http://vgl.serc.iisc.ernet.in/mscomplex/index.html`. L'ordinateur sur lequel a été programmé notre méthode a Ubuntu 12.04 LTS 64 bits comme système d'exploitation avec le noyau 3.2.0.33-generic. Il possède 8 Gio de RAM et son processeur est un Intel Xeon W3520 @ 2,67GHz possédant 8 cœurs. Notre méthode a été programmée en C/C++. Plusieurs bibliothèques ont été utilisées :

- Eigen[22] : inversion de matrices ;
- GLPK[2] : résolution d'optimisation linéaires ;
- CGAL[1] : triangulation et paramétrisation.

### 5.6.2 Visualisations

Nous allons montrer deux types de visualisations : les visualisations des complexes de Morse-Smale et représentations de fonctions. Pour visualiser les complexes de Morse-Smale , nous représenterons les points critiques par des disques dont l'index sera indiqué par la couleur. Les disques bleus représentent les minima, les verts les points cols et les rouges les maxima. Les lignes représentent les 1-cellules, les vertes sont les 1-cellules descendantes et les bleues les ascendantes.

Les fonctions sont représentées grâce à une carte des couleurs encodant la valeur de la fonction. Les valeurs basses sont représentées par du bleu foncé et va jusqu'au rouge foncé pour les valeurs hautes.

Les exemples que nous montrerons seront composés d'une image représentant la fonction initiale, puis du complexe de Morse-Smale initial. Ensuite, nous aurons une image avec le complexe de Morse-Smale simplifié, suivi du complexe de Morse-Smale simplifié dont les 1-cellules ont été lissées. Enfin deux images de la reconstruction que notre méthode a produite : la première étant la projection de la fonction dans le plan $(x, y)$ et la seconde une vue 3D de la fonction reconstruite.

### 5.6.3 Les données

Nous allons montrer les résultats de notre algorithme sur trois exemples. Le premier correspond à une fonction qui interpole cinquante points que nous avons tiré aléatoirement. Celui-ci nous a permis d'avoir les deux cas particuliers de 2-cellules ainsi que des 2-cellules correspondant au cas standard. Le deuxième est un exemple réel : c'est la topographie du Mont Rainier aux États-Unis. Il correspond à un exemple contenant beaucoup de paires de points critiques ayant une persistence faible. Le dernier exemple est une fonction $20\pi$-périodique selon les directions $x$ et $y$. Pour cet exemple, nous appliquons l'algorithme de reconstruction sans avoir fait de simplification.

#### *Exemple 1 : Shepard*

L'exemple de la figure 5.18 représente une fonction calculée par la méthode de Shepard [43] que nous avons présentée dans la section 3.1.1 (dans les méthodes de pondération inverse à la distance) avec le coefficient $k = 2$. Nous avons tiré 50 couples $(\{\mathbf{x}_i, f_i\})_{i=1,\dots,50}$ où pour tout $i = 1, \dots, 50$, nous avons $\mathbf{x}_i \in [0, 249]^2$ et $f_i \in [-50, 50]$. La fonction $S : \mathbb{R}^2 \to \mathbb{R}$ telle que pour tout $i = 1, \dots, 50$, $S(\mathbf{x}_i) = f_i$ est alors calculée par la méthode de Shepard. Cette fonction est ensuite échantillonnée sur une grille $250 \times 250$ ayant un pas de 1 et correspondant au domaine $[0, 249]^2$. Cette fonction est représentée à l'image 5.18(a). Son complexe de Morse-Smale est ensuite calculé (figure 5.18(b)) puis une simplification de 25% est appliquée sur ce complexe de Morse-Smale (figure 5.18(c)). Le complexe de Morse-Smale initial contenait 207 points critiques alors que le complexe simplifié en contient 25. Nous montrons aux figures 5.18(e,f) une reconstruction de la fonction où les 1-cellules ont été approximées par $k = 4$ courbes de Bézier cubiques au minimum. Quant aux figures 5.18(g,h), les 1-cellules ont été approximées par au minimum $k = 7$ courbes de Bézier cubiques avant que la reconstruction de la fonction sur les domaines des 2-cellules soit appliquée. Le rôle du paramètre $k$ est de contrôler la flexibilité des courbes qui approximent les 1-cellules. En effet, plus ce paramètre est grand, plus il y a de courbes de Bézier cubiques approximant un morceaux de 1-cellules. Cette approximation peut donc être plus proche de la courbe initiale lorsque le paramètre $k$ est grand. Ce paramètre joue aussi un rôle dans le nombre de surfaces de Bézier triangulaires

qui seront nécessaires par 2-cellules. En effet, si toutes les 1-cellules bordant une 2-cellule sont approximées par $k$ courbes de Bézier cubiques, alors il faut $4k^2$ patchs triangulaires de Bézier à notre méthode pour reconstruire la surface définie sur la 2-cellule. Dans les images 5.18(d) et 5.18(g), nous pouvons voir qu'il y a moins d'ondulations pour $k = 4$, mais aussi que l'approximation est moins bonne que dans le cas où $k = 7$.

### *Exemple 2*

Le second exemple (figure 5.19) correspond à la topographie du Mont Rainier[2], qui se trouve au Nord-Ouest des États-Unis. C'est le plus haut sommet des États-Unis contigus (tous les États sauf Hawaï et l'Alaska) avec un point culminant à 4392 m. Cet ensemble de données contient 140913 points répartis sur une grille régulière $459 \times 307$. Il possède 1931 points critiques, après une simplification de 2% de son complexe de Morse-Smale, il ne contient alors plus que 69 points critiques. Cela nous montre qu'un grand nombre de paires de points critiques ont une valeur de persistence faible comparée à la valeur de persistence de la fonction. Les figures 5.19(d,e,f) montrent d'abord le lissage des 1-cellules dans le plan $(x, y)$, puis la reconstruction calculée à partir du complexe de Morse-Smale simplifié.

La suppression d'un grand nombre de points critiques nous permet d'avoir des 2-cellules ayant des formes allongées. De plus, certaines 2-cellules sont bordées par des 1-cellules très proches dans le plan $(x, y)$ mais qui ne s'intersectent pas. Pour toutes ces 2-cellules, notre méthode a été capable de calculer une surface monotone à l'intérieur de celles-ci tout en interpolant les courbes de bords.

Cet exemple nous permet aussi de voir que la reconstruction est $C^0$ à travers les 1-cellules. En effet, les lignes isovaleures sont seulement continues à travers les 1-cellules.

### *Exemple 3*

La figure 5.20 correspond à la fonction $10\sin(x/10) \cdot \sin(y/10)$ échantillonnée sur une grille $100 \times 100$ correspondant au domaine $[-49, 50]^2$. Cette fonction possède 31 points critiques sur ce domaine. Nous avons appliqué notre méthode de reconstruction sur le complexe de Morse-Smale sans simplification de cette fonction. Les résultats sont visibles aux figures 5.20(d,e).

Nous voyons à nouveau que la surface est $C^0$ à travers les 1-cellules. En effet, les formes données par la carte de couleurs semblent plus anguleuses dans notre reconstruction que dans la surface initiale (voir par exemple les formes données par la couleur jaune). Notre reconstruction ne donne donc pas la même surface que celle donnée en entrée, mais nous avons une surface contenant les mêmes points critiques et n'en contenant aucun autre.

### *Temps de calcul*

Le tableau 5.2 montre les différentes statistiques sur nos exemples ainsi que les temps de calculs.

Nous avons décomposé le temps de calculs en deux parties : la première correspond au lissage des 1-cellules et le second à la reconstruction des surfaces sur les 2-cellules. Cette deuxième partie est elle-même découpée en deux parties : d'une part la triangulation et la reparamétrisation des 2-cellules et d'autre part, le calcul des surfaces sur chaque 2-cellules.

