Intersplines: A New Approach to Globally Optimal Multivariate Splines Using Interval Analysis

C. C. de Visser
Delft University of Technology, Delft, The Netherlands
c.c.devisser@tudelft.nl

E. van Kampen
Delft University of Technology, Delft, The Netherlands
e.vankampen@tudelft.nl

Q. P. Chu
Delft University of Technology, Delft, The Netherlands
q.p.chu@tudelft.nl

J. A. Mulder
Delft University of Technology, Delft, The Netherlands
j.a.mulder@tudelft.nl

Abstract

In science and engineering there often is a need for the approximation of scattered multi-dimensional data. A class of powerful scattered data approximators are the multivariate simplex B-splines. Multivariate simplex B-splines consist of Bernstein basis polynomials that are defined on a geometrical structure called a triangulation. Multivariate simplex B-splines have a number of advantages over other scattered data approximators, like neural networks and kernel methods. Firstly, the multivariate simplex B-splines have an arbitrarily high approximation power. Secondly, the simplex spline models are parametric models, which allows for efficient approximation of arbitrarily large datasets. Finally, the local basis property of the simplex B-splines results in sparse solution and evaluation systems. Until now, the disadvantage of using simplex B-splines for scattered data approximation is that a triangulation must be defined a-priori which can not be guaranteed to be optimal. This triangulation optimization problem has been a long standing issue in the field of multivariate splines. In this article, a new interval form of the multivariate simplex

B-splines is introduced which aims to solve the triangulation optimization problem. This new form is the interval spline, or Interspline, which consists of interval B-coefficients which scale interval Bernstein basis polynomials defined on interval simplices. A branch and bound optimization scheme can then be used to concurrently determine the guaranteed optimal set of B-coefficients and triangulation vertex locations given a set of scattered multidimensional data, thus solving the triangulation optimization problem.

Keywords: multivariate splines; interval analysis; triangulation optimization; function approximation; global optimization
AMS subject classifications: 65D07,65D15,65G40,90C26

1 Introduction

Accurate models of dynamic systems are essential to many applications in science and engineering. Currently, there are only a handful of methods available for modeling such systems. The most widely used of these methods are neural networks, kernel methods, polynomial blending methods, and spline methods. Neural networks are powerful function approximators that have been used successfully in the past, see e.g. [8, 29, 17, 19, 28, 25]. Neural networks are black box models, and therefore suffer from an inherent intransparency [5]. The result of this is that their performance can only be guaranteed under special circumstances [7]. Additionally, the radial basis functions in a neural network have a global influence on neural network output. As a consequence, all basis functions and coefficients need to be considered during evaluation and estimation, resulting in computationally inefficient evaluation and solution systems.

Kernel methods like the support vector machines can also be used for scattered nonlinear data modeling [9, 14, 31, 15]. While powerful in their role as categorizers, kernel methods are non-parametric in nature and require that every significant data point is associated with a single kernel function, with the result that the size of the optimization problem is proportional to the total number of data points [9]. This in turn produces computationally expensive optimization problems, especially when the modeled datasets are large.

Polynomial modeling methods are perhaps the simplest and the most widely used of all modeling methods. While a single polynomial has only a limited approximation power, accurate global models can still be created by estimating local models on subregions of the system operating domain. This, however, introduces discontinuities in the model which can present a problem if it is to be used in some model based controller. Continuity can be restored by employing a blending scheme like Takagi–Sugeno fuzzy blending [1, 5]. While fuzzy blending techniques are powerful, their construction and tuning is done based on expert knowledge which means that they will probably never result in a fully automated identification technique [5].

Finally, polynomial spline methods have been used in the past for the modeling of nonlinear systems, see e.g. [34, 20, 6]. However, these spline methods employed multivariate tensor product B-splines which due to their rectangular nature are incapable of fitting scattered data and are confined to rectangular domains, limiting their applicability to rectangular datasets [2].

Multivariate simplex B-splines are a type of multivariate splines which are capable of accurately approximating scattered nonlinear data [4, 23, 12, 11, 13]. The multivariate simplex B-splines have a stable local polynomial basis consisting of Bernstein basis

polynomials. The Bernstein basis polynomials are functions in terms of barycentric coordinates. The barycentric coordinate system is an affine local coordinate system which is native to the simplex, a geometric structure that minimally spans a given set of dimensions. As such, each simplex B-spline basis function is defined on a single simplex. The true power of the multivariate simplex B-spline comes from connecting any number of simplices into a geometric structure called a triangulation [23].

The simplex B-splines have a number of important advantages over the earlier mentioned modeling methods. Firstly, the simplex B-splines have an arbitrarily high approximation power on a global model scale. Secondly, simplex B-spline models are parametric models, which allows for efficient approximation of very large datasets. Thirdly, the simplex B-splines are linear in the parameters, which means that linear regression methods like generalized least squares and recursive least squares can be used in their estimation [12, 11]. Finally, the simplex B-splines have a local polynomial basis, which implies that only small subsets of parameters and basis polynomials need to be considered during estimation and evaluation, resulting in efficient computational schemes.

However, there exists a gap in the theory of the multivariate simplex B-spline which is the absence of a rigorous method for triangulation optimization. The difficulty with triangulation optimization lies in its essentially non-convex nature. A number of triangulation optimization methods are present in the literature, most notably the methods for 2-D constrained Delaunay triangulation (CDT) optimization presented by Ruppert [30] and Shewchuk [32], and the general N-D CDT optimization method introduced by Shewchuk in [33]. However, none of these methods are specifically designed for use with simplex B-splines in the sense that the per-simplex data content is not used as an optimization parameter. This particular optimization parameter has been found to be of fundamental importance to the conditioning and solvability of a data approximation problem with simplex B-splines [12, 11, 13].

The objective of this paper is the presentation of a new type of multivariate spline, which we call the multivariate '*Interspline*'. Intersplines are multivariate simplex B-splines with interval B-coefficients which scale individual interval Bernstein basis functions. The interval Bernstein basis functions are formulated in terms of the vertices of interval-simplices. The Intersplines allow the use of interval analysis to determine, in a single step, a globally optimal solution for the combined triangulation optimization and B-coefficient estimation problem.

The idea of using intervals in the formulation of multivariate simplex B-splines is not entirely new. Zhou presented an Interval Simplex Spline (ISS) method in which the B-coefficients of the simplex B-splines are represented by intervals for the purpose of encoding uncertainty in datasets and data reduction [37]. However, Zhou's ISS method only uses intervals for the B-coefficients of the simplex splines, and does not consider using intervals for the simplex vertices and Bernstein basis functions. As a result, the ISS method can not be used for the combined triangulation optimization and B-coefficient estimation problem for which our Interspline method has been developed.

The Intersplines are enabled by a new formulation of the Bernstein basis polynomials of the multivariate simplex B-splines in terms of global coordinates. This new formulation effectively combines the simplex vertices with the spline polynomial coefficients into a single set of optimization parameters. The resulting Interspline function produces the guaranteed optimal spline function defined on the guaranteed optimal triangulation, thereby closing the gap in multivariate simplex B-spline theory that has been present since its inception. This paper is primarily focused on the linear Interspline, which consists of linear spline pieces. The principles presented in this paper

are general, however, and can also be used to define nonlinear Intersplines.

The outline of this paper is as follows. First, in Sec. 2 an introduction into the theory of the multivariate simplex B-splines is provided. Following this, Sec. 3 presents a new formulation of the Bernstein basis polynomials of the multivariate simplex B-splines in terms of global coordinates. In Sec. 4 a brief introduction on interval analysis is given. In Sec. 5 the problem of triangulation optimization is introduced, and a theorem is presented for the non-convex nature of the combined B-coefficient and triangulation optimization problem. In Sec. 6 the multivariate Interspline is introduced. In Sec. 7 the results from a number of numerical experiments are presented. Finally, in Sec. 8 the paper is concluded and recommendations for future work are provided.

2 Preliminaries on Multivariate Simplex B-splines

This section serves as a brief introduction into the mathematical theory of the multivariate simplex B-spline. For a more in-depth coverage of the mathematical theory, we would like to refer to [23].

2.1 The Simplex and Barycentric Coordinates

The individual polynomial pieces of the simplex B-spline are defined on simplices. A simplex is a geometric structure that provides a minimal, non-degenerate span of n-dimensional space. For example, lines are 1-dimensional simplices, triangles are 2-dimensional simplices, and tetrahedrons 3-dimensional simplices. A simplex is defined as follows. Let $\mathcal{V}$ be a set of $n+1$ unique, non-degenerate, points in n-dimensional space:

$$\mathcal{V} := \{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n\} \quad \subset \mathbb{R}^n, \tag{1}$$

with each $\mathbf{v}_i$ a vector in $\mathbb{R}^n$.

The convex hull of $\mathcal{V}$ is the n-simplex t:

$$t := \langle \mathcal{V} \rangle \,, \tag{2}$$

with $\langle \bullet \rangle$ the convex hull operator. The boundary edges of a simplex are called *facets*. A facet of an n-simplex is an $(n-1)$-simplex by definition; it is constructed from all but one of the vertices of the n-simplex.

The simplex supports its own local coordinate system in the form of the barycentric coordinate system. The barycentric coordinate system is instrumental in the definition of the stable local polynomial basis for the multivariate splines. The principle of barycentric coordinates is the following: every point $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ inside or outside a simplex t, with t as in Eq. 2, can be described in terms of a unique weighted vector sum of the vertices of t. The barycentric coordinate $b(\mathbf{x}) = (b_0, b_1, \ldots, b_n)$ of $\mathbf{x}$ with respect to simplex t are these vertex weights:

$$\mathbf{x} = \sum_{i=0}^{n} b_i \mathbf{v}_i. \tag{3}$$

The barycentric coordinates are normalized, i.e.

$$\sum_{i=0}^{n} b_i = 1. \tag{4}$$

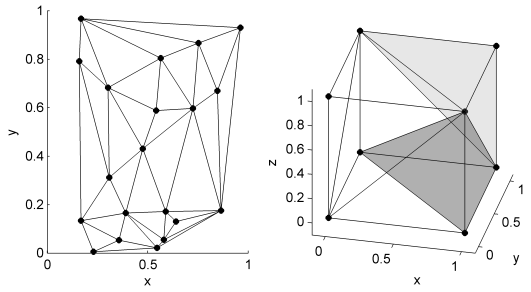

Fig. 1 2-D Delaunay triangulation consisting of 31 triangles (left) and 3-D Delaunay triangulation consisting of 6 tetrahedrons (right, two tetrahedrons are shaded for clarity).

2.2 Triangulations of Simplices

A triangulation $\mathcal{T}$ is a special partitioning of a domain into a set of J non-overlapping simplices.

$$\mathcal{T} := \bigcup_{i=0}^{J} t_i. \tag{5}$$

In a valid triangulation we have that for each pair of simplices t_i and t_j that their intersection is void or a k-simplex $\tilde{t}_{ij}$ with $0 \leq k \leq n-1$, as follows:

$$t_i \cap t_j \in \left\{\emptyset, \tilde{t}_{ij}\right\}, \quad \forall t_i, t_j \in \mathcal{T}, \tag{6}$$

One of the most common triangulation methods is the Delaunay triangulation, see Fig. 1.

Creating high quality triangulations is not a trivial task, especially in higher dimensions. Methods for creating high quality triangulations are present in the literature, such as for example Shewchuk's constrained Delaunay triangulation (CDT) method [32, 33]. These methods, however, are not designed for creating triangulations that are optimal for use with simplex splines as they do not use the per-simplex data coverage as an optimization parameter. In [11] a triangulation optimization method is presented that is specifically designed for use with simplex splines. This method, however, can never be guaranteed to produce optimal triangulations.

2.3 Spline Spaces

A spline space is the space of all spline functions s of a given degree d and continuity order r on a given triangulation $\mathcal{T}$. Such spline spaces have been studied extensively, see e.g. [24, 22, 23]. We use the definition of the spline space from [23]:

$$S_d^r(\mathcal{T}) := \{s \in C^r(\mathcal{T}) : s|_t \in \mathbb{P}_d, \ \forall t \in \mathcal{T}\}, \tag{7}$$

with $\mathbb{P}_d$ the space of all polynomials of total degree d. The definition of the spline space in Eq. 7 provides a convenient notation for stating the degree, continuity and triangulation of a spline solution without having to specify individual spline functions. For example, $S_3^1(\mathcal{T})$ is the space of all cubic spline functions with continuity C^1 defined on the triangulation $\mathcal{T}$.

2.4 The B-form of the Multivariate Simplex Spline

The Bernstein basis polynomial $B^d_\kappa(\mathbf{b})$ of degree d in terms of the barycentric coordinate $\mathbf{b} = (b_0, b_1, \ldots, b_n)$ from Eq. 3 is given by:

$$B^d_\kappa(\mathbf{b}) := \frac{d!}{\kappa!}\mathbf{b}^\kappa, \tag{8}$$

with $\kappa = (\kappa_0, \kappa_1, \ldots, \kappa_n) \in \mathbb{N}^{n+1}$ a *multi-index* with the following notation: $\kappa! := \kappa_0!\kappa_1! \cdots \kappa_n!$ and $|\kappa| := \kappa_0 + \kappa_1 + \cdots + \kappa_n$. In Eq. 8, we use the notation $\mathbf{b}^\kappa = b_0^{\kappa_0} b_1^{\kappa_1} \cdots b_n^{\kappa_n}$. The total number of Bernstein basis functions thus is equal to the total number of valid permutations of κ:

$$\hat{d} := \frac{(d+n)!}{n!d!}. \tag{9}$$

Any polynomial $p^d(\mathbf{b})$ of degree d defined on a simplex t can be written as a linear combination of Bernstein basis polynomials [10]:

$$p^d(\mathbf{b}) := \sum_{|\kappa|=d} c^t_\kappa B^d_\kappa(\mathbf{b}), \quad \mathbf{b} \geq 0, \tag{10}$$

with c^t_κ the so-called *B-coefficients* which uniquely determine $p^d(\mathbf{b})$. The right hand term in Eq. 10 is known as the *B-form* of the multivariate simplex spline in local barycentric coordinates. Note that Eq. 10 is only defined for $\mathbf{b} \geq 0$ which implies a barycentric coordinate inside t.

2.5 Vector Formulations of the B-Form

In [12] a vector formulation for the B-form of the multivariate simplex B-spline was introduced which will prove to be instrumental in the definition of the B-form in global coordinates.

