VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

APPROXIMATION OF TERRAIN DATA UTILIZING MULTIVARIATE SPLINES

MASTER'S THESIS

AUTHOR Bc. PETER TOMEK

SUPERVISOR Dr.-Ing. FLORIAN FISCH
Ing. PETER CHUDÝ, Ph.D.

BRNO – MUNICH 2012

Abstrakt

Pro optimalizaci letových trajektorií ve velmi malé nadmorské výšce, terenní vlastnosti musí být zahrnuty velice přesne. Proto rychlá a efektivní evaluace terenních dat je velice důležitá vzhledem nato, že čas potrebný pro optimalizaci musí být co nejkratší. Navyše, na optimalizaci letové trajektorie se využívájí metody založené na výpočtu gradientu. Proto musí být aproximační funkce terenních dat spojitá do určitého stupne derivace. Velice nádejná metoda na aproximaci terenních dat je aplikace víceroměrných simplex polynomů. Cílem této práce je implementovat funkci, která vyhodnotí dané terenní data na určitých bodech spolu s gradientem pomocí vícerozměrných splajnů. Program by měl vyčíslit více bodů najednou a měl by pracovat v n-dimensionálním prostoru.

Abstract

For the optimization of near-of-the-earth flight trajectories the terrain data have to be taken into account very precisely. At this, a fast and efficient evaluation of terrain data is very important since within the optimization task the computational effort for one single cost function evaluation has to be as small as possible. Furthermore, the trajectory optimization is done by gradient-based optimization methods. Thus, the approximation of the terrain data has to be continuously differentiable and also the gradients of the terrain data have to be evaluated along with the terrain data itself. A very promising approach for the approximation of the terrain data are multivariate splines based on the triangulations of the approximation domain.
The aim of this master thesis was to develop a MATLAB and C++ function that evaluates given terrain data at certain points along with the gradients of the terrain data at these points based on multivariate splines. The function supports evaluation of multiple points at once and is not limited to the three-dimensional data but should also be capable to approximate the data of any dimension.

Klíčová slova

vícerozměrný simplex splajn, triangulace, barycentrické souřadnice, Bernsteinovi polynomi, podmínky spojitosti, modelování terenních dat, lineární regrese

Keywords

multivariate simplex splines, simplex, triangulation, barycentric coordinates, Bernstein basis polynomials, B-form, B-coefficients, linear regression, equality constrained least squares estimator, smoothness matrix, continuity conditions, terrain approximation

Citation

Peter Tomek: Approximation of Terrain Data Utilizing Multivariate Splines,
master's thesis, FIT VUT v Brně, MW TU München, 2012

Approximation of Terrain Data Utilizing Multivariate Splines

Declaration

I have written this Master thesis independently and without the aid of unfair or unauthorized resources. Whenever content was taken directly or indirectly from other sources, this has been indicated and the source referenced.

....
Peter Tomek
May 23, 2012

Acknowledgement

The two people I would like to thank most are Florian Fisch from the department of Flugsystemdynamik at TU München and Peter Chudý from the department of Computer Graphics and Multimedia at VUT Brno.

Without your help this project would not have been possible.

Contents

List of Figures

Chapter 1

Introduction

In these days, in times of a huge expansion of the aviation industry in which also unmanned aerial vehicles are becoming more and more common, the exact description of the environment in which the aircraft is located, as well as a fast evaluation of the immediate environmental condition changes is essential. The term environment description is understood as a set of useful data obtained by taking measurements which are inherently discrete and scattered. The usage of various mathematical methods for assessing a wide range of interesting features assumes an evaluation of a continuous differentiable function. Thus, the need to model the observed data is obvious.

In case of the near-to-earth flight trajectories optimization, the terrain data need to be taken into account very precisely. One example can be found in the defence industry, where cruise missiles or other air force vehicles are moving in a very low altitude. Furthermore, due to the high speed of the aircrafts, the effort to evaluate the modelled data must be very fast and computationally efficient.

Today, for any kind of surface modelling, the techniques based on the extremely popular B-spines are typically used. All of them have their own strengths and weaknesses and some of them have become widely used and are considered as finalized theories. Another very promising approach for the approximation of the scattered data, although not very known so far, are multivariate splines based on triangulations of an approximation domain or simply called multivariate simplex splines. The main important properties of simplex splines are:

- a possibility of scattered data modelling,
- an approximation can be made on an non-rectangular domain,
- modelling of data in any dimension,
- an approximation power is limited only by the complexity and the density of a triangulation,
- a mathematically well defined theory,
- a computationally fast and efficient evaluation.

Despite of all these positive features above, the applications of the multivariate simplex splines are limited today. In the last thirty years there were many of other approaches [13] which looked better and were preferred for some practical reasons of those times. But today, the method of the multivariate simplex splines has found its place and is being used more and more. This background is more discussed in the section 1.1 where also some other interpolation methods are mentioned.

1.1 The related interpolation methods

Splines as an interpolation tool are widely used in thousands of different applications. Especially in the *computer graphics* are these special polynomials useful because of the simplicity of their implementation, their convenient and accurate evaluation, and their ability to approximate the complex shapes through curve fitting and an interactive curve design. In this field the B-spline and the Bézier spline are the most common and the mathematical theories describing them are well developed [19], however, in case of modelling the 3 or n-dimensional data the univariate splines are usually not enough. The further generalization to n dimensions is needed. Through the years many papers [4, 6, 18] were published trying to handle the topic of the multivariate splines and today we can see that the result is ambiguous. The elegance and the power of the mathematical theory of the univariate B-splines motivated the search for a true multivariate generalization of the theory, which proved to be much more difficult than initially expected. De Boor in his paper states: *„The generalization of univariate polynomial interpolation to the multivariate context is made difficult by the fact that one has to decide just which of the many of its nice properties to preserve, as it is impossible to preserve them all“* [7].

1.1.1 The tensor product splines

The need for 2-dimensional splines rose in the field of the car industry in 1950's, as cars made with smooth curves and surfaces had become a trend. Mathematician Paul de Casteljau, working for Citroën, invented the method for modelling smooth surfaces using Bernstein polynomials defined in terms of the barycentric coordinates on rectangular and triangular patches [12]. Later Pierre Étienne Bézier working for Renault company invented his Bézier curves and patches, those idea where similar to the previous work of de Casteljau, and reached high popularity. The first compact and widely used method for the surface interpolation by splines was presented in 1972 by [2] and is called *The multivariate tensor product splines*, or shortly TPS, which are simple Bézier patches stitched together so they form a continuous surface. The algorithmic representation of the TPS is simple and fast, thus they are widely used in CAD applications, however, they cannot be used directly for the interpolation of the scattered data for some essential reasons [1, 5]. The theory of the TPS in not a true generalization of the univariate splines as a construction of bivariate function is made by taking a tensor product of two univariate B-splines. In result, TPS is good for modelling artificial data placed on a rectangular grid and because the data obtained by taking measurements are inherently scattered for the purpose of this work it is better to use something else.

1.1.2 The thin plate splines

The inability of scattered data modelling by the tensor product splines method led the search for an alternative and the answer was called *The thin plate splines*. This technique was used by companies in industry, however it possesses certain limitations. The first is that the thin plate splines have an non-local radial basis function which leads to inefficient algorithms. Its creation and evaluation requires an evaluation of all the basis functions and also the coefficients. The interpolation algorithms are becoming slow, inefficient and for this reason the usability of the thin plate splines method is limited only to two dimensional space which for some purposes may be enough. From the other side, the presence of a global basis function is in fact a great advantage as an interpolation is smooth and differentiable up to any order. Also the thin plate spline model has no other free parameters that would require complicated manual tuning of the model. Everything summed up, even though the scattered data modelling is possible, this method is inefficient and therefore may be used only with small datasets in low dimensions. More about this topic can be found in [4].

1.1.3 The polyhedral splines

The true and mathematically elegant multivariate splines generalization was formed in the theory of *The polyhedral splines* that preserves most of the convenient properties of univariate splines. The polyhedral spline is formed by projecting a multi-dimensional polyhedron like a hypercube or a simplex onto a lower dimensional plane. The value of a multivariate polynomial at a specific location is then the volume of a slice of the polyhedron 'floating above' the given location. This concept is mathematically general and clear but has never become widely used. The reason is that the polyhedral splines are simply too general. The problem of combining polyhedrons in high dimensions to create a desired polynomial is complex and sometimes tricky. The result are computationally inefficient algorithms, and again as the previous methods, in these days the polyhedral splines technique is not suitable for scattered data modelling in practical applications. The complete and rigorous concept of the multivariate polyhedral splines theory was provided e.g. by [18].

1.2 The goals of this thesis

The idea of simplex splines is not very new, however, only few practical applications using this method can be found these days. Therefore, the goal of this thesis is to provide a practical implementation of a function that evaluates given terrain data at certain points along with the gradients at these points based on the multivariate simplex splines. Although the terrain data are obtained in a 3-dimensional space, the implemented function should be able to approximate the data in an n-dimensional space. After the implementation is done, the detail demonstration of the main features of a simplex spline function is provided. Finally, the presented approach is applied on a real terrain dataset in order to demonstrate that the terrain data modelling by simplex splines is possible and even appropriate.

In the chapter 2 the theoretical background of the simplex splines is introduced and demonstrated by many figures and examples. Later on in the next chapter 3, the implemented function is tested on 3 and 4 dimensional datasets. This chapter also discusses the input parameters of a simplex spline function and how they need to be set in order to get a high precision of a resulting simplex spline model. In the end, all these 'know how' are applied on a real terrain dataset.

Chapter 2

The multivariate simplex splines

The requirements that an ideal interpolation method would meet are an ability of the scattered data modelling in n-dimensions, a well defined mathematical concept, a computationally fast and efficient algorithmic implementation and flexibility, by means of an adaptation to different scenarios. *The multivariate simplex splines* is just another method of multivariate splines that claims to satisfy these requirements. Since now, the notion *simplex splines* determine Bernstein polynomial functions defined on simplices using barycentric coordinates. The multivariate simplex splines are based on triangular Bernstein-Bézier patches invented by de Casteljau as mentioned in the section 1.1.1.

The idea is relatively simple. Geometric n-dimensional structures called simplices each support a single B-form polynomial. A B-form polynomial is a linear combination of Bernstein basis polynomials the shapes of which are controlled by B-coefficients. The real power of the simplex splines comes when many simplices on each of which Bernstein polynomial is defined, are joined together. This is achieved by a special topological configuration called a triangulation. The triangulation subdivides the scattered input n-dimensional points into the simplices. In a valid triangulation, the simplices are allowed to meet only at their vertices and no two simplices are allowed to overlap. This condition is very important in order to obtain the continuity of individual Bernstein polynomials. As mentioned before, B-coefficients fully control a local polynomial defined on a single simplex. Therefore, finding a solution of the scattered data approximation problem on a given triangulation reduces to finding an optimal set of B-coefficients for every B-form polynomial. A possibility to model n-dimensional scattered data on a non-rectangular domain is a crucial behaviour. Furthermore, the evaluation of Bernstein polynomials is much faster these days, so there is no need to use special algorithms as e.g. de Casteljau algorithm [12].

The theoretical concept of multivariate simplex splines described in this chapter was introduced by authors of [9, 6, 14, 15]. As the implemented algorithm directly follows this theoretical background, in the following sections the rigorous mathematical concept of the simplex splines method is provided in a very detail. In the first section 2.1 some basic definitions that are important in the simplex splines theory are given. As mentioned before, the simplex spline function is always defined on a structure called the triangulation. In the section 2.2 some types of triangulations are discussed. The triangulation introduces a continuity problem among neighbouring polynomials. The solution in form of smoothness constraints is described in the section 2.3. The following section 2.4 of this chapter presents the linear regression scheme used for the interpolation of input datasets. And finally in the last section 2.5, the definitions of various differential operators are introduced.

Many examples that demonstrate an important behaviour discussed in the following sections can be found in the appendix A of this thesis.

2.1 The preliminaries of the simplex splines

This section provides basic mathematical background of the simplex splines theory. In this place, the concepts such as, the simplex in the section 2.1.1 or barycentric coordinates in the section 2.1.2 are discussed in detail. The idea of the barycentric coordinates is used with Bernstein polynomials 2.1.3. Bernstein polynomials are often and with multivariate simplex splines always denoted in the B-form as can be seen in the section 2.1.4. A uniform notation for the splines of a given polynomial degree and continuity order is defined in a form of spline spaces 2.1.6. The coefficients of a Bernstein polynomial form a spatial structure called the B-coefficient net or shortly B-net. The shape, orientation and other properties of B-nets have a strong impact on a resulting simplex spline function. Therefore, in the final section 2.1.5 of the chapter, the topic of B-nets is discussed.

2.1.1 The simplex

In geometry, a simplex is a generalization of the notion of a triangle or tetrahedron to an arbitrary dimension. Specifically, a n-simplex is a special type of n-dimensional polytopes, which is a convex hull of its $n+1$ vertices. For example, a 2-simplex is a triangle, a 3-simplex is a tetrahedron, and a 4-simplex is a pentachoron. A single point may be considered as a 0-simplex and a line segment may be considered as a 1-simplex.

The mathematical definition of the n-simplex starts with a single point or vertex in n-dimensional Euclidean space:

$$\mathbf{v}_i = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n, \tag{2.1}$$

where $i \in \mathbb{N}$ is a vertex index, and x_i is a coordinate component in n-dimensional space. The vertex index uniquely identifies each vertex within the set of vertices of the simplex. These vertices need to be lexicographically ordered. Let $\mathcal{V}_t$ be a tuple consisting of $n+1$ vertices:

$$\mathcal{V}_t = (\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n) \in \mathbb{R}^n. \tag{2.2}$$

The n-simplex t can be now defined as a *convex hull*:

$$t = \langle \mathcal{V}_t \rangle \in \mathbb{R}^n. \tag{2.3}$$

The convex hull of any *non-empty* subset of the $n+1$ points that define an n-simplex is called a *face* of the simplex. Faces are simplices themselves. Let $\mathcal{V}_{t_1}$ be any subset of $\mathcal{V}_t$ having the size $m+1$. Then from (2.3), the convex hull of $\mathcal{V}_{t_1}$ is following:

$$t_1 = \langle \mathcal{V}_{t_1} \rangle \in \mathbb{R}^{m+1}, m \geq 0. \tag{2.4}$$

Dimension n	0-faces	1-faces	2-faces	3-faces	4-faces	5-faces	6-faces	7-faces
0	1	0	0	0	0	0	0	0
1	2	1	0	0	0	0	0	0
2	3	3	1	0	0	0	0	0
3	4	6	4	1	0	0	0	0
4	5	10	10	5	1	0	0	0
5	6	15	20	15	6	1	0	0
6	7	21	35	35	21	7	1	0
7	8	28	56	70	56	28	8	1

Table 2.1: The number of faces within different simplices. Notice the structure of the Pascal's triangle.

The simplex t_1 is called m-face of n-simplex t. The 0-faces are called *vertices*, the 1-faces are called *edges*, the $(n-1)$-faces are called *facets*, and the sole n-face is the whole n-simplex itself.

This fundamental property means, that simplices have a recursive structure. Moreover, the total number of m-faces of the n-simplex can be found in the $(m+1)^{th}$ column of the $(n+1)^{th}$ row of the Pascal's triangle. Consequently, the number of m-faces is equal to the binomial coefficient $\binom{n+1}{m+1}$. So for example, the triangle (2-simplex) contains three lines (1-simplex) and three vertices (0-simplex). Take a look at the table 2.1. One important face of the n-simplex t, that has a special meaning not only in the simplex splines theory is the $(n-1)$-face, usually called the *edge facet* and denoted as $\tilde{t}$. It should be clear from the table 2.1, that any n-simplex has $n+1$ edge facets. Therefore:

$$\tilde{t}_i = \langle \mathcal{V}_t \setminus \mathbf{v}_i \rangle \in \mathbb{R}^n, \; i = 0, 1, \ldots, n. \tag{2.5}$$

The equation (2.5) states, that any edge facet $\tilde{t}_i$ can be implicitly defined in terms of the single vertex $\mathbf{v}_i$ of the n-simplex t which is not in $\tilde{t}_i$. This observation will be important in the definition of the continuity conditions between simplex polynomials on neighbouring simplices. In the figure 2.1 are pictured projections of the first 15 simplices. These simplices are regular i.e. all edge facets are of the same size.

2.1.2 The barycentric coordinates

The barycentric coordinates is a brand new concept of a coordinate system introduced by August Ferdinand Möbius (1827), in which the location of a point is specified as the center of a mass, or a barycenter of masses placed at the vertices of a given simplex. The most common system recognized today is the Cartesian coordinate system that expresses a point's location on the plane relative to the origin – in 2D to the $(0, 0)$. The barycentric coordinates despite of that express the point's location relatively to some other points on the plane. Möbius in his work proved that any coordinate in a 2-dimensional or a 3-dimensional Euclidean space can be described by barycentric coordinates. Furthermore, he found out that the new coordinates system is homogeneous, which among other things means that it is invariant to a scalar multiplication.

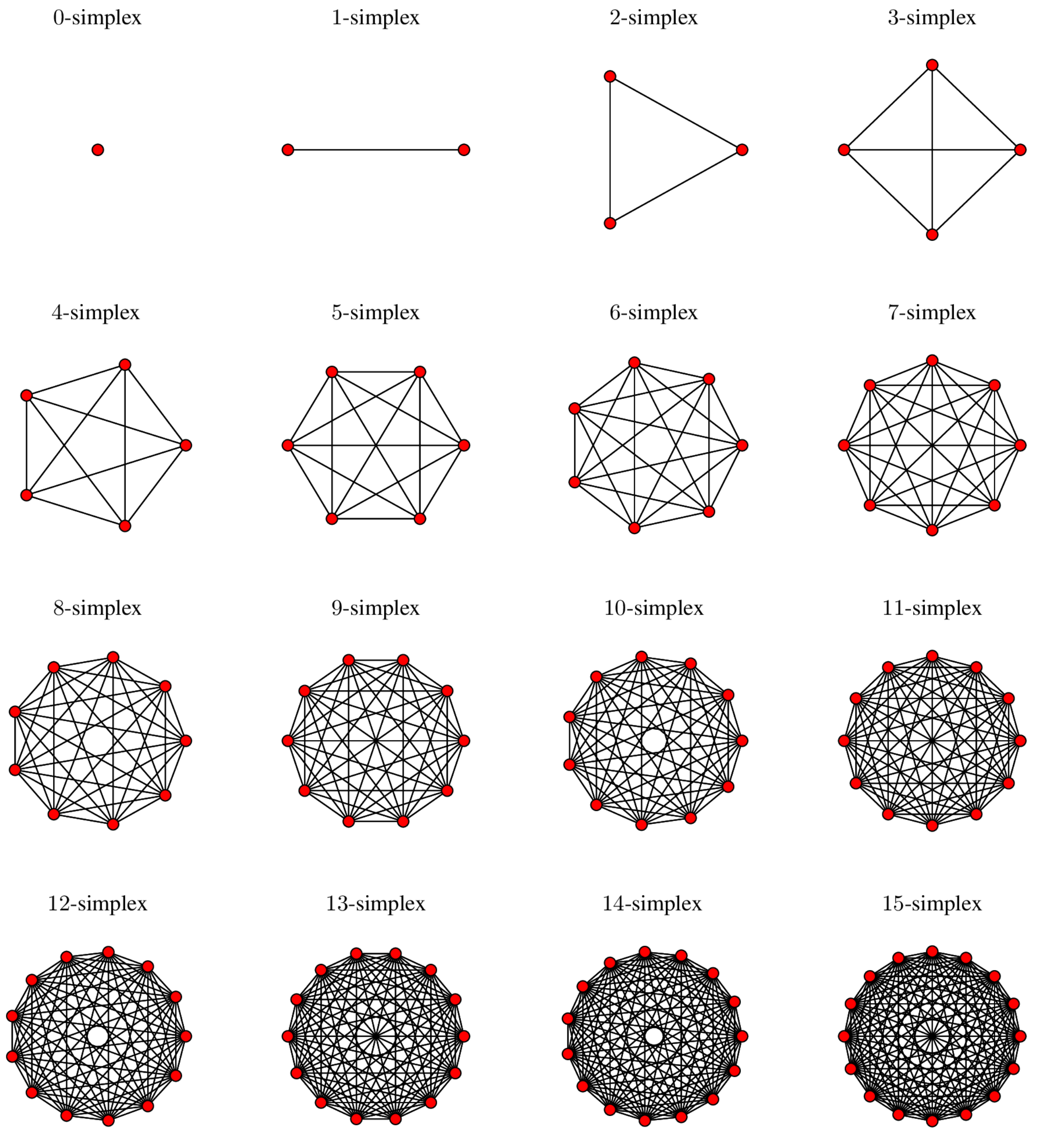


Figure 2.1: The projections on the 2-plane of the regular simplices of dimension 0 to 15. This figure was taken from [9, page 41].

As mentioned before, Möbius was concerned only about the 2 and the 3 dimensional space, but the generalization of his idea can be easily made for an n-dimensional space. As this is one of the basic topics in the simplex splines theory, in the following lines the mathematical background of the general barycentric coordinates is provided along with an example where the important behaviour is discussed.

The derivation of barycentric coordinates as proposed in [9] starts with a point $\mathbf{x}$ in n-dimensional Euclidean space:

$$\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n. \tag{2.6}$$

To express point $\mathbf{x}$ relative to the vertices of the n-simplex the expression (2.1) can be used as follows:

$$\mathbf{x} = \sum_{i=0}^{n} b_i \mathbf{v}_i, \tag{2.7}$$

with $\mathbf{v}_i \in \mathbb{R}^n$ the simplex vertex and $b_i \in \mathbb{R}$ scalar weights. The barymetric coordinate of the point $\mathbf{x}$ is the vector of vertex weights $\mathbf{b}$ given by:

$$\mathbf{b} = (b_0, b_1, \ldots, b_n) \in \mathbb{R}^{n+1}. \tag{2.8}$$

Notice please, that according to this definition, the barycentric coordinate of n-dimensional point is a vector of $n+1$ elements. In the following, the normalization property of $\mathbf{b}$ is defined:

$$\sum_{i=0}^{n} b_i = 1. \tag{2.9}$$

The example A.1 shows that weights b_i represent fractions of the area[1] of n-simplex, that is according to (2.9), the area of whole n-simplex is equal to 1. Now, the first component of $\mathbf{b}$ can be expressed in terms of the remaining components:

$$b_0 = 1 - \sum_{i=1}^{n} b_i. \tag{2.10}$$

The explicit form of the Cartesian-to-barycentric transformation can be given now.

$$\begin{aligned}
\mathbf{x} &= \sum_{i=0}^{n} b_i \mathbf{v}_i \\
&= b_0 \mathbf{v}_0 + \sum_{i=1}^{n} b_i \mathbf{v}_i \\
&= \left(1 - \sum_{i=1}^{n} b_i\right) \mathbf{v}_0 + \sum_{i=1}^{n} b_i \mathbf{v}_i \\
&= \mathbf{v}_0 + \sum_{i=1}^{n} (b_i \mathbf{v}_i - b_i \mathbf{v}_0) \\
&= \mathbf{v}_0 + \sum_{i=1}^{n} b_i (\mathbf{v}_i - \mathbf{v}_0) \\
\mathbf{x} - \mathbf{v}_0 &= \sum_{i=1}^{n} b_i (\mathbf{v}_i - \mathbf{v}_0) \qquad (2.11)
\end{aligned}$$

[1]The term 'area' is connected only with a 2-simplex of course. In 3D it is volume, etc.

The right hand side of the last equation can be rewritten to the form of vector multiplication:

$$\mathbf{x} - \mathbf{v}_0 = [\mathbf{v}_1 - \mathbf{v}_0 \quad \mathbf{v}_2 - \mathbf{v}_0 \quad \cdots \quad \mathbf{v}_n - \mathbf{v}_0] \cdot \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}. \tag{2.12}$$

The first factor of the multiplication contains vertex coordinates of n-simplex normalized by the coordinates of the first vertex. This useful entity is called a *normalized simplex vertex matrix* and is denoted as $\mathbf{A}_t$ with t a given simplex. Now, the expression (2.12) can be simplified by the substitution of the normalized simplex vertex matrix:

$$\mathbf{x} - \mathbf{v}_0 = \mathbf{A}_t \cdot \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}. \tag{2.13}$$

And finally, the vector of barycentric coordinates is given by:

$$\mathbf{A}_t^{-1} \cdot (\mathbf{x} - \mathbf{v}_0) = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}. \tag{2.14}$$

The matrix $\mathbf{A}_t$ must be invertible, which is the case when $\mathbf{v}_i - \mathbf{v}_0$ are linearly independent. If this condition is satisfied, the Cartesian-to-barycentric coordinate system transformation is a linear, one-to-one transformation. This means, that any Cartesian coordinate in the Euclidean n-plane has a unique barycentric coordinate with respect to a given n-simplex. Moreover, it is possible to determine the position of points with respect to a given n-simplex t. Let's take a look at the following expressions:

$$\forall i : i = 0 \ldots n, \; b_i \geq 0 \longrightarrow \mathbf{x} \in t \tag{2.15}$$

$$\exists i : i = 0 \ldots n, \; b_i < 0 \longrightarrow \mathbf{x} \notin t \tag{2.16}$$

Now one of the key features provided by the barycentric coordinate system may be seen. If a point is located within a simplex, all its barycentric coordinate components with respect to that simplex are greater or equal to zero. On the other hand, if at least one of the barycentric coordinate component is less than zero, point is located outside of a simplex.