2. http://data.geocomm.com/catalog/

| | **Modèle** | | **MSC** | | | **1-cellules** | | **2-cellules** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | #faces | #V | niveau simplif. | #PC | #2-cellules | k | temps I[1] | temps II Delaunay +param. | temps III géométrie | #Δ patchs Bézier |
| *Shepard* | 124002 | 62500 | 25% | 25 | 22 | 4 | 0,13 | 8,43 | 1,15 | 6672 |
| | | | 25% | 25 | 22 | 7 | 0,13 | 8,80 | 1,39 | 19024 |
| *MtRainier* | 280296 | 140913 | 2% | 69 | 53 | 7 | 0,31 | 9,64 | 2,16 | 43300 |
| *Trigo* | 19602 | 10000 | 0% | 31 | 24 | 7 | 0,03 | 2,11 | 0,11 | 6240 |

[1] temps I correspond au temps de calcul du lissage des 1-cellules dans le plan $(x, y)$ et des valeurs de hauteur.

**TABLE 5.2 –** *Statistiques des exemples.*

La partie la plus coûteuse de la reconstruction est le calcul des surfaces sur les 2-cellules. En effet, le lissage des 1-cellules correspond en moyenne à $1,6\%$ du coût de calcul et le calcul des surfaces à $98,4\%$. Mais les temps de calcul ne sont pas égaux à l'intérieur de la partie de la reconstruction sur les 2-cellules. Le temps comprenant la triangulation et la reparamétrisation correspond en moyenne à $87,8\%$ de la reconstruction sur les 2-cellules ($86,4\%$ du temps de calcul total).

L'implémentation de notre méthode passe donc la majorité de son temps de calcul à trianguler et à reparamétriser les 2-cellules.

De plus, le temps de calcul augmente lorsque le paramètre $k$ (nombre de courbes de Bézier cubiques approximant une 1-cellule) augmente.

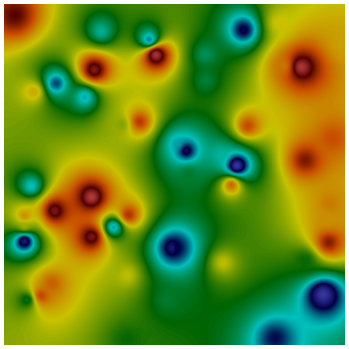
### *Comparaison avec l'état de l'art*

La surface que nous obtenons est $C^0$ à travers les 1-cellules. La reconstruction de Bremer *et al.* [7] est elle aussi $C^0$ à travers les 1-cellules. Quant à la méthode de Weinkauf *et al.* [50], la reconstruction finale est $C^1$. Notre méthode et celle de Bremer *et al.* [7] fait une reconstruction 2-cellule par 2-cellule sans utiliser de conditions sur les dérivées partielles à travers les 1-cellules, alors que la méthode de Weinkauf *et al.* [50] reconstruit la fonction sur le domaine entier grâce à une optimisation sur tout les points de la triangulation.

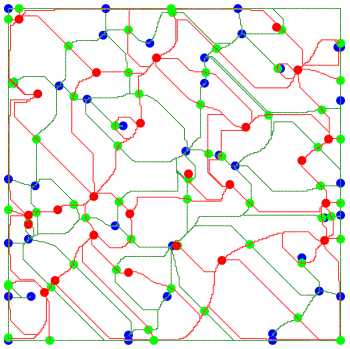
Notre méthode est indépendante du maillage ou de la grille sur lequel la fonction initiale est définie ce qui peut être un avantage dans le cas où la taille de la grille est élevée. De plus, la surface que nous obtenons peut être évaluée en tout point du domaine comprenant l'union de toutes les cellules du complexe de Morse-Smale après application du lissage dans le plan des 1-cellules, et cela sans avoir à appliquer de nouveaux calculs.

### *Reconstruction topologique vs géométrique*

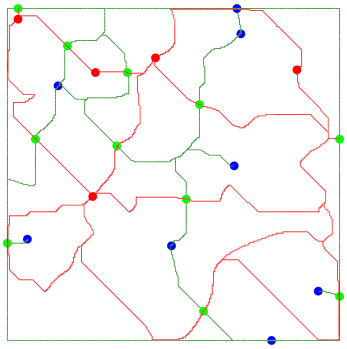
Nous avons vu avec l'exemple de reconstruction de la figure 5.20 que la reconstruction d'une fonction à partir d'un complexe de Morse-Smale non simplifié ne donne pas la même fonction que la fonction initiale. Nous pouvons généraliser cet exemple : notre méthode ne peut pas reconstruire exactement une fonction à partir d'un complexe de Morse-Smale non simplifié. En effet, nous essayons de nous en approcher en approximant les 1-cellules, mais lors de la construction des surfaces sur les 2-cellules, nous cherchons seulement à ne pas construire de nouveau point critique. Nous pourrions imaginer des stratégies pour approximer au mieux la surface initiale. Par exemple à l'étape où l'on calcule des valeurs croissantes le long des diagonales de la grille régulière, nous pourrions utiliser la paramétrisation pour approximer la valeur de la fonction initiale aux points de la grille tout en garantissant la croissance des valeurs



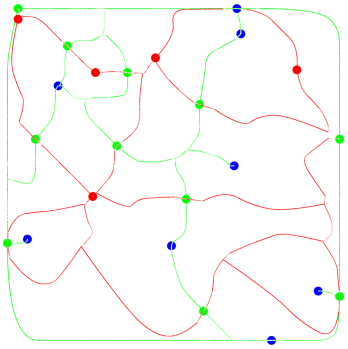
**(a)** *Fonction initiale*



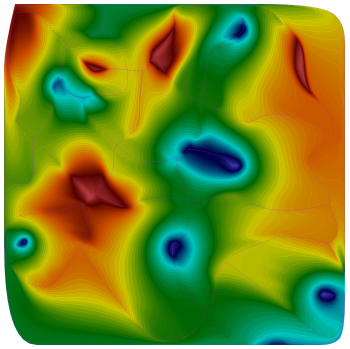
**(b)** *Complexe de Morse-Smale initial*



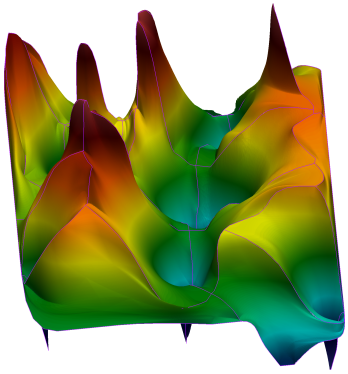
**(c)** *Complexe de Morse-Smale simplifié à 25%*



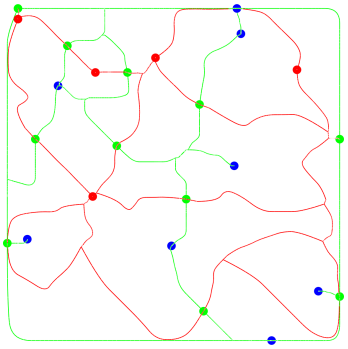
**(d)** *Complexe de Morse-Smale dont les 1-cellules ont été lissées avec $k = 4$ courbes de Bézier par morceaux de 1-cellules*



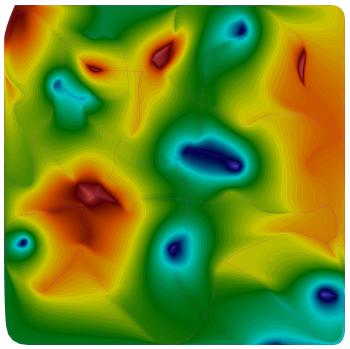
**(e)** *Reconstruction ($k = 4$)*



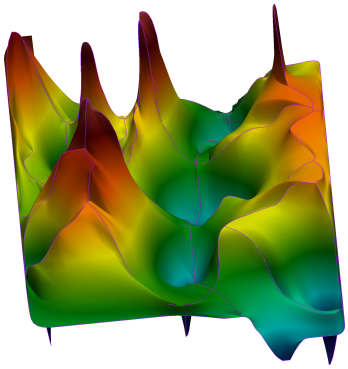
**(f)** *Reconstruction ($k = 4$)*



**(g)** *complexe de Morse-Smale dont les 1-cellules ont été lissées avec $k = 7$ courbes de Bézier par morceaux de 1-cellules*