The vector formulation of the B-form is enabled by the lexicographical sorting order on the elements of the multi-index as introduced in [18] and [23]. In general, for $\kappa = (\kappa_0, \kappa_1, \ldots, \kappa_n)$ and $|\kappa| = d$ this lexicographical sorting order is the following:

$$\begin{aligned} d, 0, 0 \cdots 0 > d-1, 1, 0 \cdots 0 > d-1, 0, 1, 0 \cdots 0 > \cdots > \\ > 0 \cdots 0, 1, d-1 > 0 \cdots 0, 0, d. \end{aligned} \tag{11}$$

The vector formulation for a B-form polynomial defined on a single simplex t is defined as follows:

$$p^d(b(\mathbf{x})) := \begin{cases} \mathbf{B}^d(b(\mathbf{x})) \cdot \mathbf{c}^t & , \mathbf{x} \in t \\ 0 & , \mathbf{x} \notin t \end{cases}, \tag{12}$$

with $b(\mathbf{x})$ the barycentric coordinate of the Cartesian vector $\mathbf{x}$.

The vector $\mathbf{B}^d(b(\mathbf{x}))$ in Eq. 12 is a vector which is constructed from lexicographically sorted basis polynomials:

$$\begin{aligned} \mathbf{B}^d(b(\mathbf{x})) \quad := \quad & \Big[\mathbf{B}^d_{d,0,0}(b(\mathbf{x})) \;\; \mathbf{B}^d_{d-1,1,0}(b(\mathbf{x})) \;\; \cdots \\ & \cdots \;\; \mathbf{B}^d_{0,1,d-1}(b(\mathbf{x})) \;\; \mathbf{B}^d_{0,0,d}(b(\mathbf{x}))\Big] \;\; \in \mathbb{R}^{1\times\hat{d}}. \end{aligned} \tag{13}$$

The vector $\mathbf{c}^t$ in Eq. 12 is the vector of lexicographically sorted B-coefficients on the simplex t:

$$\mathbf{c}^t := [c_{d,0,0} \quad c_{d-1,1,0} \quad \cdots \quad c_{0,1,d-1} \quad c_{0,0,d}]^\top \in \mathbb{R}^{\hat{d}\times 1}. \tag{14}$$

For example, let $p^2(b(\mathbf{x}))$ be a (barycentric) bivariate B-form of degree $d = 2$ on a single simplex t and with $\mathbf{x} \in t$. We then have $|\kappa| = 2$ and $\kappa \in \{(2,0),(1,1),(0,2)\}$. In this case the vector formulation of the B-form is the following:

$$\begin{aligned} p^2(b(\mathbf{x})) &= \mathbf{B}^2(b(\mathbf{x})) \cdot \mathbf{c} \\ &= \begin{bmatrix} B^2_{2,0}(b(\mathbf{x})) & B^2_{1,1}(b(\mathbf{x})) & B^2_{0,2}(b(\mathbf{x})) \end{bmatrix} \begin{bmatrix} c_{2,0} \\ c_{1,1} \\ c_{0,2} \end{bmatrix}. \end{aligned}$$

The complete simplex spline function of degree d and continuity order r for all J simplices in a triangulation $\mathcal{T}$ is then defined as follows:

$$s^r_d(b(\mathbf{x})) := \mathbf{B}^d \cdot \mathbf{c} \quad \in S^r_d(\mathcal{T}), \tag{15}$$

with $\mathbf{B}^d$ the full-triangulation vector of basis polynomials:

$$\mathbf{B}^d := \begin{bmatrix} \mathbf{B}^d_{t_1}(b(\mathbf{x})) & \mathbf{B}^d_{t_2}(b(\mathbf{x})) & \cdots & \mathbf{B}^d_{t_J}(b(\mathbf{x})) \end{bmatrix} \in \mathbb{R}^{1\times J\cdot\hat{d}}. \tag{16}$$

The full-triangulation vector of B-coefficients $\mathbf{c}$ in Eq. 15 is constructed as follows:

$$\mathbf{c} := \begin{bmatrix} \mathbf{c}^{t_1} \\ \mathbf{c}^{t_2} \\ \vdots \\ \mathbf{c}^{t_J} \end{bmatrix} \in \mathbb{R}^{J\cdot\hat{d}\times 1} \tag{17}$$

with $\mathbf{c}^{t_j}$ the per-simplex vector of lexicographically sorted B-coefficients from Eq. 14.

2.6 The B-Coefficient Net

The B-coefficients of the multivariate simplex B-splines are structured in what is called the B-coefficient net, or *B-net*. The B-net has a spatial representation which not only provides insight into the structure of B-form polynomials, but which also aids the visualization of the structure of continuity between simplices. This spatial representation of the B-net is well known in the literature, see e.g. [16, 21, 23].

The spatial representation of the B-net results from a direct relationship between the index of a B-coefficient and its spatial location, or barycentric coordinate within a simplex:

$$b_{c_\kappa} := \frac{\kappa_0\mathbf{v}_0 + \kappa_1\mathbf{v}_1 + \cdots + \kappa_n\mathbf{v}_n}{d}, \quad |\kappa| = d, \tag{18}$$

with b_{c_κ} the barycentric coordinate of B-coefficients and $\mathbf{v}_i, i = 0,1,\ldots,n$ the simplex vertices.

In Fig. 2 the graphical representation of the B-net corresponding with a fourth degree basis function (i.e. $d = 4$) defined on a triangulation consisting of the three simplices t_i, t_j and t_k is shown.

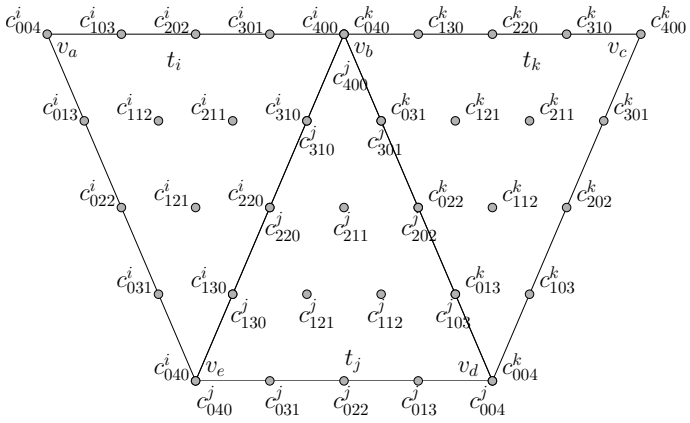

Fig. 2 B-net for a 4^{th} degree bivariate simplex spline function on a triangulation consisting of 3 simplices.

2.7 Continuity between Simplices

A spline function is a piecewise polynomial function with C^r continuity between its pieces. Continuity between the polynomial pieces is enforced by continuity conditions which are defined for every facet shared by two neighboring simplices. The formulation of the continuity conditions in this subsection are well known in the literature see e.g. [10, 3, 23, 11], but are repeated here for completeness. Let two neighboring n-simplices t_i and t_j, differing by only the vertex $\mathbf{u}$ be defined as follows:

$$t_i := \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}, \mathbf{u} \rangle , \quad t_j := \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}, \mathbf{v}_n \rangle , \tag{19}$$

then t_i and t_j meet along the facet $\tilde{t}$ given by:

$$\tilde{t} := t_i \cap t_j = \langle \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1} \rangle . \tag{20}$$

In [4, 23] the following formulation for the continuity conditions between t_i and t_j is used:

$$c^i_{(\kappa_0,\ldots,\kappa_{n-1},m)} = \sum_{|\gamma|=m} c^j_{(\kappa_0,\ldots,\kappa_{n-1},0)+\gamma} B^m_\gamma(\mathbf{u}), \quad 0 \le m \le r, \tag{21}$$

with $\gamma = (\gamma_0, \gamma_1, \ldots, \gamma_n)$ a multi-index independent of κ, and with $\mathbf{u}$ the vertex in t_i that is not in the edge $\tilde{t}$. The operator '+' in Eq. 21 adds the components of the two multi-indices κ and γ in an element by element fashion, e.g. $\kappa_0 + \gamma_0$.

It was shown in [12] and [13] that the formulation from Eq. 21 is not accurate for B-nets of general orientation. Instead, Eq. 21 needs to be generalized such that the location of the constants in the multi-indices (i.e. the 0 and m) are equal to the locations of the single non-zero values in the multi-indices of B-coefficients located at the out-of-edge vertices $\mathbf{u}$ and $\mathbf{v}_n$, respectively. Such a generalization is presented in [11].

The total number of continuity conditions for C^r continuity on a single edge facet is:

$$R := \sum_{m=0}^{r} \frac{(d-m+n-1)!}{(n-1)!(d-m)!}. \tag{22}$$

By equating all continuity conditions to zero, the following matrix form is obtained:

$$\mathbf{Hc} = 0, \tag{23}$$

with the matrix $\mathbf{H}$ the so-called smoothness matrix.

3 A New Formulation of the B-form in Global Coordinates

The barycentric coordinate system is instrumental in the definition of the B-form basis polynomials of the multivariate simplex B-splines. Because the barycentric coordinate system is a local coordinate system, defined on a per-simplex basis, the polynomials defined in terms of these coordinates have a local interpretation only. In this section, a new formulation of the B-form of the multivariate simplex spline in terms of a global coordinate system will be presented. This formulation will prove to be instrumental in the definition of the multivariate Intersplines.

3.1 Barycentric Coordinates Revisited

Before starting the derivation of the B-form in global coordinates, a better insight into the barycentric coordinate system is required. It was shown in Sec. 2 that the global coordinate $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ can be written as follows:

$$\mathbf{x} = \sum_{i=0}^{n} b_i \mathbf{v}_i, \tag{24}$$

with $\mathbf{v}_i \in \mathbb{R}^{n\times 1}$ the i-th vertex of the n-simplex t and b_i the i-th barycentric component. This expression can be rewritten in matrix formulation as follows:

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \cdots \mathbf{v}_n \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix} \\ &= \mathbf{V} \cdot \mathbf{b}, \end{aligned} \tag{25}$$

with $\mathbf{V} \in \mathbb{R}^{n\times(n+1)}$ a singular matrix containing the $n+1$ vertices of t as columns. The singular nature of $\mathbf{V}$ shows that the system is under determined. This is where the barycentric normalization property from Eq. 4 is introduced:

$$b_0 = 1 - \sum_{i=1}^{n} b_i. \tag{26}$$

Substitution of Eq. 26 in Eq. 25 results in:

$$\mathbf{x} = \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \cdots \mathbf{v}_n \end{bmatrix} \begin{bmatrix} 1 - \sum_{i=1}^{n} b_i \\ b_1 \\ \vdots \\ b_n \end{bmatrix}. \tag{27}$$

Expanding Eq. 27, leads to the following expression:

$$\mathbf{x} = \begin{bmatrix} (\mathbf{v}_1 - \mathbf{v}_0) & (\mathbf{v}_2 - \mathbf{v}_0) & \cdots (\mathbf{v}_n - \mathbf{v}_0) \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} + \mathbf{v}_0. \tag{28}$$

So finally,

$$\begin{aligned} \mathbf{x} - \mathbf{v}_0 &= \begin{bmatrix} (\mathbf{v}_1 - \mathbf{v}_0) & (\mathbf{v}_2 - \mathbf{v}_0) & \cdots (\mathbf{v}_n - \mathbf{v}_0) \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \\ &= \tilde{\mathbf{V}} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \end{aligned} \tag{29}$$

with the matrix $\tilde{\mathbf{V}}$ defined as:

$$\tilde{\mathbf{V}} = \begin{bmatrix} (\mathbf{v}_1 - \mathbf{v}_0) & (\mathbf{v}_2 - \mathbf{v}_0) & \cdots (\mathbf{v}_n - \mathbf{v}_0) \end{bmatrix} \in \mathbb{R}^{n \times n}. \tag{30}$$

It is now easy to check that the matrix $\tilde{\mathbf{V}}$ is invertible when the n-simplex has $n+1$ unique non-degenerate vertices. Now define $\mathbf{\Lambda}$ as the inverse of $\tilde{\mathbf{V}}$ as follows:

$$\mathbf{\Lambda} = \tilde{\mathbf{V}}^{-1} \in \mathbb{R}^{n \times n}. \tag{31}$$

The barycentric components $(b_1, b_2, \ldots, b_n)$ of $\mathbf{x}$ with respect to the simplex t then are:

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \mathbf{\Lambda}(\mathbf{x} - \mathbf{v}_0) \tag{32}$$

Now let $\mathbf{z} = (z_1, z_2, \ldots, z_n) \in \mathbb{R}^n$ be the global coordinate of $\mathbf{x}$ with respect to t as follows:

$$\mathbf{z} = \mathbf{x} - \mathbf{v}_0 \in \mathbb{R}^n. \tag{33}$$

With the relative coordinate, Eq. 32 can be simplified as follows:

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = \mathbf{\Lambda}\mathbf{z} \tag{34}$$

Using the normalization property of the barycentric coordinates from Eq. 26, the b_0 component becomes:

$$b_0 = 1 - |\mathbf{\Lambda z}|\,, \tag{35}$$

with $|\bullet|$ the 1-norm of a vector.

In the following, let $\mathbf{v}$ be the vector consisting of the n vertex components for each of the $n+1$ vertices of the simplex t as follows:

$$\mathbf{v} := \begin{bmatrix} \mathbf{v}_{0_x} & \mathbf{v}_{0_y} & \cdots \mathbf{v}_{n_x} & \mathbf{v}_{n_y} \end{bmatrix}^\top \in \mathbb{R}^{n\cdot(n+1)\times 1} \tag{36}$$

Combining Eq. 34 with Eq. 35 then results in the following definition for the barycentric coordinate transformation of $\mathbf{z}$:

$$b(\mathbf{v},\mathbf{z}) := \begin{bmatrix} 1 - |\mathbf{\Lambda z}| \\ \mathbf{\Lambda z} \end{bmatrix}, \tag{37}$$

with $\mathbf{v}$ the vector introduced in Eq. 36. Note that $\mathbf{v}$ is included in Eq. 37 because the triangulation vertex locations are optimization parameters.