In the real-world applications, the input dataset of points that is being modelled e.g. terrain dataset, is always defined in the Cartesian coordinate system and as the simplex B-form polynomials 2.1.4 are defined in terms of the barycentric coordinates, the input points need to be transformed to the barycentric concept. This transportation is the first step of the implemented algorithm. For better understanding, the behaviour of barycentric coordinates is demonstrated by the example A.1.

2.1.3 The Bernstein basis polynomials

In the next few sections the mathematical concept of Bernstein polynomials will be described. Bernstein polynomials are used as an interpolation function on every single simplex.

Maybe not clear so far, however, a simplex spline function consists of a set of neighbouring simplices on each of which a simplex polynomial is defined. The simplex polynomial is a Bernstein polynomial, and therefore consists of *Bernstein basis polynomials.*

The theory of Bernstein polynomials as an approximation method is in general well known, however, in his paper [9, section 2.2.3] C.C. Visser proposes the derivation of Bernstein polynomials from *multinomial theorem*, which will turn out to be essential in a definition of the multivariate simplex spline function.

In the beginning, the general form of the multinomial theorem is given:

$$(a_0 + a_1 + \cdots + a_n)^d = \sum_{k_0+k_1+\cdots+k_n=d} \frac{d!}{k_0!k_1!\cdots k_n!} a_0^{k_0} a_1^{k_1} \cdots a_n^{k_n} \tag{2.17}$$

Let $\mathbf{b} = (b_0, b_1, \ldots, b_n)$ be a barycentric coordinate in $\mathbb{R}^{n+1}$. Then from (2.17) a polynomial of degree $d \in \mathbb{N}$ can be defined as follows:

$$(b_0 + b_1 + \cdots + b_n)^d = \sum_{k_0+k_1+\cdots+k_n=d} \frac{d!}{k_0!k_1!\cdots k_n!} b_0^{k_0} b_1^{k_1} \cdots b_n^{k_n}. \tag{2.18}$$

This expression may be simplified by introduction of a new entity called *multi-index*. The multi-index k of dimension n is a tuple defined as follows:

$$k = (k_0, k_1, \ldots, k_n) \in \mathbb{N}^{n+1}. \tag{2.19}$$

The 1-norm and the factorial of multi-index k is then:

$$|k| = k_0 + k_1 + \cdots + k_n = d, \quad d \geq 0, \tag{2.20}$$

$$k! = k_0!k_1!\cdots k_n!. \tag{2.21}$$

Using equations above the multinomial form from (2.18) may be simlified:

$$(b_0 + b_1 + \cdots + b_n)^d = \sum_{|k|=d} \frac{d!}{k!} \mathbf{b}^k \tag{2.22}$$

The number of polynomial terms in the sum is determined by the number of *valid permutations* of k. The total number of valid permutations of k for a given dimension n and polynomial degree d is $\hat{d}$ defined as follows:

$$\hat{d} = \binom{d+n}{d} = \frac{(d+n)!}{n!d!}. \tag{2.23}$$

This expression has been proposed in [6]. The total number of valid permutations $\hat{d}$ also determines the number of free variables of the polynomial. The number $\hat{d}$ has a very special meaning in the multivariate splines theory as almost all important constants or matrices are defined by this number.

Everything is in place to define the form of Bernstein basis polynomial:

$$B_k^d(\mathbf{b}) = \frac{d!}{k!} \mathbf{b}^k \tag{2.24}$$

and further the (2.22) may be simplified as:

$$(b_0 + b_1 + \cdots + b_n)^d = \sum_{|k|=d} B_k^d(\mathbf{b}), \tag{2.25}$$

where the right hand side sum defines a Bernstein polynomial. As was mentioned before, the left hand side of (2.25) represents a barycentric coordinate, which according to normalization property from (2.9) is equal to one. This gives us the final expression of this part:

$$1 = \sum_{|k|=d} \frac{d!}{k!}\mathbf{b}^k. \tag{2.26}$$

In words, the total sum of all Bernstein basis polynomials is equal to one. And while the Bernstein polynomial is used for interpolation of given n-simplex, this outcome is applied at every location within an n-simplex.

The main point of this section was to derive the Bernstein polynomial from the multinomial theorem. The expression from (2.22) will be used in next section, where the B-form is introduced.

2.1.4 The B-form

The Bernstein basis polynomials of degree d form a basis for the space of polynomials of degree less than or equal to d. In other words, Bernstein basis polynomials form a vector space of polynomials of a given degree. This in turn means, that all Bernstein basis polynomials of a degree d are linearly independent and therefore, any polynomial of the degree d may be expressed as a unique linear combination of Bernstein basis polynomials. The proof of this theorem was provided by Carl de Boor in 1986 [6]. The B-form provides a convenient notation for the linear combination of Bernstein basis functions on a single n-simplex.

$$p(\mathbf{x}) = \sum_{|k|=d} c_k B_k^d(b(\mathbf{x})), \tag{2.27}$$

with $p(x)$ an arbitrary polynomial, $b(\mathbf{x}) = \mathbf{b}$ the implicit transformation from the Cartesian to barycentric coordinate system, and most importantly with c_k control coefficients or *B-coefficients*. The B-coefficients control the shape of a B-form polynomial by scaling the individual basis functions. And consequently, there are as much B-coefficients as there are basis polynomial terms. According to (2.23), the number of them is equal to $\hat{d}$. The example A.2 demonstrates the simple trivariate B-form polynomial along with the visualization.

2.1.5 The B-coefficient net

The spatial location of B-coefficients within a simplex is not random, but forms a structure called B-*coefficient net* or shortly the B-*net*. The concept of B-net was widely recognized in literature e.g. see [11, 6, 14]. The location in barycentric coordinates of a given B-coefficient c_k within an n-simplex is given by:

$$\mathbf{b} = \frac{k}{d}, \quad |k| = d, \tag{2.28}$$

which is also equal to the location of the maximum of the associated Bernstein basis polynomial $B^d_{|k|=d}(\mathbf{b})$. In the figure 2.2 the spatial location of the B-coefficients is shown. Notice, that according to the number of the B-coefficients from (2.23), the density of the B-net increases with higher polynomial degree. The structure of B-coefficients has an important meaning in the theory of the simplex splines as smoothness constraints between neighbouring polynomials are defined on the B-net. This will be discussed later in the section 2.3, however, to make an effective algorithmic implementation of the multivariate simplex splines function, the B-coefficients and the corresponding B-net need to be ordered in some way. Therefore, the formal orientation rule will be introduced now.

The orientation of the B-net within a simplex depends mainly on the indices of the B-coefficients located at the simplex vertices. The vertices of all simplices in a triangulation need to have a globally unique index. After that, every simplex can be defined by vertices that are lexicographically sorted. De Visser in [9, definition 3] introduces a formal definition of the orientation rule that will be repeated here. An simple example can be found in the appendix A.3.

Definition 1 (The B-net Orientation Rule) *Let every n-simplex t consist of a tuple $\mathcal{V}_t$ of $(n+1)$ globally indexed vertices. The vertices in the set are ordered using their indices. A new set is now introduced which consists of tuples of vertices and B-coefficients as follows:*

$$\mathcal{V}_{\mathcal{B}} = \{(c_{d,0,0,\dots,0}, \mathbf{v}_{p_0}), (c_{0,d,0,\dots,0}, \mathbf{v}_{p_1}), \dots, (c_{0,0,0,\dots,d}, \mathbf{v}_{p_n})\}, \tag{2.29}$$

with $p_0 > p_1 > p_n$ the ordered indices. After the B-coefficients located at simplex vertices are known, the other B-coefficients are calculated by the expression (2.28).

2.1.6 The spline spaces

An important notation that generalize a degree and a continuity order of a simplex spline function is called a *spline space*. Formally, the spline space is a space of all spline functions s of a given polynomial degree d and continuity order C^r on a given triangulation $\mathcal{T}$ denoted as follows:

$$\mathcal{S}^r_d(\mathcal{T}) = \{s \in C^r(\mathcal{T}) : s|_t \in \mathbb{P}_d, \forall t \in \mathcal{T}\}, \tag{2.30}$$

with $\mathbb{P}_d$ the space of polynomials of degree d. For example, $\mathcal{S}^2_1(\mathcal{T})$ is the space of all quadratic spline functions with continuity order C^1 defined on the triangulation $\mathcal{T}$.

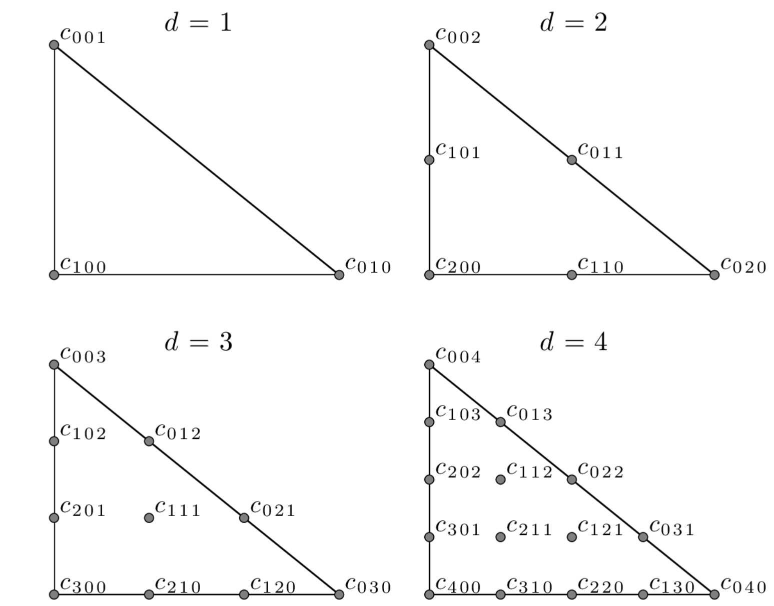


Figure 2.2: The spatial location of the B-coefficients within the simplex. In this case, the B-net is of the 2-dimensional simplex polynomial for various polynomial degrees.

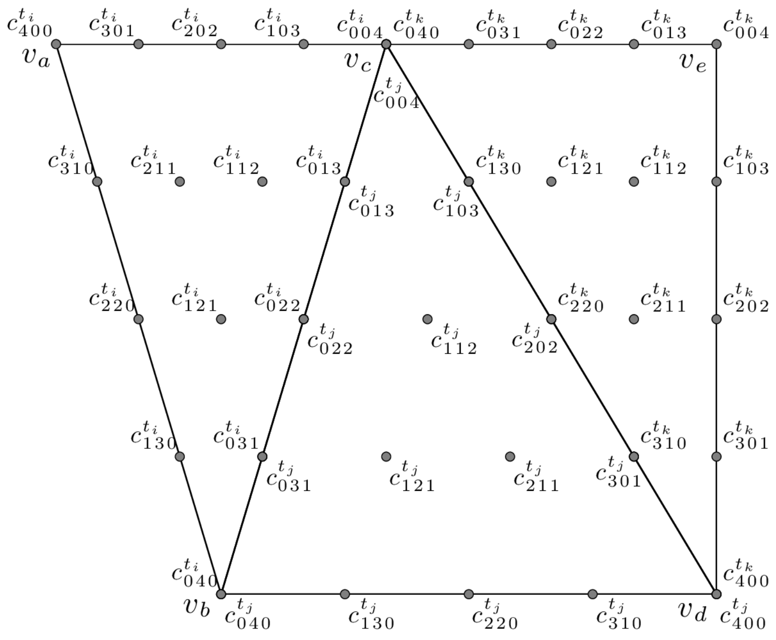


Figure 2.3: Spatial locations of the B-coefficients and the B-nets orientation on three simplices. The B-coefficients correspond to the 4th degree Bernstein polynomial.

2.2 The triangulations

So far we talked only about the interpolation with Bernstein polynomials on a single n-simplex. In the real world applications where many multi-dimensional scattered data need to be interpolated, one single simplex is not enough. The power of the simplex splines theory comes with the combination of neighbouring simplices on each of which a single simplex polynomial is defined. The combination of the simplices means also the combination of the simplex polynomials. The most important constraint that must be met by polynomials defined on the neighbouring simplices is the smoothness.

Triangulation gives a powerful method for subdividing n-dimensional objects into n-simplices. Some properties of triangulation are explained in next sections. Firstly in the section 2.2.1, the formal definition is provided. The simplest kinds of triangulation – regular triangulations are presented in the section 2.2.2. Then in the section 2.2.3, the common method of Delaunay triangulation is described.

2.2.1 Formal definition of the triangulation

In the most general meaning, the triangulation may be defined as a subdivision of geometric objects into simplices. In a 2-dimensional plane it is a subdivision into triangles, hence the name. In a formal definition, a triangulation $\mathcal{T}$ is a *set* consisting of J simplices defined as:

$$\mathcal{T} = \bigcup_{i=1}^{J} t_i,\ t_i \cap t_j \in \{\emptyset, \tilde{t}\},\ \forall (t_i, t_j) \in \mathcal{T},\ i \neq j \tag{2.31}$$

with $\tilde{t}$ a k-face of the n-simplices t_i and t_j, with $0 \leq k \leq n-1$. This expression was taken from literature [9, section 2.3.2]. Simplex $\tilde{t}$ also called *face* connects two or more n-simplices, and its dimension is lower than n. One special case of $\tilde{t}$ would be with $k = n-1$, when $\tilde{t}$ is called *edge facet* as discussed in the section 2.1.1. In this case, two neighbouring simplices share edge facet $\tilde{t}$, or in other words, $\tilde{t}$ connects two neighbouring simplices. Another special case of $\tilde{t}$ is with $k = 0$, in which $\tilde{t}$ is just a single vertex. The definition (2.31) also says that any two n-simplices of a triangulation are allowed to overlap only on their *faces* of dimension lower than n or not overlap at all.

Notice, that for a given set of points there may be more than just one possible triangulation and therefore triangulation have not to be necessarily unique. This gives us a need to measure how the given triangulation is „good“ or „bad“. The objective performance measure of the entire triangulation is required. One good approach is to count the ratio between the radius of the circumsphere and the shortest simplex ridge. This ratio is also called SRSC as a shortcut for *Ratio radius circumsphere and shortest ridge.* Without any more discussion the formula of SRSC is given as follows:

$$P_{\mathcal{T}} = \frac{1}{J} \sum_{i=1}^{J} \frac{r_{\Theta_j}}{\min |\mathbf{v}_u - \mathbf{v}_w|}, \tag{2.32}$$

with r_{Θ_j} the radius of the circum-hypersphere of the n-simplex t_j, with $\tilde{t}$ an edge facet of t_j and with $\mathbf{v}_u$ and $\mathbf{v}_w$ vertices in the edge facet $\tilde{t}$. There are also many other useful metrics but this topic goes beyond the content of this work.

2.2.2 The regular triangulations

The simplest forms of triangulations are so-called *the regular triangulations*. The construction of a regular triangulation is easy: every cell of the n-dimensional grid is filled with a regularly triangulated n-cube. The n-dimensional grid may or may not be uniform. In the second case the triangulated n-cube prototype has to be scaled or even rotated to fill the grid cell. This is shown by the figure 2.4 where two possible triangulations of a square are used. These triangulations are non-uniform but regular. The left hand side of the figure shows maybe the simplest triangulation possible while in the right hand side, the extra vertex is included to the middle of each cell. In the figure 2.5 the triangulated 3-cube is pictured. This triangulation consists of six tetrahedrons that are symmetric along the ridge of the 3-cube. All these tetrahedrons are of the same size and volume i.e. they are regular. In the regular 3-dimensional triangulation this cube can be used for filling individual grid cells.

A great advantage of the regular triangulation method, as was described above, is that it avoids the generation of badly defined simplices according to the SRSC metric from (2.32). As maybe not clear at first, the construction of a regular triangulation is not limited only for rectangular domains. For example, the cube from the figure 2.5 can be extended in any direction by attaching extra tetrahedrons of the same size as others.

One last note about the complexity of a implementation of a regular triangulation algorithm. In low dimensions the calculation of a triangulation is fast, and even faster than other kinds of triangulations e.g. Delaunay discussed in the next section. However, in higher dimensions, more than 8^{th}, the structure of a hypercube is becoming much more complex and computationally ineffective.

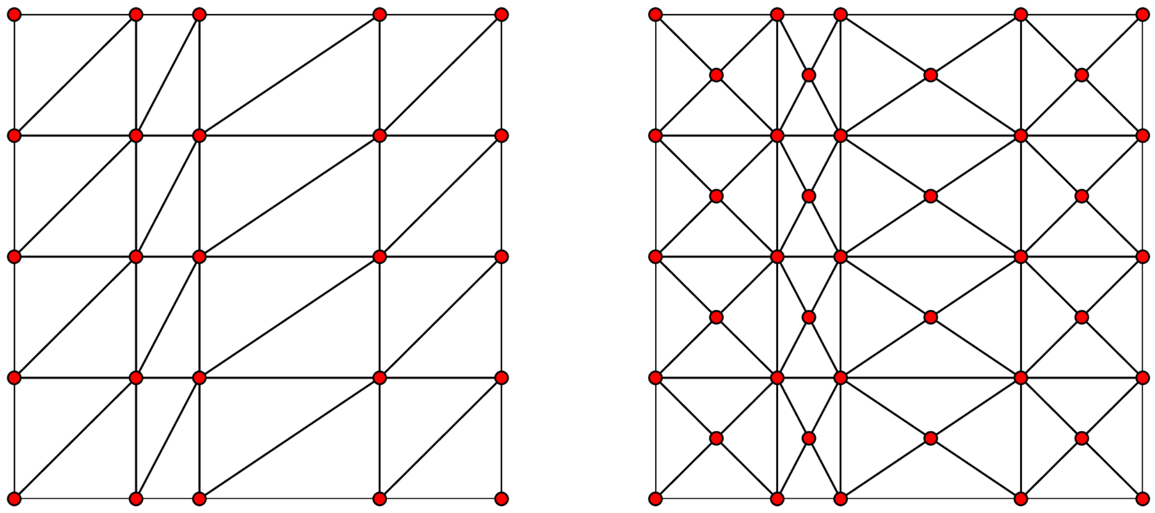


Figure 2.4: Non-uniform regular triangulations defined on a rectangular domain. Taken from [9, Section 2.3.3].

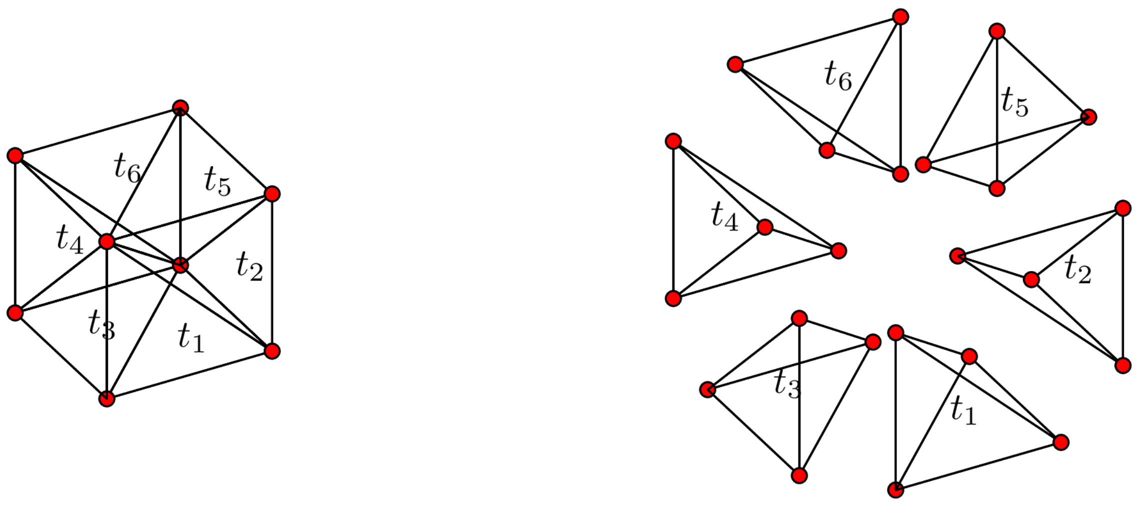


Figure 2.5: The regularly triangulated 3-cube (left) consisting of 6 tetrahedrons and its exploded view (right). Taken from [9, Section 2.3.3].

2.2.3 The Delaunay triangulations

Many methods and techniques were proposed through the years to calculate a triangulation of a discrete set of points. Every one of them has its own application but the most powerful seems to be *the Delaunay triangulation method* invented by a Russian mathematician Boris Delaunay in 1934. The most important advantage of the Delaunay triangulation is that for any set of discrete points the outcome triangulation can be counted if points are non-degenerate and therefore is suitable for using with simplex splines.

The formal definition of the Delaunay triangulation is usualy derived from a dual operation called the Voronoi tessellation. The complete proof may be found in literature [17]. In a valid Delaunay triangulation, the vertices of the n-simplex t_j must be located exactly on and only on the circum-hypersphere of t_j. This is called the *Delaunay condition* for simplices and says, if Θ_{t_k} is a circum-hypersphere of the n-simplex t_k, and $\mathcal{V}_j$ are the vertices of t_j, then

$$\mathcal{V}_j \in \Theta_{t_k} \iff j = k. \tag{2.33}$$

In the figure 2.6 the comparison of the Delaunay triangulation and the Voronoi tessellation is pictured. In the triangulation the original points are connected, while the lines in a Voronoi diagram are the equidistance points of two neighbouring vertices. The Voronoi diagram therefore provides a graphical depiction of the closest neighbour set of a set of vertices. This closest neighbour set can be used to create the Delaunay triangulation of the set of vertices by constructing simplices which each contain the most compact set of $n + 1$ vertices.

As can be seen, the Delaunay triangulation provides many important features and furthermore the triangulation algorithms are simple and fast with complexity $O(n \log n)$. The Delaunay triangulation method also tends to avoid bad defined „skinny“ simplices, but especially along the boundaries of the triangulation domain the problem still preserves and optimization techniques have to be used.

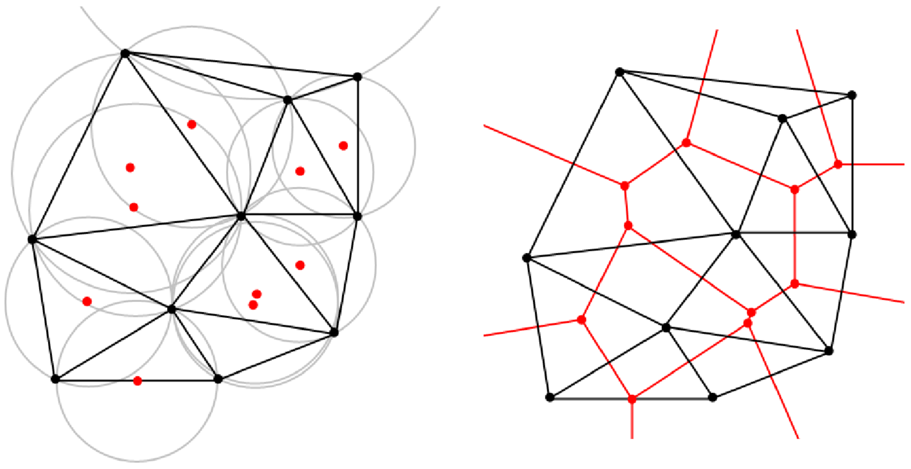


Figure 2.6: On the left hand side, the Delaunay triangulation with all the circumcirces and their centres in red color. Connecting the centres of the circumcircles produces the Voronoi diagram (in red) on the right hand side.

2.3 The smoothness constrains

With the triangulation the problem of continuity among neighbouring simplices is introduced. The simplex spline function is by definition a piecewise defined polynomial function with C^r continuity between its pieces. the smoothness constraints formed by continuity conditions somehow link B-coefficients of neighbouring polynomials. These conditions are then used as a linear equality constraint in a linear regression scheme.

This section presents mathematical formulation of the smoothness constraints and also some consequences of their use which might not be evident at first sight. In 1979 Gerald E. Farin in his paper [10] and later in [11] gave the statement of continuity of the polynomial pieces across an interface between two simplices. This topic as presented in the section 2.3.1 was later studied by many authors e.g. [14, 15, 3], however, de Visser found their formulation not general and introduced some modifications which are summarized in 2.3.2. All continuity conditions for all simplices of a triangulation are in the end included in a global smoothness matrix 2.3.3.

2.3.1 Definition of continuity conditions

The continuity between the polynomial pieces of the simplex spline function are enforced by special equations called the continuity conditions which are defined for every edge facet shared by two neighbouring simplices. An edge facet is as mentioned in the section 2.1.1 an $(n-1)$-simplex that defines the edge of the n-simplex. The number of edges of any n-simplex is equal to $n+1$. As long as simplices are joined in a triangulation, the neighbouring simplices share their edge facets between each other. As a result, any set of two joined simplices shares one and only one unique edge facet. Now, let t_i and t_j be two neighbouring

n-simplices defined as:

$$t_i = \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}, \mathbf{v}_{i,j} \rangle \,,$$
$$t_j = \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}, \mathbf{v}_{j,i} \rangle \,. \tag{2.34}$$

They share one edge facet $\tilde{t}_{i,j}$ given by:

$$\tilde{t}_{i,j} = t_i \cap t_j = \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1} \rangle \,. \tag{2.35}$$

It should be clear that $\tilde{t}_{i,j}$ is an $(n-1)$-simplex as it consists of n vertices (see table 2.1). Moreover, both simplices individually have one, and only one, unshared vertex, for t_i it is vertex $\mathbf{v}_{i,j}$ and for t_j it is $\mathbf{v}_{j,i}$ (note that indices are not same). These vertices are called *out of edge vertices*, as they do not participate on the edge facet. Formally are defined as:

$$\tilde{\mathbf{v}}_{i,j}, \tilde{\mathbf{v}}_{j,i} \notin \tilde{t}_{i,j}. \tag{2.36}$$

Having (2.36), the edge facet $\tilde{t}_{i,j}$ can be defined also indirectly by out of edge vertices $\mathbf{v}_{i,j}$ and $\mathbf{v}_{j,i}$ as following:

$$\tilde{t}_{i,j} = \begin{cases} t_i \setminus \mathbf{v}_{i,j} \\ t_j \setminus \mathbf{v}_{j,i} \end{cases} \tag{2.37}$$

An application of this observation will be shown in later section during the implementation part.