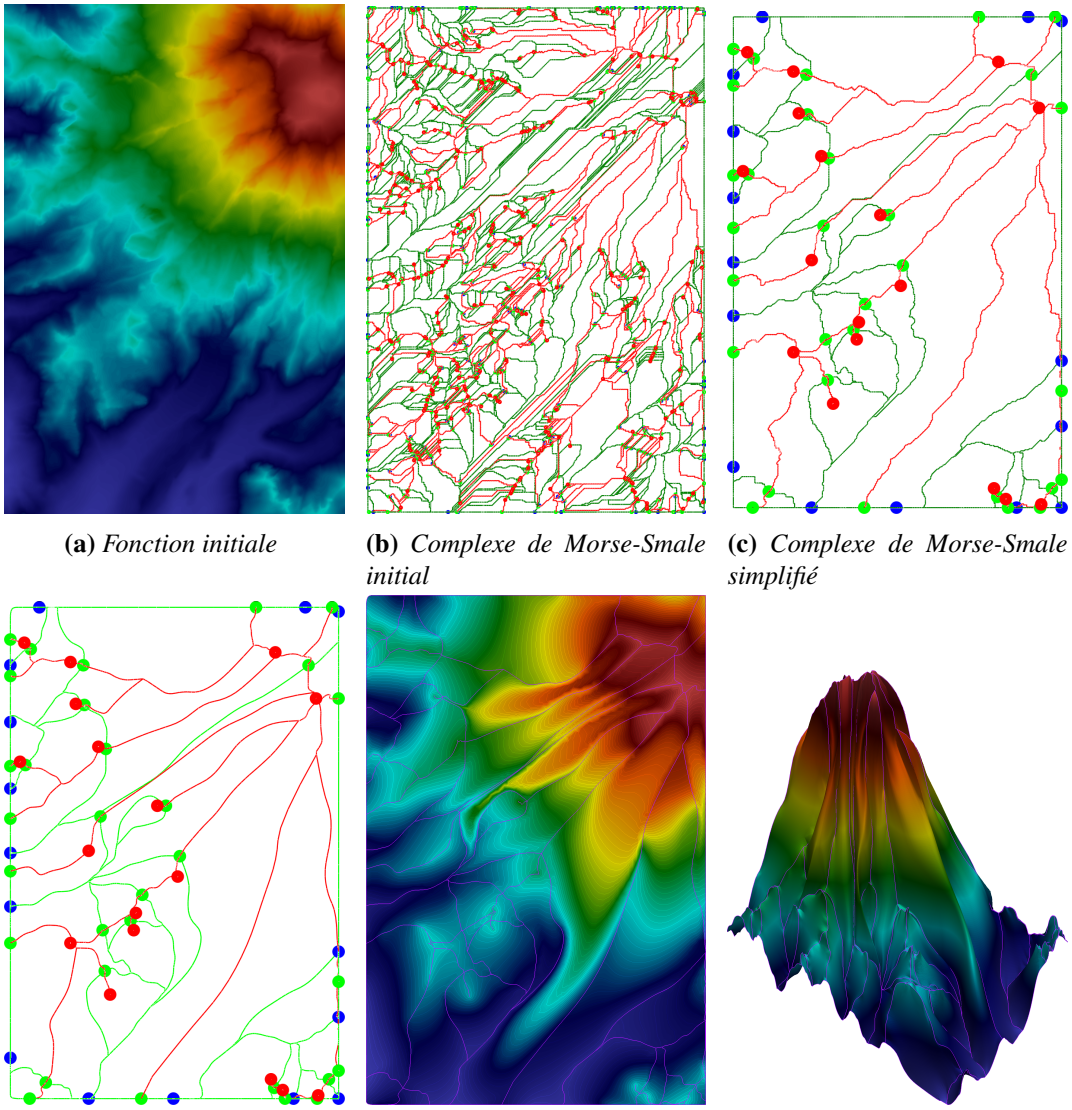


**(h)** *Reconstruction ($k = 7$)*



**(i)** *Reconstruction ($k = 7$)*

**FIGURE 5.18 –** *Application de notre algorithme sur une fonction que nous avons générée en utilisant la méthode d'interpolation de données éparses de Shepard. Nous avons testé notre algorithme avec deux valeurs différentes de $k$ correspondant au nombre de courbes de Bézier par morceaux de 1-cellules. Dans les deux cas, notre méthode reconstruit un champ scalaire qui correspond à la structure topologique du complexe de Morse-Smale.*



**(a)** *Fonction initiale*

**(b)** *Complexe de Morse-Smale initial*

**(c)** *Complexe de Morse-Smale simplifié*

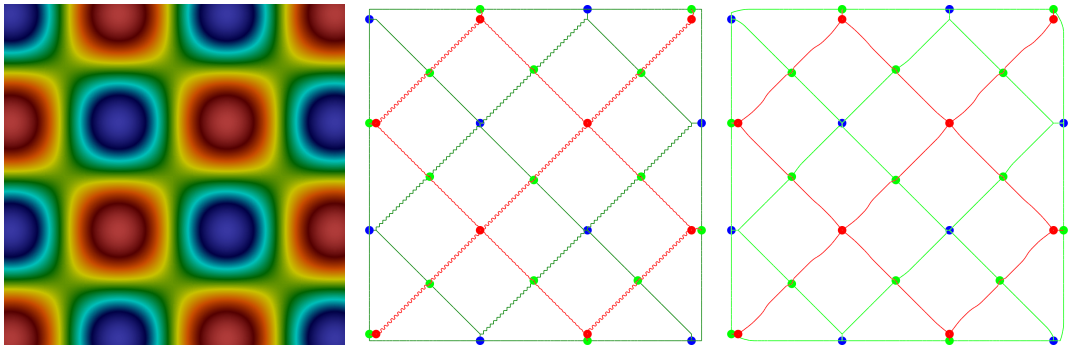
**(d)** *Complexe de Morse-Smale dont les 1-cellules ont été lissées*

**(e)** *Reconstruction*

**(f)** *Reconstruction*

**FIGURE 5.19 –** *Nous avons appliqué notre algorithme à un champ scalaire correspondant à la topographie du Mont Rainier. Son complexe de Morse-Smale initial contient 1931 points critiques. Nous reconstruisons le champs scalaire correspondant à une simplification de 2% qui contient 69 points critiques.*

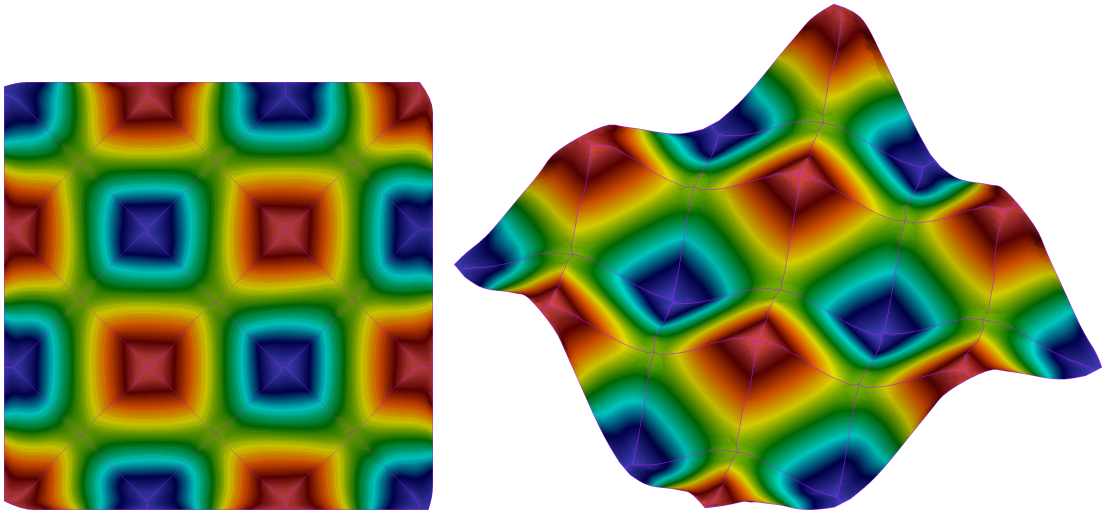
le long des diagonales. Mais ceci n'est pas dans l'esprit de cette méthode : nous souhaitons faire une reconstruction topologiquement cohérente avec le complexe de Morse-Smale . Notre reconstruction garde donc la position et la valeur des points critiques du complexe de Morse-Smale simplifié et approxime les 1-cellules. Ensuite, nous calculons une surface interpolant ces 1-cellules et ne fait plus d'approximation à l'intérieur de la 2-cellule.