3.2 A B-form in Global Coordinates

In Eq. 12 the vector formulation of the B-form was introduced. The vector formulation of the B-form of degree d in terms of the global coordinate $\mathbf{z}$ from Eq. 33 and the vertex vector $\mathbf{v}$ from Eq. 36 is defined as follows:

$$p^d(\mathbf{v},\mathbf{z}) := \begin{cases} \mathbf{Z}^d(\mathbf{v},\mathbf{z})\cdot\mathbf{c}^t, & (\mathbf{z}+\mathbf{v}_0)\in t \\ 0, & (\mathbf{z}+\mathbf{v}_0)\notin t \end{cases}, \tag{38}$$

with $\mathbf{Z}^d(\mathbf{v},\mathbf{z}) \in \mathbb{R}^{1\times\hat{d}}$ the vector of basis polynomials in terms of the vertex vector $\mathbf{v}$ and the global coordinate $\mathbf{z}$. Note that in Eq. 38 we can replace $p^d(b(\mathbf{v},\mathbf{z}))$ with $p^d(\mathbf{v},\mathbf{z})$ as $b(\mathbf{v},\mathbf{z})$ is a function of only $\mathbf{v}$ and $\mathbf{z}$ as shown in Eq. 37.

At this point, the actual elements of $\mathbf{Z}^d(\mathbf{v},\mathbf{z})$ are unknown. In the following two theorems, expressions will be derived for the basis polynomials in terms of global coordinates.

Theorem 1 *Any B-form polynomial of degree $d = 1$ on the n-simplex t has the following representation in the global coordinate $\mathbf{z}$ from Eq. 33:*

$$p^1(\mathbf{v},\mathbf{z}) = \mathbf{Z}^1(\mathbf{v},\mathbf{z})\cdot\mathbf{c}^t, \tag{39}$$

with $\mathbf{Z}^1(\mathbf{v},\mathbf{z}) \in \mathbb{R}^{1\times(n+1)}$ the vector of Bernstein basis polynomials from Eq. 38 for $d = 1$. The vector of first degree Bernstein basis polynomials in the global coordinate $\mathbf{z}$ is then given by:

$$\mathbf{Z}^1(\mathbf{v},\mathbf{z}) = \begin{bmatrix} 1-\Lambda_1 z_1 - \Lambda_2 z_2 - \cdots - \Lambda_n z_n \\ \Lambda_{1,1} z_1 + \Lambda_{1,2} z_2 + \cdots + \Lambda_{1,n} z_n \\ \Lambda_{2,1} z_1 + \Lambda_{2,2} z_2 + \cdots + \Lambda_{2,n} z_n \\ \vdots \\ \Lambda_{n,1} z_1 + \Lambda_{n,2} z_2 + \cdots + \Lambda_{n,n} z_n \end{bmatrix}^\top, \tag{40}$$

with $\Lambda_{i,j}$ the individual components of the matrix $\mathbf{\Lambda}$ from Eq. 31 and with Λ_j the sum of all n elements of the j^{th} column of $\mathbf{\Lambda}$ as follows:

$$\Lambda_j := \sum_{i=1}^{n} \Lambda_{i,j} \tag{41}$$

Proof:

Expanding the vector form of the B-form from Eq. 12 in barycentric coordinates for a first degree simplex polynomial leads to:

$$\begin{aligned} p^1(\mathbf{b}) &= [b^\kappa]_{|\kappa|=1} \cdot \mathbf{c}^t \\ &= [b_0^{\kappa_0} b_1^{\kappa_1} \cdots b_n^{\kappa_n}]_{|\kappa|=1} \cdot \mathbf{c}^t \\ &= \begin{bmatrix} b_0 & b_1 & \cdots & b_n \end{bmatrix} \cdot \mathbf{c}^t \end{aligned} \tag{42}$$

Expanding the matrix multiplication from Eq. 34 results in the following expression for the individual barycentric components b_i with $i > 0$:

$$b_i = \Lambda_{i,1} z_1 + \Lambda_{i,2} z_2 + \cdots + \Lambda_{i,n} z_n, \quad i > 0, \tag{43}$$

Using Eq. 35, the b_0 component can be reformulated as follows:

$$\begin{aligned} b_0 &= 1 - |\mathbf{\Lambda z}| \\ &= 1 - [(\Lambda_{1,1} + \Lambda_{2,1} + \cdots + \Lambda_{n,1}) z_1 + \\ &\quad (\Lambda_{1,2} + \Lambda_{2,2} + \cdots + \Lambda_{n,2}) z_2 + \cdots + \\ &\quad (\Lambda_{1,n} + \Lambda_{2,n} + \cdots + \Lambda_{n,n}) z_n] \\ &= 1 - \sum_{j=1}^{n} (\Lambda_{1,j} + \Lambda_{2,j} + \cdots + \Lambda_{n,j}) z_j \end{aligned} \tag{44}$$

Using Eq. 41, Eq. 44 can be simplified as follows:

$$\begin{aligned} b_0 &= 1 - \sum_{j=1}^{n} \Lambda_j z_j \\ &= 1 - \Lambda_1 z_1 - \Lambda_2 z_2 - \cdots - \Lambda_n z_n \end{aligned} \tag{45}$$

Substitution of the expression for b_0 from Eq. 45 and the expression for $(b_1, b_2, \ldots, b_n)$ from Eq. 43 in Eq. 42 then immediately results in Eq. 40, which proves the theorem. □

□

Higher order polynomials in terms of the global coordinate $\mathbf{z}$ can be found easily by recursively multiplying the vector of basis polynomials from Eq. 40. Based on **Theorem** 1, the following theorem for the general B-form in global coordinates can be introduced.

Theorem 2 *The B-form polynomial of degree d on the n-simplex t has the following representation in the global coordinate $\mathbf{z}$:*

$$p^d(\mathbf{v}, \mathbf{z}) = \mathbf{Z}^d(\mathbf{v}, \mathbf{z}) \cdot \mathbf{c}^t, \tag{46}$$

with $\mathbf{Z}^d(\mathbf{v}, \mathbf{z})$ the vector of basis polynomials of degree d in the global coordinate $\mathbf{z}$ which is defined as follows:

$$\mathbf{Z}^d(\mathbf{v}, \mathbf{z}) := \left[\frac{d!}{\kappa!} \cdot (\mathbf{Z}^1(\mathbf{v}, \mathbf{z}))^\kappa \right]_{|\kappa|=d} \in \mathbb{R}^{1 \times \hat{d}}, \tag{47}$$

with $\mathbf{Z}^1(\mathbf{v}, \mathbf{z})$ the vector of basis polynomials of degree 1 from Eq. 40

Proof:

By treating every individual polynomial basis function in $\mathbf{Z}_i^1(\mathbf{v},\mathbf{z})$ as a single term, the multinomial theorem can be used to create a new polynomial of any degree d:

$$\sum_{|\kappa|=d} \frac{d!}{\kappa!}(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^\kappa \quad = \quad \left(\mathbf{Z}_1^1(\mathbf{v},\mathbf{z}) + \mathbf{Z}_2^1(\mathbf{v},\mathbf{z}) + \cdots + \mathbf{Z}_{n+1}^1(\mathbf{v},\mathbf{z})\right)^d \tag{48}$$

Using the method for constructing the vector of basis polynomials from Eq. 12, the right hand term in Eq. 48 can be represented in the vector formulation of Eq. 47, thereby proving the theorem. □

□

The simplex spline function in terms of global coordinates that is the equivalent of Eq. 15 for all J simplices in a triangulation then is:

$$s_d^r(\mathbf{w},\mathbf{z}) := \mathbf{Z}^d(\mathbf{w},\mathbf{z}) \cdot \mathbf{c} \quad \in \mathcal{S}_d^r(\mathcal{T}), \tag{49}$$

with $\mathbf{w}$ a vector constructed from a total of J vectors of the form Eq. 36 as follows:

$$\mathbf{w} := \left[\mathbf{v_{t_1}}^\top \quad \mathbf{v_{t_2}}^\top \quad \cdots \quad \mathbf{v_{t_J}}^\top\right]^\top, \tag{50}$$

and with $\mathbf{Z}^d(\mathbf{w},\mathbf{z})$ the full-triangulation vector of B-form regressors of the form Eq. 38 as follows:

$$\mathbf{Z}^d(\mathbf{w},\mathbf{z}) \quad := \quad \left[\mathbf{Z}^{d,t_1}(\mathbf{v}_{t_1},\mathbf{z}_{t_1}) \quad \mathbf{Z}^{d,t_2}(\mathbf{v}_{t_2},\mathbf{z}_{t_2}) \quad \cdots \quad \mathbf{Z}^{d,t_J}(\mathbf{v}_{t_J},\mathbf{z}_{t_J})\right]. \tag{51}$$

Example 1 *The global B-form for a first degree polynomial on a* 1*-dimensional simplex.*

A 1*-dimensional simplex is formed by the convex hull of two vertices:*

$$t = \langle v_0, v_1 \rangle \tag{52}$$

In this case we have $n = 1$*,* $d = 1$ *and* $\kappa \in \{(1,0),(0,1)\}$*. The transformation matrix* $\mathbf{\Lambda}$ *from Eq. 31 is actually a scalar:*

$$\mathbf{\Lambda} = \tilde{\mathbf{V}}^{-1} = \frac{1}{v_1 - v_0}, \tag{53}$$

which implies that $\mathbf{\Lambda} = \Lambda_1 = \Lambda_{1,1}$*. The global coordinate in this case is* $\mathbf{z} = z_1 = x - v_0$*, which is a scalar. Using Eq. 40 the following vector of basis polynomials is obtained:*

$$\begin{aligned} p^1(\mathbf{v},\mathbf{z}) &= \begin{bmatrix} 1 - \Lambda_{1,1} z_1 \\ \Lambda_{1,1} z_1 \end{bmatrix}^\top \cdot \mathbf{c}^t \\ &= \begin{bmatrix} 1 - \frac{1}{v_1 - v_0} z_1 \\ \frac{1}{v_1 - v_0} z_1 \end{bmatrix}^\top \cdot \mathbf{c}^t \end{aligned}$$

Example 2 *The global B-form of a second degree polynomial on a* 2*-dimensional simplex.*

In this example the simplified construction method for B-form polynomials in global coordinates will be demonstrated. In this case, the global B-form polynomial of degree $d = 2$ *on a single* 2*-simplex will be derived using* ***Theorem*** *2.*

First, we need the first degree basis function vector in global coordinates from Eq. 40 for $n = 2$:

$$\mathbf{Z}^1(\mathbf{v},\mathbf{z}) = \begin{bmatrix} 1-\Lambda_1 z_1 - \Lambda_2 z_2 \\ \Lambda_{1,1} z_1 + \Lambda_{1,2} z_2 \\ \Lambda_{2,1} z_1 + \Lambda_{2,2} z_2 \end{bmatrix}^\top$$

Using Eq. 47, together with the valid values for κ we can derive $\mathbf{Z}^2(\mathbf{v},\mathbf{z})$:

$$\begin{aligned} \mathbf{Z}^2(\mathbf{v},\mathbf{z}) &= \left[\frac{2!}{\kappa!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^\kappa\right]_{|\kappa|=2}, \quad \mathbf{Z}^2(\mathbf{v},\mathbf{z}) \in \mathbb{R}^{1\times 6} \\ &= \begin{bmatrix} \frac{2!}{2!0!0!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{2,0,0} \\ \frac{2!}{1!1!0!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{1,1,0} \\ \frac{2!}{1!0!1!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{1,0,1} \\ \frac{2!}{0!2!0!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{0,2,0} \\ \frac{2!}{0!1!1!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{0,1,1} \\ \frac{2!}{0!0!2!}\cdot(\mathbf{Z}^1(\mathbf{v},\mathbf{z}))^{0,0,2} \end{bmatrix}^\top \\ &= \begin{bmatrix} (1-\Lambda_1 z_1 - \Lambda_2 z_2)^2 \\ 2\left((1-\Lambda_1 z_1 - \Lambda_2 z_2)(\Lambda_{1,1} z_1 + \Lambda_{1,2} z_2)\right) \\ 2\left((1-\Lambda_1 z_1 - \Lambda_2 z_2)(\Lambda_{2,1} z_1 + \Lambda_{2,2} z_2)\right) \\ (\Lambda_{1,1} z_1 + \Lambda_{1,2} z_2)^2 \\ 2\left((\Lambda_{1,1} z_1 + \Lambda_{1,2} z_2)(\Lambda_{2,1} z_1 + \Lambda_{2,2} z_2)\right) \\ (\Lambda_{2,1} z_1 + \Lambda_{2,2} z_2)^2 \end{bmatrix}^\top \end{aligned}$$

4 Preliminaries on Interval Analysis

4.1 Interval Arithmetic

Interval analysis is the theory dealing with interval numbers and the arithmetic operations on them [27]. An interval number $[x]$ is defined by an ordered pair of real numbers $[x] = [a, b]$ with $a \le b$. Operations applied to the ordinary number system can be extended to cover interval numbers, for example the basic computational operations of addition, subtraction, multiplication and division:

$$[a,b] + [c,d] = [a+c, b+d]\,, \tag{54}$$

$$[a,b] - [c,d] = [a-d, b-c]\,, \tag{55}$$

$$[a,b]\cdot[c,d] = [\min(ac,ad,bc,bd), \max(ac,ad,bc,bd)]\,, \tag{56}$$

$$\frac{[a,b]}{[c,d]} = [a,b]\cdot[1/d, 1/c] \qquad if \quad 0 \notin [c,d]\,. \tag{57}$$

The lower bound of an interval is called the infimum and is denoted as $\inf([x])$ or as $\underline{x}$. The upper bound of an interval is called the supremum and is denoted as $\sup([x])$ or as $\bar{x}$.

The core of interval analysis is to use interval arithmetic to form an inclusion function $[f]([\mathbf{x}])$ of any function $f(\mathbf{x})$. This property of interval arithmetic follows from the inclusion function theorem given by R.E. Moore [26, 27]:

Theorem 3
If $[f]([x_1],[x_2],...,[x_n])$ *is a rational expression in the interval variables* $[x_1],[x_2],...,[x_n]$, *i.e., a finite combination of* $[x_1],[x_2],...,[x_n]$ *and a finite set of constant intervals with interval arithmetic operations, then:*

$$[x_1]' \subset [x_1], [x_2]' \subset [x_2], ..., [x_n]' \subset [x_n], \tag{58}$$

implies:

$$[f]\left([x_1]', [x_2]', ..., [x_n]'\right) \subset [f]([x_1],[x_2],...,[x_n]), \tag{59}$$

for every set of interval numbers $[x_1],[x_2],...,[x_n]$ *for which the interval arithmetic operations in* $[f]$ *are defined.*

If we take $[x_1]', [x_2]', ..., [x_n]'$ to be the crisp numbers $x_1, x_2, ..., x_n$ and apply the theorem, then we obtain:

$$f(x_1, x_2, ..., x_n) \in [f]([x_1],[x_2],...,[x_n]) \tag{60}$$

for $x_1 \in [x_1], x_2 \in [x_2], ..., x_n \in [x_n]$. It states that if the function arguments lie within the corresponding intervals we can use interval arithmetic to produce an interval for the output of the function which is guaranteed to contain the crisp function output $f(\mathbf{x})$.