The main reason why it is so important to obtain continuity of a given order between the polynomial pieces of the spline function is a matching of derivatives. When directional derivatives up to order C^r between two B-form polynomials on neighboring simplices are equal at every point of the edge facet, the continuity of order C^r is achieved. Formally, given the two simplices t_i and t_j and their edge facet $\tilde{t}_{i,j}$ and given the two B-form polynomials p_{t_i} and p_{t_j} associated with t_i, t_j respectively, the following expression must hold for all points $\mathbf{x}$ in $\tilde{t}_{i,j}$:

$$D^m_{\mathbf{u}} p_{t_i}(b(\mathbf{x})) = D^m_{\mathbf{u}} p_{t_j}(b(\mathbf{x})), \; \forall \mathbf{x} \in \tilde{t}_{i,j}, \; m = 0, 1, \ldots, r \,, \tag{2.38}$$

with vector $\mathbf{u}$ a direction of derivative and $b(\mathbf{x})$ the transformation from Cartesian coordinate system to barycentric one. Equation (2.38) formally says what is meant by continuity between two polynomials but does not mention how this continuity would be achieved. As said before, smoothness constraints are expressed in a form of continuity conditions. Continuity conditions relates B-coefficients of neighbouring B-form polynomials. The most prevalent formulation of this „relation“ was provided by Awanou [3]:

$$c^{t_i}_{m,\ldots,k_n} = \sum_{|\gamma|=m} c^{t_j}_{(0,\ldots,k_n)+\gamma} B^m_\gamma(\tilde{\mathbf{v}}_{i,j}), \; 0 \le m \le r, \tag{2.39}$$

with $\gamma = (\gamma_0, \gamma_1, \ldots, \gamma_n) \in \mathbb{R}^n$ a multi-index independent of k. Note, that the basis function $B^m_\gamma(\tilde{\mathbf{v}}_{i,j})$ is a polynomial in terms of the barycentric coordinate $\tilde{\mathbf{v}}_{i,j}$ with respect to simplex t_j. In other words, location of the point $\tilde{\mathbf{v}}_{i,j}$ is not within the simplex t_j and therefore at least one of its barycentric coordinates is negative. The geometric interpretation of (2.39) for various continuity orders C^0, C^1, C^2 and C^3 is shown in the figure 2.7. It can be seen how subsimplices across the edge facets link B-coefficients of both sides. In the figure a dashed lines represent the left hand side of (2.39) and a solid lines the right hand side.

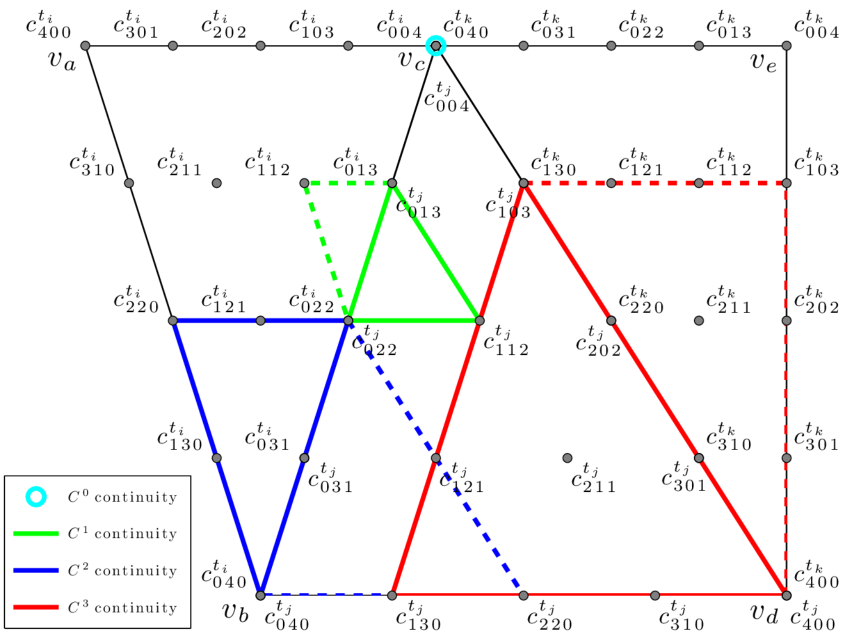


Figure 2.7: The spatial interpretation of the structure of the continuity showed for 4 different continuity orders. Three simplices t_i, t_j and t_k are covered by 4th degree B-net.

For example, the C^2 continuity condition between simplices t_j and t_i, plotted with blue, is formulated for the B-coefficient $c^{t_j}_{2,2,0}$ and involves six B-coefficients of the simplex t_i which are used in the right hand side of the expression (2.39). The single B-coefficient for which the continuity condition is formulated, in this case $c^{t_j}_{2,2,0}$, is called the *continuity point* B-coefficient and related B-coefficients of the simplex t_i are called *continuity body* as they form the polynomial body of the continuity condition. The higher is the continuity order the more B-coefficients are related together. For C^0 continuity only B-coefficients located at the edge facet are taken into the account. The figure 2.7 shows only parts of the continuity structures for given orders. Per entire edge between simplices t_j and t_i the full formulation of the C^2 continuity involves continuity points $c^{t_j}_{2,2,0}$, $c^{t_j}_{2,1,1}$, $c^{t_j}_{2,0,2}$ and continuity bodies related to them. It is clear, that for different continuity orders the number of continuity conditions is not same. In general, the total number of continuity conditions R_r for C^r continuity depends on the dimension n, polynomial degree d, and continuity order of a spline function and is given by following expression:

$$R_r = \sum_{m=0}^{r} \frac{(d-m+n-1)!}{(n-1)!(d-m)!}, \tag{2.40}$$

This equation was introduced and proved by de Visser in [9, Theorem 5, page 111]. As the concepts discussed above are very important for next sections some practical demonstrations are presented by examples A.4 and A.5.

The formulation of the continuity conditions from (2.39) is valid as long as the orientation of the B-nets of two neighbouring simplices is very specific, which is the orientation of the B-nets of t_i and t_j from the figure 2.7. This kind of orientation is called *maximum degree* *symmetric orientation* and requires, that the B-coefficient with highest lexicographical sorting order, i.e. $c_{d,0,\ldots,0}$, should be located at the out of edge vertex. The problem raises due to the fixed position of the 0 and m constants within the multi-indices of the B-coefficients in (2.39). An incompleteness of the expression (2.39) is demonstrated by the example A.6.

2.3.2 The reformulation of continuity conditions

As was shown in the previous section, the formulation of the continuity conditions from (2.39) is not general, and works only in some special situations which are usually not achievable. The main problem is about the indexing scheme of B-coefficients that produces invalid results. Therefore some new rules need to be introduced to this indexing scheme. De Visser [9] provides one solution that is relatively easy to implement and works in any number of dimensions. This new formulation of continuity conditions is based on insights gained from the graphical interpretation of the continuity conditions from the figures 2.7 and A.4.

In the beginning some observations should be noticed. First of all, it can be seen from the figure 2.7 that for a globally indexed B-net, the location of the constant in the multi-index (i.e. the m and 0) need to be the same as the location of the single non-zero value in the multi-index of the B-coefficient located at the out of edge vertex. In (2.39) the location of the constants m and 0 in multi-indices is fixed at the first position and for B-coefficients $c^{t_i}_{4,0,0}$, and $c^{t_j}_{4,0,0}$ associated with out of edge vertices the single non-zero value is also located at the first position. To determine the location of the non-zero value in the multi-index of the B-coefficient at the out-of-edge vertex the index function based on the global vertex indexation is introduced as follows:

$$\rho(\mathbf{v}_i) = (n+2) - \sum_{j=0}^{i} k_j, \; \begin{cases} k_j = 1 \text{ if } \mathbf{v}_j \in \mathcal{V}_t \\ k_j = 0 \text{ if } \mathbf{v}_j \notin \mathcal{V}_t \end{cases} \tag{2.41}$$

This expression simply gives the order of the vertex within simplex set $\mathcal{V}_t$ with respect to global vertex index. Now, when there is a way to determine the position of the single non-zero value in multi-indices the tuple function $\mathcal{M}(w,k)$ can be defined as follows:

$$\mathcal{M}(m,k) = \begin{pmatrix} (w, k_0, k_1, \ldots, k_{n-1}), \\ (k_0, w, k_1, \ldots, k_{n-1}), \\ (k_0, k_1, \ldots, w, k_{n-1}), \\ (k_0, k_1, \ldots, k_{n-1}, w) \end{pmatrix}, \; |\mathcal{M}_i(w,k)| = d - m + w \tag{2.42}$$

where the i^{th} multi-index is given by $\mathcal{M}_i(w,k)$. Letting $w = m$, $\mathcal{M}(w,k)$ produces all posible multi-indices of the continuity point B-coefficients, while if $w = 0$ then $\mathcal{M}(w,k) + \gamma$ produces all possible multi-indices of the continuity body B-coefficients. De Visser in [9, Example 14, page 116] provides some step-by-step examples which demonstrate how are multi-indices actually generated. In this thesis a practical algorithm 1 is proposed that can be directly implemented, however, one more definition or re-definition of the expression (2.39) needs to take a place:

$$c^{t_i}_{\mathcal{M}_{\rho(\mathbf{v}_{i,j})}(m,k)} = \sum_{|\gamma|=m} c^{t_j}_{\left(\mathcal{M}_{\rho(\mathbf{v}_{j,i})}(0,k)+\gamma\right)} B^m_\gamma(\mathbf{v}_{i,j}), \; 0 \leq m \leq r, \tag{2.43}$$

with C^r continuity between simplices, $\mathcal{M}_{\rho(\bullet)}(\bullet)$ the tuple function from (2.42) and $B_\gamma^m(\mathbf{v}_{i,j})$ the Bernstein basis polynomial in terms of the barycentric coordinate of $\mathbf{v}_{i,j}$ with respect to t_j. The proof of this expression can be found in [9, Theorem 6, page 118].

2.3.3 The estimation of the smoothness matrix

The algorithm 1 implements an estimation of the so-called smoothness matrix. The smoothness matrix contains all continuity conditions for all edge facets shared by the simplex pairs in a triangulation. Each row of the smoothness matrix $\mathbf{H}$ represents a single continuity condition from (2.43) equated to zero:

$$-\,c^{t_i}_{\mathcal{M}_{\rho(\mathbf{v}_{i,j})}(m,k)} + \sum_{|\gamma|=m} c^{t_j}_{\left(\mathcal{M}_{\rho(\mathbf{v}_{j,i})}(0,k)+\gamma\right)} B_\gamma^m(\mathbf{v}_{i,j}) = 0, \quad 0 \le m \le r. \tag{2.44}$$

Formulation of all continuity conditions in a matrix form leads to the following expression:

$$\mathbf{H}\mathbf{c} = \mathbf{0}. \tag{2.45}$$

The dimension of the smoothness matrix is:

$$\mathbf{H} \in \mathbb{R}^{(E \cdot R_r) \times (J \cdot \hat{d})}, \tag{2.46}$$

with E the total number of shared facets in a triangulation, with J the number of simplices, with R_r from (2.40) and $\hat{d}$ from (2.23).

An estimation of the smoothness matrix is proposed by the algorithm 1. In the following, this algorithm will be discussed step by step. The input of the algorithm is a valid triangulation $\mathcal{T} \in \mathbb{R}^{J \times n}$and a continuity order C^r. It is assumed that every row in the triangulation $\mathcal{T}$ represents a tuple of vertices of a single simplex. These vertices are ordered according to a global indexation rule.

The algorithm 1 starts with the allocation of the smoothness matrix $\mathbf{H}$, the dimensions of which are specified by (2.46). After that, on the line 2 of the algorithm, the function *edgeAttachments* is called. This function finds all possible simplex edges in the triangulation $\mathcal{T}$ that are shared by two neighbouring simplices. These simplex pairs are then returned and saved in the variable $\mathcal{A}_\mathcal{T}$. In the next part of the algorithm, the continuity conditions are calculated per-continuity order i.e. continuity conditions C^0 are calculated for all edge facets, then the continuity conditions of order C^1 again for all edges, etc. For that purpose, two consecutive loops are used. The first one listed on the line 3 is running through the continuity orders, while the second one on the line 4 iterates over all simplex pairs in the set $\mathcal{A}_\mathcal{T}$. Now, the continuity conditions of a given order m for an edge facet between simplices t_i and t_j need to be calculated. Firstly, the out-of-edge vertex $\mathbf{v}_{i,j}$ is found. This is done by the function $outofedgeVertex(t_i, t_j)$ listed on the line 5. This function simply applies the symmetric set difference operator on an input simplex pair. The symmetric operator Δ of two sets A, B is defined as follows:

$$A \,\Delta\, B = (A \setminus B) \cup (B \setminus A). \tag{2.47}$$

The notation of out-of-edge vertices is the same as in (2.37). The simplex pair defines two out-of-edge vertices, one for every simplex, so the result is a pair of the vertices.

Algorithm 1 The estimation of the smoothness matrix $\mathbf{H}$

```
Require: $\mathcal{T}, C^r$
 1: $\mathbf{H} \in \mathbb{R}^{(E \cdot R_r) \times (J \cdot \hat{d})}$
 2: $\mathcal{A}_{\mathcal{T}} = \text{edgeAttachments}(\mathcal{T})$
 3: for $m = 0 \ldots r$ do
 4:     for all simplex pairs $p = (t_i, t_j) \in \mathcal{A}_{\mathcal{T}}$ do
 5:         $(\mathbf{v}_{i,j}, \mathbf{v}_{j,i}) = \text{outofedgeVertices}(t_i, t_j)$
 6:         $\mathcal{C}_{point} = c^{t_i}_{\mathcal{M}_{\rho(\mathbf{v}_{i,j})}(m,k)}, \; |\mathcal{M}_{\rho(\mathbf{v}_{i,j})}(m,k)| = d$
 7:         $\mathcal{C}_{body} = c^{t_j}_{\mathcal{M}_{\rho(\mathbf{v}_{j,i})}(0,k)+\gamma}, \; |\mathcal{M}_{\rho(\mathbf{v}_{j,i})}(0,k)| = d - m, \; |\gamma| = m$
 8:         $\mathbf{h} \in \mathbb{R}^{J \cdot \hat{d}}$
 9:         for all continuity point $cp \in \mathcal{C}_{point}$ do
10:             Get associated continuity body $cb \in \mathcal{C}_{body}$
11:             Find index in $\mathbf{H}$ matrix for $cp$ and indices for $cb$
12:             $\mathbf{h}[i_{cpoint}] = -1$
13:             $\mathbf{h}[i_{cbody}] = [B^m_\gamma(\mathbf{v}_{i,j})]$
14:         end for
15:         $\mathbf{H}(\text{end}) = \mathbf{h}$
16:     end for
17: end for
```

Everything is in order to calculate the continuity conditions as were defined in (2.43). A note must be made at this point as the values of B-coefficients are not know yet. The smoothness matrix does not contain B-coefficients values but rather the multipliers based on the continuity conditions, see (2.45). Therefore, only the location of every B-coefficient within the globally indexed B-net is needed. As long as the orientation rule from the section 2.1.5 is used, the location of every B-coefficient is unambiguously determined by its multi-index. This index is then used in the smoothness matrix. In each row of the smoothness matrix $\mathbf{H}$, every single B-coefficient of a global B-net is represented exactly by one element. The dimensionality of a single row of the smoothness matrix then is $\mathbf{h} \in \mathbb{R}^{J \cdot \hat{d}}$.

On the line 6 of the algorithm, the B-coefficients of the continuity point are estimated. The tuple function from (2.42) together with the index function (2.41) will generate all possible valid multi-indices. On the next line the situation is quite different. The tuple and the index function are also used but they do not produce directly valid multi-indices of the continuity body B-coefficients, and they even can not, as the sum of all elements of the multi-index is in this case equal to $(d - m)$. By the definition from (2.20), the 1-norm of the multi-index must be equal to the polynomial degree d. Therefore, an independent multi-index permutation γ is introduced. The number of the continuity body B-coefficients for a single continuity point depends directly on the total number of valid permutations $|\gamma|$.

Having both, the continuity point B-coefficients as well as the continuity body B-coefficients, the corresponding position in the row vector $\mathbf{h}$ can be found. From (2.44) can be observed that the continuity point is a negative which corresponds to the line 12 of the algorithm. On the line 13, the continuity body B-coefficients are scaled by Bernstein basis polynomial or vice versa. This corresponds to the right hand side of the (2.43). The polynomial order of the Bernstein polynomial is equal to the actual continuity order m and is defined in

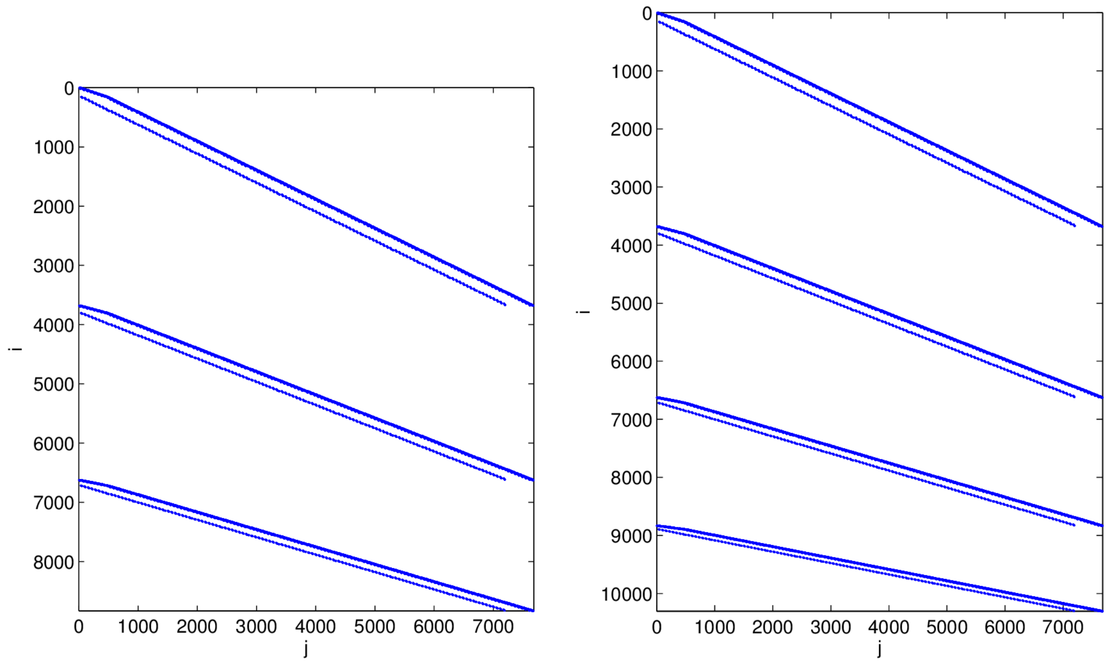
$\mathcal{C}^r$	conditions	non-zero elem.	elements $\times 10^6$	spareness
$\mathcal{C}^0$	3 680	7 360	≈ 28	97.4 %
$\mathcal{C}^1$	6 624	17 216	≈ 51	96.6 %
$\mathcal{C}^2$	8 832	28 352	≈ 68	95.8 %
$\mathcal{C}^3$	10 304	38 784	≈ 79	95.1 %

Table 2.2: Some important characteristics of the smoothness matrix. The spline space assumed in all four cases is $\mathcal{S}_4^r(\mathcal{T}_{512})$

terms of the barycentric coordinate $\mathbf{v}_{i,j}$ with respect to the simplex t_j. After all, the row vector $\mathbf{h}$ is pushed into the end of the resulting smoothness matrix $\mathbf{H}$.

In the figure 2.8 the smoothness matrices for C^2 and C^3 continuities of the spline function of the polynomial degree 4 are plotted. In these graphs every non-zero element of the smoothness matrix is visualized by a blue dot. The stripes from the top to the bottom correspond to the individual continuity conditions of a given order. The left hand side figure 2.8(a) shows three big stripes. The first at the top represents the continuity contritions of the order C^0, the second one in the middle is for continuity order C^1, and the third one for the order C^3. Each stripe consists of two lines. This is caused by the index distance between the simplices in a row of the smoothness matrix. The right hand side of the figure 2.8 shows the same situation but now the continuity conditions are defined up to the order C^3 and therefore four big stripes are present. As can be seen in the table 2.2, the spareness factor of the smoothness matrix is very high and is slightly decreasing with higher continuity order. In the table also some other important values are shown. From the left it is the total number of continuity condition i.e. the number of rows of the smoothness matrix, then the number of non-zero elements of the smoothness matrix, and finally, the total number of all elements in the smoothness matrix, which is in these cases equal to tens of millions. After the smoothness matrix is known, it is used as a linear equality constraint in linear regression scheme, therefore $\mathbf{H}$ needs to be of full rank. As Lai and Schumaker observed in [15] the rank deficiency of $\mathbf{H}$ is caused by some redundant continuity condition that are easily produced even for small and simple triangulations and low continuity orders. However, the construction of a smoothness matrix of a full rank is not a trivial task. Especially during the implementation part this step would be critical, therefore some optimizations are needed.

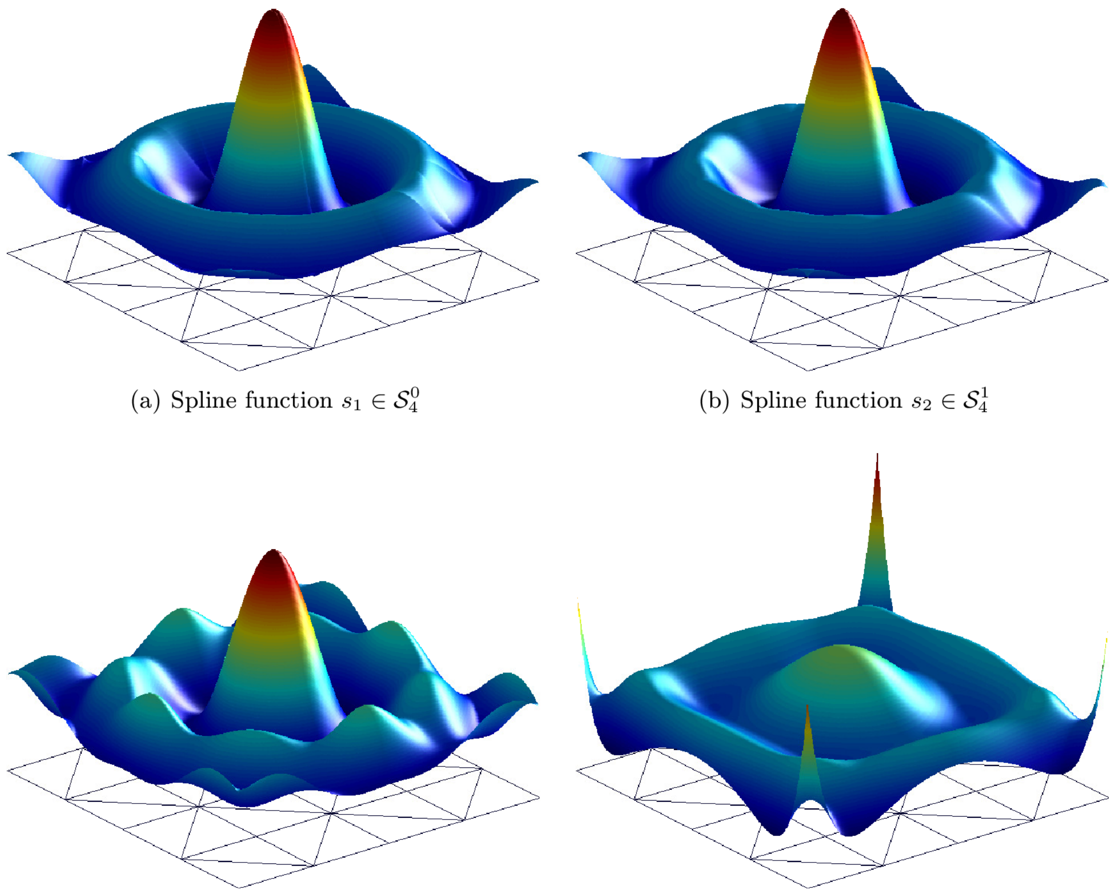
In the figure 2.9 the spline function of 4$^{\text{th}}$ degree for various continuity orders is shown. The objective function that is being approximated is well known as the Mexican Hat function. The figure shows the effect of the smoothness. One important behaviour that need to be recognized is a global smoothing effect caused by the continuity conditions. With increasing continuity order the global smoothing effect is more and more significant. By comparing C^0 and C^3 this behaviour is evident. As a result, with higher continuity order of the simplex spline function, the approximation power of the function is decreasing while a smoothing of the function is getting stronger. The first two C^0 and C^1 continuous splines approximate the objective function quite sufficiently, values of the C^3 spline are much out of the range as there are fewer degrees of freedom available.