**(a)** *Fonction initiale*

**(b)** *Complexe de Morse-Smale initial*

**(c)** *Complexe de Morse-Smale dont les 1-cellules ont été lissées*



**(d)** *Reconstruction*

**(e)** *Reconstruction*

**FIGURE 5.20 –** *La surface à la figure (a) correspond à l'échantillonage sur une grille régulière de la fonction* $10\sin(x/10) \cdot \sin(y/10)$. *La reconstruction est appliquée sur le complexe de Morse-Smale sans simplification topologique..*

***Discussions***

Nous avons décrit une méthode qui est complètement automatique pour reconstruire une fonction à partir d'une fonction initiale et un complexe de Morse-Smale simplifié.

Le paramètre $k$ de notre méthode est modifiable. Celui-ci correspond au nombre de courbes de Bézier cubiques utilisées pour approximer les 1-cellules. Plus il est élevé, plus les approximations des 1-cellules peuvent être proches des 1-cellules du complexe de Morse-Smale . Mais lorsque ce paramètre augmente, le nombre de patchs de Sibson modifiés augment de façon quadratique, d'où une augmentation du temps de calculs.

De l'exemple de la figure 5.18, nous avons pu tirer le cas particulier des quasi-cellules poches. Dans la théorie des complexes de Morse-Smale discrets, celles-ci sont considérés comme des cellules normales, mais nous avons pu voir en section 5.5.2 que de notre point de vue, celles-ci correspondent à un cas proche des cellules poches. En effet, deux des 1-cellules sont fusionnées sur une partie de leur trajet et ceci en étant à l'intérieur de l'adhérence de la 2-cellule. La différence avec les cellules poches est que dans le cas des cellules poches, ces

1-cellules sont fusionnées de l'extremum au point col, alors que dans les cas des quasi-cellules poches, celles-ci sont communes de l'extremum au premier point de jonction.

## 5.7 CONCLUSION

Dans ce chapitre, nous avons présenté une approche originale pour construire un champs scalaire à partir d'un complexe de Morse-Smale simplifié. Celle-ci permet de calculer une surface polynomiale par morceaux préservant les points critiques du complexe de Morse-Smale simplifié ainsi que la connectivité entre ceux-ci. Notre méthode est composée de deux étapes principales : la première est le lissage des 1-cellules dans le plan $(x, y)$ ainsi que des valeurs de hauteur. La deuxième étant la construction de surfaces monotones pour chaque 2-cellules qui interpolent les 1-cellules bordant les 2-cellules. Elles sont calculées grâce à un reparamétrisation de la 2-cellule sur le carré unité. Cette méthode utilise les résultats du chapitre 4 afin de construire des surfaces monotones sur le carré unité.

Un problème de notre méthode est l'utilisation de la paramétrisation par l'algorithme de Floater Mean Value Coordinates. En effet, cette méthode nous donne une paramétrisation $C^0$ entre le domaine de la 2-cellule et le carré unité. Notre surface finale n'est donc que $C^0$. Il existe des méthodes de paramétrisation $C^1$ d'un domaine défini par quatre courbes de bords dans le plan $\mathbb{R}^2$[32]. Mais ces techniques ne parviennent pas à gérer correctement toutes les 2-cellules que nous pouvons obtenir à partir d'un complexe de Morse-Smale . L'apport d'une technique de paramétrisation $C^1$ pour des domaines arbitraires nous permettrait d'améliorer l'aspect lisse de notre surface finale. Nous pourrions, de plus, chercher des techniques pour construire une surface globalement $C^1$, notamment à travers les 1-cellules.

Le second problème concerne les temps de calculs. Pour diminuer ceux-ci, nous pourrions paralléliser notre algorithme. La reconstruction s'effectue 2-cellule par 2-cellule et cela de façon indépendante. Ce calcul peut donc être fait de manière parallèle pour chaque 2-cellule. Une deuxième analyse nous montre que l'étape la plus coûteuse en temps de notre méthode est le calcul de la triangulation d'une 2-cellule puis sa paramétrisation sur le carré unité. Nous aimerions nous passer de cette étape et aller travailler directement dans les 2-cellules.

CHAPITRE

# 6

# CONCLUSION

## 6.1 BILAN

Dans cette thèse, nous avons présenté tout d'abord une modification de l'interpolant de Sibson permettant de créer une surface monotone dans la direction $(x + y)$ sur un domaine rectangulaire. Sur une grille 2D où une valeur de fonction est définie en chaque point, l'utilisation de cet interpolant sur chaque rectangle de la grille permet de construire une surface globalement $C^1$. Nous avons dérivé les conditions suffisantes sur les valeurs de fonctions et les valeurs de dérivées partielles dans les directions $x$ et$y$ pour chaque point de la grille afin que la surface définie par l'interpolant de Sibson modifié vérifiant ces conditions ne possède pas de point critique. De plus, cette surface est croissante dans la direction $(x + y)$. Nous avons construit deux algorithmes permettant d'avoir des valeurs de dérivées partielles vérifiant les conditions suffisantes : le premier permet de modifier des valeurs de dérivées partielles existantes et le second construit les valeurs de dérivées partielles à partir des valeurs de fonctions aux points de la grille.

Nous avons ensuite présenté une méthode permettant de construire une fonction polynomiale par morceaux à partir d'un complexe de Morse-Smale simplifié. Nous reconstruisons les 1-cellules du complexe de Morse-Smale simplifié en les approximant par des courbes de Bézier cubiques. Ensuite, chaque 2-cellule est reparamétrisée sur le carré unité $[0, 1]^2$. L'interpolant de Sibson modifié est alors utilisé pour construire une surface ne contenant pas de points critiques sur $]0, 1[^2$. En utilisant la reparamétrisation et cette surface sur le carré unité, une fonction est construite sur le domaine de la 2-cellule considérée. Cette fonction ne contient alors pas de point critique non voulu sur le domaine de la 2-cellule. L'union des fonctions sur toutes les 2-cellules du complexe de Morse-Smale simplifié correspond à une fonction cohérente avec ce complexe de Morse-Smale .

## 6.2 PERSPECTIVES

Nous allons présenter ici les perspectives que nous laissent les méthodes présentées dans cette thèse.

### 6.2.1 Interpolation monotone de données sur une grille

***Interpolation monotone de données sur un maillage triangulaire***

L'article de Han et Schumaker [27] permet de construire une surface monotone qui interpole des données éparses. Pour cela, ils construisent une grille irrégulière passant par les données. Comme les points ne sont pas à-priori sur une grille, il leur faut donc construire des valeurs à interpoler aux points de la grille où aucune valeur n'est connue. Ils commencent par construire une surface interpolant les points connus. Ensuite, la surface ainsi construite est évaluée aux points de la grille. En un point où la valeur n'est pas connue, il faut regarder si la valeur donnée satisfait les conditions de monotonie selon les directions $x$ et $y$. Si ce n'est pas le cas, elle est changée de façon à ce qu'elle satisfasse les conditions. Les deux problèmes soulevés dans cet article sont qu'à partir de $n$ points, il se peut que de l'ordre de $n^2$ rectangles soient nécessaires pour créer la grille. De plus, ces rectangles peuvent être très petit dans une direction, mais aussi très grand. Une solution serait de déplacer les points dans un cercle de rayon donné par l'utilisateur. Nous aimerions chercher du côté d'un interpolant permettant d'interpoler de façon monotone des données échantillonnées sur un maillage triangulaire.