4.2 Global Nonlinear Optimization using Interval Analysis

Using the properties of intervals as described in the previous section, this section will show how intervals can be used to solve nonlinear optimization problems. Interval analysis has been successfully applied to a number of nonlinear optimization problems, ranging from spacecraft trajectory optimization to human perception modeling[36, 35].

Consider a nonlinear cost function J as a function of variables $x_1, .., x_n$ and optimization parameters $p_1, .., p_m$:

$$J = J(x_1, .., x_n; p_1, .., p_m). \tag{61}$$

The parameters and the resulting cost function can be replaced with intervals:

$$[J] = [J]([x_1], .., [x_n]; [p_1], .., [p_m]), \tag{62}$$

where for each parameter the interval is set to cover the search space in which it needs to be optimized. The complete search space is defined by the set of all interval parameters, which can be combined into a single interval vector or box:

$$[P] = \begin{pmatrix} [p_1] \\ \vdots \\ [p_m] \end{pmatrix}. \tag{63}$$

A global minimum can only be found if it resides within the chosen search space, defined by the interval boundaries. When the true global minimum is not inside the search space, a suboptimal local minimum will be found.

The optimization problem can now be written as:

$$\min [J](x_1, .., x_n; [P]) \qquad \forall p_i \in [p_i] \quad i = 1..m. \tag{64}$$

The goal of the branch and bound algorithm is to efficiently remove sub boxes from the initial parameter space, for which it is guaranteed that they do not contain the parameter combination for which the cost function has a global minimum. When the cost function is evaluated for a particular parameter box, the resulting interval can be wider than the range of values obtained when computing the cost function for each combination of parameters in the parameter box.

Let $[J_1]$ and $[J_2]$ be the cost function evaluation for parameter sub boxes $[P_1]$ and $[P_2]$ respectively. Assume that $[P_1]$ and $[P_2]$ are formed by a bisection of the complete search space $[P]$, i.e. a partition into two subsets. Due to dependency it is not possible to say that the true minimum value of J is in the interval with the lowest lower bound. So if $\inf([J_1]) < \inf([J_2])$ the true minimum value of J can still be in $[J_2]$.

Only if $\sup([J_1]) < \inf([J_2])$ it is guaranteed that the global minimum of the cost function is in the interval $[J_1]$ and thus formed by a combination of parameters inside $[P_1]$. This method of eliminating sub boxes that cannot contain the global minimum is a fundamental element of the interval branch and bound algorithm.

The first step of the interval branch and bound algorithm is to define the parameter search space as a multi-dimensional interval box $[P]$. Next the cost function is evaluated for this complete search space, resulting in $[J_P]$. This interval is likely to be very wide, but it is guaranteed that the global minimum is contained in it. As was explained above, it is not possible to say that the global minimum is equal to $\inf([J_P])$. The best estimate J^*_{min} for the global minimum of the cost function at this point in the algorithm is: $J^*_{min} = \sup([J_P])$.

The next step is to split the initial search space into sub boxes. There are many possible ways to split an interval box and there is no consensus in the literature which method is best. Although the type of box splitting influences the efficiency of the algorithm, the same global minimum will be found each time. Some possibilities are:

- Bisect in all dimensions, resulting in 2^m new sub boxes, where m is the number of dimensions in the original interval box.
- Bisect in the dimension with the widest interval, while not bisecting in the remaining dimensions, resulting in 2 new sub boxes.
- Split a random dimension into a random number of subintervals.

The new boxes obtained by splitting are put into a list ordered by the lower bound on the cost function evaluation of these boxes. Every time a new box is added to the list, the current best estimate of the cost function J^*_{min} can be lowered if the upper bound of the cost function corresponding to the added sub-box is lower than the current best estimate. When J^*_{min} is lowered, a check is done to see if any of the boxes in the list have a lower bound on the cost function that is higher than the current best estimate. If so, then this box *and all following boxes* can be removed from the ordered list, thereby reducing the search space.

Splitting, evaluating and discarding of sub boxes continues until the list of sub boxes is empty. This requires some criteria to be set on the minimal width of the parameter sub boxes:

$$\max_i \left(w\left([P_i]\right)\right) < \varepsilon_P, \qquad w\left([x]\right) = \sup\left([x]\right) - \inf\left([x]\right). \tag{65}$$

When sub boxes reach this criteria, they will not be split again and they will be put in a list of solutions. ε_P is chosen as a trade-off between the required resolution of the solution and the amount of memory that is available to store interval-boxes. With a small ε_P it is possible that multiple solutions are found which are all joined. In this

case the hull of the remaining solutions is taken. A different type of stopping criteria is the width of the evaluated cost function for a sub-box:

$$w\left([J_i]\right) < \varepsilon_J. \tag{66}$$

Usually a combination of both stopping criteria is used, depending on the shape of the cost function. A flowchart of the basic interval branch and bound algorithm is given in Fig. 3.

5 Combined B-Coefficient and Triangulation Optimization

Triangulation optimization is an essentially non-convex optimization problem. The non-convexity of the optimization problem is caused by the fact that the combined B-coefficient and triangulation optimization problem is nonlinear in the parameters.

In this section, a theorem for the non-convexity of the cost function for the combined B-coefficient and triangulation optimization problem will be introduced. This theory justifies the use of interval analysis for solving the combined B-coefficient and triangulation optimization problem. Additionally, the non-convex nature of the combined triangulation optimization and B-coefficient estimation problem is demonstrated with two numerical experiments.

5.1 A General Proof of Non-Convexity

It is well known in the literature that least squares B-coefficient optimization is a convex optimization problem [4]. However, when the parameters of the triangulation should also be optimized, the problem becomes non-convex. In the following a theorem will be presented for the non-convexity of the combined B-coefficient and triangulation optimization problem. For this, we first define the combined B-coefficient and triangulation vertex vector as follows:

$$\mathbf{q} = \begin{bmatrix} \mathbf{c}^\top & \mathbf{w}^\top \end{bmatrix}^\top, \tag{67}$$

with $\mathbf{c}$ the vector of B-coefficients for all simplices from Eq. 17, and with $\mathbf{w}$ the vertex vector from Eq. 50.

The Bernstein basis polynomials of the intersplines do not only depend on the data point locations, but also on the coordinates of the triangulation vertices. Therefore, the problem cannot be stated in the form of a linear regression problem such as that presented in [12]. Using the global formulation of the B-form from Eq. 39 the nonlinear regression problem is the following:

$$\epsilon_i(\mathbf{q}) = \mathbf{y}_i - \mathbf{Z}_i^d(\mathbf{w}, \mathbf{z})\mathbf{c}, \tag{68}$$

with $\mathbf{Z}_i(\mathbf{w}, \mathbf{z})$ a single row of B-form regressors of the form Eq. 51.

In this paper, a Sum of Absolute Errors cost function (SAE) will be used in the optimization:

$$J_1(\mathbf{q}) = \sum_{i=1}^{N} \left|\epsilon_i(\mathbf{q})\right|. \tag{69}$$

All prerequisites are now in place to introduce the theory for non-convexity and non-concavity of the combined B-coefficient and triangulation optimization problem.

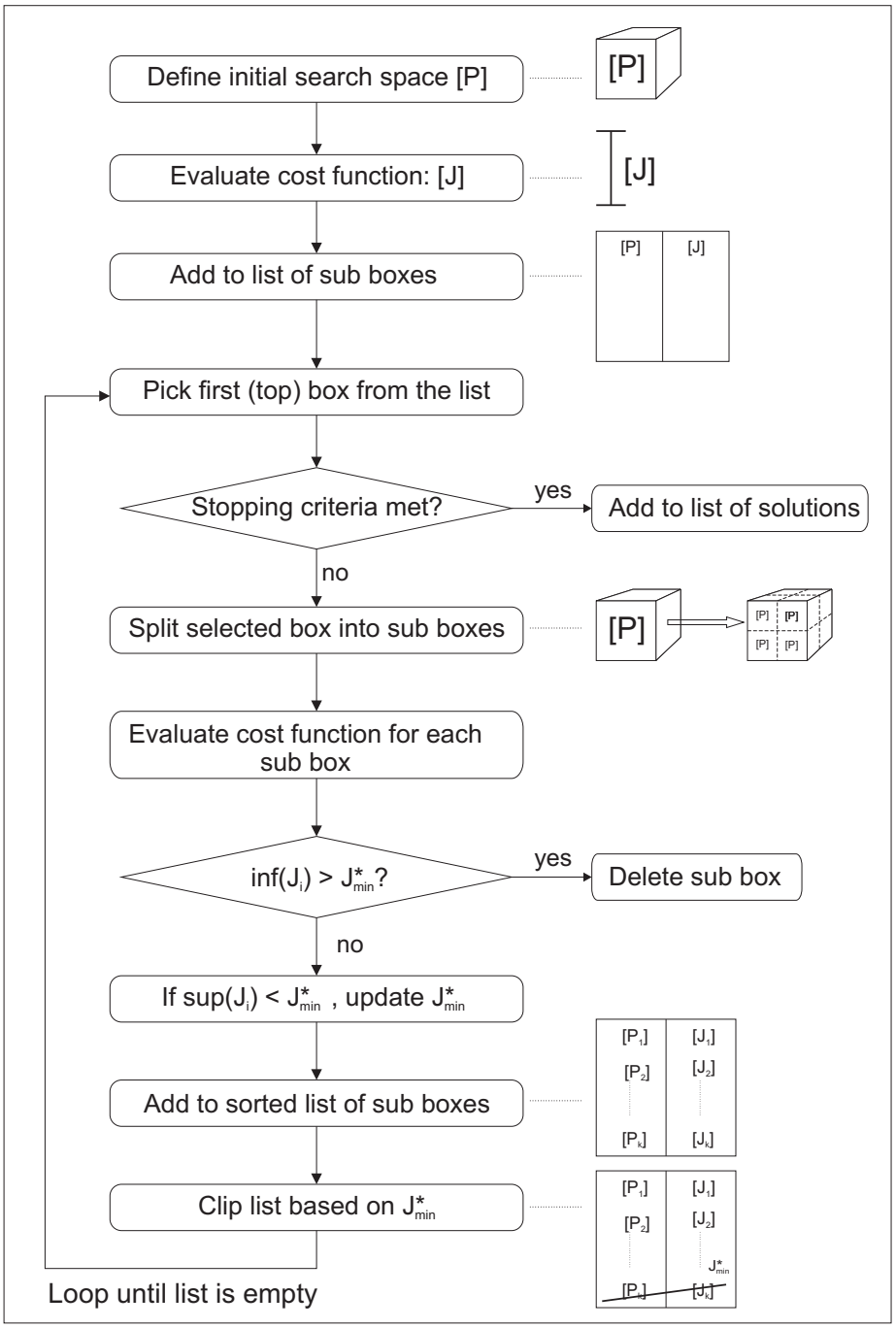

Fig. 3 Schematic flowchart of the interval branch and bound algorithm.

Theorem 4 *The cost function $J_1(\mathbf{q})$ from Eq. 69 is non-convex on the set $\mathcal{A} \cup \mathcal{B} \subset \mathbb{R}^n$, with $\mathcal{A} \cap \mathcal{B} = \emptyset$, if the following holds:*

$$\sum_{i=1}^{N} \left(sgn(\mathbf{q}) \frac{\partial^2 \mathbf{Z}_i(\mathbf{w}, \mathbf{z})}{\partial \mathbf{q}_j^2} \mathbf{c} \right) \quad \begin{cases} < 0 & \text{if } \mathbf{q} \in \mathcal{A}, \\ > 0 & \text{if } \mathbf{q} \in \mathcal{B}, \end{cases} \tag{70}$$

with $\mathbf{Z}_i(\mathbf{w}, \mathbf{z})$ from Eq. 51, with $\mathbf{c}$ as in Eq. 17, with $\mathbf{q}$ as in Eq. 67, and with $\mathcal{A}$ and $\mathcal{B}$ two non-overlapping sets in $\mathbb{R}^n$.

Proof:

The proof of the theorem is based on the second derivative test for the cost function $J_1(\mathbf{q})$ from Eq. 69. The first derivative of $J_1(\mathbf{q})$ with respect to $\mathbf{q}$ is:

$$\frac{\partial J_1(\mathbf{q})}{\partial \mathbf{q}_j} = \sum_{i=1}^{N} \text{sgn}(\mathbf{q}) \frac{\partial \epsilon_i(\mathbf{q})}{\partial \mathbf{q}_j}. \tag{71}$$

with sgn the sign function.

Using Eq. 68 the first derivative of $\epsilon_i(\mathbf{q})$ with respect to $\mathbf{q}$ is:

$$\frac{\partial \epsilon_i(\mathbf{q})}{\partial \mathbf{q}_j} = \begin{cases} -\mathbf{Z}_i(\mathbf{w}, \mathbf{z}) & \text{if } \mathbf{q}_j \in \mathbf{c} \\ -\frac{\partial \mathbf{Z}_i(\mathbf{w}, \mathbf{z})}{\partial \mathbf{q}_j} \mathbf{c} & \text{if } \mathbf{q}_j \in \mathbf{w} \end{cases}. \tag{72}$$

Using the results from Eq. 71 and Eq. 72 we find the following for the second derivative of $J_1(\mathbf{q})$ with respect to $\mathbf{q}$:

$$\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2} = \begin{cases} \text{undefined} & \text{if } \mathbf{q} = 0 \\ \sum_{i=1}^{N} \left(\text{sgn}(\mathbf{q}) \frac{\partial^2 \epsilon_i(\mathbf{q})}{\partial \mathbf{q}_j^2} \right) & \text{if } \mathbf{q} \neq 0 \end{cases}, \tag{73}$$

with the second derivative of $\epsilon_i(\mathbf{q})$ with respect to $\mathbf{q}$ given by:

$$\frac{\partial^2 \epsilon_i(\mathbf{q})}{\partial \mathbf{q}_j^2} = \begin{cases} 0 & \text{if } \mathbf{q}_j \in \mathbf{c} \\ -\frac{\partial^2 \mathbf{Z}_i(\mathbf{w}, \mathbf{z})}{\partial \mathbf{q}_j^2} \mathbf{c} & \text{if } \mathbf{q}_j \in \mathbf{w} \end{cases}. \tag{74}$$

From Eq. 73 it is clear that an analysis of the sign of the second derivative of $J_1(\mathbf{q})$ can be restricted to domains where $J_1(\mathbf{q})$ is smooth, that is, the domain $\mathbf{q} \neq 0$.