(a) The smoothness matrix $\mathbf{H}_3 \in \mathbb{R}^{8\,832 \times 7\,680}$ of the spline function $s_3 \in \mathcal{S}_4^2(\mathcal{T}_{512})$

(b) The smoothness matrix $\mathbf{H}_4 \in \mathbb{R}^{10\,304 \times 7\,680}$ of the spline function $s_4 \in \mathcal{S}_4^3(\mathcal{T}_{512})$

Figure 2.8: The visualization of the smoothness matrices of two different continuity orders. This figure was plotted by `spy` function in MATLAB.



(a) Spline function $s_1 \in \mathcal{S}_4^0$

(b) Spline function $s_2 \in \mathcal{S}_4^1$

(c) Spline function $s_3 \in \mathcal{S}_4^2$

(d) Spline function $s_4 \in \mathcal{S}_4^3$

Figure 2.9: The simplex spline function of various degree. The input dataset $\mathcal{X}$ contains $|\mathcal{X}| = 10\,000$ points.

2.4 A linear regression scheme

After the triangulation and the smoothness matrix are known, all B-coefficients of all B-form polynomials defined on every simplex need to be calculated. In this section, the generalized linear regression scheme for estimation of B-coefficients is provided. This regression form allows the formulation of B-form basis polynomials of the simplex spline function in a form of a standard regression matrix. The smoothness matrix then acts as an linear equality constraint. This method was proposed by authors of [8]. However, as the mathematical concept of this problem is very complex only some important parts had been chosen.

2.4.1 The linear regression using B-form polynomial

In this section the formulation of a standard linear regression scheme using the B-form polynomials is introduced. It is assumed that the dimension, the triangulation and the spline space of the problem are already known according to the previous sections as well as the input dataset of all observations. All observations are in a form of vector-scalar pairs, where the vector defines input parameters of the spline function while the response of the function or the output is defined by the scalar value. For example, an input dataset of a terrain data approximation problem contains two elements of latitude and longitude in a vector part while the scalar value gives the altitude. Note that the number of elements in a vector part of a pair is always equal to the dimension less one.

In general, the elements of a vector-scalar pair $(\mathbf{x}(i), y(i))$ are related as:

$$y(i) = f(\mathbf{x}(i)) + r(i), \quad i = 1, 2, \dots N \ , \tag{2.48}$$

with f an unknown function and $r(i)$ a residual term. Now, given a triangulation consisting of J simplicies and given polynomial degree d, an unknown function f can be substituted by already well known simplex spline function as follows:

$$y(i) = \sum_{j=1}^{J} \sum_{|k|=d} c_k^{t_j} B_k^d(\mathbf{b}(i)) + r(i), \tag{2.49}$$

with $\mathbf{b}(i) = b(\mathbf{x}(i))$ the barycentric coordinate of $\mathbf{x}(i)$ with respect to the simplex t_j. For a single datapoint $\mathbf{x}(i)$ expression (2.49) is evaluated with respect to every simplex in a triangulation. Eventually we want to find a value of (2.49) only with respect to the simplex in which the point $\mathbf{x}(i)$ is located. In order to obtain a per-simplex interpolation scheme, a simplex membership operator $\delta_{\mathbf{x},\mathcal{T}_j}$ need to be introduced:

$$\delta_{\mathbf{x},\mathcal{T}_j} = \begin{cases} 1 \text{ if } \mathbf{x} \in \mathcal{T}_j \\ 0 \text{ if } \mathbf{x} \notin \mathcal{T}_j \end{cases} \tag{2.50}$$

with $\mathbf{x}$ an arbitrary point and with $\mathcal{T}_j$ a j^{th} simplex of a triangulation. Expression (2.50) returns 1 if data point $\mathbf{x}$ is located within a simplex $\mathcal{T}_j$. This membership function is now included into (2.49) which leads to following:

$$y(i) = \sum_{j=1}^{J} \left(\delta_{\mathbf{x},\mathcal{T}_j} \sum_{|k|=d} c_k^{t_j} B_k^d(\mathbf{b}(i)) \right) + r(i). \tag{2.51}$$

The expression (2.51) is an entirely valid linear regression structure, however, a matrix formulation is more usual. For that, a vector formulation of a d^{th} degree B-form polynomial from (2.27) is derived in following lines. Firstly, a vector of lexicographically sorted basis polynomial terms of a simplex t_j is:

$$\mathbf{B}_{t_j}^{d}(i) = \left[B_{k}^{d,t_j}(\mathbf{b}(i)) \right]_{|k|=d} \in \mathbb{R}^{1\times\hat{d}}. \tag{2.52}$$

Using expression (2.52) the full triangulation basis function vector is defined as follows:

$$\mathbf{B}^{d}(i) = \left[\mathbf{B}_{t_1}^{d}(i), \mathbf{B}_{t_2}^{d}(i), \ldots, \mathbf{B}_{t_J}^{d}(i) \right] \in \mathbb{R}^{1\times J\cdot\hat{d}}, \tag{2.53}$$

with J the number of simplices of a triangulation. The per-simplex diagonal matrix form of a membership operator is defined as:

$$\mathbf{D}_{t_j}(i) = \left[\left(\delta_{\mathbf{x},\mathcal{T}_j} \right)_{q,q} \right]_{q=1}^{\hat{d}} \in \mathbb{R}^{\hat{d}\times\hat{d}}. \tag{2.54}$$

The block diagonal full-triangulation data membership matrix $\mathbf{D}(i)$ for a single observation is a matrix with $\mathbf{D}_{t_j}(i)$ blocks on the main diagonal:

$$\mathbf{D}(i) = \left[\left(\mathbf{D}_{t_j}(i) \right)_{j,j} \right]_{j=1}^{J} \in \mathbb{R}^{(J\cdot\hat{d})\times(J\cdot\hat{d})}. \tag{2.55}$$

Using all these equations together, the matrix form of a B-form simplex spline function for a single observation i defined on a complete triangulation is:

$$P(\mathbf{b}(i)) = \mathbf{B}^{d}(i) \cdot \mathbf{D}(i) \cdot \mathbf{c}, \tag{2.56}$$

with $\mathbf{c}$ the lexicographically sorted B-coefficient and with $P(\mathbf{b}(i))$ an arbitrary B-form polynomial. Another assumption can be made, let $\mathbf{X}(i)$ be a single row in the full-triangulation resression matrix for all observations as follows:

$$\mathbf{X}(i) = \mathbf{B}^{d}(i) \cdot \mathbf{D}(i) \in \mathbb{R}^{1\times J\cdot\hat{d}}. \tag{2.57}$$

The expression (2.56) can be rewritten as:

$$y(i) = \mathbf{X}(i) \cdot \mathbf{c} + r(i). \tag{2.58}$$

And now the final step is:

$$\mathbf{Y} = \mathbf{X}\mathbf{c} + \mathbf{r} \in \mathbb{R}^{N\times 1}, \tag{2.59}$$

that is a well-known formulation of the linear regression scheme. There are many methods to solve (2.59), however, most used is a standard least square estimator.

2.4.2 The equality constrained least squares estimator

The outcome of the previous section was a formulation of a simplex spline function in a form of the linear regression scheme. In this section an estimator that can be used to solve the linear regression model is described. As there are equality constraints in the form of the smoothness matrix, the so-called linear equality-constrained least squares (LSE) approach needs to be used. This optimization method is well known in literature [16, Chapter 20]. The following lines will provide a short mathematical concept of LSE.

In the beginning, the p-norm of a vector $\mathbf{x}$ need to be introduced as follows:

$$\|\mathbf{x}\|_p = \left(\sum_{i\in\mathbb{N}} |x_i|^p\right)^{\frac{1}{p}}, \quad p \geq 1. \tag{2.60}$$

With LSE is used the case of $p = 2$ which is also called the Euclidean norm or the Euclidean distance. Now, the LSE problem can be stated as follows:

$$\min_{\mathbf{c}} \|\mathbf{X}\mathbf{c} - \mathbf{Y}\|_2 \quad \text{subject to} \quad \mathbf{H}\mathbf{c} = \mathbf{0}, \tag{2.61}$$

with dimensions assumed as follows:

$$\mathbf{Y} \in \mathbb{R}^{N\times 1}, \mathbf{X} \in \mathbb{R}^{N\times(J\cdot\hat{d})}, \mathbf{H} \in \mathbb{R}^{(E\cdot R_r)\times(J\cdot\hat{d})}, \mathbf{0} \in \mathbb{R}^{(E\cdot R_r)\times 1}. \tag{2.62}$$

In the following will be assumed that $n = J \cdot \hat{d}$ and $p = E \cdot R_r$. It is important to mention, that the expression (2.61) has one unique solution if and only if the smoothness matrix is of a full rank i.e. $\text{rank}(\mathbf{H}) = p$.

From the literature, the estimation of the vector $\mathbf{c}$ of regression coefficients, or in this case of B-coefficients starts by calculating the QR-factorization of $\mathbf{H}^\top$:

$$\mathbf{H}^\top = \mathbf{Q}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix} \text{ with } \mathbf{Q} \in \mathbb{R}^{n\times n}, \mathbf{R} \in \mathbb{R}^{p\times p}, \mathbf{0} \in \mathbb{R}^{(n-p)\times p}, \tag{2.63}$$

where the matrix $\mathbf{Q}$ is an orthogonal (unitary) matrix while the matrix $\mathbf{R}$ is upper triangular and non-singular. The QR-decomposition method also introduces following expressions:

$$\mathbf{A}\mathbf{Q} = [\mathbf{A}_1\mathbf{A}_2], \tag{2.64}$$

$$\mathbf{Q}^\top\mathbf{c} = \begin{bmatrix}\mathbf{y}\\ \mathbf{z}\end{bmatrix}. \tag{2.65}$$

After the substitution this gives:

$$\mathbf{H}\mathbf{c} = \left(\mathbf{Q}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix}\right)^\top \mathbf{c} = \left[\mathbf{R}^\top\mathbf{0}\right]\mathbf{Q}^\top\mathbf{c} = \left[\mathbf{R}^\top\mathbf{0}\right]\begin{bmatrix}\mathbf{y}\\ \mathbf{z}\end{bmatrix} = \mathbf{R}^\top\mathbf{y}. \tag{2.66}$$

Using the formulation $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}$:

$$\mathbf{X}\mathbf{c} = (\mathbf{X}\mathbf{Q})\left(\mathbf{Q}^\top\mathbf{c}\right) = [\mathbf{A}_1\mathbf{A}_2]\begin{bmatrix}\mathbf{y}\\ \mathbf{z}\end{bmatrix} = \mathbf{A}_1\mathbf{y} + \mathbf{A}_2\mathbf{z}. \tag{2.67}$$

So the problem becomes:

$$\min \|\mathbf{A}_1\mathbf{y} + \mathbf{A}_2\mathbf{z} - \mathbf{Y}\|_2 \quad \text{subject to} \quad \mathbf{R}^\top\mathbf{y} = \mathbf{0}, \tag{2.68}$$

with $\mathbf{y}$ determined directly from the equality constraints, and then inserted into the LSE problem as follows:

$$\min \|\mathbf{A}_2\mathbf{z} - (\mathbf{Y} - \mathbf{A}_1\mathbf{y})\|_2. \tag{2.69}$$

This leads to the final expression in this part giving the vector $\mathbf{z}$ which can be used to find the answer. Using (2.65):

$$\mathbf{c} = \mathbf{Q}\begin{bmatrix}\mathbf{y}\\ \mathbf{z}\end{bmatrix}. \tag{2.70}$$

The implementation of LSE estimator can be found for example in MATLAB Optimization toolbox. The function is named `lsqlin` and provides many features e.g inequality constraints, bound constraints, etc.

2.5 The basic operations with simplex splines

The multivariate simplex spline theory as defined in the previous chapters is truly an effective approximation tool. In this section some basic operations with the simplex spline will be presented. For the purpose of this thesis the differential operators are especially important, however, many other e.g. sums, integrals or inner products of the B-form polynomials can be found in literature [15, 9].

In the first section, the concept of de Casteljau recursion in introduced. The important structure in the definitions of the differential operators on the B-form polynomials is the so-called de Casteljau *one-step* matrix derived in the section 2.5.3. In the final section 2.5.4, the one-step matrix is used in the definition of differential operators such as the directional derivative, the gradient, Laplacian and the divergence.

2.5.1 The De Casteljau algorithm

The De Casteljau algorithm is a recursive method to evaluate polynomials in Bernstein form named after its inventor Paul de Calsteljau. The algorithm was invented in 1959 when the computing performance of computers of that times was very low. The De Casteljau algorithm was widely used in many areas of industry and research. Today, however, the performance of computers reached the level when it is faster to evaluate the Bernstein polynomial directly through ALU of a processor. Despite of this, many operations on the simplex splines are defined with the use of the De Casteljau algorithm concept and therefore some basics of this method are presented.

A B-form polynomial can be expressed in terms of the m^{th} de Casteljau iteration as follows:

$$p(\mathbf{b}) = \sum_{|k|=d-m} c_k^{(m)}(\mathbf{b}) B_k^{d-m}(\mathbf{b}), \tag{2.71}$$

in which the B-coefficient of iteration m is related to the B-coefficient of iteration $m-1$ as follows:

$$c_k^{(m)}(\mathbf{b}) = \sum_{|\gamma|=1} b_\gamma c_{k+\gamma}^{(m-1)}(\mathbf{b}), m \leq d, \tag{2.72}$$

with $c_k^{(0)}(\mathbf{b}) = c_k$ and with $b_\gamma = b_{\gamma_0} b_{\gamma_1} \ldots b_{\gamma_n}$ a first order polynomial defined in terms of the barycentric coordinate $\mathbf{b}$. The final value of the polynomial at the evaluation point $\mathbf{b}$ is $c_0^{(d)}(\mathbf{b})$, which is the last iteration of de Casteljau algorithm. For the practical demostration of the de Casteljau recursion see A.7.

2.5.2 A multi-degree formulation of the de Casteljau recursion

The de Casteljau recursion in a multi-degree form was presented in [9, 8]. Now the derivation of this form will be repeated here as this topic is essential for the calculation of derivatives of the multivariate simplex spline function.

The polynomial basis function $B^d_k(\mathbf{b})$ from (2.24) can be expanded as follows:

$$\begin{aligned} B^d_k(\mathbf{b}) &= b_0 B_{k_0-1,k_1,\dots,k_n}(\mathbf{b}) + b_1 B_{k_0,k_1-1,\dots,k_n}(\mathbf{b}) + \cdots + \\ &\qquad + b_n B_{k_0,k_1,\dots,k_n-1}(\mathbf{b}),\ |k| = d \\ &= \sum_{|\gamma|=1} b_\gamma B^{d-1}_{k-\gamma}(\mathbf{b}), \end{aligned} \tag{2.73}$$

with $\gamma = (\gamma_0, \gamma_1, \dots, \gamma_n)$ a multi-index dependent of k as follows:

$$|\gamma| = d - |k|. \tag{2.74}$$

In general, for any degree $m \leq d$ a single basis function term is:

$$\begin{aligned} B^d_k(\mathbf{b}) &= P^m_\gamma(\mathbf{b}) B^{d-m}_{k_0-m,k_1,\dots,k_n}(\mathbf{b}) + P^m_\gamma(\mathbf{b}) B^{d-m}_{k_0,k_1-m,\dots,k_n}(\mathbf{b}) + \cdots + \\ &\quad + P^m_\gamma(\mathbf{b}) B^{d-m}_{k_0,k_1,\dots,k_n-m}(\mathbf{b}) \\ &= \sum_{|\gamma|=m} P^m_\gamma(\mathbf{b}) B^{d-m}_{k-\gamma}(\mathbf{b}), \end{aligned} \tag{2.75}$$

with the m^{th} degree basis function $P^m_\gamma(\mathbf{b})$ defined as follows:

$$P^m_\gamma(\mathbf{b}) = \frac{m!}{\gamma!} \mathbf{b}^m_\gamma. \tag{2.76}$$

This formulation produces also polynomial terms with negative multi-indices that are by definition equal to zero as follows:

$$B^{d-m}_{k-\gamma}(\mathbf{b}) = 0 \quad \text{if} \quad l \in (k - \gamma)\ \wedge\ l < 0. \tag{2.77}$$

After substitution of (2.75) into (2.27) we have following expression:

$$p(\mathbf{b}) = \sum_{|k|=d} \left[c_k \sum_{|\gamma|=m} P^m_\gamma(\mathbf{b}) B^{d-m}_{k-\gamma}(\mathbf{b}) \right]. \tag{2.78}$$

This expression still produces negative multi-indicise, however, letting $|k| = d - m$ the reformulation is possible as follows:

$$p(\mathbf{b}) = \sum_{|k|=d-m} \sum_{|\gamma|=m} c_{k+\gamma} P^m_\gamma(\mathbf{b}) B^{d-m}_k(\mathbf{b}). \tag{2.79}$$

Finally, the multi-degree de Casteljau algorithm, which relates the B-coefficients of interation q with those of iteration $m + q$ is:

$$c^{(q+m)}_k(\mathbf{b}) = \sum_{|\gamma|=m} P^m_\gamma(\mathbf{b}) c^{(q)}_{k+\gamma}(\mathbf{b}),\ q \geq 0, \tag{2.80}$$

with $c^{(0)}_k(\mathbf{b}) = c_k$. This theorem was proposed and proved in [9].

2.5.3 A one-step de Casteljau matrix

The definition of a derivative of the simplex spline function requires one more concept that is discussed in the following paragraphs. The one-step matrix form of the algorithm generalizes the de Casteljau algorithm and is defined in a matrix form. The de Casteljau matrix is a function of degree m, which reduces a set of B-coefficients of a degree d into a set of B-coefficients of degree $d-m$, is defined as follows:

$$\mathbf{P}^{d,d-m}(\mathbf{b}) \in \mathbb{R}^{\hat{d}^* \times \hat{d}}, \tag{2.81}$$

with $\hat{d}^*$ the total number of basis function terms for degree $d-m$ and dimension n:

$$\hat{d}^* = \frac{(d-m+n)!}{(d-m)!n!}. \tag{2.82}$$

The de Casteljau matrix has the following structure:

$$\left[\mathbf{P}^{d,d-m}(\mathbf{b})\right]_{i(k),i(\theta)} = P^m_{\theta-k}(\mathbf{b}),\ |\theta| = d,\ |k| = d-m, \tag{2.83}$$

with $i(k)$ and $i(\theta)$ index functions for the rows and columns of $\mathbf{P}^{d,d-m}(\mathbf{b})$ respectively. The index function $i(k)$ for the rows and the index function $i(\theta)$ for the columns are defined as follows:

$$i(k) = \sum_{|\gamma|=d-m} 1,\ \gamma \leq k,\ |k| = d-m \tag{2.84}$$

$$i(\theta) = \sum_{|\gamma|=d} 1,\ \gamma \leq \theta,\ |\theta| = d. \tag{2.85}$$

In his paper [9] De Visser demonstrated the calculation of the one-step de Casteljau matrix by an easy example. As this part is essential in many basic calculations with simplex splines the demonstration example is provided by A.8.

2.5.4 The directional derivatives of the B-form polynomials

As will be shown now, the de Casteljau matrix provides a very elegant and simple formulation of the directional derivatives of B-form polynomials. This concept was well explored by [15].

The directional derivative is defined for a differentiable function $f(\mathbf{x})$ with $\mathbf{x} \in \mathbb{R}^n$ and for a directional vector $\mathbf{u} \in \mathbb{R}^n$. Now, at the point $\mathbf{x}$ the directional derivative can be written as:

$$D_{\mathbf{u}} f(\mathbf{x}) = \frac{d}{dt} f(\mathbf{x} + t\mathbf{u}), \tag{2.86}$$

with both coordinates $\mathbf{x}$ and $\mathbf{u}$ in the Cartesian form. The transformation of a single point from the Cartesian coordinate system to the barycentric system was discussed in the section 2.1.2, and works well. However, the same transformation of a directional vector $\mathbf{u}$ will not work. Instead of that, the simple definition of the vector can be used. Let $\mathbf{x}_1$ and $\mathbf{x}_2$

be two points in Euclidean space $\mathbb{R}^n$. After the transformation of them to the barycentric system the directional vector in barycentric coordinates is defined as follows:

$$b(\mathbf{u}) = \mathbf{a} = b(\mathbf{x}_2) - b(\mathbf{x}_1) \in \mathbb{R}^{n+1}, \tag{2.87}$$

with $b(\mathbf{x})$ the transformation from Cartesian coordinate system to barycentric one.

From the literature, the m^{th} derivative of the B-form polynomial $p(\mathbf{b})$ in the direction $\mathbf{u}$ can be stated as follows:

$$D_{\mathbf{u}}^m p(\mathbf{b}) = \frac{d!}{(d-m)!}\mathbf{B}^{d-m}(\mathbf{b})\mathbf{P}^{d,d-m}(\mathbf{a}) \cdot \mathbf{c}^{t_j}, \tag{2.88}$$

with $\mathbf{P}^{d-m}(\mathbf{a})$ the de Casteljau matrix from (2.83) of degree d to $d-m$ expressed in terms of the directional barycentric vector $\mathbf{a} = b(\mathbf{u})$, with $\mathbf{B}^{d-m}(\mathbf{b})$ the vector form of the basis polynomials, and with $\mathbf{c}^{t_j}$ the vector of B-coefficients for a single simplex t_j.

The theorem (2.88) was proved by de Visser in his paper [9, Theorem 4, page 103] where an easy to reproduce example can be found as well.

After the derivation of the B-form polynomial is known, the formulation of other differential operator is very easy as the formal definition can be used.

The gradient of a spline function is given by:

$$\begin{aligned}\nabla p(b(\mathbf{x}))_{t_i} =& \left(D_{x_1}^1 p(b(\mathbf{x}))_{t_j}, D_{x_2}^1 p(b(\mathbf{x}))_{t_j}, \dots, D_{x_n}^1 p(b(\mathbf{x}))_{t_j}\right) \\ =& d\mathbf{B}^{d-1}(\mathbf{b}) \cdot \Big(\mathbf{P}^{d,d-1}(\mathbf{a}_1) \cdot \mathbf{c}^{t_j}, \mathbf{P}^{d,d-1}(\mathbf{a}_2) \cdot \mathbf{c}^{t_j}, \dots, \\ & \mathbf{P}^{d,d-1}(\mathbf{a}_n) \cdot \mathbf{c}^{t_j}\Big),\end{aligned} \tag{2.89}$$

with $\mathbf{a}_1, \dots, \mathbf{a}_n$ the barycentric directional vectors along the axis respectively.

The Laplacian operator is defined as follows:

$$\begin{aligned}\Delta p(b(\mathbf{x}))_{t_i} =& \left(D_{x_1}^2 p(b(\mathbf{x}))_{t_j}, D_{x_2}^2 p(b(\mathbf{x}))_{t_j}, \dots, D_{x_n}^2 p(b(\mathbf{x}))_{t_j}\right) \\ =& \frac{d!}{(d-2)!}\mathbf{B}^{d-2}(\mathbf{b}) \cdot \Big(\mathbf{P}^{d,d-2}(\mathbf{a}_1) \cdot \mathbf{c}^{t_j}, \mathbf{P}^{d,d-2}(\mathbf{a}_2) \cdot \mathbf{c}^{t_j}, \dots, \\ & \mathbf{P}^{d,d-2}(\mathbf{a}_n) \cdot \mathbf{c}^{t_j}\Big).\end{aligned} \tag{2.90}$$

The divergence operator:

$$\begin{aligned}\operatorname{div} p(b(\mathbf{x}))_{t_i} =& \left(D_{x_1}^1 p(b(\mathbf{x}))_{t_j} + D_{x_2}^1 p(b(\mathbf{x}))_{t_j} + \dots + D_{x_n}^1 p(b(\mathbf{x}))_{t_j}\right) \\ =& d\mathbf{B}^{d-1}(\mathbf{b}) \cdot \Big(\mathbf{P}^{d,d-1}(\mathbf{a}_1) \cdot \mathbf{c}^{t_j} + \mathbf{P}^{d,d-1}(\mathbf{a}_2) \cdot \mathbf{c}^{t_j} + \ldots + \\ & + \mathbf{P}^{d,d-1}(\mathbf{a}_n) \cdot \mathbf{c}^{t_j}\Big).\end{aligned} \tag{2.91}$$

Chapter 3

The numerical experiments with the simplex splines

In this section some essential properties of multivariate simplex splines are demonstrated by various numerical experiments. The accuracy of the interpolation by simplex splines depends on many factors. For example, in the section 2.3.3 was written that the approximation power of the simplex splines is decreasing with higher continuity order. This is in general very much true, however, not always the case. The polynomial degree can be set to higher value to increase the precision of the interpolation as well as higher number of elements in input dataset can provide better precision. Also the different number of simplices in the triangulation may have a positive or a negative effect. In all these cases, the estimation of the resulting simplex spine function or also so-called the training process takes non-zero amount of time. The computational cost depends on many factors and sometimes may become critical.

The algorithm used in following experiments was implemented in MATLAB, therefore the results, especially the computational time, must be understood in the context of this programming language. It is true that the implementation in a different low-level language would provide different characteristics and in fact better (faster) results. However, the evaluation of the simplex spline function model needs to be as fast as possible. Therefore, this part of the algorithm was implemented in C++ and is highly optimized for computational speed.

In the beginning, some methods are proposed in the section 3.1 for assessing the quality of the multivariate simplex spline models. These methods provide a unified testing tool of the spline functions and are used in the following sections. An interpolation of a simple bivariate function is presented in the section 3.2. In this section the parameters are described that need to be set before any training process of the simplex spline model as well as their effects on the resulting model.