#### *Surface interpolante $C^2$*

L'interpolant que nous avons proposé permet de construire une surface de continuité $C^1$, cela grâce à l'utilisation de surfaces de Bézier triangulaires de degré 3. En augmentant le degré des surfaces de Bézier au degré 5, il serait possible d'avoir une surface de continuité $C^2$. Il nous faudrait donc trouver les valeurs des points de contrôles, puis chercher les conditions sur les dérivées premières, mais aussi secondes aux points de la grille pour avoir une surface monotone.

### 6.2.2 Reconstruction polynomiale par morceaux de champs scalaires à partir de complexes de Morse-Smale simplifiés

#### *Optimisation des algorithmes*

Il existe deux possibilités d'amélioration de l'algorithmique pour pouvoir réduire les temps de calcul de notre méthode de reconstruction. La première est de travailler sur l'algorithme de paramétrisation des cellules poches et des quasi-cellule poches. En effet, pour le cas des cellules standards, nous utilisons la bibliothèque cgal [1] pour calculer la paramétrisation. En entrée, nous donnons à la bibliothèque seulement le maillage à paramétriser et les valeurs de paramètres le long du bord du domaine. Dans les cas particuliers, nous avons besoin de construire nous même le bord de la 2-cellule, c'est-à-dire l'union du bord du domaine avec la courbe se trouvant à l'intérieur du domaine. Nous devons donc tout d'abord retrouver la courbe interne dans la triangulation de la 2-cellule, mais aussi pour tout triangle en contact avec cette courbe, il nous faut définir de quel côté de la courbe celui-ci se trouve pour que la paramétrisation soit appliquée à un domaine défini par un maillage planaire. Cette étape de découpe est coûteuse en temps de calcul dans notre implémentation, le calcul de la reconstruction de telles cellules prend en moyenne deux fois plus de temps que la reconstruction pour une 2-cellule standard. Nous devrions donc pouvoir améliorer nos algorithmes concernant ce découpage pour faire baisser nos temps de calculs pour les complexes de Morse-Smale contenant des 2-cellules particulières.

La deuxième possibilité d'améliorer notre méthode serait de paralléliser l'algorithme de reconstruction des 2-cellules. En effet, celui-ci travail 2-cellule par 2-cellule et chaque traitement d'une 2-cellule est indépendant du traitement d'une autre. Il serait donc possible de faire la reconstruction des 2-cellules non pas de manière séquentielle, mais en parallèle.

#### *Continuité $G^1$ ou $C^1$ de la reconstruction à travers les 1-cellules*

Notre reconstruction est de continuité $C^0$ à travers les 1-cellules. Cela se manifeste, par exemple, par le fait que les lignes de niveaux ne sont pas lisses lorsque l'on traverse les 1-cellules. Nous pourrions remédier à cela en faisant en sorte que deux 2-cellules adjacentes partagent aussi des informations à propos des dérivées transverses aux 1-cellules. La difficulté vient du fait que les 2-cellules sont paramétrisées sur le carré unité, il nous faudrait donc aussi transformer les informations donnés sur les dérivées transverses à l'intérieur du carré unité. Cela nous permettrait alors d'avoir une continuité $G^1$ ou $C^1$ à travers les 1-cellules.

#### *Amélioration de la paramétrisation*

La méthode de paramétrisation que nous utilisons est seulement $C^0$, ce qui entraîne que notre reconstruction est de continuité $C^0$ par morceaux. La surface que nous calculons sur le carré unité est bien $C^1$, l'utilisation d'une méthode de paramétrisation $C^1$ nous permettrait

d'avoir la même continuité pour la surface définie sur une 2-cellule. Il existe des méthodes de paramétrisation $C^1$ basées sur des splines $C^1$ [32, 52], mais celles-ci ne gèrent pas des domaines quelconques ayant quatre courbes de bords. Or, nous n'avons pas de restrictions sur la forme des 2-cellules des complexes de Morse-Smale. Comme notre méthode n'est pas dépendante d'un certain type de paramétrisation, il nous suffirait d'avoir une paramétrisation $C^1$ qui gérerait toutes les 2-cellules que nous pouvons trouver pour que nous puissions avoir une surface $C^1$ sur les 2-cellules.

# ANNEXES

ANNEXE

# A

# OPTIMISATION AYANT UN OBJECTIF EN SOMME DE VALEURS ABSOLUES

Soit $n \in \mathbb{N}$. Soit $\mathscr{A}$ un ensemble inclus dans $\mathbb{R}^n$ pouvant être défini par des contraintes linéaires et soit $x_1, \ldots, x_n \in \mathbb{R}$. Nous cherchons à minimiser la fonction $f : \mathscr{A} \to \mathbb{R}$ définie par :

$$f(\bar{x}) = \sum_{i=1}^{n} |\bar{x}_i - x_i| \tag{A.1}$$

sur l'ensemble admissible $\mathscr{A}$. L'optimisation s'écrit alors sous la forme :

$$f(\bar{x}) = \sum_{i=1}^{n} |\bar{x}_i - x_i| \longrightarrow min \tag{A.2}$$

sous les contraintes :

$$\bar{x} \in \mathscr{A}.$$

La fonction-objectif $f$ n'étant pas linéaire, nous ne pouvons utiliser les méthodes développer dans le cadre de l'optimisation linéaire. Il est cependant possible de transformer cette optimisation en un problème linéaire. Nous allons ajouter des variables supplémentaires $(r_i^+)_{1 \leqslant i \leqslant n}$ et $(r_i^-)_{1 \leqslant i \leqslant n}$. Posons ensuite un deuxième problème qui est linéaire :

$$h\left(\begin{pmatrix} r_1^- \\ \vdots \\ r_n^- \end{pmatrix}, \begin{pmatrix} r_1^+ \\ \vdots \\ r_n^+ \end{pmatrix}\right) = \sum_{i=1}^{n}(r_i^+ + r_i^-) \longrightarrow min \qquad \text{(A.3)}$$

sous les contraintes :

$$\begin{aligned} & r_i^+ \geqslant 0 \\ & r_i^- \geqslant 0 \\ & r_i^+ - r_i^- = \bar{x}_i - x_i \\ & \bar{x} \in \mathscr{A}, \end{aligned}$$

avec $h : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.

Dans l'optimisation linéaire A.3, la valeur absolue de la fonction $f$ de l'optimisation A.1 a été linéarisée. La linéarisation de la valeur absolue consiste à faire les changements qui suivent. Soit $x \in \mathbb{R}$, on cherche à linéariser la valeur absolue de $x$. Soit $X_{min}, X_{max} \in \mathbb{R}^{+*}$ tels que $-X_{min} \leqslant x \leqslant X_{max}$. Soit $y$ une variable binaire et $x^+, x^- \in \mathbb{R}$. Nous avons alors :

$$z = |x| \Rightarrow \begin{cases} y & > \dfrac{x}{2X_{max}} \\ y & \leqslant \dfrac{x}{2X_{min}} + 1 \\ x & = x^+ - x^- \\ 0 & \leqslant x^+ \leqslant yX_{max} \\ 0 & \leqslant x^- \leqslant (1-y)X_{min} \\ z & = x^+ + x^- \end{cases} .$$

Cette relation permet de linéariser une valeur absolue dans une optimisation. En pratique, cette relation est seulement utile lorsque l'on veut linéariser les contraintes en valeur absolue du problème. Lorsque la valeur absolue est dans la fonction-objectif, alors il est possible de ne prendre qu'une partie de la linéarisation :

$$z = |x| \Rightarrow \begin{cases} x & = x^+ - x^- \\ 0 & \leqslant x^+ \\ 0 & \leqslant x^- \\ z & = x^+ + x^- \end{cases} .$$

Pour passer de l'optimisation A.1 au problème linéaire A.3, les trois premières équations ont été passées en contraintes et la valeur absolue dans la fonction-coût a été remplacée par $z = x^+ + x^-$.