For the case $\mathbf{q} \neq 0$ we substitute Eq. 74 in Eq. 73 resulting in:

$$\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2} = \begin{cases} 0 & \text{if } \mathbf{q}_j \in \mathbf{c} \\ -\sum_{i=1}^{N} \left(\text{sgn}(\mathbf{q}) \frac{\partial^2 \mathbf{Z}_i(\mathbf{w}, \mathbf{z})}{\partial \mathbf{q}_j^2} \mathbf{c} \right) & \text{if } \mathbf{q}_j \in \mathbf{w} \end{cases} \tag{75}$$

From Eq. 75 it follows that the sign of $\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2}$ is determined by the signs of the elements of $\mathbf{q}$. Therefore, there exists a parameter vector $\mathbf{q_1} \in \mathcal{A}$ of the form Eq. 67 for which $\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2} > 0$. Equivalently, there exists an alternative parameter vector $\mathbf{q_2} \in \mathcal{B}$ of the form Eq. 67 for which $\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2} < 0$. It then follows that $\frac{\partial^2 J_1(\mathbf{q})}{\partial \mathbf{q}_j^2}$ switches sign on $\mathcal{A} \cup \mathcal{B}$, which implies that J_1 is non-convex on $\mathcal{A} \cup \mathcal{B}$, proving the theorem.□

□

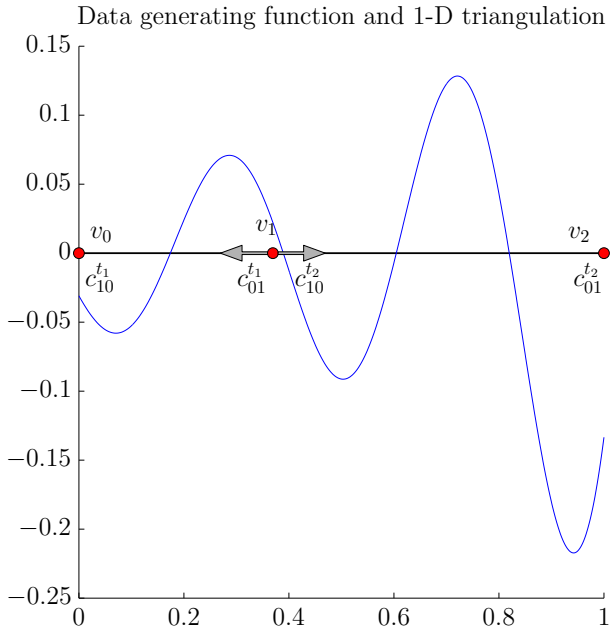

Fig. 4 The data generating function $f(x_0)$ together with the triangulation consisting of two 1-simplices in which v_1 has a variable location, together with the B-net for $d = 1$.

5.2 Numerical Analysis of the 1-D Cost Function

In this section the non-convex nature of the cost function of the combined B-coefficient and triangulation optimization problem for the 1-D case is demonstrated with a numerical experiment. First, let $\mathcal{X}_N$ be a univariate dataset consisting of N points which have a uniform random distribution in Cartesian $\mathbb{R}^1$ as follows:

$$\mathcal{X}_N = \bigcup_{i=1}^{N} \{x_0(i)\} \quad \in \{U(0,1)\} \tag{76}$$

The data generating function is the 1-D Mexican hat function (see Fig. 4):

$$f(x_0) = \frac{\sin\left(k_1\pi\sqrt{k_2(x_0-.5)^2+\nu}\right)}{k_1\pi\sqrt{k_2(x_0-.5)^2+\nu}} \tag{77}$$

with $k_1 = 15$ and $k_2 = 3$ scaling constants, and with $\nu > 0$ some small number (i.e. $\nu = 10^{-3}$).

For this experiment, a simple triangulation consisting of two simplices (line segments) is used, in which only the location of the center vertex is a variable, see Fig. 4.

The cost function used in the analysis is the SAE cost function from Eq. 69.

In Fig. 5 the cost function for varying locations of v_1 for the S_1^0 spline space is shown, clearly demonstrating its non-convex nature.

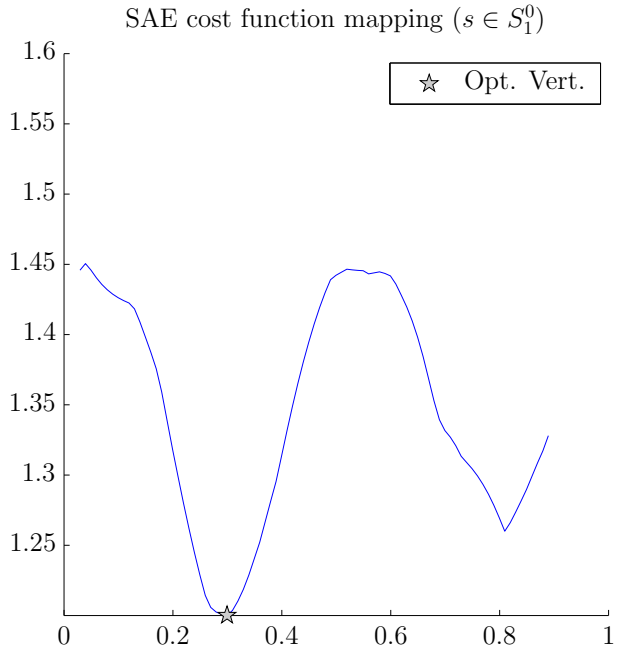

Fig. 5 Mapping of the SAE cost function for different values of the central vertex $\mathbf{w}_c$ for the spline function of degree 1 with 0^{th} order continuity.

5.3 Numerical Analysis of the 2-D Cost Function

In this section, the non-convex nature of the cost function for the 2-D case will be demonstrated with a numerical experiment.

First, let $\mathcal{X}_N$ be a bivariate dataset consisting of N points which have a uniform random distribution in Cartesian $\mathbb{R}^2$ as follows:

$$\mathcal{X}_N = \bigcup_{i=1}^{N} \{x_0(i), x_1(i)\} \quad \in \{U(0,1), U(0,1)\} \tag{78}$$

The data generating function is the 2-D Mexican hat function:

$$f(x_0, x_1) = \frac{\sin\left(k_1\pi\sqrt{k_2(x_0-.5)^2 + k_2(x_1-.5)^2} + \nu\right)}{k_1\pi\sqrt{k_2(x_0-.5)^2 + k_2(x_1-.5)^2} + \nu} \tag{79}$$

with $k_1 = 15$ and $k_2 = 1$ scaling constants, and with $\nu > 0$ some small number (i.e. $\nu = 10^{-3}$). In Fig. 7 the Mexican hat function is shown.

In this numerical experiment, a triangulation consisting of four 2-simplices is used, see Fig. 6. The central vertex in this triangulation (v_4 in the figure) has a variable location. The spline spaces S_1^0, S_2^1, and S_5^2 are used in the demonstration. The SAE cost function from from Eq. 69 is used in the optimization.

In Fig. 8, Fig. 9, and Fig. 10 the cost functions of respectively the spline spaces S_1^0, S_2^1, and S_5^2 are plotted as a function of the location of the center vertex v_4. In all cases the cost function is non-convex.

For low volume data sets the specific distribution of the data points in the spline domain have a significant influence on the shape of the cost function. The result is that the optimal location of the center vertex ($\mathbf{v}_4$) depends not only on the spline space and data generating function, but also on the particular configuration of data points. In Fig. 11 this fact is demonstrated for the S_1^0 spline space. The figure shows the optimal vertex locations for 1000 different realizations of the dataset from Eq. 78 consisting of 50 data points. Clearly, there is a significant spread in the optimal location of $\mathbf{v}_4$. In some cases, the optimal location of $\mathbf{v}_4$ for one data realization coincides with the least optimal location of $\mathbf{v}_4$ for a different realization.

6 The Multivariate Interspline

In this section the multivariate Interspline is introduced. Additionally, the setup of the numerical experiment conducted in the next section is discussed in detail.

6.1 The Interval B-Form in Global Coordinates

The multivariate Interspline is enabled by introducing interval variables in the global formulation of the B-form from Eq. 38. This results in the interval B-form as follows:

$$[p^d(\mathbf{v}, \mathbf{z})] = [\mathbf{Z}^d(\mathbf{v}, \mathbf{z})] \cdot [\mathbf{c}^t], \tag{80}$$

with $[\mathbf{Z}^d(\mathbf{v}, \mathbf{z})]$ the interval-Bernstein regressor matrix in global coordinates, and with $[\mathbf{c}^t]$ the interval B-coefficients for a single simplex. The interval simplex spline function is derived using Eq. 49 as follows:

$$[s_d^r(\mathbf{w}, \mathbf{z})] = [\mathbf{Z}^d(\mathbf{w}, \mathbf{z})] \cdot [\mathbf{c}] \quad \in [\mathcal{S}_d^r(\mathcal{T})] \tag{81}$$

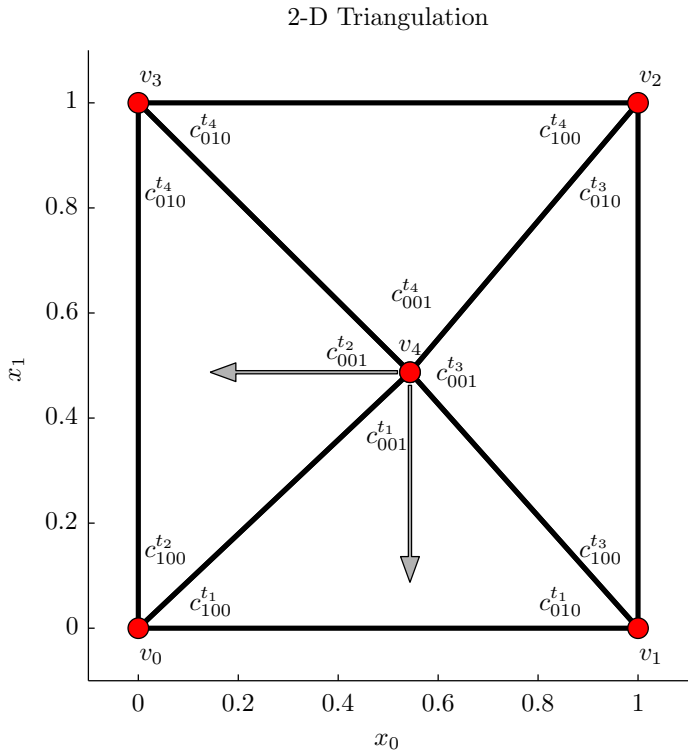

Fig. 6 The Type-II triangulation consisting of four 2-simplices in which $\mathbf{v}_4$ has a variable location, together with the B-net for $d = 1$.

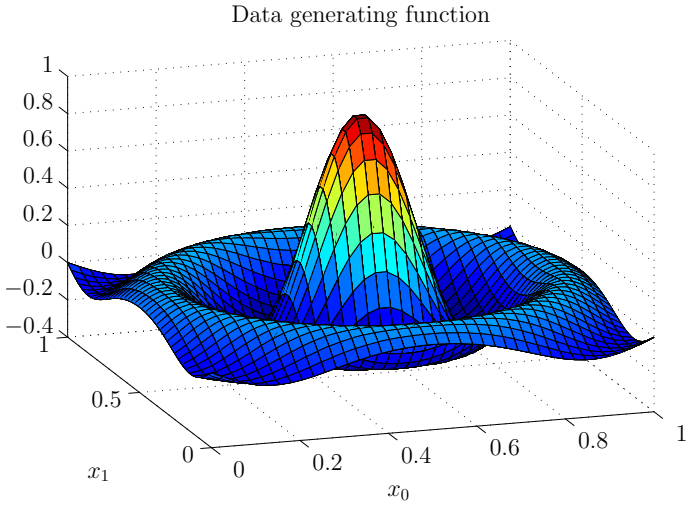

Fig. 7 The Mexican hat data generating function $f(x_0, x_1)$.

with $\mathbf{Z}^d(\mathbf{w}, \mathbf{z})$ the full-triangulation B-form regressor matrix in global coordinates from Eq. 51.

The formulation in Eq. 81 is valid for both linear and nonlinear Intersplines. In general, Eq. 81 describes a manifold in $\mathbb{R}^n$ containing all possible spline functions of a given degree.

The remainder of this paper is focused primarily on the linear Interspline defined on Type-II triangulations, such as that shown in Fig. 6. The linear Interspline is in fact a non-convex polytope which faces are formed by planes spanned by the extremum values of the B-coefficients at the corner vertices and the extremum values of the central vertex and central B-coefficient. This polytope then contains all possible linear spline polynomials.

Determining the geometry of the Interspline inclusion polytope is a non-trivial task. In Fig. 12 a schematic of the algorithm for determining these faces is shown. In the figure, the square is the volume formed by the intersection of the interval of a vertex location $[\mathbf{v}^*]$ and the interval of the B-coefficient $[c^*]$ located at $\mathbf{v}^*$.

The principle of the algorithm is that all possible B-form polynomials of a given degree on a simplex must be contained by the inclusion polytope. In the linear case, this means that the interval B-coefficient $[c_0]$ must be connected to $[c^*]$ by a linear polynomial, thereby forming the plane $P([c_0]) \in \mathbb{R}^n$. This plane is defined by the intersection point of the vertical ridges of $[c^*] \cap [\mathbf{v}^*]$ with $P([c_0])$ and all remaining (interval) B-coefficients in the simplex, see Fig. 12. The solid gray lines in the figure are valid inclusion planes, while the dashed gray lines are not. It is easy to check that the solid gray lines indeed contain all possible linear polynomials between the interval B-coefficient $[c_0]$ and $[c^*] \cap [\mathbf{v}^*]$. All inclusion planes for all simplices together then form the Interspline inclusion polytope.