One of the most important features of the multivariate simplex splines is the ability of data approximation in an n-dimensional space. The visualization of the 4 and higher dimensional simplex spline function is a tricky task, however, in the section 3.3, the interpolation of a tri-variate simplex spline function is shown. With higher polynomial degree the estimation of the simplex spline function is becoming slower and therefore, some complexity issues are also discussed in this section. Later, in section 3.4 the topic of the terrain data modelling is discussed and demonstrated on a real terrain dataset.

3.1 Model quality assessment

To compare the precision of the simplex splines models, the objective comparative method needs to be introduced. In this section some simple but powerful techniques are described.

As the linear regression scheme from 2.4 is used to interpolate input dataset it is possible to asses the quality of the simplex spline model directly by the residue $\mathbf{r}$ from (2.59) which leads to the following expression:

$$\mathbf{r} = \mathbf{Y} - \mathbf{X}\mathbf{c}. \tag{3.1}$$

The model residual analysis is the analysis of the difference between the observations and the spline model output at the same locations. To statistically measure the varying magnitude of the residual, the root mean square (RMS) method can be used. The formulation of RMS for this case is as follows:

$$RMS(\mathbf{r}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\mathbf{r}_i^2}, \tag{3.2}$$

with $N = J \cdot \hat{d}$ the total number of B-coefficients of simplex spline function and $\mathbf{r}_i$ the scalar value of a residue at location defined by the corresponding B-coefficient.

Another useful metric for assessment of the global quality of a spline model is based on the variances of all B-coefficients in a spline function. At the first in this method the variances of all B-coefficients located in each single simplex are estimated. After that, the mean of all these variances is calculated as follows:

$$Var(\mathbf{c}) = \frac{1}{J}\sum_{i=1}^{J} var(\mathbf{c}_i), \tag{3.3}$$

with J the number of the simplices in the triangulation and $\mathbf{c}_i$ the vector of B-coefficients for a simplex i. This method will be used to analyse the dependency of B-coefficients in various spline spaces.

3.2 The interpolation of bivariate function

Using the multivariate simplex splines as an approximation tool, the accuracy of the modeled data and the interpolation speed are affected by many factors. In the following list the most important factors are mentioned:

- the total number of the elements of the input dataset,
- the degree of the polynomials defined on every simplex,

- the continuity order between adjoining polynomials,
- the number of the simplices of the triangulation.

In this section, all these parameters will be analysed very closely to get a good overview of the simplex splines method.

In the beginning, the section 3.2.1 provides the formulation of an objective function. The objective function is then used to generate the input dataset. In the following sections 3.2.2, 3.2.3 and 3.2.4, the input parameters and their impact on a resulting spline model are discussed step-by-step.

3.2.1 The objective function

The behaviour of the multivariate simplex splines will be in this case demonstrated on a very simple Mexican hat function defined as follows:

$$O(x_1, x_2) = \operatorname{sinc}\left(h \cdot \sqrt{\left(\frac{x_1}{\pi}\right)^2 + \left(\frac{x_2}{\pi}\right)^2}\right) \quad \text{with} \quad \operatorname{sinc}(x) = \frac{\sin(x)}{x}, \tag{3.4}$$

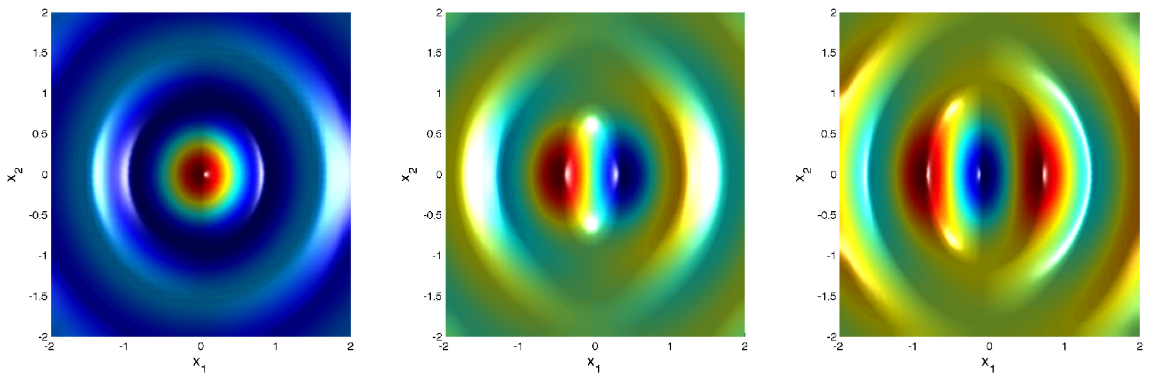
and with $h = 5$ the scale factor. This function is suitable for this example as the difference of the approximation in various spline spaces is significant and the 1^{st} and the 2^{nd} order derivatives are in general well known. In the figure 3.1 the objective function O is plotted along with its first and second order derivatives. The first row 3.1(a) shows a projection to the 2-dimensional plane while the values of the function are represented by different colors. The contours of the function are clearly visible in this figure. The second row 3.1(b) shows usual 3-dimensional visualization.

In the following, all input datasets are generated by the uniform random number generator.

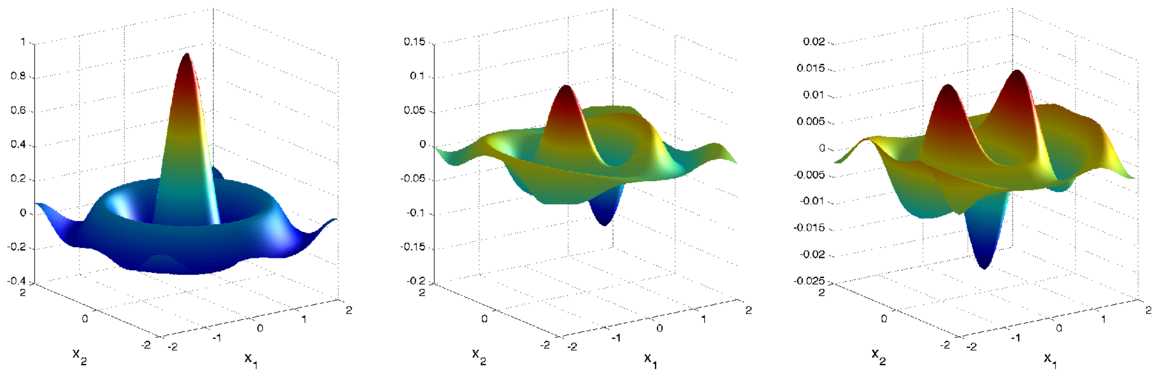
3.2.2 Effects of the triangulation

The shape and the number of simplices of the triangulation have a strong impact on a resulting simplex spline model. First of all, all simplices of the triangulation should be well-defined as was described in the section 2.2.1. In order to obtain a good accuracy of the approximation on a badly-defined simplex, a simplex polynomial must be of a high degree – in general the degree must be much higher than it would be with a well-defined simplex. This is mainly caused by the fact, that in order to obtain good approximation on a sliver simplex, the associated simplex polynomial needs more available degrees of freedom and therefore needs to be of a higher degree.

Another important task is to find out an optimal number of simplices of a triangulation. The number of simplices can affect the speed of the training process as well as the quality of a simplex spline model itself. The higher number of simplices of a triangulation, the longer it takes to approximate an input dataset. However, in some cases more simplices may increase an accuracy of a simplex spline model. In the figure 3.2 three simplex spline



(a) Projection from 3 to 2 dimensional plane.



(b) 3D visualization

Figure 3.1: Objective Mexican hat function O (3.4). From the right hand side the original function followed by the first order $\dfrac{\mathrm{d}O}{\mathrm{d}x}$ and the second order $\dfrac{\mathrm{d}^2O}{\mathrm{d}x^2}$ derivatives.

functions of the same spline space $\mathcal{S}_2^0$ are plotted. The only difference between these spline functions is a number of simplices of the triangulation they are defined on. Each simplex is interpolated by a 2nd degree polynomial i.e. quadratic function. The input dataset which contains 1 000 randomly generated elements, is visualized by red dots. The first triangulation $\mathcal{T}_8$, 3.2(a), contains 8 simplices. Every simplex contains enough input points so the system from (2.59) is overdetermined, however, the approximation is not very accurate. Simplices of the triangulation are relatively large and as all of them are interpolated by pieces of a quadratic function, some shapes of the Mexican hat function cannot be sufficiently modeled. Therefore, in order to obtain higher accuracy of the simplex spline model from the figure 3.2(a)(b), more degrees of freedom of simplex polynomial must be available, thus, the higher polynomial order of the simplex spline function is needed.

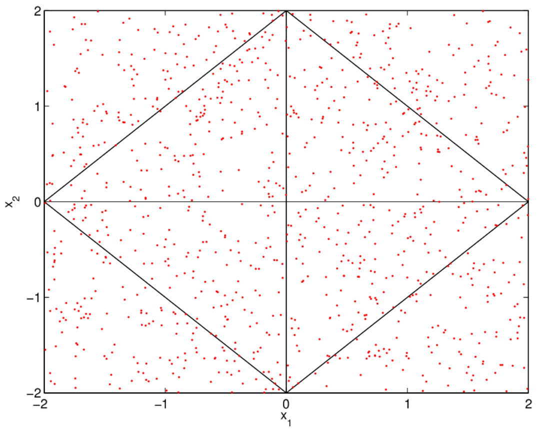
Second row of the figure 3.2 shows the same case as the previous one, but the triangulation $\mathcal{T}_{32}$ consists of 32 simplices. Four time more quadratic polynomial pieces participate on the resulting simplex spline model, so the accuracy of the spline function is much higher what corresponds with the observation 3.2(d). In the last case 3.2(e)(f) the triangulation $\mathcal{T}_{128}$ contains again 4-times more simplices than the previous one and the same pattern of the observation is still followed. Now the resulting simplex spline function 3.2(f) consists of 128 quadratic polynomial pieces, therefore the shape of the model is much more accurate according to the objective function from (3.4).

It could seem, that in order to increase an exactness of the simplex spline model, increasing the number of simplices of a triangulation would be enough. In the example from 3.2 it is very much true, however, not always the case. As was mentioned before, to interpolate an input dataset by the linear least square estimator from (2.61) the system has to be overdetermined to find an approximate solution, that means the number of points within every simplex has to be greater than or equal to the number of independent variables of an interpolation polynomial. In the multivariate simplex splines theory, independent variables of a polynomial are well known B-coefficients. The number of them was defined by (2.23), in this case $\hat{d} = 6$. If this condition is not satisfied, resulting model will become unstable as the one from the figure 3.3. It can be seen that many simplices do not contain any points of the input dataset, so the corresponding polynomials are not bounded by any value, but by continuity conditions across the edges only. The result 3.3(b) is clearly useless.

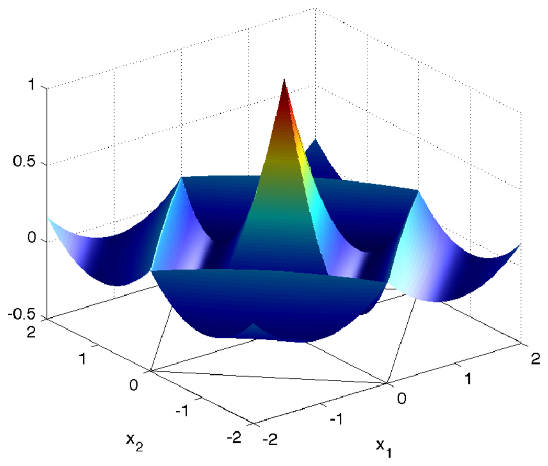
In the table 3.1 some observed values for the different kinds of metrics are listed. The first column represents triangulations from figures 3.2 and 3.3 with the total number of simplices of a triangulation denoted as J. The second column shows the precision of each simplex spline model. The best precision of a model was obtained for the triangulation $\mathcal{T}_{128}$ while the worst for the triangulation $\mathcal{T}_{512}$. This corresponds with observations from the figures 3.2 and 3.3. In the third column a relative number of points within every simplex is listed. This number is calculated as the number of points of the input dataset, in this case $|\mathcal{X}| = 1\,000$, divided by the number of simplices J. As mentioned before, the 2nd degree simplex spline function consists of quadratic polynomial pieces, therefore, at least two $\hat{d} = 6$ points within each simplex are needed. In the case of $\mathcal{T}_{512}$ this condition is corrupted. The last column simply shows the computational time needed to train a spline model. It can be observed, that with the increasing number of simplices of the triangulation the time spent by training process of the model is growing rapidly. This is caused mainly by the smoothness matrix. The more simplices of the triangulation, the more continuity conditions need to be calculated. The smoothness matrix is then much larger and the estimation of B-coefficients by the least squares takes more computational time.

Triangulation $\mathcal{T}_J$	RSM	$^{\lvert\mathcal{X}\rvert}/_{J}$	time
$\mathcal{T}_8$	0.0820	125.00	0.1120
$\mathcal{T}_{32}$	0.0442	31.25	0.3689
$\mathcal{T}_{128}$	0.0083	7.81	4.0941
$\mathcal{T}_{512}$	1.6364	1.95	77.3013

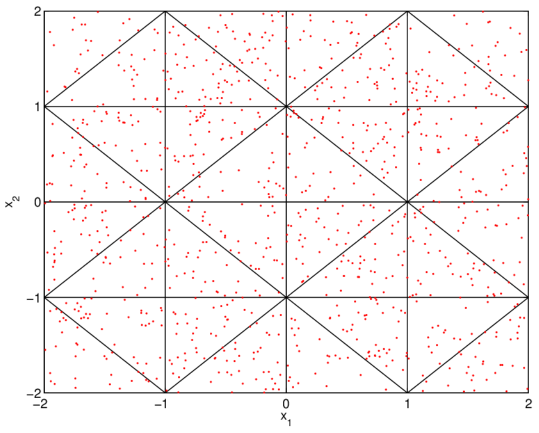
Table 3.1: Overview of some useful metrics for different triangulations $\mathcal{T}_J$, with J the number of simplices of a triangulation.



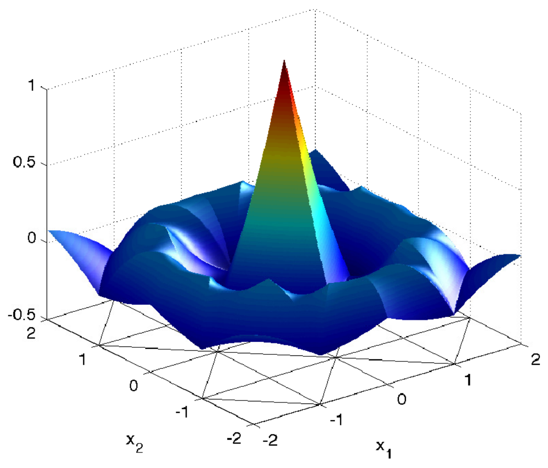
(a) Triangulation $\mathcal{T}_8$



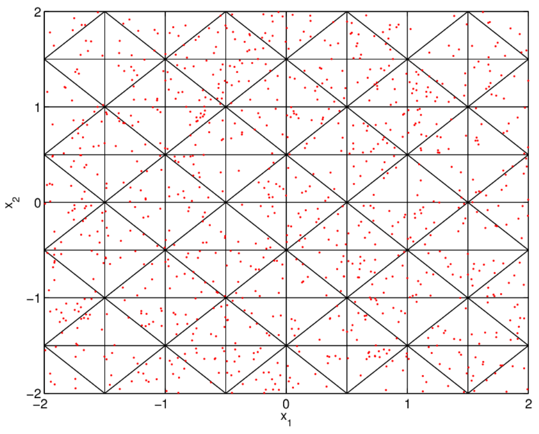
(b) $s_1 \in \mathcal{S}_2^0(\mathcal{T}_8)$



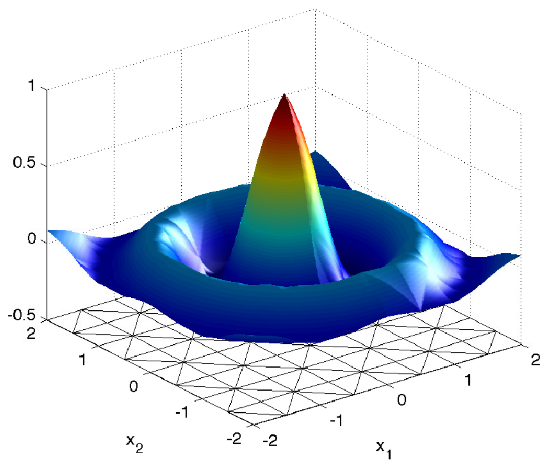
(c) Triangulation $\mathcal{T}_{32}$



(d) $s_2 \in \mathcal{S}_2^0(\mathcal{T}_{32})$

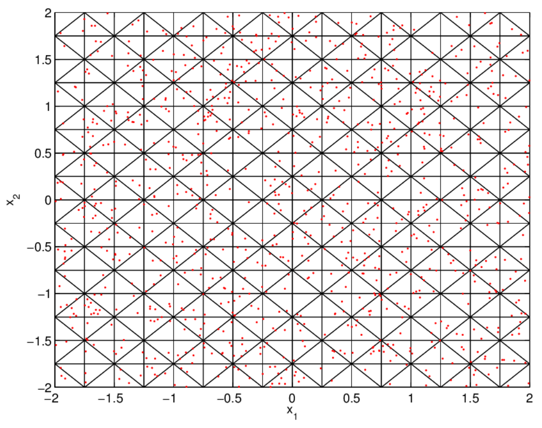


(e) Triangulation $\mathcal{T}_{128}$



(f) $s_3 \in \mathcal{S}_2^0(\mathcal{T}_{128})$

Figure 3.2: Demonstration of simplex splines models defined on triangulations with different number of simplices. In this figure all simplex spline function are of 2nd degree and of 0 continuity order.


(a) Triangulation $\mathcal{T}_{512}$

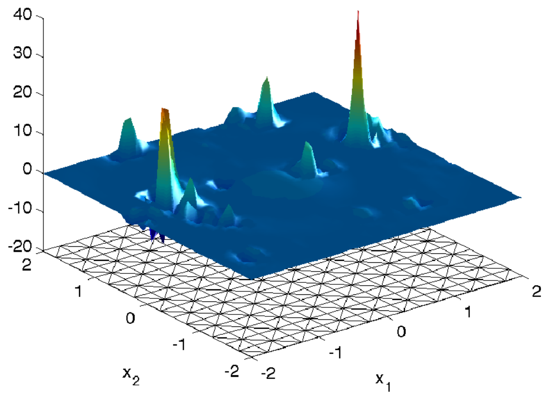

(b) $s_4 \in \mathcal{S}_2^0(\mathcal{T}_{512})$

Figure 3.3: Unstable simplex spline function. Simplices of the triangulation $\mathcal{T}_{512}$ do not contain enough input points. In this case the minimal number of points contained by each simplex is $\hat{d} = 6$.

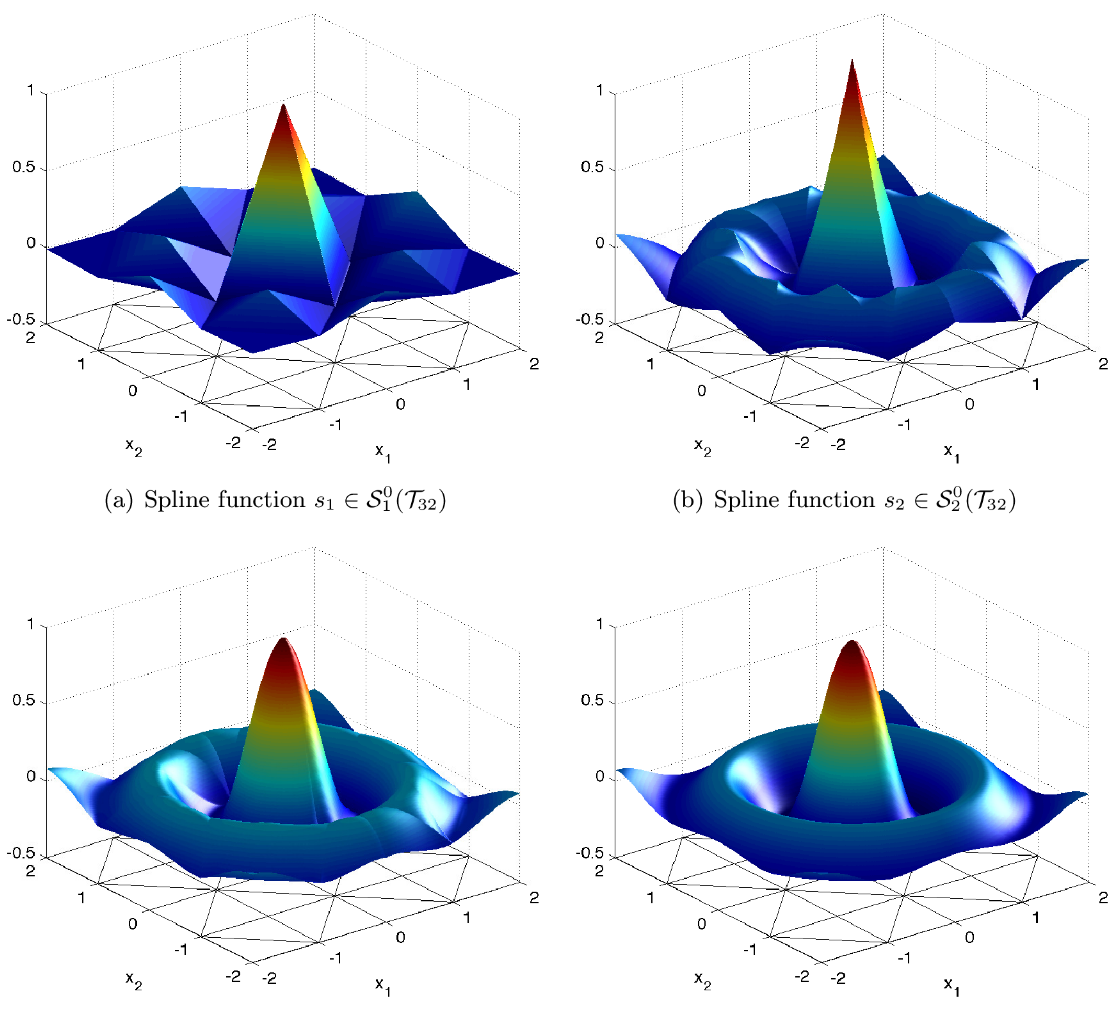
3.2.3 Effects of the polynomial degree

The values of the input dataset are interpolated on every single simplex by a piece of a polynomial function, therefore, a whole simplex spline function is a piecewise defined polynomial. A degree of a simplex spline function determines a degree of all polynomial pieces that are associated with every simplex. The higher the degree of a polynomial function, the more degrees of freedom are available to interpolate points of the input dataset and higher precision of the approximation is obtained. In the figure 3.4 the simplex spline function for various polynomial degrees is shown. Note, that the triangulation remains the same in all four cases as well as the input data set $|\mathcal{X}| = 10\,000$. In the first plot 3.4(a), the interpolation is made only by simple planes as the degree is set to 1. In next two plots 3.4(b), 3.4(c) the simplex spline model is getting more and more accurate. And finally, the plot 3.4(d) with polynomial degree 7 shows the simplex spline model having the error value very low.

While the polynomial degree is increasing, the computational time required to train the simplex spline model is increasing as well. This fact is shown in the figure 3.5 by the blue line. The red line of the same plot shows the RMS of the residue for different polynomial degrees. Note that the logarithm was used to scale the plots.

3.2.4 Effects of the smoothness

So far the continuity order was always supposed to be equal to zero, therefore, the continuity of the simplex spline function was enforced only across the simplex edges. In this section, the effects of the continuity conditions on a resulting spline model will be explained in a very detail.



(a) Spline function $s_1 \in \mathcal{S}_1^0(\mathcal{T}_{32})$ (b) Spline function $s_2 \in \mathcal{S}_2^0(\mathcal{T}_{32})$

(c) Spline function $s_3 \in \mathcal{S}_3^0(\mathcal{T}_{32})$ (d) Spline function $s_4 \in \mathcal{S}_7^0(\mathcal{T}_{32})$

Figure 3.4: Simplex spline function of various polynomial degree. The input dataset $\mathcal{X}$ contains $|\mathcal{X}| = 10\,000$ points.

Recalling back from the chapter 2.3, the smoothness constraints in a form of continuity conditions link B-coefficients across a shared edge of two neighbouring simplices. The number of B-coefficients involved in a definition of a continuity conditions depends on a continuity order C^r (2.40). If the derivatives up to the given order of a simplex spline function are required, continuity conditions have to be used. For example, a simplex spline function of the continuity order C^0 is differentiable up to order 0, that is, the continuity of a original function is satisfied. This is the case of all functions in the figures 3.2 and 3.4. Clearly, these functions are continuous, but the first order derivative is discontinuous and therefore the function is not differentiable. A spline function of the C^1 continuity order is then differentiable up to the order 1, a function of continuity order C^2 up to the order 2, etc.