ANNEXE

# B

# DÉRIVATION DES COORDONNÉES BARYCENTRIQUES D'UN TRIANGLE PAR RAPPORT AUX COORDONNÉES *X* ET *Y*

Soit un triangle défini par trois points $P1 = (x1, y1)$, $P_2 = (x_2, y_2)$ et $P_3 = (x_3, y_3)$ et soit $P = (x, y)$ un point se trouvant à l'intérieur du triangle. Il existe alors un triplet $(r, s, t) \in \mathbb{R}^3$ appelé coordonnées barycentriques tel que :

$$\begin{cases} r+ & s+ & t = 1 \\ P_1 r + P_2 & s + P_3 & t = P \end{cases} \tag{B.1}$$

Le but est de calculer les différentes dérivées partielles des coordonnées barycentriques par rapport à $x$ et à $y$.

Nous avons donc que :

$$\begin{cases} r+ & s+ & t = 1 \\ x_1 r + x_2 & s + x_3 & t = x \\ y_1 r + y_2 & s + y_3 & t = y \end{cases} \tag{B.2}$$

Cela équivaut à

$$\begin{cases} t = 1 - r - s & & \\ (x_1 - x_3)r + (x_2 - x_3) & s & = x - x_3 \\ (y_1 - y_3)r + (y_2 - y_3) & s & = y - y_3 \end{cases} \tag{B.3}$$

Les deux dernières lignes de l'équation (B.3) se transforment en le système linéaire qui suit.

$$\begin{pmatrix} x_1 - x_3 & x_2 - x_3 \\ y_1 - y_3 & y_- y_3 \end{pmatrix} \begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} x - x_3 \\ y - y_3 \end{pmatrix} \tag{B.4}$$

La déterminant de la matrice est

$$\begin{aligned} Det\begin{pmatrix} x_1 - x_3 & x_2 - x_3 \\ y_1 - y_3 & y_- y_3 \end{pmatrix} &= (x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_3) \\ &= \overrightarrow{P_3P_1} \wedge \overrightarrow{P_3P_2} \\ &= 2A \end{aligned}$$

avec $A$ l'aire du triangle$P_1P_2P_3$.

Le système peut donc être résolu lorsque l'aire du triangle est différente de $0$. Nous nous plaçons donc dans un tel cas. En résolvant le système, nous obtenons alors la valeur de $r$ et de $s$ en fonction de $x$ e de $y$ :

$$\begin{cases} t = 1 - r - s \\ r = \dfrac{1}{2A}(y_2 - y_3) \quad (x - x_3) + (x_3 - x_2)(y - y_3) \\ s = \dfrac{1}{2A}(y_3 - y_1) \quad (x - x_3) + (x_1 - x_3)(y - y_3) \end{cases} \tag{B.5}$$

En dérivant par rapport à $x$, nous obtenons

$$\begin{cases} \dfrac{\partial r}{\partial x} = \dfrac{y_2 - y_3}{2A} \\ \dfrac{\partial s}{\partial x} = \dfrac{y_3 - y_1}{2A} \\ \dfrac{\partial t}{\partial x} = \dfrac{y_1 - y_2}{2A} \end{cases} \tag{B.6}$$

De même, en dérivant par rapport à $y$, on obtient :

$$\begin{cases} \dfrac{\partial r}{\partial y} = \dfrac{x_3 - x_2}{2A} \\ \dfrac{\partial s}{\partial y} = \dfrac{x_1 - x_3}{2A} \\ \dfrac{\partial t}{\partial y} = \dfrac{x_2 - x_1}{2A} \end{cases} \tag{B.7}$$

Ce qui est le résultat voulu.

ANNEXE

# C

# INTERSECTION DE DEUX COURBES DE BÉZIER CUBIQUES PAR LA MÉTHODE D'IMPLITICISATION

La méthode que nous présentons ici est la méthode de détection de deux courbes de Bézier cubiques par impliticisation des courbes [42]. Nous avons choisi d'utiliser cette méthode car elle est la plus rapide dans les cas de courbes de degré 3 [41]. L'idée de cette méthode est de calculer l'équation implicite d'une des deux courbes paramétriques, y injecter la paramétrisation de la deuxième et obtenir les paramètres correspondants aux points d'intersections en calculant les racines de cette dernière équation.

Soient $P(s) = \sum_{i=0}^{3} P_i B_i^3(s), P_i = (P_i^x, P_i^y)^t \in \mathbb{R}^2$ avec $s \in [0,1]$ et $Q(t) = \sum_{i=0}^{3} Q_i B_i^3(t)$, $Q_i = (Q_i^x, Q_i^y)^t \in \mathbb{R}^2$ avec $t \in [0,1]$ deux courbes de Bézier cubiques. Pour calculer l'équation implicite de la courbe $P(s) = (x(s), y(s))^t$, on utilise la résultante de Bézout des deux polynômes définis par :

$$D(x,y) = \begin{vmatrix} l_{3,2}(x,y) & l_{3,1}(x,y) & l_{3,0}(x,y) \\ l_{3,1}(x,y) & l_{3,0}(x,y) + l_{2,1}(x,y) & l_{2,0}(x,y) \\ l_{3,0}(x,y) & l_{2,0}(x,y) & l_{1,0}(x,y) \end{vmatrix}$$

où

$$l_{i,j}(x,y) = \begin{pmatrix} 3 \\ i \end{pmatrix} \begin{pmatrix} 3 \\ j \end{pmatrix} \begin{vmatrix} x & y & 1 \\ P_i^x & P_i^y & 1 \\ P_j^x & P_j^y & 1 \end{vmatrix}$$

L'équation $D(x,y) = 0$ est alors l'équation implicite de $P(s)$. Elle est de degré 3 en $x$ et $y$.

La courbe $Q(t) = (x(t), y(t))^t$ avec :

$$x(t) = (-Q_0^x + 3Q_1^x - 3Q_2^x + Q_3^x)t^3 + (3Q_0^x - 6Q_1^x + 3Q_2^x)t^2 + (-3Q_0^x + 3Q_1^x)t + Q_0^x$$
$$y(t) = (-Q_0^y + 3Q_1^y - 3Q_2^y + Q_3^y)t^3 + (3Q_0^y - 6Q_1^y + 3Q_2^y)t^2 + (-3Q_0^y + 3Q_1^y)t + Q_0^y$$

est ensuite injectée dans l'équation $D(x, y) = 0$.

Il en résulte un déterminant d'une matrice $3 \times 3$ dont les éléments sont des polynômes de degré 3 en $t$. Le déterminant est donc un polynôme de degré 9 en $t$. Nous devons maintenant trouver les racines de ce polynôme appartenant à l'intervalle $[0, 1]$ car le paramètres $t$ de définition de la courbe $Q$ est compris entre 0 et 1. Pour cela, nous considérons que le polynôme ainsi obtenu $D(t) = D(x(t), y(t))^t$ est le polynôme caractéristique d'une matrice. Cette matrice est la matrice compagnon du polynôme.

**Définition C.1** (Matrices compagnons)**.** Soit $D = \sum_{i=0}^{n} d_i X^i$ avec $d_n \neq 0$ un polynôme de degré $n$. On définit la matrice compagnon de $D$ par :

$$\begin{pmatrix} 0 & 0 & \dots & 0 & -d_0 \\ 1 & 0 & \dots & 0 & -d_1 \\ 0 & 1 & \dots & 0 & -d_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & -d_n \end{pmatrix}$$

Le polynôme caractéristique de cette matrice est le polynôme $D$. Il suffit alors de chercher les valeurs propres de la matrice compagnon de $D$ pour trouver les racines de $D$.

Les racines $(t_i)_{i=1,\dots,9}$ correspondent bien à des points sur la courbe paramétrique $Q(t)$. Or, nous cherchons des intersections uniquement des 2 morceaux de courbes $P(s)$ et $Q(t)$ limités aux intervalles de paramètres $snt \in [0, 1]$. Nous retenons donc uniquement les racines $t_i \in [0, 1]$ comme possibles points d'intersection entre les courbes $P$ e $Q$. Cette vérification s'appelle l'inversion d'un point par rapport à la courbe $P$.