In this paper the described ridge plane intersection algorithm is used for Type-I to Type-II triangulation refinement (see e.g. [11]). In this case the Type-II triangulation is formed from the Type-I triangulation by inserting a single vertex inside the grid

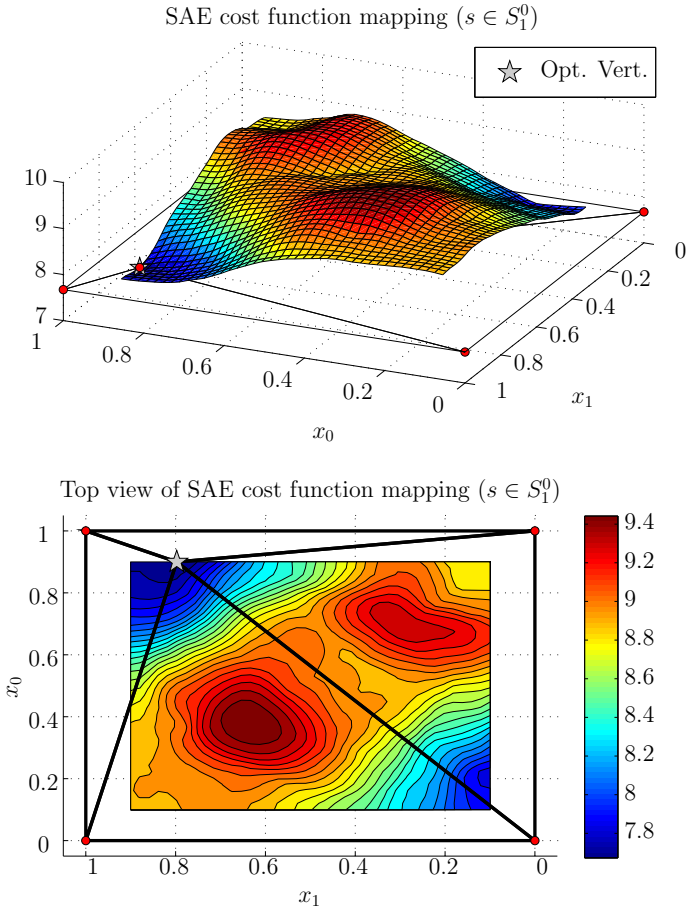

Fig. 8 Mapping of the SAE cost function for different values of the central vertex $\mathbf{v}_4$ for the spline function of degree 1 with 0^{th} order continuity.

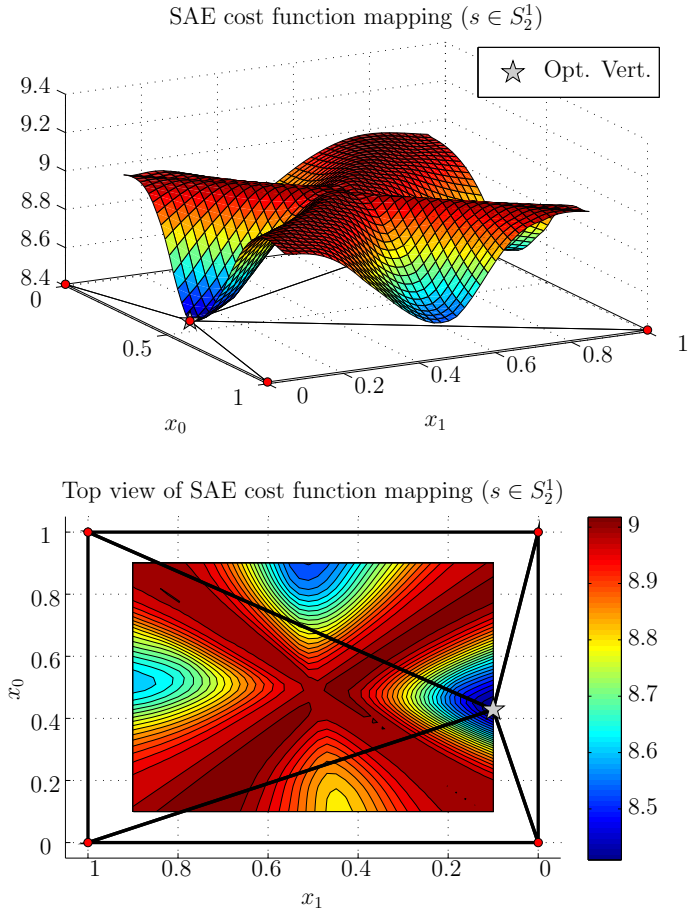

Fig. 9 Mapping of the SAE cost function for different values of the central vertex $\mathbf{v}_4$ for the spline function of degree 2 with 1^{st} order continuity.

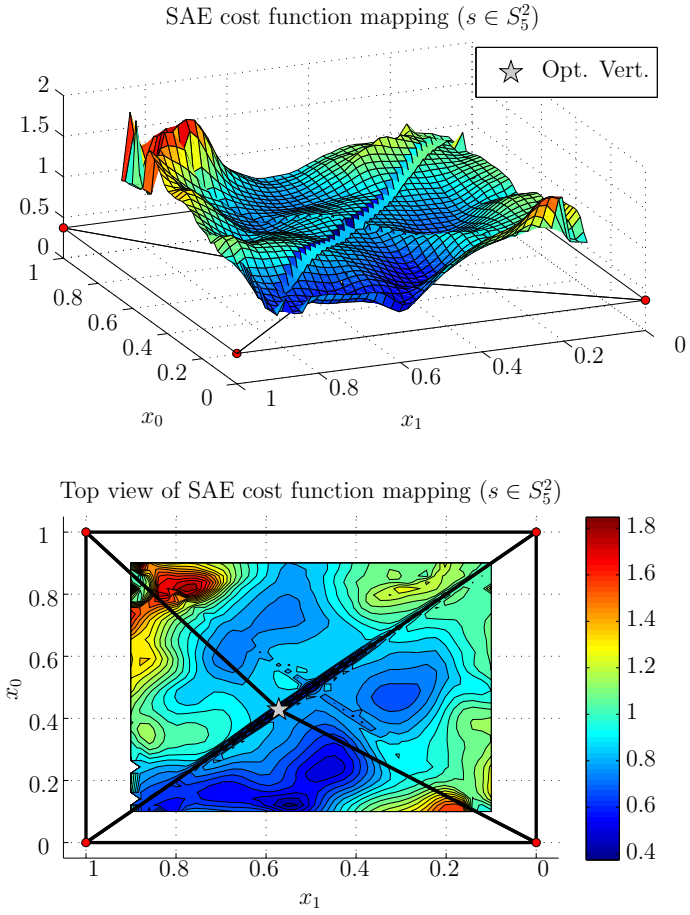

Fig. 10 Mapping of the SAE cost function for different values of the central vertex $\mathbf{v}_4$ for the spline function of degree 5 with 2^{nd} order continuity.

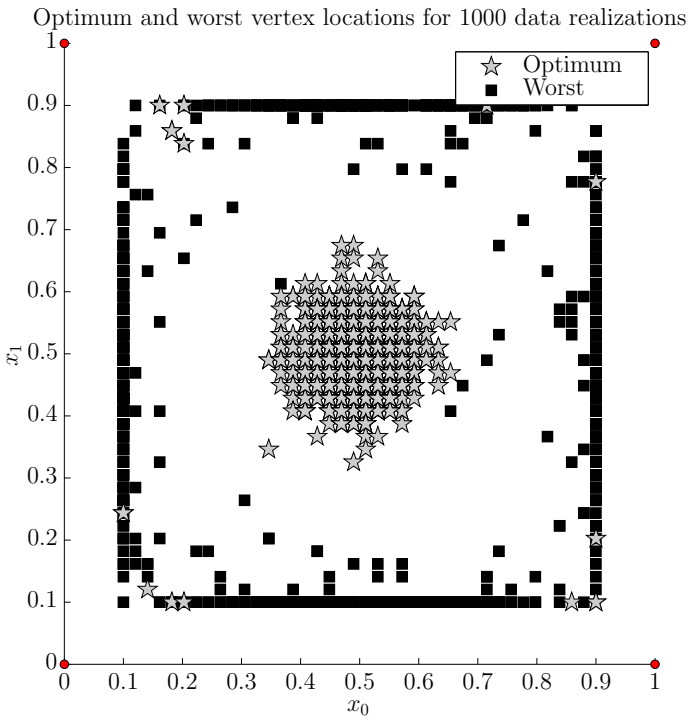

Fig. 11 Optimal and least optimal vertex locations for 1000 random realizations of a dataset consisting of 50 data points using the Mexican hat data generating function.

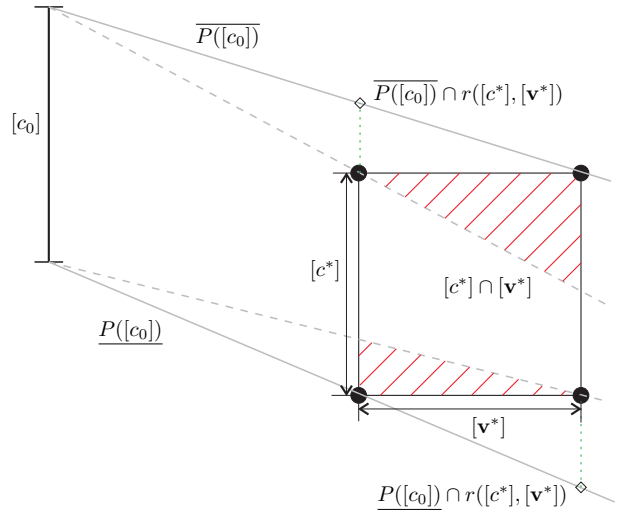

Fig. 12 The ridge-plane intersection for linear inclusion. Solid gray lines indicate valid inclusion planes, dashed gray lines indicate invalid inclusion planes, while the hatched regions denote the inclusion error when invalid inclusion planes are selected.

cells of the Type-I triangulation.

6.2 1-Dimensional Basis

The demonstration of the 1-dimensional Interspline uses linear Bernstein basis function on a triangulation consisting of the two simplices $t_1 = \langle v_0, v_1 \rangle$ and $t_2 = \langle v_1, v_2 \rangle$. Zeroth order continuity is enforced between the two resulting spline pieces. In this case, the simplex spline function in global coordinates from Eq. 49 becomes:

$$\begin{aligned} s_1^0(\mathbf{w}, \mathbf{z}) &= \left[Z^{1,t_1}(\mathbf{v_{t_1}}, \mathbf{z_{t_1}}) \;\; Z^{1,t_2}(\mathbf{v_{t_2}}, \mathbf{z_{t_2}}) \right] \cdot \mathbf{c}, \\ &= \left[1 - \frac{x - v_0}{v_1 - v_0} \quad \frac{x - v_0}{v_1 - v_0} \quad 1 - \frac{x - v_1}{v_2 - v_1} \quad \frac{x - v_1}{v_2 - v_1} \right] \cdot \mathbf{c}, \end{aligned} \tag{82}$$

with the vector of B-coefficients equal to:

$$\mathbf{c} = [c_{10}^{t_1} \;\; c_{01}^{t_1} \;\; c_{10}^{t_2} \;\; c_{01}^{t_2}]^\top. \tag{83}$$

The complete optimization problem for Eq. 82 thus consists of 7 parameters. However, zeroth order continuity requires that $c_{01}^{t_1} = c_{10}^{t_2}$ in accordance with Eq. 21. Additionally, it is assumed that both v_0 and v_2 are constants. This leaves a total of 4 optimization parameters: v_1, $c_{10}^{t_1}$, $c_{01}^{t_1}$, and $c_{01}^{t_2}$. In the following, the notation $v^* = v_1$, $c_0 = c_{10}^{t_1}$, $c^* = c_{01}^{t_1}$, and $c_1 = c_{01}^{t_2}$ are used for these parameters. The parameters v^*, c_0, c^*, and c_1 are all defined as intervals. The center vertex v^* is chosen as an interval $[v^*]$ with lower bound larger than v_0 and upper bound smaller than v_1, see Fig. 13.

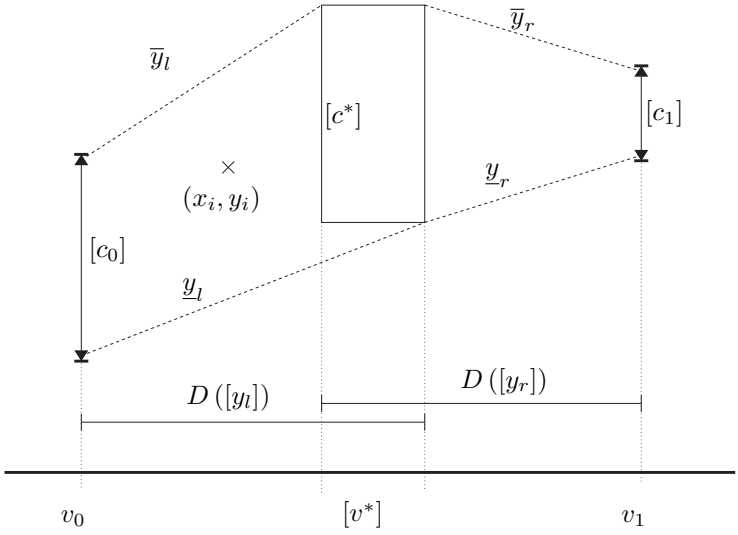

Fig. 13 One-dimensional spline basis geometry.

Goal of the optimization is the simultaneous optimization of the location of the center vertex $\mathbf{v}^*$ and the spline coefficients c_0, c^*, c_1, with c^* the B-coefficient located at the center vertex $\mathbf{v}^*$.

The first step in the optimization is to define a lower and upper bound of the enclosing polynomials, $\underline{y}$ and $\overline{y}$ respectively, see Fig. 13. The complete domain is split into two parts, $D([y_l])$ and $D([y_r])$, which are partly overlapping.