A powerful method for visually inspecting the continuity of a simplex spline function is based on the divergence operator defined in (2.91). A visualisation of the divergence vector

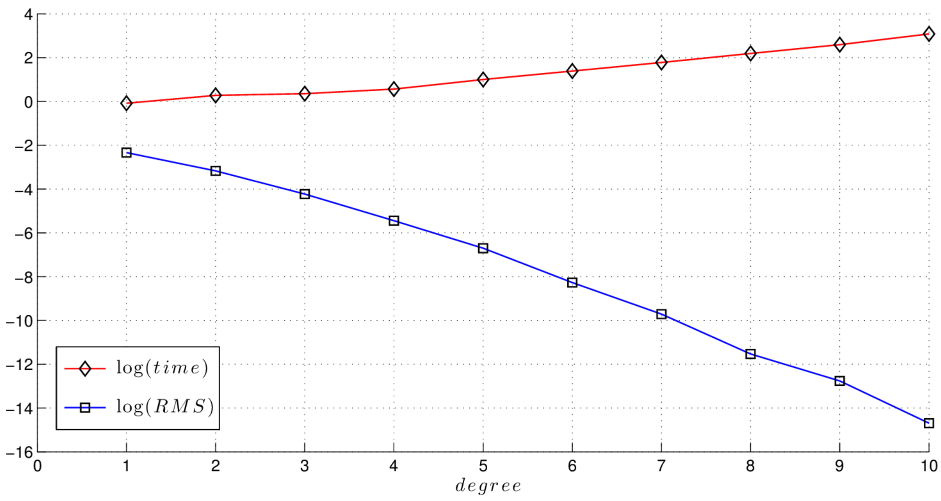


Figure 3.5: Computational time (red) and resulting RMS of the residue (blue) of the simplex spline model for various polynomial degrees.

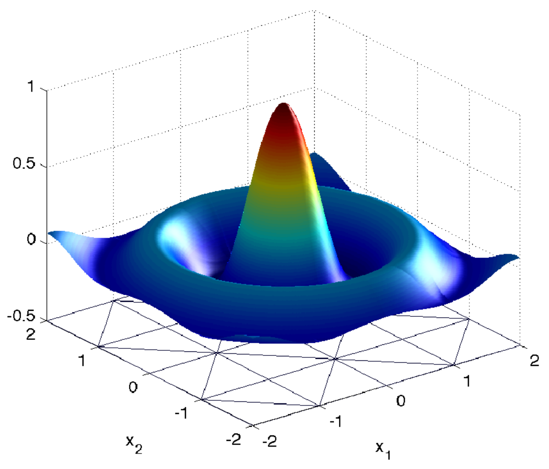
field may show some discontinuities that do not need to be apparent from a normal surface plot. In the left hand plots of the figure 3.6 three simplex spline functions differing only in continuity order are shown. The right hand side plots of the same figure show the visualization of a divergence vector field of each function. Beginning from the top, plots 3.6(a)(b) represent the simplex spline model of the continuity C^0. The surface plot 3.6(a) looks quite smooth and any discontinuities are not clear. The divergence operator 3.6(b), otherwise, shows strong discontinuities that coincide with simplex edges. The second row of the same figure shows the similar situation with the continuity order equal to C^1. It can be seen in 3.6(d), that isolines are smoother which corresponds with higher continuity order. Despite of that, very small discontinuities that again coincide with simplex edges still remain. As the expression from (2.43) is correct, the reason of this strange behaviour is inaccuracy of computer calculations. Especially round-off errors may have a strong impact on a result of a computation. The last two plots 3.6(e) and 3.6(f) represent the simplex spline function of C^2 continuity order. The discontinuities across the simplex edges are almost gone, while the precision of the interpolation is still good.

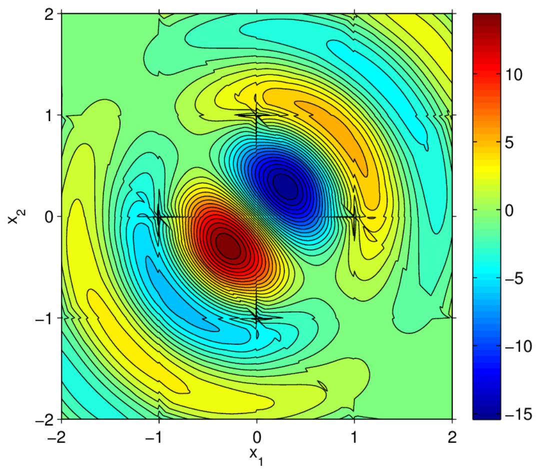
As mentioned in the chapter 2.3 and then demonstrated by the figure 2.9, with increasing continuity order the approximation power of a simplex spline function is being reduced. This effect is the consequence of the loss of degrees of freedom of interpolation polynomial caused by continuity conditions. Another, more exact visualization of this behaviour is provided by the figure 3.7(a). In order to get a better overview, a number of simple spline models in various spline spaces were trained. By words, there are 5 different continuity orders together with 10 different polynomial degrees that makes 40 various spline spaces. Note please, in the figure, the plot of the continuity order C^r starts always from the polynomial degree $d = r + 1$. The y-axis represents the residual RMS. It can be seen that while the polynomial degree is increasing, the residual RMS is being reduced and converges to the zero value, as expected. From the graph 3.7(a) also another important observation can be made. In order to obtain a good accuracy of approximation, the continuity order C^r and polynomial degree of the simplex spline function d should satisfy the condition as follows:

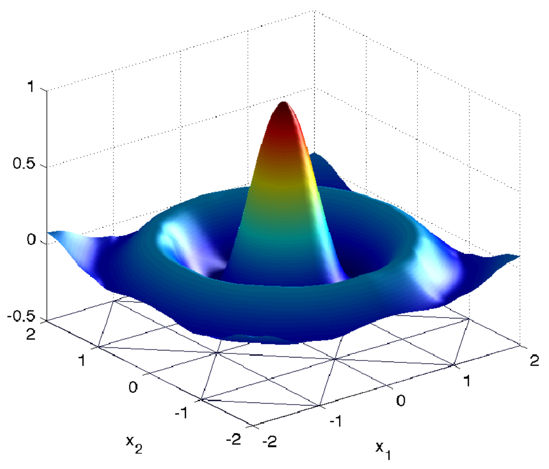
$$d - r > 3. \tag{3.5}$$

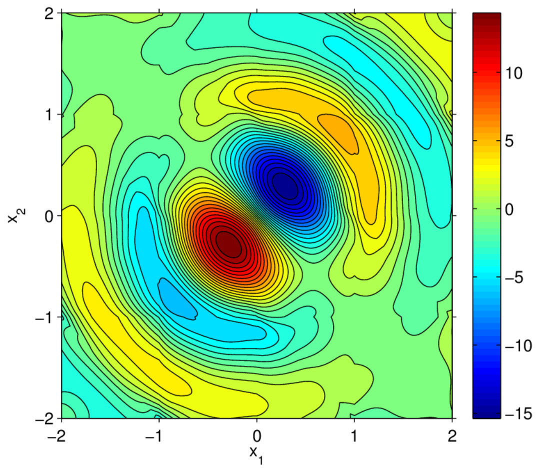
For example, the residual RMS of the spline model $\mathcal{S}_d^1$ in the figure 3.7(a) is decreasing rapidly for $d = 2 \ldots 4$. For higher $d > 4$ polynomial degrees, however, the residual error is slowly getting close to zero.

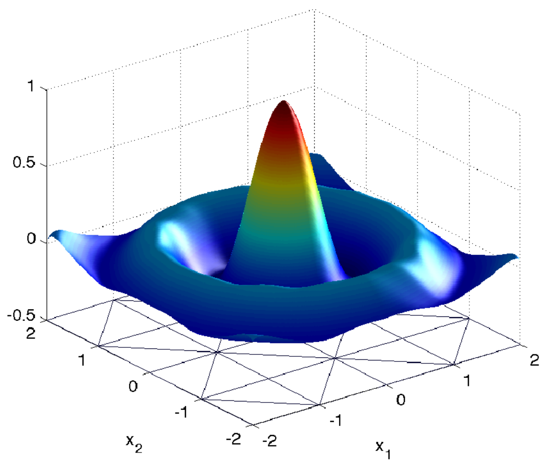
Another useful measurement of the quality of a simplex spline model was proposed by (3.3). The figure 3.7(b) shows the mean of variances of all B-coefficients located in each simplex of the spline function. The spline spaces are the same as in the previous case. According to the graph, the mean variance of B-coefficients for low polynomial degrees depends heavily on the continuity order. For higher degrees though, the variances are more and more stable. This can be explained as follows. After the polynomial degree of simplex spline function is high enough to interpolate input dataset with very low error the shape of the function remains the same also for even higher degrees and therefore, for a given degree d the variance of B-coefficients remains the same regardless of the continuity order as long as the condition from (3.5) is satisfied.


(a) Spline function $s_1 \in \mathcal{S}_5^0(\mathcal{T}_{32})$


(b) div s_1


(c) Spline function $s_2 \in \mathcal{S}_5^1(\mathcal{T}_{32})$


(d) div s_2


(e) Spline function $s_3 \in \mathcal{S}_5^2(\mathcal{T}_{32})$

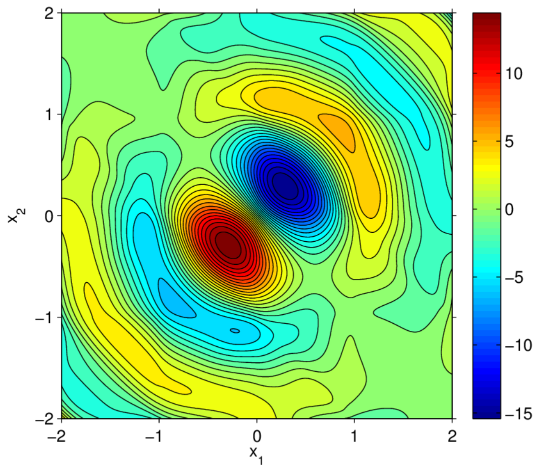
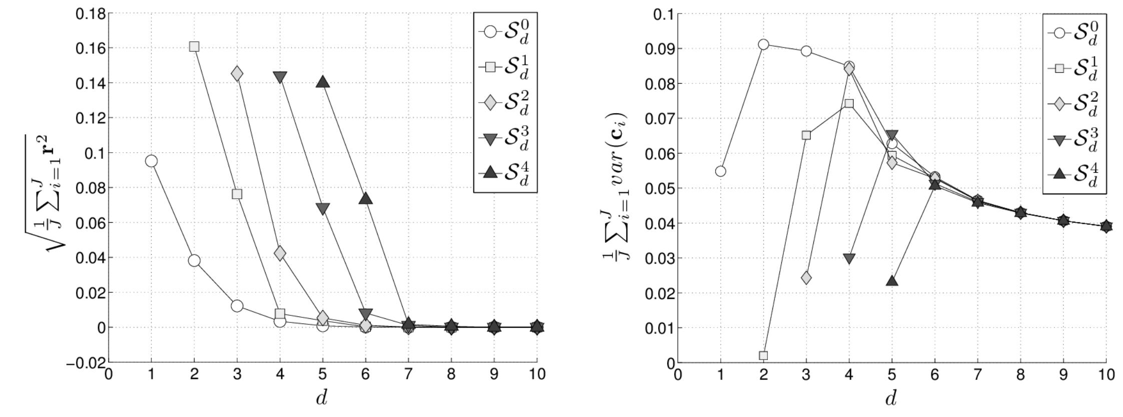

(f) div s_3

Figure 3.6: Three simplex spline functions of variout continuity order C^r and visualisation of associated divergence operators. Note, the isolines are smoother with higher continuity order.



(a) Higher polynomial degree d provides better approximation.

(b) Mean of B-coefficients variances

Figure 3.7: The residual RMS and mean variances of B-coefficients for various spline spaces.

3.3 Interpolation in higher dimensions

The theory of the multivariate simplex splines can be used as an interpolation tool in any number of dimensions. The higher dimensional problems are found in many areas of industry. For example, observed aerodynamic or thermodynamic data often span to the fourth or higher dimension. The dimensionality of a problem determines the number of input variables of the interpolation polynomial e.g. 4-dimensional input dataset needs to be approximated by the simplex spline function of three variables that represent first three dimensions while the output of the function represents the fourth dimension. In these days, visualization of data of four dimensions is not as straightforward as the visualization of any surface for example. One convenient way of showing 4-dimensional data, or also the so-called volume data, on a screen is shown in this section. Along with that, approximation of a simple function of three variables is presented.

3.3.1 Experiment setup

For the numerical experiment 10 000 data points $\mathcal{X} = \{(x_1, x_2, x_3) \in \mathbb{R}^3\}$ on the interval $\langle -2, 2\rangle$ were generated using a uniform random number generator. The response values for each triple were generated by the simple first order derivative of 3-variate Gaussian as follows:

$$O(x_1, x_2, x_3) = x_1 e^{-x_1^2 - x_2^2 - x_3^2}. \tag{3.6}$$

The Delaunay triangulation was then applied on a 3-dimensional grid. The grid forms a cube while all generated data points are located inside this cube. In the figure 3.8 the triangulation of the 3-dimensional grid with resolution 2 is shown. Data points $\mathcal{X}$ were dropped for better visualization in this figure, however, as all point of the set $\mathcal{X}$ were generated on the interval $\langle -2, 2\rangle$ they must be located inside the grid from the figure 3.8.

3.3.2 Experiment results

To interpolate the dataset $\mathcal{X}$ through the response values from (3.6) the equality constrained least squares estimator as proposed in the section 2.4.2 was used. Two simplex spline models in the spline spaces $s_1 \in \mathcal{S}_4^0(\mathcal{T}_{48})$ and $s_2 \in \mathcal{S}_4^3(\mathcal{T}_{48})$ were trained during this experiment. In both cases the triangulation $\mathcal{T}_{48}$ remained the same as in the figure 3.8. The plot of $\mathcal{S}_4^0$ is shown in the figure 3.9. In this figure a slice plane takes on colouring based on the values of the simplex spline function in the region where the slice is positioned. The first plot of the figure 3.9(a) shows only one single plane along the axis x_1 and x_2, while the third component x_3 is equal to the constant 0. In other plots the slices are always visualized in different positions to get a better overview of the simplex spline function shape. Red colors are corresponding with higher values of the simplex spline function while the blue colors with lower values. It should be clear from the figure, how the maxima and the minima of the function are distributed in space and that first order derivative of the 3-variate Gaussian would look similar.

The another figure 3.10 shows the second trained simplex spline function s_2. The point of this figure is to show the global smoothing effect caused by the continuity order. The

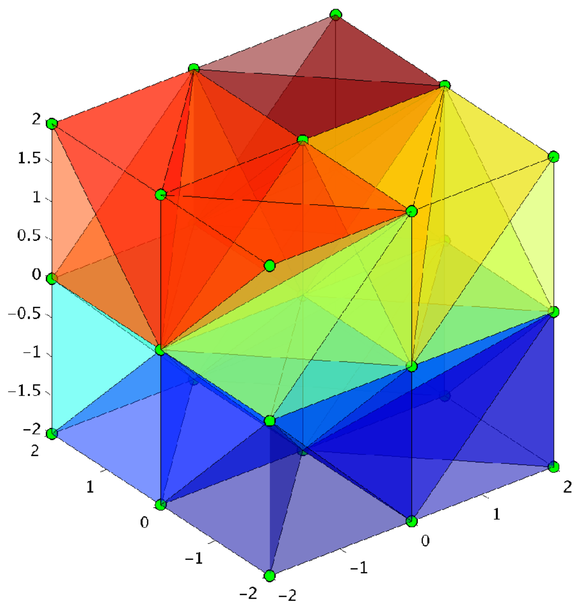


Figure 3.8: Demonstration of the Delaunay triangulation $\mathcal{T}_{48}$ of the 3-dimensional grid. The Delaunay triangulation, by definition, does not need to be regular and therefore this one is non-regular and contains 48 simplices. The resolution of the grid is 2.

difference between the polynomial degree and the continuity order is in this case equal to 1. The minima and the maxima are not so strong as in the figure 3.9 and some deviations are present across the domain edges. This corresponds with figure 2.9(d) where the same pattern can be observed.

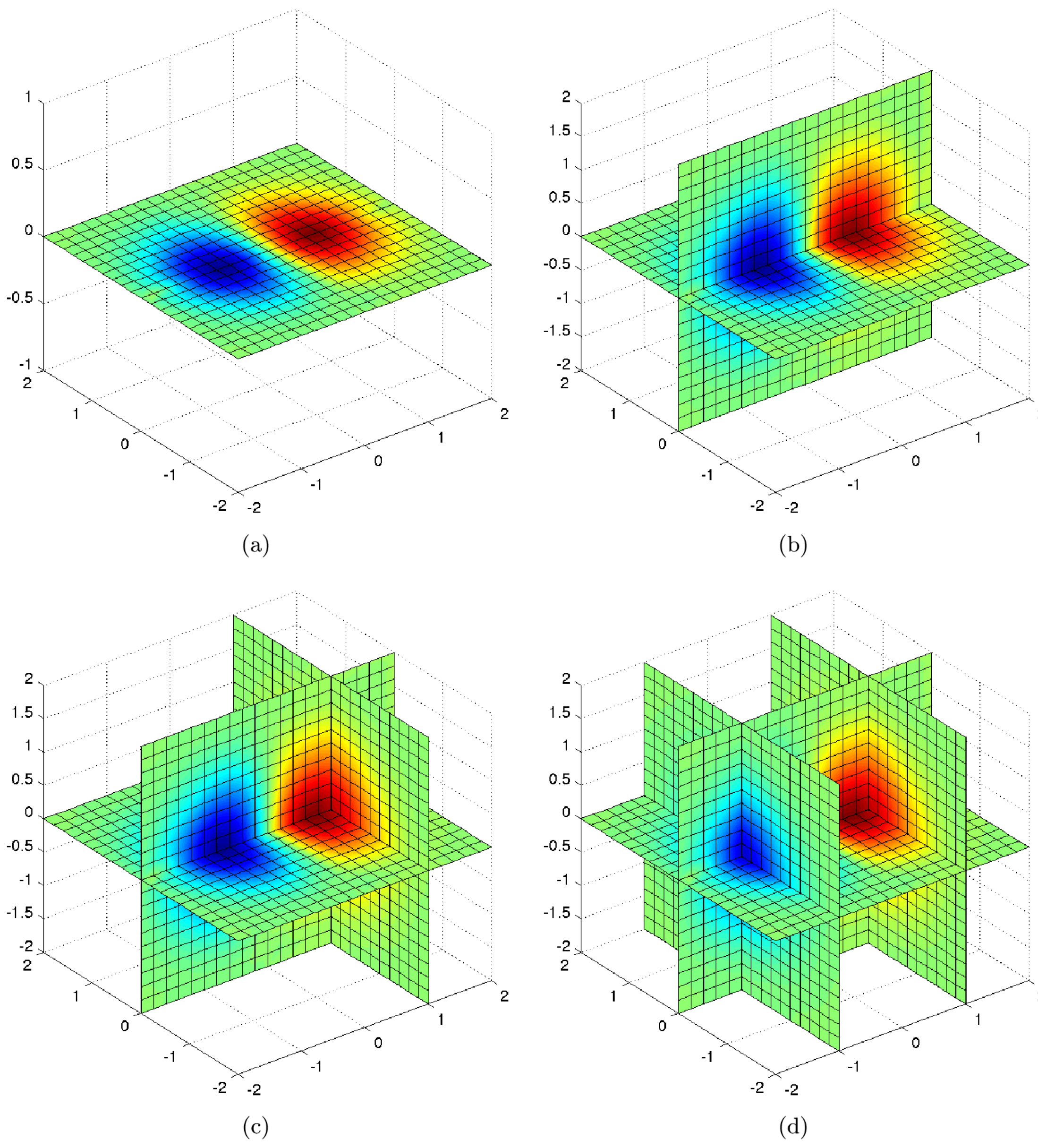


Figure 3.9: The simplex spline function $s_1 \in \mathcal{S}_4^0(\mathcal{T}_{48})$. The input data $|\mathcal{X}| = 10\,000$ set was generated by the 3-variate function from (3.6).

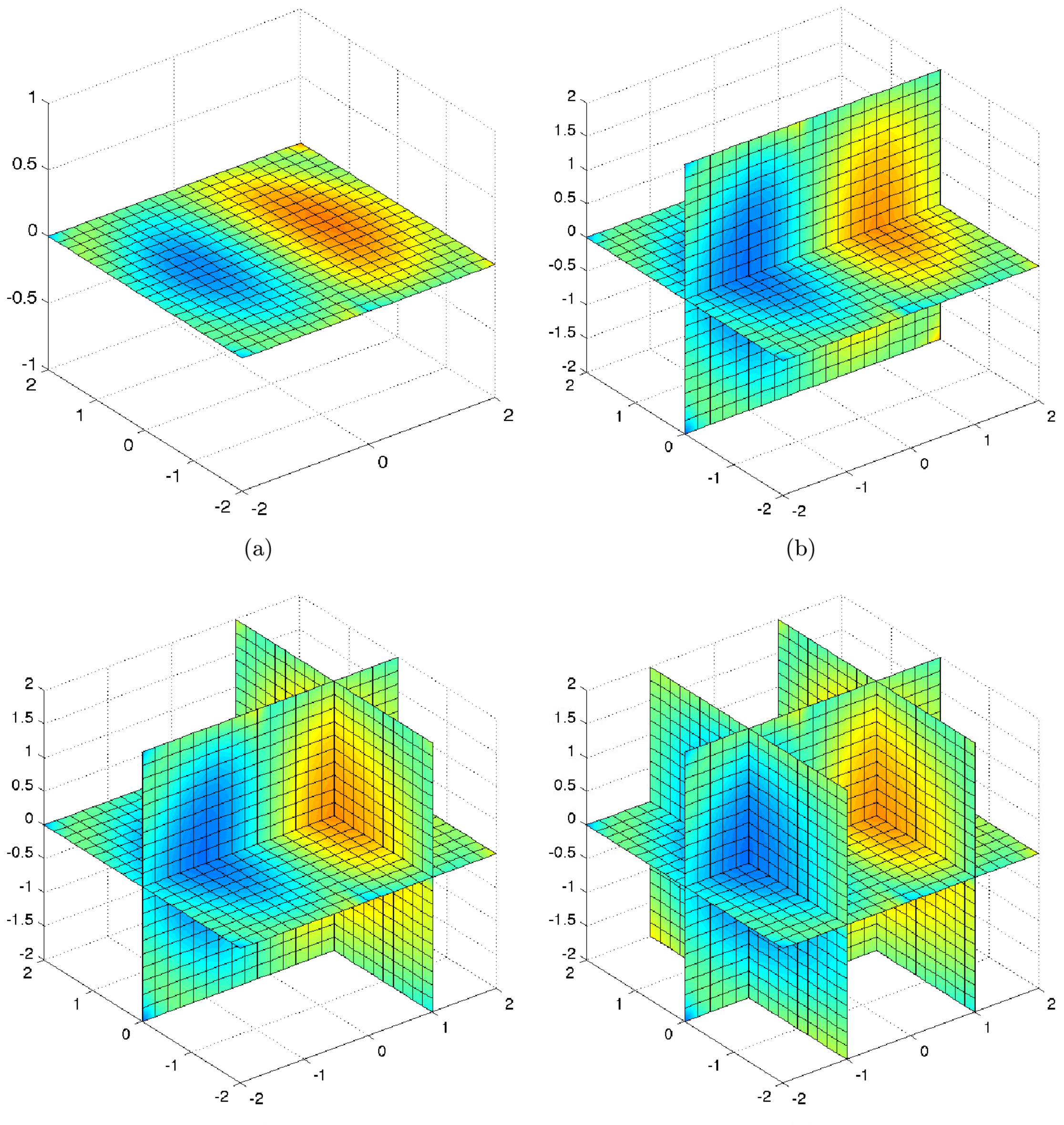


Figure 3.10: The simplex spline function $s_2 \in \mathcal{S}_4^3(\mathcal{T}_{48})$. The input data is the same as in 3.9. As the difference between polynomial degree and continuity order is very low the smoothness effect caused by the continuity conditions is obvious.

3.4 Approximation of terrain data

The multivariate simplex splines are indeed the real and powerful interpolation tool. In this section an application of this approach is demonstrated on a dataset consisting of the real terrain data. It should be clear now how the resulting simplex spline model can be tuned by different input parameters.

The following example starts with the experiment setup 3.4.1 where mentioned parameters are chosen and then in the section 3.4.2, the resulting models are analysed.

3.4.1 Experiment setup

The input dataset $\mathcal{X}_t$ in this experiment consists of 20 000 triples where the first two values, latitude and longitude, represent a point and the third value, altitude, represents response value. All points are randomly distributed in the space and form a rectangular region with intervals, $lat \in \langle 47.15\,°, 47.50\,° \rangle$ and $long \in \langle 11.15\,°, 11.50\,° \rangle$. The figure 3.11 shows the input dataset $\mathcal{X}_t$ together with the triangulation $\mathcal{T}_{98}$. The Delaunay triangulation method was used to calculate this triangulation. Notice strong irregularities of the triangulation grid. As mentioned many times, the Delaunauy triangulation does not have to be regular. In the multivariate simplex spline theory the shape of a triangulation is in general not so important as long as all simplices are well defined. As the terrain data are of three dimensions the resulting simplex spline function $s_{\mathcal{T}}$ will be bivariate:

$$alt = s_{\mathcal{T}}(lat, long) \in \mathcal{S}_d^r(\mathcal{T}_{98}), \tag{3.7}$$

with the output value alt the altitude. The generalized least square estimator from 2.4.2 was used to train individual simplex spline functions. Again as in previous examples, the simplex spline function was trained in various spline spaces; for each continuity order $C^r = 0 \ldots 4$ and the degree $d = (r+1) \ldots 10$. Together it makes 40 different spline spaces $\mathcal{S}_d^r(\mathcal{T}_{98})$.

3.4.2 Model analysis

After the coefficients of the simplex spline function are estimated, associated residues representing the difference between interpolated and observed values are available. The model residual analysis is performed using the theory of the section 3.1.