L'inversion consiste donc à trouver le paramètre $s$ du point $A = (x_A, y_A)^t$ par rapport à la courbe de Bézier cubique $P$. Il faut résoudre un système linéaire utilisant les $l_{i,j}$ définis précédemment.

$$\begin{pmatrix} l_{3,2}(x_A, y_A) & l_{3,1}(x_A, y_A) & l_{3,0}(x_A, y_A) \\ l_{3,1}(x_A, y_A) & l_{3,0}(x_A, y_A) + l_{2,1}(x_A, y_A) & l_{2,0}(x_A, y_A) \\ l_{3,0}(x_A, y_A) & l_{2,0}(x_A, y_A) & l_{1,0}(x_A, y_A) \end{pmatrix} \begin{pmatrix} s^2 \\ s(1-s) \\ (1-s)^2 \end{pmatrix} = 0 \quad \text{(C.1)}$$

Dans le cas où les points $P_1, P_2$ et $P_3$ ne sont pas alignés, nous définissons deux constantes $c_1$ et $c_2$ par :

$$c_1 = \frac{\begin{vmatrix} P_0^x & P_0^y & 1 \\ P_1^x & P_1^y & 1 \\ P_3^x & P_3^y & 1 \end{vmatrix}}{\begin{vmatrix} P_0^x & P_0^y & 1 \\ P_2^x & P_2^y & 1 \\ P_3^x & P_3^y & 1 \end{vmatrix}}, \; c_2 = \frac{\begin{vmatrix} P_0^x & P_0^y & 1 \\ P_2^x & P_2^y & 1 \\ P_3^x & P_3^y & 1 \end{vmatrix}}{\begin{vmatrix} P_0^x & P_0^y & 1 \\ P_2^x & P_2^y & 1 \\ P_3^x & P_3^y & 1 \end{vmatrix}}$$

En sommant $c_1$ fois la première ligne du système (C.1) et $c_2$ multiplié par la deuxième ligne avec la troisième ligne, on obtient alors :

$$\begin{pmatrix} l_{3,2}(x_A, y_A) & l_{3,1}(x_A, y_A) & l_{3,0}(x_A, y_A) \\ l_{3,1}(x_A, y_A) & l_{3,0}(x_A, y_A) + l_{2,1}(x_A, y_A) & l_{2,0}(x_A, y_A) \\ 0 & l_a(x_A, y_A) & l_b(x_A, y_A) \end{pmatrix} \begin{pmatrix} s^2 \\ s(1-s) \\ (1-s)^2 \end{pmatrix} = 0 \quad \text{(C.2)}$$

avec

$$l_a(x,y) = c_1 l_{3,1}(x,y) + c_2(l_{3,0}(x,y) + l_{2,1}(x,y)) + l_{2,0}(x,y)$$
$$l_b(x,y) = c_1 l_{3,0}(x,y) + c_2 l_{2,0}(x,y) + l_{1,0}(x,y)$$

De la troisième ligne du système (C.2), on obtient :

$$l_a(x_A, y_A)s(1-s) + l_b(x_A, y_A)(1-s)^2 = 0$$

La valeur du paramètre $s$ est alors :

$$s = \frac{l_b(x_A, y_A)}{l_b(x_A, y_A) - l_a(x_A, y_A)}$$

Pour le cas où les points sont alignés, il suffit de faire des changements d'indice pour avoir un diviseur différent de 0 dans les fractions définissant $c_1$ et $c_2$.

Les courbes s'intersectent alors si les paramètres $t$ et $s$ appartiennent bien à l'intervalle $[0, 1]$.

# BIBLIOGRAPHIE


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

[2] GLPK : GNU Linear Programming Kit. http://www.gnu.org/software/glpk/glpk.html.

[3] Léo Allemand-Giorgis, Georges-Pierre Bonneau, et Stefanie Hahmann. Reconstruction polynomiale par morceaux de fonctions à partir de complexes de Morse-Smale simplifiés. In *AFIG 2014 - 27es journées de l'Association Française d'Informatique Graphique*, Actes des Journées AFIG, Reims, France, November 2014. URL https://hal.inria.fr/hal-01100263.

[4] Léo Allemand-Giorgis, Georges-Pierre Bonneau, et Stefanie Hahmann. Piecewise Polynomial Reconstruction of Functions from Simplified Morse-Smale complex. In *IEEE Visualization conference 2014, SciVis Posters*, Paris, France, November 2014. IEEE. URL https://hal.inria.fr/hal-01100280.

[5] Léo Allemand-Giorgis, Georges-Pierre Bonneau, Stefanie Hahmann, et Fabien Vivodtzev. Piecewise polynomial monotonic interpolation of 2D gridded data. In Janine Bennett, Fabien Vivodtzev, et Valerio Pascucci, editors, *Topological and Statistical Methods for Complex Data*, Mathematics and Visualization. Springer, 2014.

[6] R.K.T. Beatson et Z. Ziegler. Monotonicity Preserving Surface Interpolation. *SIAM J. Numer. Anal.*, 22(2) :401–411, 1985.

[7] P. T. Bremer, B. Hamann, H. Edelsbrunner, et V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4) :385–396, jul 2004.

[8] Ken Brodlie et Petros Mashwama. Controlled Interpolation for Scientific Visualization. In Gregory M. Nielson, Hans Hagen, et Heinrich Müller, editors, *Scientific Visualization, Overviews, Methodologies, and Techniques, Dagstuhl, Germany, May 1994*, pages 253–276. IEEE Computer Society, 1994. ISBN 0-8186-7777-5.

[9] R. E. Carlson et F. N. Fritsch. A Bivariate Interpolation Algorithm for Data That Are Monotone in One Variable. *SIAM J. Sci. Stat. Comput.*, 12(4) :859–866, May 1991. ISSN 0196-5204. doi : 10.1137/0912046. URL http://dx.doi.org/10.1137/0912046.

[10] R.E. Carlson et F.N. Fritsch. An Algorithm for Monotone Piecewise Bicubic Interpolation. *SIAM Journal on Numerical Analysis*, 26(1) :230–238, 1989.

[11] Ray W Clough et James L Tocher. Finite element stiffness matrices for analysis of plates in bending. In *Proceedings of conference on matrix methods in structural analysis*, pages 515–545, 1965.

[12] Boris Delaunay. Sur la sphère vide. A la mémoire de Georges Voronoï. *Bulletin de l'Académie des Sciences de l'URSS*, (6) :793–800, 1934.

[13] H. Edelsbrunner, D. Letscher, et A. Zomorodian. Topological persistence and simplification. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, pages 454–, Washington, DC, USA, 2000. IEEE Computer Society.

[14] Herbert Edelsbrunner, John Harer, et Afra Zomorodian. Hierarchical Morse-Smale Complexes for Piecewise Linear 2-Manifolds. *Discrete & Computational Geometry*, 30(1) : 87–107, 2003.

[15] Gerald E. Farin. *Curves and Surfaces for Computer-Aided Geometric Design : A Practical Code*. Academic Press, Inc., Orlando, FL, USA, 4th edition, 1996.

[16] Gerald E. Farin. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3(2) :83–127, 1986.

[17] Michael S. Floater. Mean Value Coordinates. *Comput. Aided Geom. Des.*, 20(1) :19–27, 2003. ISSN 0167-8396.

[18] Michael S. Floater et Kai Hormann. Surface Parameterization : a Tutorial and Survey. In N. A. Dodgson, M. S. Floater, et M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 157–186. Springer Verlag, 2005. URL http://vcg.isti.cnr.it/Publications/2005/FH05.

[19] MichaelS. Floater et J.M. Peña. Tensor-product monotonicity preservation. *Advances in Computational Mathematics*, 9(3-4) :353–362, 1998.

[20] Robin Forman. A User's Guide To Discrete Morse Theory. In *Proc. of the 2001 Internat. Conf. on Formal Power Series and Algebraic Combinatorics, A special volume of Advances in Applied Mathematics*, page 48, 2001.