Left part, $y_l(x)$ $\quad x \in \left[0, \overline{v^*}\right]$:

$$\begin{array}{lcl}
\overline{c^*} \geq \overline{c_0} & \rightarrow & \overline{y_l(x)} = \frac{\overline{c^*} - \overline{c_0}}{v^*}(x) + \overline{c_0} \\
\overline{c^*} < \overline{c_0} & \rightarrow & \overline{y_l(x)} = \frac{\overline{c^*} - \overline{c_0}}{\overline{v^*}}(x) + \overline{c_0} \\
\underline{c^*} \geq \underline{c_0} & \rightarrow & \underline{y_l(x)} = \frac{\underline{c^*} - \underline{c_0}}{\overline{v^*}}(x) + \underline{c_0} \\
\underline{c^*} < \underline{c_0} & \rightarrow & \underline{y_l(x)} = \frac{\underline{c^*} - \underline{c_0}}{\underline{v^*}}(x) + \underline{c_0}
\end{array}$$

Right part, $y_r(x)$ $\quad x \in [\underline{v^*}, 1]$:

$$\begin{array}{lcl}
\overline{c_1} \geq \overline{c^*} & \rightarrow & \overline{y_r(x)} = \frac{\overline{c_1} - \overline{c^*}}{1 - \underline{v^*}}(x - \underline{v^*}) + \overline{c^*} \\
\overline{c_1} < \overline{c^*} & \rightarrow & \overline{y_r(x)} = \frac{\overline{c_1} - \overline{c^*}}{1 - \overline{v^*}}\left(x - \overline{v^*}\right) + \overline{c^*} \\
\underline{c_1} \geq \underline{c^*} & \rightarrow & \underline{y_r(x)} = \frac{\underline{c_1} - \underline{c^*}}{1 - \overline{v^*}}\left(x - \overline{v^*}\right) + \underline{c^*} \\
\underline{c_1} < \underline{c^*} & \rightarrow & \underline{y_r(x)} = \frac{\underline{c_1} - \underline{c^*}}{1 - \underline{v^*}}(x - \underline{v^*}) + \underline{c^*}
\end{array}$$

The next step is to combine the left and right domains in order to obtain a continuous description for the enclosing polynomials, $\underline{y}$ and $\overline{y}$.

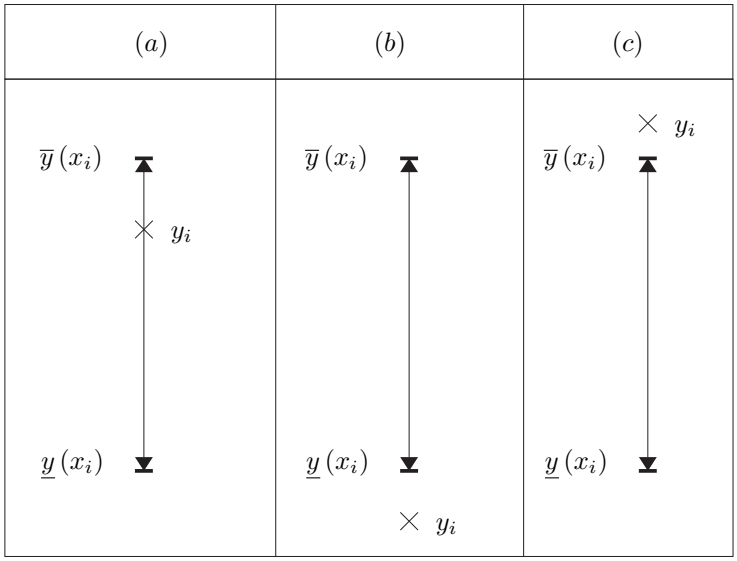

Fig. 14 Determination of the distance between the data point y_i and the enclosing interval $[y]$ for three cases.

$$\overline{y(x)} = \begin{cases} \overline{y_l(x)} & x_i \in [0, \underline{v^*}] \\ \overline{y_r(x)} & x_i \in [\overline{v^*}, 1] \\ \widehat{y}(x) & x_i \in [v^*] \end{cases}$$

$$\widehat{y}(x) = \begin{cases} \overline{c^*} & \overline{c^*} \geq \overline{c_0} \wedge \overline{c^*} \geq \overline{c_1} \\ \overline{y_r(x)} & \overline{c^*} \geq \overline{c_0} \wedge \overline{c^*} < \overline{c_1} \\ \overline{y_l(x)} & \overline{c^*} < \overline{c_0} \wedge \overline{c^*} \geq \overline{c_1} \\ \max\left(\overline{y_l(x)}, \overline{y_r(x)}\right) & \overline{c^*} < \overline{c_0} \wedge \overline{c^*} < \overline{c_1} \end{cases}$$

$$\underline{y(x)} \begin{cases} \underline{y_l(x)} & x_i \in [0, \underline{v^*}] \\ \underline{y_r(x)} & x_i \in [\overline{v^*}, 1] \\ \tilde{y}(x) & x_i \in [v^*] \end{cases}$$

$$\tilde{y}(x) = \begin{cases} \underline{c^*} & \underline{c^*} < \underline{c_0} \wedge \underline{c^*} < \underline{c_1} \\ \underline{y_r(x)} & \underline{c^*} < \underline{c_0} \wedge \underline{c^*} \geq \underline{c_1} \\ \underline{y_l(x)} & \underline{c^*} \geq \underline{c_0} \wedge \underline{c^*} \geq \underline{c_1} \\ \min\left(\underline{y_l(x)}, \underline{y_r(x)}\right) & \underline{c^*} \geq \underline{c_0} \wedge \underline{c^*} < \underline{c_1} \end{cases}$$

Once the equations for the lower and upper bound of the enclosing polynomials have been found, a distance measure from the data-point to this enclosure must be defined. In Fig. 14, three cases are presented:

- (a) The data point y_i is located inside the enclosing interval $\left[\underline{y}(x_i), \overline{y}(x_i)\right]$.
- (b) The data point y_i is located below the enclosing interval $\left[\underline{y}(x_i), \overline{y}(x_i)\right]$.
- (c) The data point y_i is located above the enclosing interval $\left[\underline{y}(x_i), \overline{y}(x_i)\right]$.

The corresponding error measures $[\epsilon_i]$ are given by:

$$\begin{array}{ll} (a) & [\epsilon_i] = \left[0, \max\left\{\text{abs}\left(\overline{y\left(x_i\right)} - y_i\right), \text{abs}\left(\underline{y\left(x_i\right)} - y_i\right)\right\}\right] \\ (b) & [\epsilon_i] = \left[\underline{y\left(x_i\right)} - y_i, \overline{y\left(x_i\right)} - y_i\right] \\ (c) & [\epsilon_i] = \left[y_i - \overline{y\left(x_i\right)}, y_i - \underline{y\left(x_i\right)}\right] \end{array} \tag{84}$$

The cost function defined as the summation of the absolute distances from the data point to the interspline as described by Eq. 84:

$$[J] = \sum_i [\epsilon_i] \tag{85}$$

6.3 2-Dimensional Basis

The demonstration of the 2-dimensional Interspline uses linear bivariate Bernstein basis function on a triangulation consisting of the four simplices $t_1 = \langle \mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_4 \rangle$, $t_2 = \langle \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_4 \rangle$, $t_3 = \langle \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4 \rangle$, and $t_4 = \langle \mathbf{v}_0, \mathbf{v}_3, \mathbf{v}_4 \rangle$, see also Fig. 6. Zeroth order continuity is enforced between the four spline pieces. In this case, the simplex spline function in global coordinates from Eq. 49 is:

$$\begin{aligned} s_1^0(\mathbf{w}, \mathbf{z}) \quad = \quad & \left[Z^{1,t_1}(\mathbf{v_{t_1}}, \mathbf{z_{t_1}}) \;\; Z^{1,t_2}(\mathbf{v_{t_2}}, \mathbf{z_{t_2}}) \right. \\ & \left. Z^{1,t_3}(\mathbf{v_{t_3}}, \mathbf{z_{t_3}}) \;\; Z^{1,t_4}(\mathbf{v_{t_4}}, \mathbf{z_{t_4}})\right] \cdot \mathbf{c} \;\in\; \mathcal{S}_1^0, \end{aligned} \tag{86}$$

with each $Z^{1,t_i}(\mathbf{v}, \mathbf{z})$ a vector of Bernstein basis polynomials as shown in Eq. 47. For example, $Z^{1,t_1}(\mathbf{v}, \mathbf{z})$, which is defined on the simplex t_1, is given by:

$$Z^{1,t_1}(\mathbf{v}, \mathbf{z}) = \left[\begin{array}{c} \frac{v_{1_x} v_{4_y} - v_{1_y} v_{4_x} + v_{1_y} x - v_{1_x} y - v_{4_y} x + v_{4_x} y}{v_{0_x} v_{1_y} - v_{0_y} v_{1_x} - v_{0_x} v_{4_y} + v_{0_y} v_{4_x} + v_{1_x} v_{4_y} - v_{1_y} v_{4_x}} \\ -\frac{v_{0_x} v_{4_y} - v_{0_y} v_{4_x} + v_{0_y} x - v_{0_x} y - v_{4_y} x + v_{4_x} y}{v_{0_x} v_{1_y} - v_{0_y} v_{1_x} - v_{0_x} v_{4_y} + v_{0_y} v_{4_x} + v_{1_x} v_{4_y} - v_{1_y} v_{4_x}} \\ \frac{v_{0_x} v_{1_y} - v_{0_y} v_{1_x} + v_{0_y} x - v_{0_x} y - v_{1_y} x + v_{1_x} y}{v_{0_x} v_{1_y} - v_{0_y} v_{1_x} - v_{0_x} v_{4_y} + v_{0_y} v_{4_x} + v_{1_x} v_{4_y} - v_{1_y} v_{4_x}} \end{array}\right]^\top \tag{87}$$

The vector of B-coefficients in Eq. 86 is equal to:

$$\begin{aligned} \mathbf{c} \quad = \quad & \left[c_{100}^{t_1} \;\; c_{010}^{t_1} \;\; c_{001}^{t_1} \;\; c_{100}^{t_2} \;\; c_{010}^{t_2} \;\; c_{001}^{t_2} \;\; \cdots \right. \\ & \left. \cdots \;\; c_{100}^{t_3} \;\; c_{010}^{t_3} \;\; c_{001}^{t_3} \;\; c_{100}^{t_4} \;\; c_{010}^{t_4} \;\; c_{001}^{t_4}\right]^\top. \end{aligned} \tag{88}$$

In the demonstration, it is assumed that the corner vertices $\mathbf{v}_0$, $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ are fixed. Additionally, C^0 continuity requires that the B-coefficients at the center vertex $\mathbf{v}_4$ (see Fig. 6) have equal values according to Eq. 21: $c_{001}^{t_1} = c_{001}^{t_2} = c_{001}^{t_3} = c_{001}^{t_4}$. As a final simplification step, all B-coefficients located at the corner vertices are assumed to be equal to zero: $c_{100}^{t_1} = c_{010}^{t_1} = c_{100}^{t_2} = c_{010}^{t_2} = c_{100}^{t_3} = c_{010}^{t_3} = c_{100}^{t_4} = c_{010}^{t_4} = 0$. This results in a global optimization problem with interval parameters $\mathbf{v}_4 \in [\mathbf{v}^*]$ and $c^* \in \left[c_{001}^{t_1}\right]$. The interval center vertex $[\mathbf{v}^*]$, is located somewhere in the rectangle formed by the corner vertices (see Fig. 15). $[\mathbf{v}^*]$ is a 2-dimensional interval:

$$[\mathbf{v}^*] = \left(\begin{array}{c} [v_x^*] \\ [v_y^*] \end{array}\right) \tag{89}$$

By connecting the corner vertices to the center vertex, the rectangle is divided into 4 triangles which each support an Interspline piece. The coefficients of the interspline

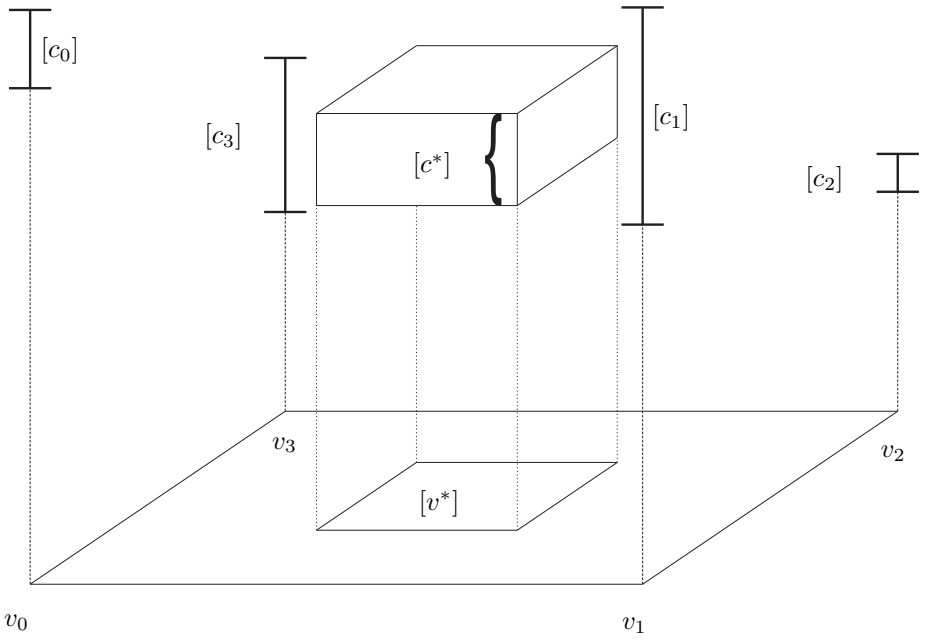

Fig. 15 2-Dimensional Interspline geometry

are intervals as well, $([c_0], [c_1], [c_2], [c_3])$ for the corner vertices, and $[c^*]$ for the center vertex. A first order interspline is now defined as the collection of all sets of planes created by connecting any two crisp points in the coefficients of two adjoining corner vertices with a crisp point in the rectangular cuboid formed by the interval center vertex and its interval coefficient. An example of such a set of planes is given in Fig. 16, where the supremum of each interval coefficient is selected as well as the supremum of the center vertex in each of the two dimensions.

Similar to the 1-dimensional case, the cost function is defined as the summation of the absolute distances from a data point to the upper and lower inclusion planes, which are determined by the procedure shown in Fig. 12.

7 Numerical Experiments with Multivariate Intersplines

In this section the results from a numerical experiment with the multivariate intersplines are presented.

7.1 2-Dimensional Interspline Demonstration

The demonstration in this section concerns finding the optimal vertex location and spline coefficient value for a bivariate dataset, obtained by randomly sampling 50 points of the Mexican hat function (Fig. 7), according to Eq. 78 [1].

[1] Upon request the specific dataset can be provided by the authors

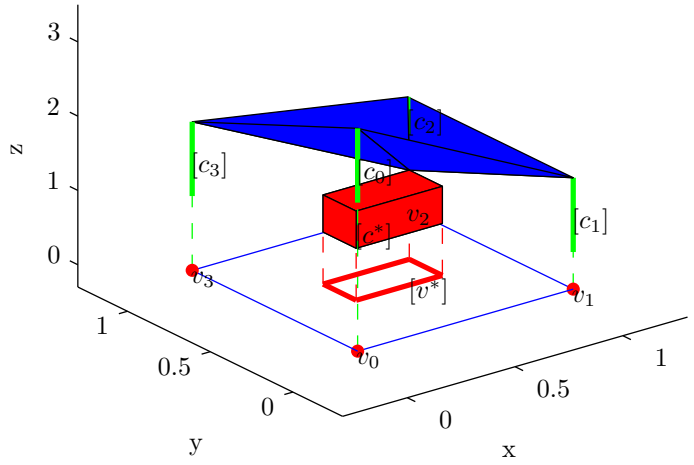

Fig. 16 2-D Interspline geometry: example of a crisp spline that is inside the Interspline.