In the figure 3.12(a) the RMS of the residue is plotted. It is clear, that with higher polynomial degree of the simplex spline function, the total residue is decreasing and slowly converges to the ideal zero value. Although the speed of convergence is slower, the pattern of the plot is basically the same as the one from 3.7(a). It is important to say, that in demonstration examples of the section 3.2, as well as of the section 3.3, very simple objective functions were used. High precision interpolation of the input dataset was obtained even with low polynomial degrees of the spline function. In this case however, the input dataset consists of measured terrain data which are highly non-linear. It is hard or even not possible, due to the time complexity, to interpolate input dataset with such a high precision

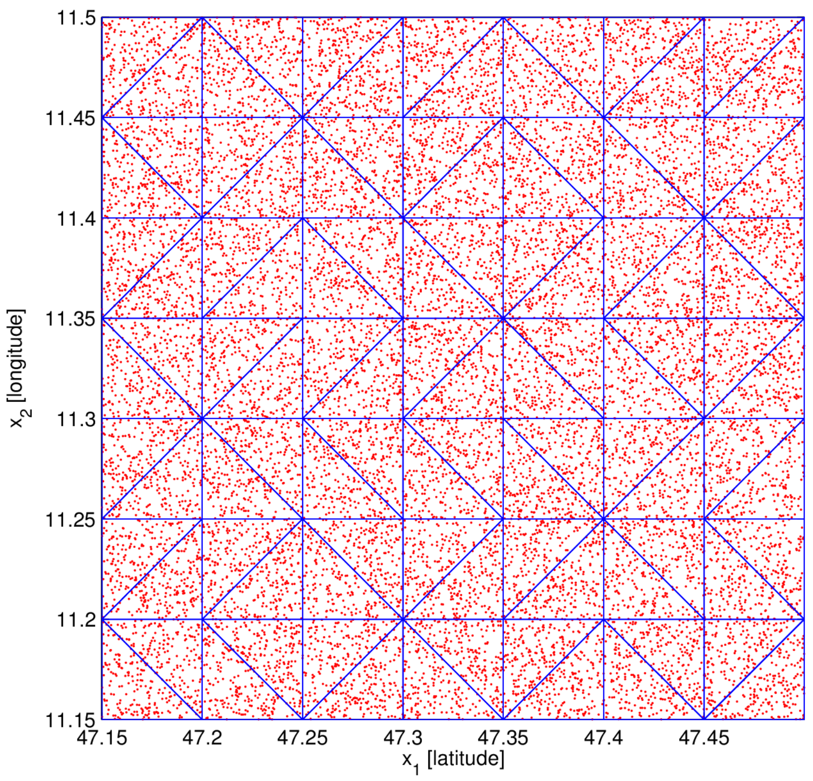
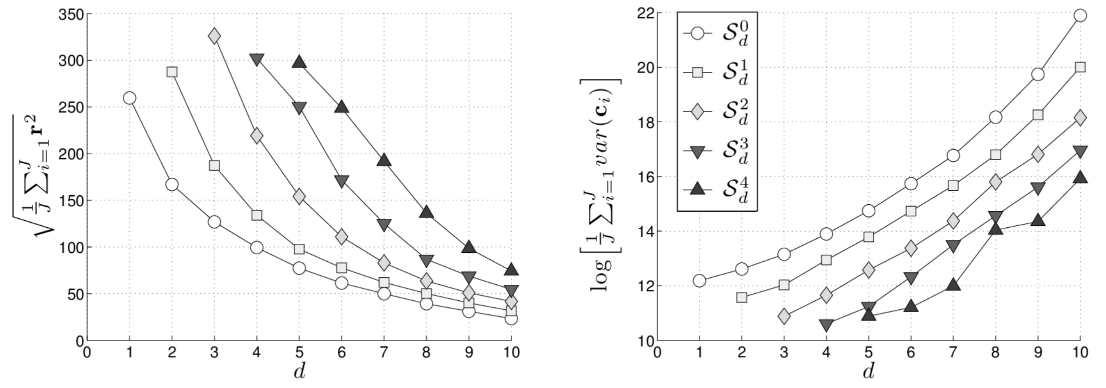


Figure 3.11: The input dataset $\mathcal{X}_t$ consisting of 20 000 points (red) and the Delaunay triangulation $\mathcal{T}_{98}$ consisting of 98 simplices (blue).

as in 3.4(d) for example. That is the reason why in 3.7(a) the residue is more-less zero for 7$^{\text{th}}$ degree and in the figure 3.12(a) it is not even for 10$^{\text{th}}$ degree.

The right hand side plot 3.12(b) shows the mean variance of B-coefficients of the spline function. Note please, that for a better visualization the logarithm function was used in this case to scale individual plots. Interestingly, the variance of B-coefficients is increasing with higher polynomial degree. This is caused, as already mentioned, by highly non-linear input dataset. From the figure 3.7(a) can be observed that after the polynomial degree is high enough the B-coefficients variance remains the same for all continuity orders. In this case however, for the degree equal to 10 the difference of the variance for various continuity orders is still significant, therefore the simplex spline function of a higher degree need to be used. High polynomial degree implies the inevitable price in the form of the time complexity. The plot 3.13 shows the computational time spent by the training process of the simplex spline function. It is clear that with higher polynomial degree the time complexity is growing rapidly as well as with increasing continuity order. For example, training of the spline function $\mathcal{S}_{10}^{4}$ takes more than twice more time than of the function $\mathcal{S}_{10}^{0}$. This is caused mainly by the size of the smoothness matrix, which is much larger for the continuity order C^4, see 2.2.

In the figure 3.14 the simplex spline function $s_{\mathcal{T}} \in \mathcal{S}_6^2$ is pictured. The rugged terrain



(a) The residual RMS is decreasing with higher polynomial degree.

(b) The B-coefficients variances are still increasing even for high polynomial degrees.

Figure 3.12: The residual RMS for various spline spaces (a) and the mean variance of B-coefficients (b).

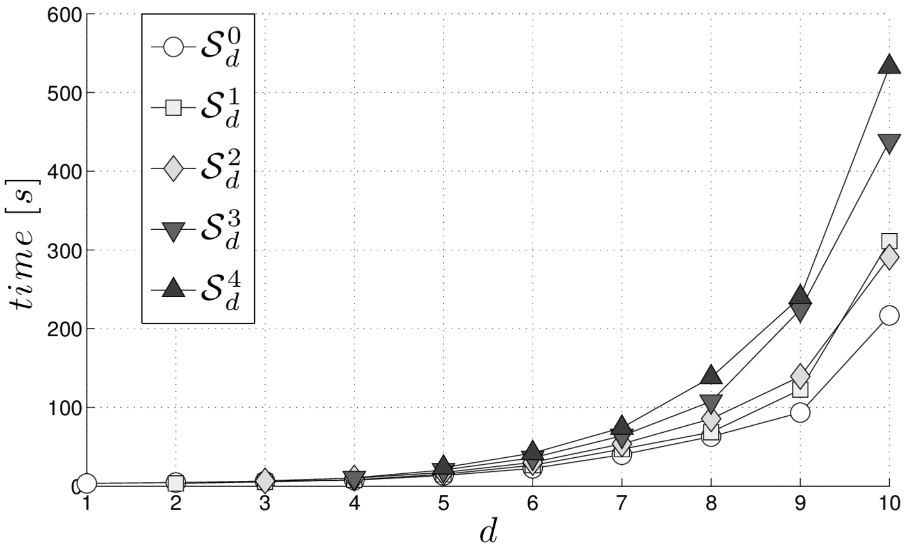


Figure 3.13: The time complexity of the training process is rapidly growing with increasing polynomial degree as well as with the continuity order.

surface can be seen together with the contour plot below. Isolines (black) represent points with the same altitude. In the next figure 3.15, gradients of the function s_τ are pictured as introduced in 2.5.4. In this figure the color of the plot represents the magnitude of the gradient. Peaks and valleys are coloured with blue which corresponds to gradients of low value. On the other hand, at slopes where the altitude is changing rapidly the red color is dominant.

It may be also interesting to analyse the smoothness using the divergence operator as proposed in the section 3.2.4. The result is pictured in the figure 3.16. In this figure are again areas with low gradients coloured by blue while the slopes are coloured by the red. It

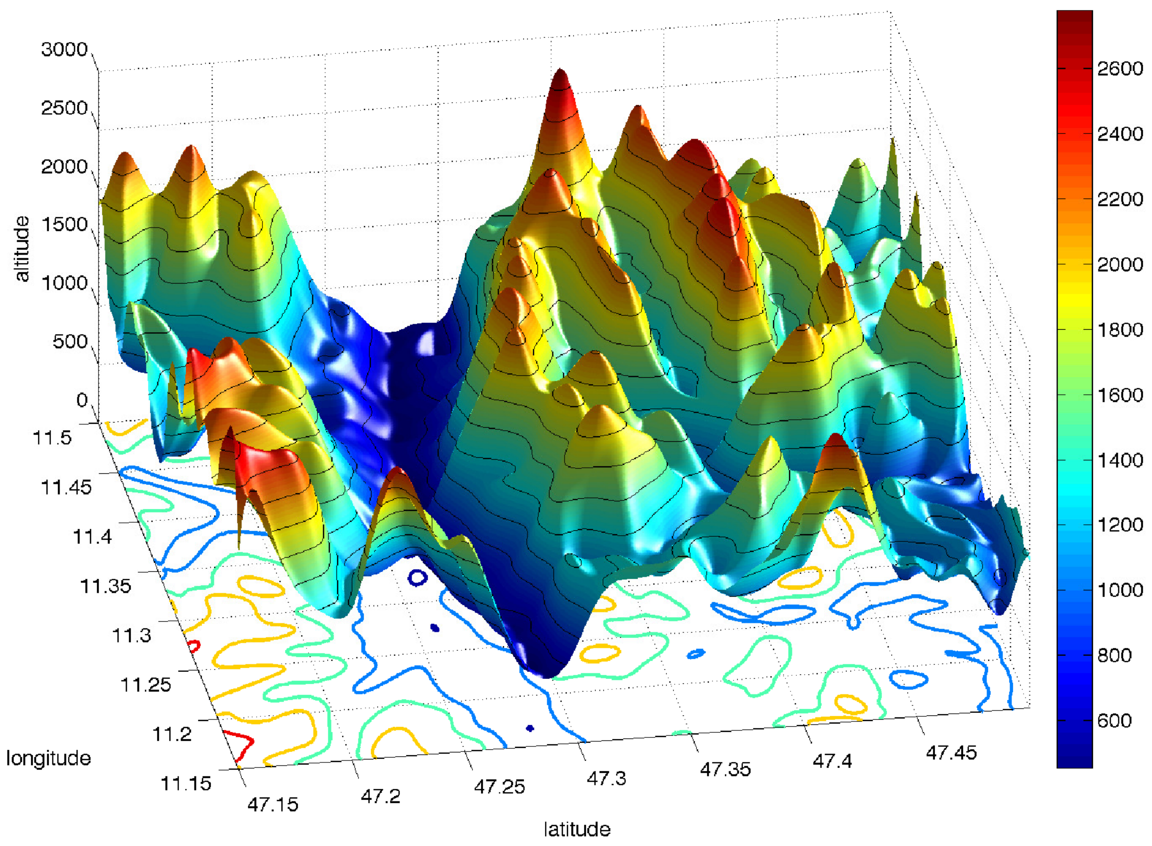


Figure 3.14: Example of terrain data approximation. Simplex spline function $s_{\mathcal{T}} \in \mathcal{S}_6^2$ together with the contour plot below.

can be seen that all isolines are nicely smooth and continuous what corresponds with the second continuity order. Also the structure of the triangulation is not visible, as it was in the figure 3.6. All this implies the good precision of the approximation even up to the 2 order derivative.

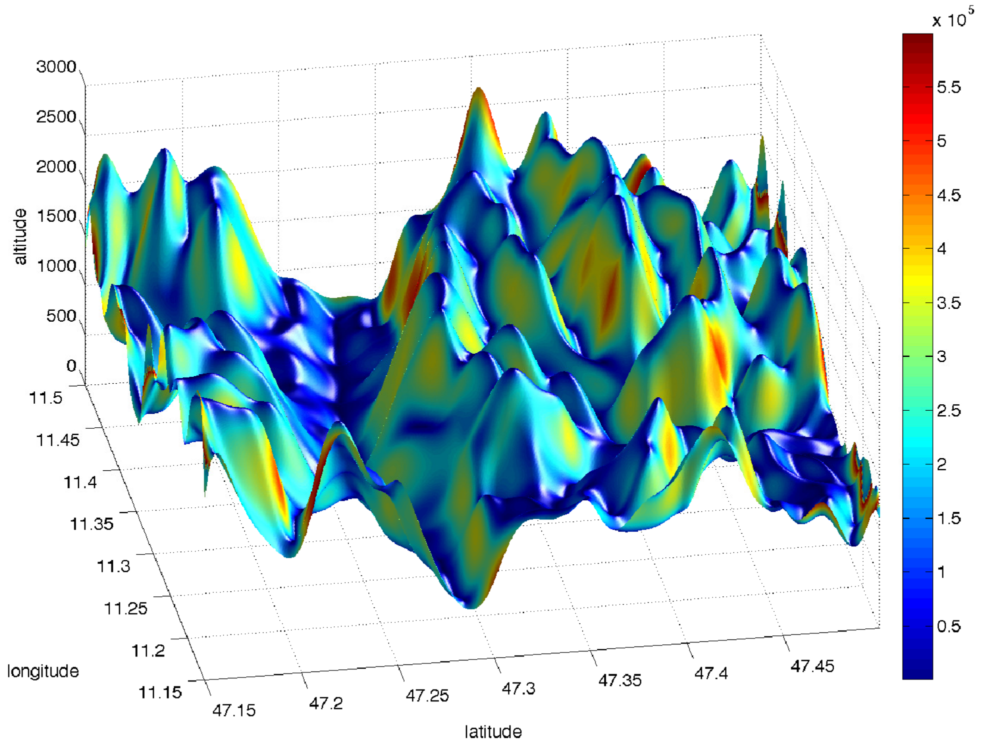


Figure 3.15: Visualization of gradients of the simplex spline function $s_{\mathcal{T}} \in \mathcal{S}^2_6$. The surface takes colouring based on the gradients magnitudes.

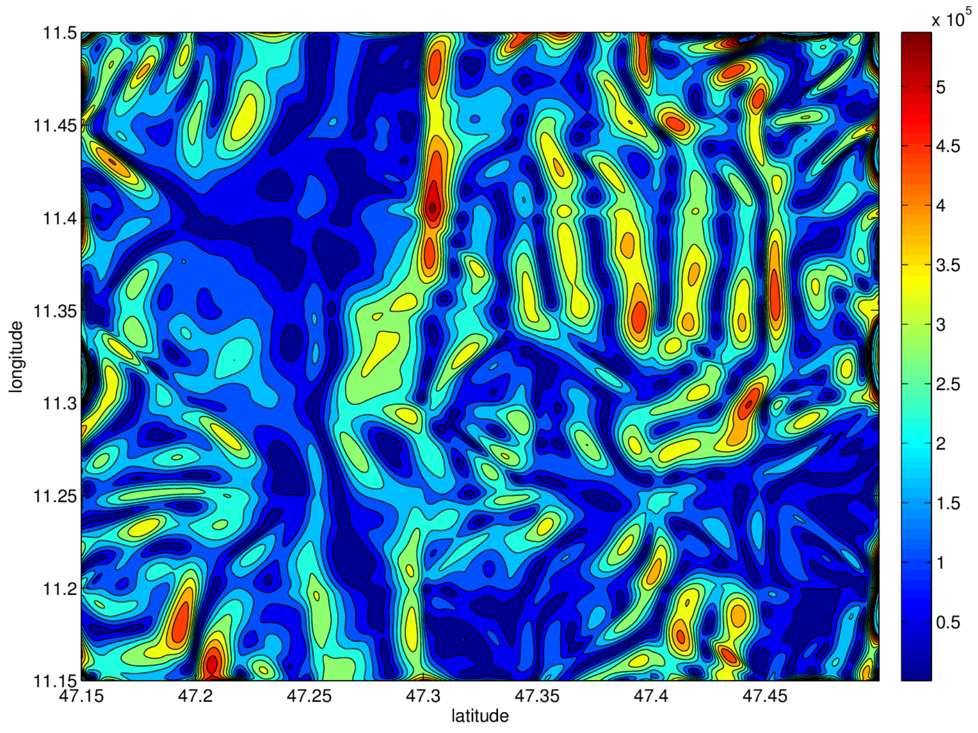


Figure 3.16: Visualization of the divergence vector field of the spline function $s_{\mathcal{T}} \in \mathcal{S}^2_6$. The smooth isolines (black) represent high smoothness of the simplex spline function.

Chapter 4

Conclusions

The aim of this thesis was to introduce the multivariate simplex splines theory into the framework of the terrain data interpolation.

The theoretical background of the simplex splines was provided in the chapter 2. This part was based mainly on the papers of de Visser [9] and Lai & Schumaker [15] where the concept of multivariate simplex splines is fully introduced together with some practical applications. The idea of the theory is relatively old, but was never widely used. The concept of the B-form polynomials defined in terms of barycentric coordinates on many simplices is dated back to Farin and to de Boor, to the mathematicians who published first papers on this topic in 1980s. Their definitions were in general correct, but for practical usage some reformulations are needed. This is the case of B-net orientation rule discussed in the section 2.1.5 as well as the smoothness constrains 2.3.2. The estimation of the smoothness matrix can be considered as the critical part of the implemented algorithm as the calculation takes a lot of computational time. Therefore, in the section 2.3.3 the practical algorithm was proposed. Finally, the formulation of the simplex spline function in the form of the linear regression model was derived in 2.4, that allows to use well known linear equality-constrained least squares estimator for calculation of the B-coefficients 2.4.2.

The multivariate simplex splines are truly a general interpolation and approximation tool. In the chapter 3 many numerical experiments were presented that aimed to demonstrate the behaviour of this method more closely. Input parameters having a strong impact on the resulting simplex spline model were described in individual sections. In the final section of this thesis the interpolation of the terrain dataset was presented. It was proved that the simplex spline theory provides an effective tool for terrain approximation by a continuous function, resulting to the models of a high precision.

The application of the implemented algorithm is obvious. As many approaches for the data analyses are based on various differential methods that require the continuous model, the implemented function can be used as a part of these systems. It should be clear by now, that the simplex splines have very strong potential in many different areas. As all important parts of the theory are defined in an n-dimensional space e.g. the B-form polynomials, the triangulation, the linear regression, the concept works easily in any number of dimensions. Moreover, the B-form polynomials provide very elegant formulation of many mathematical operations as can be seen in the section 2.5. It is also important to mention, that one of the main advantages of the multivariate simplex splines is the scattered data modelling as the triangulation can be calculated on a domain of any shape.

The multivariate simplex splines could find its place also in some parts of computer graphics. The linear regression provides very effective approximation tool as the minimal number of points within every simplex is equal to the number of free variables of an interpolation polynomial. The number of points required for complex polygonal models could be possibly smaller. Moreover, the real-time approximation is very actual these days. In order to obtain fast real-time implementation the modification of the least square estimator is needed. All of these ideas and many others will be subject to the future researches.

Appendix A

Demostration examples

Example A.1 *Given a 2-simplex, that is a triangle, with vertices* $A = \left(1, \frac{1}{2}\right)$*,* $B = (4, 1)$ *and* $C = (2, 3)$*. We want to find barycentric coordinates of point* $X = \left(2, \frac{3}{2}\right)$ *with respect to the* $\triangle ABC$*.*
Firstly, the $\mathbf{A}_t$ *matrix from* (2.14) *need to be constructed:*

$$\begin{aligned}\mathbf{A}_t &= \left[\mathbf{B}-\mathbf{A} \quad \mathbf{C}-\mathbf{A}\right] \\ &= \left[\begin{bmatrix} 4 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ \frac{1}{2} \end{bmatrix} \quad \begin{bmatrix} 2 \\ 3 \end{bmatrix} - \begin{bmatrix} 1 \\ \frac{1}{2} \end{bmatrix}\right] \\ &= \begin{bmatrix} 3 & 1 \\ \frac{1}{2} & \frac{5}{2} \end{bmatrix}.\end{aligned}$$

After inverting matrix $\mathbf{A}_t$*, we can find last two elements of* $\mathbf{b}$*:*

$$\begin{aligned}\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} &= \mathbf{A}_t^{-1} \cdot [\mathbf{X}-\mathbf{A}] \\ &= \begin{bmatrix} 0.3571 & -0.1429 \\ -0.0714 & 0.42861 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 0.2143 \\ 0.3571 \end{bmatrix}.\end{aligned}$$

And finally, after using (2.10)*:*

$$\begin{aligned}\mathbf{b} &= \left(1 - (b_1 + b_2), b_1, b_2\right) \\ &= (0.4286, 0.2143, 0.3571).\end{aligned}$$

All components of the barycentric coordinate vector $\mathbf{b}$ *are greater than zero, thereby, the point* X *is located within the* 2*-simplex* ABC*.*

As mentioned before, every component of the barycentric vector $\mathbf{b}$ *defines a subarea of the given* 2*-simplex* ABC*. In the figure A.1 the* $\triangle ABC$ *and point* X *are pictured. The first component of the barycentric vector* $\mathbf{b}$*,* $b_0 = 0.4286$ *corresponds to the area of* $\triangle XBC$*, the second component* $b_1 = 0.2143$ *to* $\triangle XCA$ *and, the third component* $b_2 = 0.3571$ *to* $\triangle XAB$*.*

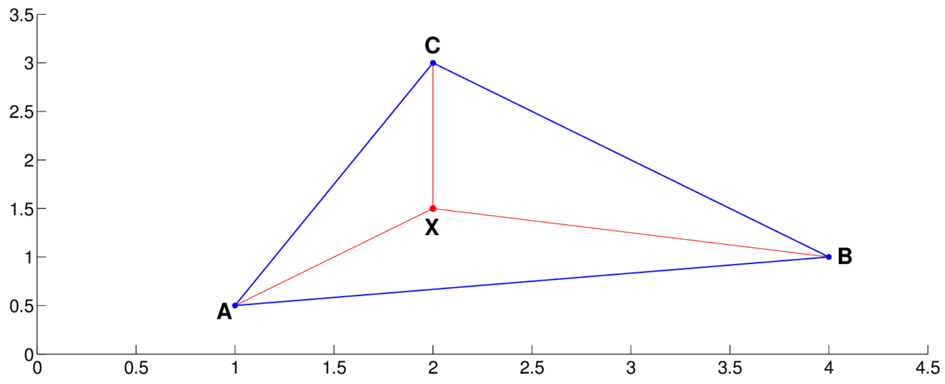


Figure A.1: Triangle ABC and location of point X.

Finally, the barycentric coordinates of vertices of $\triangle ABC$ *are* $A = (1,0,0), B = (0,1,0), C = (0,0,1)$. *Point* G *placed in the center of mass of* $\triangle ABC$ *has coordinates* $G = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ *as areas of all subtriangles must be same.*

Example A.2 *Demonstration of a trivariate third degree Bernstein polynomial. The B-form of any trivariate polynomial is given by* (2.27):

$$p(\mathbf{x}) = \sum_{|k|=3} c_k B^3_k(b(\mathbf{x})), \tag{A.1}$$

where $b(\mathbf{x}) = \mathbf{b}$ *is transformation from Cartesian system to barycentric one. Notice please, that the input point* $\mathbf{x}$ *contains only two elements, but as the Bernstein polynomial is of three variables the third input value is obtained from the transformation* $b(\mathbf{x})$. *According to* (2.26) *the B-form may be expanded as follows:*

$$\begin{aligned}
p(\mathbf{x}) = &\sum_{|k|=3} c_k B^3_k(b(\mathbf{x})) \\
=&c_{300}B^3_{300}(\mathbf{b}) + \\
&c_{210}B^3_{210}(\mathbf{b}) + c_{201}B^3_{201}(\mathbf{b}) + \\
&c_{120}B^3_{120}(\mathbf{b}) + c_{111}B^3_{111}(\mathbf{b}) + c_{102}B^3_{102}(\mathbf{b}) + \\
&c_{030}B^3_{030}(\mathbf{b}) + c_{021}B^3_{021}(\mathbf{b}) + c_{012}B^3_{012}(\mathbf{b}) + c_{003}B^3_{003}(\mathbf{b}) \\
=&c_{003}b_0^3 + c_{210}3b_0^2b_1 + c_{201}3b_0^2b_2 + c_{120}3b_0b_1^2 + c_{111}6b_0b_1b_2 + \\
&c_{102}3b_0b_2^2 + c_{030}b_1^3 + c_{021}3b_1^2b_2 + c_{012}3b_1b_2^2 + c_{003}b_2^3
\end{aligned}$$

To visualize every individual Bernstein basis polynomial $B^3_k(b(\mathbf{x}))$ *defined in terms of the barycentric coordinates a corresponding* 2*-simplex need to be given. Consider a very simple set* $\mathcal{X}_3$ *of three points and the* 2*-simplex* t *according to* (2.3):

$$\begin{aligned}
\mathcal{X}_3 &= \{(-1,-1), (-1,1), (1,-1)\}, \\
t &= \langle \mathcal{X}_3 \rangle .
\end{aligned}$$

In the figure A.2 is plotted by red line the 2-simplex t with vertices from the set $\mathcal{X}_3$. All plots in this figure show Bernstein basis polynomials $B_k^3(\mathbf{b})$. The sum of these polynomial is equal to 1 at every lcoation within the simplex t as was mentioned in the end of the section 2.1.3. This concept is called partition of unity of the polynomial basis and is demonstrated in the figure A.3.