[21] F. N. Fritsch et R. E. Carlson. Monotone Piecewise Cubic Interpolation. *SIAM Journal on Numerical Analysis*, 17(2) :238–246, 1980.

[22] Gaël Guennebaud, Benoît Jacob, *et al.* Eigen v3. http ://eigen.tuxfamily.org, 2010.

[23] David Günther, Alec Jacobson, Jan Reininghaus, Hans-Peter Seidel, Olga Sorkine-Hornung, et Tino Weinkauf. Fast and Memory-Efficient Topological Denoising of 2D and 3D Scalar Fields. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Scientific Visualization / Information Visualization 2014)*, 20(12) :to appear, December 2014.

[24] David Günther, Jan Reininghaus, Hans Peter Seidel, et Tino Weinkauf. Notes on the Simplification of the Morse-Smale Complex. In Peer-Timo Bremer, Ingrid Hotz, Valerio Pascucci, et Ronald Peikert, editors, *Topological Methods in Data Analysis and Visualization III*, Mathematics and Visualization, pages 135–150. Springer International Publishing, 2014.

[25] A. Gyulassy, P.-T. Bremer, B. Hamann, et V. Pascucci. A Practical Approach to Morse-Smale Complex Computation : Scalability and Generality. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6) :1619–1626, 2008.

[26] A. Gyulassy, P. Bremer, et V. Pascucci. Computing Morse-Smale Complexes with Accurate Geometry. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12) : 2014–2022, Dec 2012. ISSN 1077-2626. doi : 10.1109/TVCG.2012.209.

[27] Lu Han et Larry L. Schumaker. Fitting Monotone Surfaces to Scattered Data Using C1 Piecewise Cubics. *SIAM J. Numer. Anal.*, 34(2) :569–585, 1997.

[28] R.L. Hardy. Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968–1988. *Computers & Mathematics with Applications*, 19(8) :163–208, 1990. ISSN 0898-1221. doi : 10.1016/0898-1221(90)90272-L. URL http://www.sciencedirect.com/science/article/pii/089812219090272L.

[29] Rolland L. Hardy. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76(8) :1905–1915, 1971. ISSN 2156-2202. doi : 10.1029/JB076i008p01905. URL http://dx.doi.org/10.1029/JB076i008p01905.

[30] Xuming He et Peide Shi. Monotone B-spline Smoothing. *Journal of the American Statistical Association*, 93 :643–650, 1996.

[31] Alec Jacobson, Tino Weinkauf, et Olga Sorkine. Smooth Shape-Aware Functions with Controlled Extrema. *Comp. Graph. Forum*, 31(5) :1577–1586, 2012. ISSN 0167-7055.

[32] Noah Jaxon et Xiaoping Qian. Isogeometric analysis on triangulations. *Computer-Aided Design*, 46(0) :45 – 57, 2014. 2013 {SIAM} Conference on Geometric and Physical Modeling.

[33] Henry King, Kevin Knudson, et Neža Mramor. Generating Discrete Morse Functions from Point Data. *Experiment. Math.*, 14(4) :435–444, 2005.

[34] Thomas Lewiner, Hélio Lopes, et Geovan Tavares. Optimal discrete Morse functions for 2-manifolds. *Computational Geometry*, 26(3) :221 – 233, 2003. ISSN 0925-7721. doi : http://dx.doi.org/10.1016/S0925-7721(03)00014-2. URL http://www.sciencedirect.com/science/article/pii/S0925772103000142.

[35] D. H. McLain. Errata. *The Computer Journal*, 19(4) :384, 1976. doi : 10.1093/comjnl/19.4.384. URL http://comjnl.oxfordjournals.org/content/19/4/384.1.short.

[36] D. H. McLain. Two Dimensional Interpolation from Random Data. *The Computer Journal*, 19(2) :178–181, 1976. doi : 10.1093/comjnl/19.2.178. URL http://comjnl.oxfordjournals.org/content/19/2/178.abstract.

[37] T. Nguyen et B. Jüttler. Parameterization of Contractible Domains using Sequences of Harmonic Maps. In J.-D. Boissonnat, P. Chenin, A. Cohen, C. Gout, T. Lyche, M.-L. Mazure, et L. Schumaker, editors, *Curves and Surfaces*, volume 6920 of *Lecture Notes in Computer Science*, pages 501–514. Springer, 2012. doi : 10.1007/978-3-642-27413-8_32.

[38] M. J. D. Powell et M. A. Sabin. Piecewise Quadratic Approximations on Triangles. *ACM Trans. Math. Softw.*, 3(4) :316–325, December 1977. ISSN 0098-3500. doi : 10.1145/355759.355761. URL http://doi.acm.org/10.1145/355759.355761.

[39] Jan Reininghaus, David Günther, Ingrid Hotz, Tino Weinkauf, et Hans-Peter Seidel. Combinatorial Gradient Fields for 2D Images with Empirically Convergent Separatrices. *CoRR*, abs/1208.6523, 2012. URL http://arxiv.org/abs/1208.6523.

[40] Larry L. Schumaker et Cornelis Traas. Fitting scattered data on spherelike surfaces using tensor products of trigonometric and polynomial splines. *Numerische Mathematik*, 60 (1) :133–144. ISSN 0945-3245. doi : 10.1007/BF01385718. URL http://dx.doi.org/10.1007/BF01385718.

[41] Thomas W Sederberg et Scott R Parry. Comparison of three curve intersection algorithms. *Computer-Aided Design*, 18(1) :58 – 63, 1986. ISSN 0010-4485. doi : http://dx.doi.org/10.1016/S0010-4485(86)80013-6. URL http://www.sciencedirect.com/science/article/pii/S0010448586800136.

[42] T.W Sederberg, D.C Anderson, et R.N Goldman. Implicit representation of parametric curves and surfaces. *Computer Vision, Graphics, and Image Processing*, 28 (1) :72 – 84, 1984. ISSN 0734-189X. doi : http://dx.doi.org/10.1016/0734-189X(84)90140-3. URL http://www.sciencedirect.com/science/article/pii/0734189X84901403.

[43] D. Shepard. A two-dimensional interpolation function for irregularly spaced data. In *Proceedings of the Symposium on Geometry Processing*, Proceedings of the 1968 23rd ACM National Conference, page 517–523. iACM, 1968.

[44] Nithin Shivashankar, Senthilnathan M, et Vijay Natarajan. Parallel Computation of 2D Morse-Smale Complexes. *IEEE Transactions on Visualization and Computer Graphics*, 18(10) :1757–1770, 2012. ISSN 1077-2626.

[45] Robin Sibson. A brief description of natural neighbour interpolation. *Interpreting multivariate data*, 21 :21–36, 1981.

[46] Robin Sibson et Graeme Thomson. A seamed quadratic element for contouring. *THe Computer Journal*, 24(4) :378–382, 1981.

[47] Gabriel Taubin. A Signal Processing Approach to Fair Surface Design. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 351–358, New York, NY, USA, 1995. ACM. ISBN 0-89791-701-4. doi : 10.1145/218380.218473. URL http://doi.acm.org/10.1145/218380.218473.

[48] W. T. Tutte. HOW TO DRAW A Graph, 1963.

[49] Florencio Utreras et M Leonor Varas. Monotone interpolation of scattered data in R s. *Constructive Approximation*, 7(1) :49–68, 1991.

[50] Tino Weinkauf, Yotam Gingold, et Olga Sorkine. Topology-based smoothing of 2D scalar fields with C1-continuity. In *Proceedings of the 12th Eurographics/IEEE-VGTC conference on Visualization*, EuroVis'10, pages 1221–1230, 2010.

[51] Karin Willemans et Paul Dierckx. Smoothing scattered data with a monotone Powell-Sabin spline surface. *Numerical Algorithms*, 12(1) :215–231, 1996.

[52] Gang Xu, Bernard Mourrain, Régis Duvigneau, et André Galligo. Parameterization of computational domain in isogeometric analysis : Methods and comparison. *Computer Methods in Applied Mechanics and Engineering*, 200(23–24) :2021 – 2031, 2011.