The initial search space $[X_0]$ for the location of the global optimum is a 7-dimensional space:

$$[X_0] = \left([c_0], [c_1], [c_2], [c_3], [c^*], [v_x^*], [v_y^*]\right)^T. \tag{90}$$

In order to reduce the computation time, some of the spline coefficients (c_0, c_1, c_2, c_3) will be fixed to a constant(crisp) value, see also Sec. 6.3. Although this makes the problem less time consuming, the difficulty of simultaneously optimizing for the vertex location $(\mathbf{v}^*)$ and the spline coefficient (c^*) is still present.

The vertex location is searched for within the square:

$$[\mathbf{v}^*] = \begin{pmatrix} [0.3, 0.7] \\ [0.3, 0.7] \end{pmatrix}, \tag{91}$$

while the coefficient c^* is searched for within the interval $[-1, 1]$, resulting in the initial search space:

$$[X_0] = ([0,0], [0,0], [0,0], [0,0], [-1,1], [0.3,0.7], [0.3,0.7])^T. \tag{92}$$

For this initial search space, the inclusion planes can be set-up and the cost function can be computed according to the procedure described in section Sec. 6.1:

$$[J]([X_0]) = [0.1941, 40.6573]. \tag{93}$$

Next, an interval branch and bound algorithm (see Fig. 3) is used to remove parts of the search space that cannot contain the global minimum of the cost function. After running the branch and bound algorithm, a large set of boxes remains that possibly contain the global minimum, but these boxes are considered too small to subdivide further. By taking the hull of all these boxes, one interval box remains that

is guaranteed to contain the global minimum:

$$X_{final} = \begin{pmatrix} [0,0] \\ [0,0] \\ [0,0] \\ [0,0] \\ [0.1384, 0.1426] \\ [0.4858, 0.4982] \\ [0.4831, 0.4949] \end{pmatrix}. \tag{94}$$

By continuing the branch and bound loop, the width of the final solution can be reduced further, but there is a limit in the achievable minimum width, which depends on the machine precision of the computation device and on the number and type of operations that are required to evaluate the cost function.

The cost function for this final solution is:

$$[J]([X_{final}]) = [11.1796, 11.3073]. \tag{95}$$

This solution agrees with the interval inclusion theorem (**Theorem** 3):

$$[X_{final}] \subset [X_0] \rightarrow [J]([X_{final}]) \subset [J]([X_0]). \tag{96}$$

The results obtained with the Intersplines were compared with results obtained with ordinary simplex B-splines that were optimized using a grid search optimization algorithm for $\mathbf{v}^*$ and c^*. The grid search used a $100 \times 100 \times 100$ three-dimensional grid inside the search interval $(\mathbf{v}^*, c^*) = ([0.30.7], [0.30.7], [-11])$. The resulting optimization parameters were found to be:

$$X_{grid} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.140 \\ 0.4898 \\ 0.4898 \end{pmatrix}. \tag{97}$$

The resulting optimization parameters from the grid search clearly are located within the final interval of the Interspline parameters from Eq. 94.

The final cost function value obtained by using the grid search algorithm is:

$$J_{grid}(X_{grid}) = 11.258. \tag{98}$$

The final cost function value found using the grid search algorithm is located within the interval cost function value from Eq. 95.

In Fig. 17 the location of the central vertex $\mathbf{v}^*$ found using the grid search and ordinary simplex B-splines is shown. In the same figure, the location of $\mathbf{v}^*$ found using interval analysis and Intersplines is shown. Clearly, the crisp value for $\mathbf{v}^*$ found using the grid search is located within the interval found using the Intersplines.

The optimal location of $\mathbf{v}^*$ found using the grid search cannot be guaranteed to be the true global optimum, however, as a grid search with a higher resolution may result in an even lower cost function value. The interval provided by the Intersplines is guaranteed to contain the optimal location of $\mathbf{v}^*$, and therefore provides a capability that cannot be matched by any grid search algorithm.

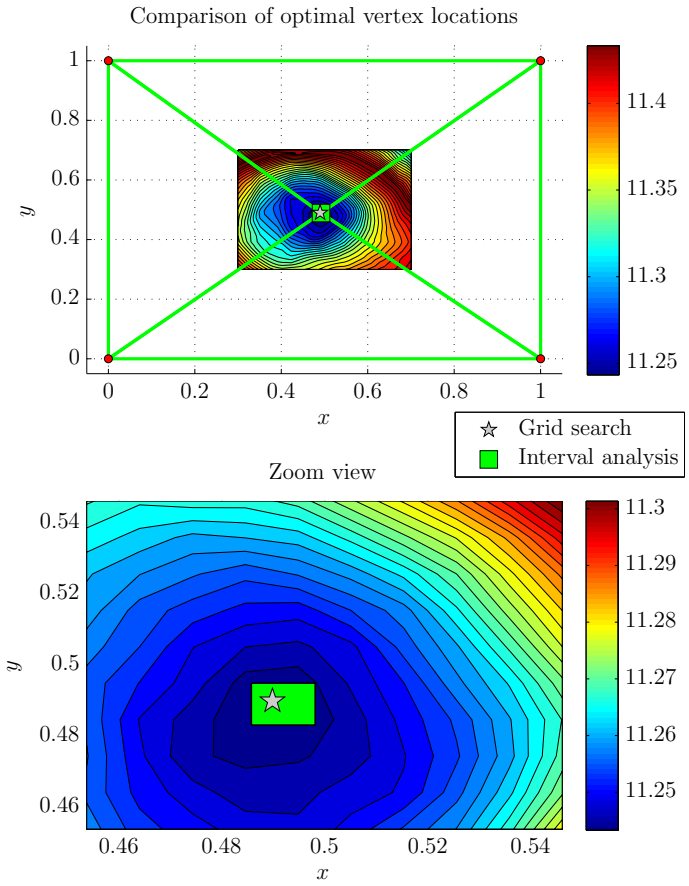

Fig. 17 The SAE cost function, together with the optimal vertex location found using a grid search, and the interval guaranteed to contain the global optimal vertex location determined using interval analysis.

8 Conclusions and Recommendations

8.1 Conclusions

This paper presents multivariate Intersplines, which are a class of globally optimal multivariate splines.

Multivariate Intersplines are the result of the synthesis of multivariate simplex B-spline theory and interval analysis. This synthesis is made possible by a new formulation in global coordinates of the Bernstein basis polynomials of the simplex B-splines. A multivariate Interspline is formed by changing the otherwise crisp B-coefficients and triangulation parameters into intervals variables. Then, using methods of interval optimization, globally optimal values for these intervals can be determined. The multivariate Intersplines solve the triangulation optimization problem, which is a long standing problem in multivariate spline theory.

The multivariate Intersplines are demonstrated with a bivariate scattered data approximation problem. In this demonstration, a bivariate linear Interspline is used to approximate a scattered bivariate dataset on four simplices. The results from the Interspline approximation are compared and validated with the results obtained by performing a grid search with ordinary simplex B-splines. The results show that the optimal spline solution found using the grid search is contained within the Interspline.

8.2 Recommendations

The application part of this paper is focused primarily on linear Intersplines. The theory presented in this paper, however, is general and can be applied to Intersplines of any degree. Nonlinear Intersplines do present a challenge in the sense that the inclusion geometry is not a polytope but a non-convex manifold. Determining the geometry of this manifold is a standing but in our view solvable problem.

The combined B-coefficient and triangulation optimization problem presented in the demonstration part of this paper actually represents a Type-I to Type-II triangulation refinement. This type of refinement is rather limited in utility as it can be applied to a Type-I triangulation only once. A more general approach would be simplex subdivision, in which a single vertex is placed inside an n-simplex, producing $n+1$ new, smaller n-simplices. This process could be then be run in a recursive fashion. However, the geometry of the inclusion planes would be more complex than that for Type-I to Type-II triangulation refinement, because the central element would not be a cuboid but a prismoid. The cuboid is natively compatible with interval analysis, as any vector of interval numbers can be represented geometrically in the form of a hypercube. A prismoid, on the other hand, contains a non rectangular subset of this hypercube, and can therefore not be represented directly in terms of interval numbers. It is recommended that more research should be performed in this more general refinement scheme.

Interval optimization methods can be combined with other optimization methods to speed up the algorithm. For example, genetic algorithms can be used to find a lower value of the estimated cost function minimum. This value can then be used to remove more boxes from the search space. Also, as soon as it can be verified that the cost function is convex over a given part of the search space, one can switch to gradient based optimization methods in order to improve computational performance.

References

[1] J. Abonyi, R. Babuška, and Ferenc Szeifert. Fuzzy modeling with multivariate membership functions: Gray-box identification and control design. *IEEE Trans. On Systems, Man, and Cybernetics*, 31:755–767, 2001.

[2] I. J. Anderson, M. G. Cox, and J. C. Mason. Tensor-product spline interpolation to data on or near a family of lines. *Numerical Algorithms*, 5:193–204, 1993.

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

[4] G. Awanou, M. J. Lai, and P. Wenston. The multivariate spline method for scattered data fitting and numerical solutions of partial differential equations. In G. Chen and M. J. Lai, editors, *Wavelets and Splines*, pages 24–75, 2005.

[5] R. Babuška and H. Verbruggen. Neuro-fuzzy methods for nonlinear system identification. *Annual Reviews in Control*, 27:73–85, 2003.

[6] P. D. Bruce and M. G. Kellett. Modelling and identification of non-linear aerodynamic functions using b-splines. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 214:27–40, 2000.

[7] C. Cao and N. Hovakimyan. Novel L1 neural network adaptive control architecture with guaranteed transient performance. *IEEE Transactions on Neural Networks*, 18(4):1160–1171, 2007.

[8] S. Chen and S.A. Billings. Neural networks for nonlinear dynamic system modelling and identification. *Int. J. Control*, 56:319–346, 1992.

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[10] C. de Boor. B-form basics. In G. Farin, editor, *Geometric modeling: algorithms and new trends*. SIAM, 1987.

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

[12] C. C. de Visser, Q. P. Chu, and J. A. Mulder. A new approach to linear regression with multivariate splines. *Automatica*, 45(12):2903–2909, 2009.

[13] C. C. de Visser, Q. P. Chu, and J. A. Mulder. Differential constraints for bounded recursive identification with multivariate splines. *Automatica*, 47:2059–2066, 2011.

[14] M. Espinoza, J.A.K. Suykens, and B. de Moor. Kernel based partial linear models and nonlinear identification. *IEEE Transactions On Automatic Control*, 50:1602–1606, 2005.

[15] H. Y. Fan, G. S. Dulikravich, and Z. X. Han. Aerodynamic data modeling using support vector machines. *Inverse Problems in Science and Engineering*, 13:261–278, 2005.

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

[17] S. Ferrari and R. F. Stengel. Smooth function approximation using neural networks. *IEEE Transactions On Neural Networks*, 16(1):24–38, 2005.

[18] X. L. Hu, D. F. Han, and M. J. Lai. Bivariate splines of various degrees for numerical solution of partial differential equations. *SIAM Journal on Scientific Computing*, 29:1338–1354, 2007.

[19] R. V. Jategaonkar. *Flight Vehicle System Identification*, volume 216 of *Progress in Astronautics and Aeronautics*. AIAA, 2006.

[20] V. Klein and J. G. Batterson. Determination of airplane model structure from flight data using splines and stepwise regression. Technical Report 2126, NASA, 1983.

[21] M. J. Lai. Geometric interpretation of smoothness conditions of triangular polynomial patches. *Computer Aided Geometric Design*, 14:191–199, 1997.

[22] M. J. Lai and L. L. Schumaker. On the approximation power of bivariate splines. *Advances in Computational Mathematics*, 9:251–279, 1998.

[23] M. J. Lai and L. L. Schumaker. *Spline Functions on Triangulations*. Cambridge University Press, 2007.

[24] M.J. Lai. Some sufficient conditions for convexity of multivariate Bernstein-Bézier polynomials and box spline surfaces. *Studia Scient. Math. Hung.*, 28:363–374, 1990.

[25] D. T. Mirikitani and N. Nikolaev. Recursive bayesian recurrent neural networks for time-series modeling. *IEEE Transactions on Neural Networks*, 21:262–274, 2010.

[26] R.E. Moore. *Interval Analysis*. Prentice-Hall, Inc., 1966.

[27] R.E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA, 1979.

[28] S. K. Oh, W. D. Kim, W. Pedrycz, and B. J. Park. Polynomial-based radial basis function neural networks (P-RBF NNs) realized with the aid of particle swarm optimization. *Fuzzy Sets and Systems*, 163(1):54–77, 2011.

[29] B. J. Park, W. Pedrycz, and S. K. Oh. Fuzzy polynomial neural networks: Hybrid architectures of fuzzy modeling. *IEEE Transactions on Fuzzy Systems*, 10:607–621, 2002.

[30] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of Algorithms*, 18:548–585, 1995.

[31] B. Schölkopf, J. Giesen, and S. Spalinger. Kernel methods for implicit surface modeling. In *Advances in Neural Information Processing Systems 17*, pages 1193–1200. MIT Press, 2005.

[32] J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry*, 22:21–74, 2001.

[33] J. R. Shewchuk. General-dimensional constrained Delaunay and constrained regular triangulations I: Combinatorial properties. *Discrete and Computational Geometry*, 39(1):580–637, 2008.

[34] P. L. Smith. Curve fitting and modeling with splines using statistical variable selection techniques. Contractor Report NASA-CR-166034, NASA, 1982.

[35] E. van Kampen. *Global Optimization using Interval Analysis*. PhD thesis, Delft University of Technology, 2010.

[36] E. van Kampen, P.M.T. Zaal, E. de Weerdt, Q.P. Chu, and J.A. Mulder. Optimization of human perception modeling using interval analysis. *Journal of Guidance, Control, and Dynamics*, 33:42–52, 2010.

[37] J. Zhou. *Interval Simplex Splines for Scientific Databases*. PhD thesis, Massachusetts Institute of Technology, 1995.