Example A.3 (The B-net Orientation Rule) *Determine the orientation of B-nets of three simplices $t_i = \langle v_a, v_b, v_c \rangle$, $t_j = \langle v_b, v_c, v_d \rangle$, $t_j = \langle v_c, v_d, v_e \rangle$ with vertices globally sorted as follows:*

$$v_a > v_d > v_b > v_c > v_e.$$

Using the expression (2.29), it is easy to determine B-coefficients located at every vertex. The result is figured in 2.3. Spatial locations of other B-coefficients are calculated using (2.28) always with respect to the associated simplex.

Example A.4 *The total number of continuity conditions for a C^r continuity is given by (2.40). In this example all 4 continuity orders as shown in the figure 2.7 are considered for $n = 2$ and $d = 4$.*

For C^0 continuity we have:

$$R_0 = \frac{5!}{1!4!} = 5. \tag{A.2}$$

Five continuity conditions are required for the C^0 as all B-coefficients laying in the edge facet are involved.

For C^1 continuity we have:

$$R_1 = \sum_{m=0}^{1} \frac{(4-m+2-1)!}{(2-1)!(4-m)!} = 5 + 4 = 9. \tag{A.3}$$

A fundamental observation should be noticed at this point. Let's consider the line through locations of B-coefficients $c^{t_i}_{1,3,0}$, $c^{t_i}_{1,2,1}$, $c^{t_i}_{1,1,2}$ and through $c^{t_i}_{1,0,3}$. This line is clearly parallel to the edge facet $\tilde{t}_{i,j}$. Every B-coefficient lying in this line together with every B-coefficient located in the edge facet $\tilde{t}_{i,j}$ is subject to a single continuity condition see the figure A.4. For higher continuity orders this pattern is followed. In general there are $d+1$ parallel planes of dimension $d-1$. The proof of (2.40) is based on this observation.

For completeness, the continuity orders C^2 and C^3 require:

$$R_2 = \sum_{m=0}^{2} \frac{(4-m+2-1)!}{(2-1)!(4-m)!} = 5 + 4 + 3 = 12,$$
$$R_3 = \sum_{m=0}^{3} \frac{(4-m+2-1)!}{(2-1)!(4-m)!} = 5 + 4 + 3 + 2 = 14$$

continuity conditions.

Example A.5 *According to (2.39) the parts of continuity conditions up to order C^2 for the edge facet $\tilde{t}_{i,j}$ between t_i and t_j and for continuity points $c^{t_i}_{0,0,4}$, $c^{t_i}_{1,1,2}$, $c^{t_j}_{2,2,0}$ from the figure 2.7 will be derived.*

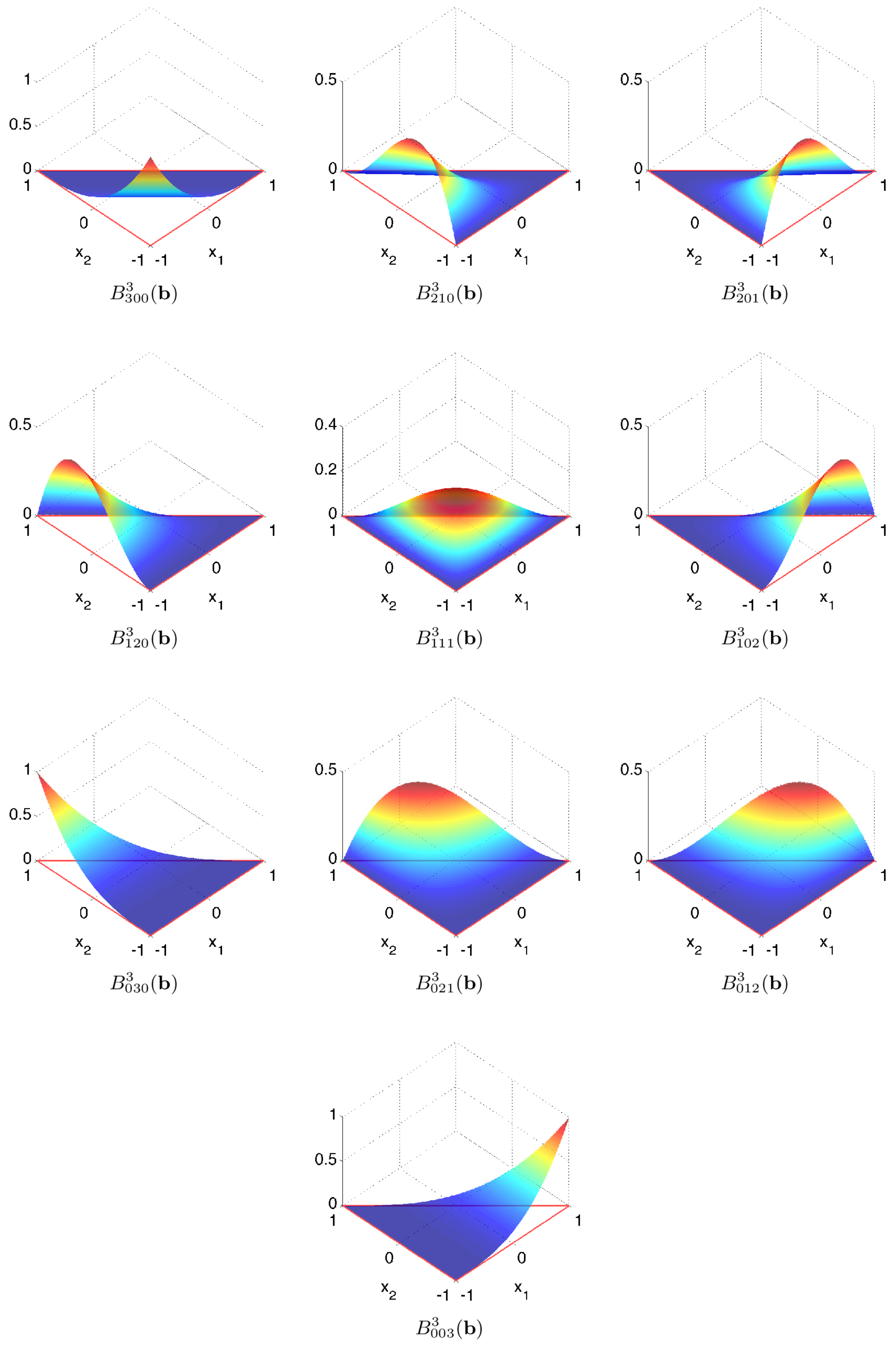


Figure A.2: Visualization of the individual Bernstein basis polynomials $B^3_k(\mathbf{b})$ defined on a single 2-simplex that is plotted by the red line.

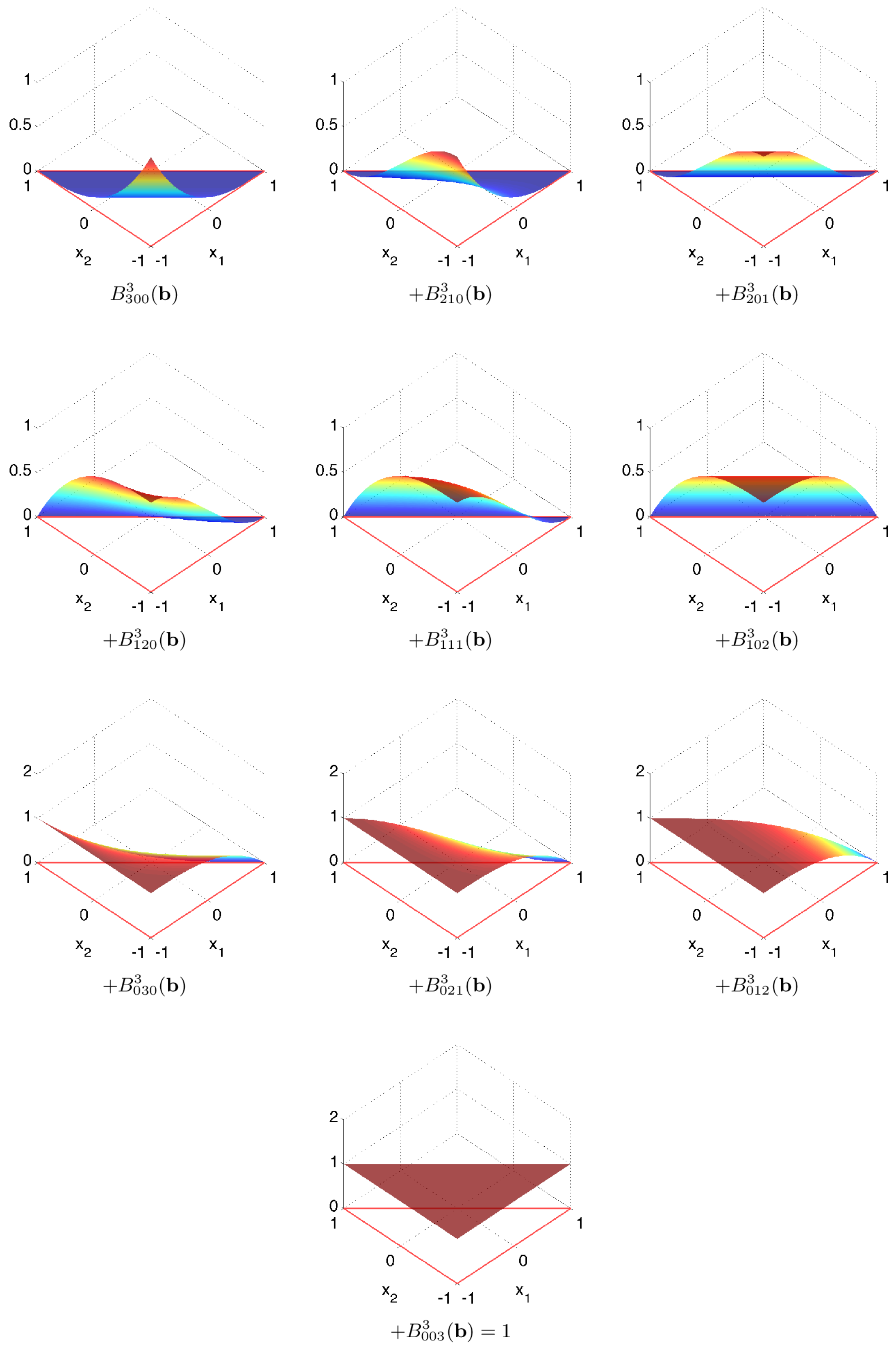


Figure A.3: Demonstration of the partition of unity property. The Bernstein polynomial is the same as in the previous figure i.e. $\sum_{|k|=3} B^3_k(\mathbf{b})$.

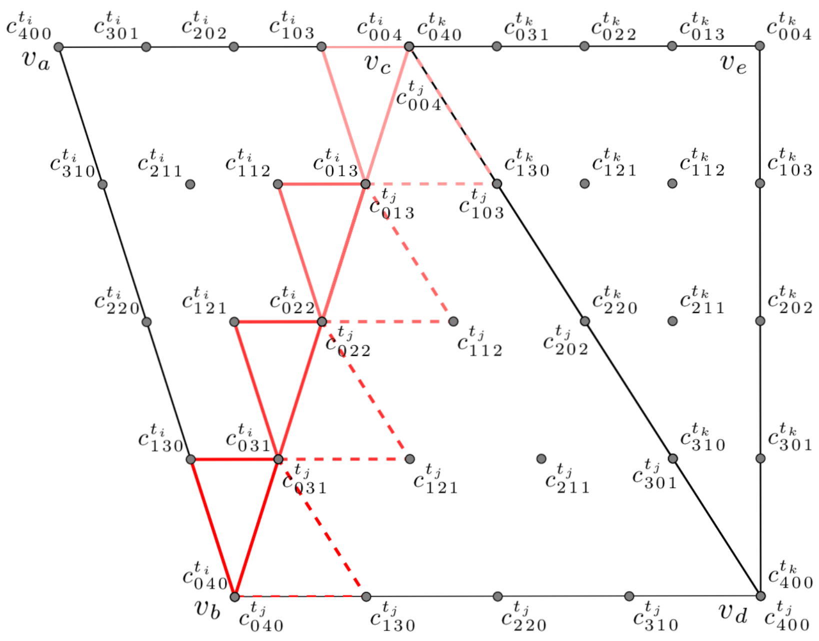


Figure A.4: The spatial interpretation of continuity conditions C^1 on a Bernstein polynomial of degree 4. There are 4 continuity points (dashed line) and corresponding continuity bodies (solid line). Together it makes 9 continuity conditions.

For C^0 continuity at $c^{t_i}_{0,0,4}$ we have:

$$\gamma \in \{(0,0,0)\}.$$

For this case there is only single continuity condition:

$$\begin{aligned} c^{t_i}_{0,0,4} &= \sum_{|\gamma|=0} c^{t_j}_{(0,0,4)+\gamma} B^0_\gamma(\mathbf{v}_a) \\ &= c^{t_j}_{(0,0,4)} \end{aligned}$$

For C^1 continuity at $c^{t_i}_{1,1,2}$ we have:

$$\gamma \in \{(1,0,0),(0,1,0),(0,0,1)\}.$$

The C^1 continuity condition then is:

$$\begin{aligned} c^{t_i}_{1,1,2} &= \sum_{|\gamma|=1} c^{t_j}_{(0,1,2)+\gamma} B^1_\gamma(\mathbf{v}_a) \\ &= c^{t_j}_{(1,1,2)} B^1_{1,0,0}(\mathbf{v}_a) + c^{t_j}_{(0,2,2)} B^1_{0,1,0}(\mathbf{v}_a) + c^{t_j}_{(0,1,3)} B^1_{0,0,1}(\mathbf{v}_a). \end{aligned}$$

At last, for C^2 continuity at point $c^{t_j}_{2,2,0}$:

$$\gamma \in \{(2,0,0),(1,1,0),(1,0,1),(0,2,0),(0,1,1),(0,0,2)\}.$$

and continuity condition then is:

$$\begin{aligned} c^{t_j}_{2,2,0} &= \sum_{|\gamma|=2} c^{t_i}_{(0,2,0)+\gamma} B^2_\gamma(\mathbf{v}_d) \\ &= c^{t_i}_{(2,2,0)} B^2_{2,0,0}(\mathbf{v}_d) + c^{t_i}_{(1,3,0)} B^2_{1,1,0}(\mathbf{v}_d) + c^{t_i}_{(1,2,1)} B^2_{1,0,1}(\mathbf{v}_d) + \\ &\quad + c^{t_i}_{(0,4,0)} B^2_{0,2,0}(\mathbf{v}_d) + c^{t_i}_{(0,3,1)} B^2_{0,1,1}(\mathbf{v}_d) + c^{t_i}_{(0,2,2)} B^2_{0,0,2}(\mathbf{v}_d). \end{aligned}$$

A quick look at the figure 2.7 will show that calculated results are truly valid and therefore it is possible to estimate C^r continuity conditions between simplices t_i and t_j by using (2.39). However, there are cases when this expression fails and resulting continuity body for given continuity point is wrong. One example is the edge facet $\tilde{t}_{j,k}$ between t_j and t_k. This observation is demonstrated by the example A.6 together with a short discussion. The solution of this problem will be subject to the following sections.

Example A.6 *Demonstration of incompleteness of expression of continuity conditions from (2.39) as it holds only for specific orientation of the B-nets between adjoined simplices. In this example the continuity conditions between t_k and t_j from the figure 2.7 is calculated and invalid results will be proved.*

Given C^0 continuity (cyan color) at $c^{t_k}_{0,4,0}$ we have single multi-index permutation, $m = 0$:

$$\gamma = (0,0,0).$$

According to the (2.39) the C^0 continuity condition then is:

$$\begin{aligned} c^{t_k}_{0,4,0} &= \sum_{|\gamma|=0} c^{t_j}_{(0,4,0)+\gamma} B^0_\gamma(\mathbf{v}_e) \\ &= c^{t_j}_{0,4,0} \end{aligned}$$

which is not a valid result as $c^{t_j}_{0,4,0}$ is not located at the edge facet $\tilde{t}_{k,j}$.

In another case with continuity C^3 at $c^{t_k}_{1,0,3}$ valid multi-indices are, $m = 1$:

$$\gamma \in \{(1,0,0),(0,1,0),(0,0,1)\}$$

The C^3 continuity order now is:

$$\begin{aligned} c^{t_k}_{1,0,3} &= \sum_{|\gamma|=3} c^{t_j}_{(0,0,3)+\gamma} B^3_\gamma(\mathbf{v}_e) \\ &= c^{t_j}_{1,0,3} B^1_{1,0,0}(\mathbf{v}_e) + c^{t_j}_{0,1,3} B^1_{0,1,0}(\mathbf{v}_e) + \dots \end{aligned}$$

which is totally wrong as the second right hand term already contains a B-coefficient index that is not at the edge facet $\tilde{t}_{k,j}$, and the number of B-coefficients in continuity body is 3 while according to the (2.40) it must be 9 as was demonstrated by example A.4.

Example A.7 *In this example, iterations of de Casteljau algorithm are demonstrated by evaluating a trivariate B-form polynomial of degree $d = 3$ at the barycentric coordinate $\mathbf{b} = (b_0, b_1, b_2)$. The B-form polynomial is therefore same as the one from A.2.*

Using (2.71), the iteration of the de Casteljau algorithm are:

$$\begin{aligned}
p(\mathbf{b}) &= \sum_{|k|=3} c_k B^3_k(\mathbf{b}) \\
&= \sum_{|k|=2} c^{(1)}_k(\mathbf{b}) B^2_k(\mathbf{b}) \\
&= \sum_{|k|=1} c^{(2)}_k(\mathbf{b}) B^1_k(\mathbf{b}) \\
&= c^{(3)}_{000}(\mathbf{b})
\end{aligned}$$

Now, expansion of all B-coefficients of every iteration according to (2.72) is given. Note, that in (2.72) is the multi-index permutation $|\gamma| = 1$ therefore for all iterations this set is equal to:

$$\gamma \in \{(1,0,0),(0,1,0),(0,0,1)\}.$$

Expansion of $c^{(1)}_k(\mathbf{b})$ with $k \in \{(2,0,0),(1,1,0),(1,0,1),(0,2,0),(0,1,1),(0,0,2)\}$:

$$\begin{aligned}
c^{(1)}_{200}(\mathbf{b}) &= b_0 c_{300} + b_1 c_{210} + b_2 c_{201} \\
c^{(1)}_{110}(\mathbf{b}) &= b_0 c_{210} + b_1 c_{120} + b_2 c_{111} \\
c^{(1)}_{101}(\mathbf{b}) &= b_0 c_{201} + b_1 c_{111} + b_2 c_{102} \\
c^{(1)}_{020}(\mathbf{b}) &= b_0 c_{120} + b_1 c_{030} + b_2 c_{021} \\
c^{(1)}_{011}(\mathbf{b}) &= b_0 c_{111} + b_1 c_{021} + b_2 c_{012} \\
c^{(1)}_{002}(\mathbf{b}) &= b_0 c_{102} + b_1 c_{012} + b_2 c_{003}
\end{aligned}$$

For $c^{(2)}_k(\mathbf{b})$ and for $k \in \{(1,0,0),(0,1,0),(0,0,1)\}$ we have:

$$\begin{aligned}
c^{(2)}_{100}(\mathbf{b}) &= b_0 c^{(1)}_{200}(\mathbf{b}) + b_1 c^{(1)}_{110}(\mathbf{b}) + b_2 c^{(1)}_{101}(\mathbf{b}) \\
c^{(2)}_{010}(\mathbf{b}) &= b_0 c^{(1)}_{110}(\mathbf{b}) + b_1 c^{(1)}_{020}(\mathbf{b}) + b_2 c^{(1)}_{011}(\mathbf{b}) \\
c^{(2)}_{001}(\mathbf{b}) &= b_0 c^{(1)}_{101}(\mathbf{b}) + b_1 c^{(1)}_{011}(\mathbf{b}) + b_2 c^{(1)}_{002}(\mathbf{b})
\end{aligned}$$

And finally, the last iteration $c^{(3)}_k(\mathbf{b})$ with $k \in \{(0,0,0)\}$ is:

$$c^{(3)}_{000}(\mathbf{b}) = b_0 c^{(2)}_{100}(\mathbf{b}) + b_1 c^{(1)}_{010}(\mathbf{b}) + b_2 c^{(2)}_{001}(\mathbf{b}).$$

It is easy to check now, that after substitution $c^{(1)}_k(\mathbf{b})$ and $c^{(2)}_k(\mathbf{b})$ to $c^{(3)}_k(\mathbf{b})$ the result is exactly the same as from A.2.

Example A.8 *Construction of the de Casteljau one-step matrix is demonstrated by this example using expression from (2.83). The de Casteljau matrix $\mathbf{P}^{3,1}(\mathbf{b})$ reduces a set of B-coefficients of degree $d = 3$ to degree $d - m = 1$, such that $m = 2$. The bivariate B-form polynomial if defined in terms of barycentric coordinate $\mathbf{b} = (b_0, b_1)$.*

In this case we have for multi-index k:

$$k \in \{(1,0),(0,1)\},\ |k| = d - m.$$

The index function $i(k)$ from (2.84) for the rows of $\mathbf{P}^{3,1}(\mathbf{b})$ therefore has the following values:

$$i(1,0)=1,\ i(0,1)=2.$$

The valid values for the multi-index θ are:

$$\theta \in \{(3,0),(2,1),(1,2),(0,3)\},\ |\theta|=d,$$

with which the index function $i(\theta)$ from (2.85) for the columns of $\mathbf{P}^{3,1}(\mathbf{b})$ has following values:

$$i(3,0)=1,\ i(2,1)=2,\ i(1,2)=3,\ i(3,0)=4.$$

Expanding row and column indices generated by $i(k)$ and $i(\theta)$, and calculating the multi-index of the corresponding polynomials, we get:

$i(k)$	$i(\theta)$	θ	k	$\theta-k$	$P^m_{\theta-k}(\mathbf{b})$
1	1	$(3,0)$	$(1,0)$	$(2,0)$	$P^2_{2,0}(\mathbf{b})$
1	2	$(2,1)$	$(1,0)$	$(1,1)$	$P^2_{1,1}(\mathbf{b})$
1	3	$(1,2)$	$(1,0)$	$(0,2)$	$P^2_{0,2}(\mathbf{b})$
1	4	$(0,3)$	$(1,0)$	$(-1,3)$	0
2	1	$(3,0)$	$(0,1)$	$(3,-1)$	0
2	2	$(2,1)$	$(0,1)$	$(2,0)$	$P^2_{2,0}(\mathbf{b})$
2	3	$(1,2)$	$(0,1)$	$(1,1)$	$P^2_{1,1}(\mathbf{b})$
2	4	$(0,3)$	$(0,1)$	$(0,2)$	$P^2_{0,2}(\mathbf{b})$

By rewriting the table values to matrix form the following result is obtained:

$$\mathbf{P}^{3,1}(\mathbf{b}) = \begin{bmatrix} P^2_{2,0}(\mathbf{b}) & P^2_{1,1}(\mathbf{b}) & P^2_{0,2}(\mathbf{b}) & 0 \\ 0 & P^2_{2,0}(\mathbf{b}) & P^2_{1,1}(\mathbf{b}) & P^2_{0,2}(\mathbf{b}) \end{bmatrix}.$$

Bibliography

[1] I. Anderson, M. Cox, and J. Mason. Tensor-product spline interpolation to data on or near a family of lines. *Numerical Algorithms*, pages 193–204, 1993.

[2] A.R. and Forrest. Interactive interpolation and approximation by bézier polynomials. *Computer-Aided Design*, 1990.

[3] Gerard M. Awanou. *Energy Methods in 3D spline approximations of the Navier-Strokes Equations.* PhD thesis, University of Georgia, 2003.

[4] F.L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:567–585, 1989.

[5] C. W. Clenshaw and J. G. Hayes. Curve and surface fitting. *IMA Journal of Applied Mathematics*, pages 164–183, 1965.

[6] Carl de Boor. B–form basics. In G. E. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 131–148. SIAM Publications, 1986.

[7] Carl de Boor and Amos Ron. On multivariate polynomial interpolation, 1990.

[8] C. C. de Visser, Q. P. Chu, and J. A. Mulder. A new approach to linear regression with multivariate splines. *Automatica*, 2009.

[9] C.C. de Visser. *Global Nonlinear Model Identification with Multivariate Splines.* PhD thesis, Delft University of Technology, 2011.

[10] G.E. Farin. *Subsplines über Dreiecken.* Braunschweig, 1979.

[11] Gerald Farin. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3:83–127, 1986.

[12] Gerald Farin. *Curves and Surfaces for CAGD.* Morgan Kaufmann, 2001.

[13] J.D. Foley. *Computer Graphics: Principles and Practice.* Addison-Wesley, 1996.

[14] Ming-J Un Lai. Geometric interpretation of smoothness conditions of triangular polynomial patches. *CAGD*, 14:191–199, 1997.

[15] Ming-Jun Lai and Larry L. Schumaker. *Spline Functions on Triangulations.* Cambridge University Press, 2007.

[16] C.L. Lawson and R.J. Hanson. *Solving Least Squares Problems.* Classics in Applied Mathematics. SIAM, 1974.

[17] D. Lee and A. Lin. Generalized delaunay triangulation for planar graphs. *Discrete and Computational Geometry*, pages 201–217, 1986.

[18] Michael David McCool. *Analytic Signal Processing for Computer Graphics using Multivariate Polyhedral Splines.* PhD thesis, University of Toronto, 1995.

[19] Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-Spline Techniques.* Springer, 2002.