Faculty of Mathematical Sciences

University of Twente

University for Technical and Social Sciences

P.O. Box 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893400
Fax: +31-53-4893114
Email: memo@math.utwente.nl

Memorandum No. 1450

A linear approach to shape preserving spline approximation

F. Kuijt and R.M.J. van Damme

July 1998

ISSN 0169-2690

A Linear Approach to Shape Preserving Spline Approximation

Frans Kuijt* and Ruud van Damme

July 1998

Abstract

This report deals with approximation of a given scattered univariate or bivariate data set that possesses certain shape properties, such as convexity, monotonicity, and/or range restrictions. The data are approximated for instance by tensor-product B-splines preserving the shape characteristics present in the data.

Shape preservation of the spline approximant is obtained by additional linear constraints. Constraints are constructed which are local *linear sufficient* conditions in the unknowns for convexity or monotonicity. In addition, it is attractive if the objective function of the minimisation problem is also linear, as the problem can be written as a linear programming problem then. A special linear approach based on constrained least squares is presented, which reduces the complexity of the problem in case of large data sets in contrast with the ℓ_∞ and the ℓ_1-norms.

An algorithm based on iterative knot insertion which generates a sequence of shape preserving approximants is given. It is investigated which linear objective functions are suited to obtain an efficient knot insertion method.

Keywords: spline approximation, linear constraints, linearisation of shape constraints, convexity, monotonicity

1991 Mathematics Subject Classification: 65D07, 65D15, 41A15, 41A29

1 Introduction

The problem of scattered data fitting under constraints is important in many applications, and a large number of methods for solving this problem have been proposed in the literature. Often, the problem can be posed as the following optimisation problem:

$$\begin{cases} \min\limits_{d} \|Ad - f\|, \\ \text{s.t. } C(d) \geq b. \end{cases} \tag{1.1}$$

The vector f contains the M given data values, and d is the vector containing the N unknowns, *e.g.,* the spline coefficients. The matrix A is of dimension $M \times N$, and the elements of A are

*financially supported by the Dutch Technology Foundation (STW)

determined by the approximation method, *e.g.,* a spline approximation. Constraints are given by a multivariate function C and a known vector b of dimension L. The inequality constraints are nonlinear in the unknowns, in general.

An example of a problem that can be solved by an optimisation program as in (1.1) is convexity-preserving approximation of scattered bivariate data using tensor-product B-splines, as we will discuss in more detail in section 6.

For practical reasons, it is attractive if the constraints in problem (1.1) are linear in the unknowns, or if they can be suitably approximated by sufficient linear constraints. In that case we have a constraint *matrix* C of dimension $L \times N$, and the optimisation problem simplifies to:

$$\begin{cases} \min\limits_{d} \|Ad - f\|, \\ \text{s.t. } Cd \geq b. \end{cases} \tag{1.2}$$

This report focusses on problem (1.2), and the following questions naturally arise:

1. concerning the objective function: which approximation method, *i.e.,* which norm, should be chosen?

2. concerning the constraints: how to linearise the shape constraints?

When these two questions have been answered, the constrained approximation problem can be solved. The goal is to approximate the data arbitrarily well, but this process generates only an approximation in general, therefore:

3. concerning convergence: can we give a sequence of approximants that come arbitrarily close to an interpolant?

These three issues are subsequently discussed in this report, which is organised as follows:

1. The first part of this report treats the choice of the approximation method, or more specifically, suitable choices for the norm in the objective function of problem (1.2). Here, we assume that the constraints are linear in the unknowns, and that they admit a feasible solution. Section 2 reviews some constrained ℓ_p-approximation methods. Approximation in the ℓ_2-norm, the ℓ_∞-norm and the ℓ_1-norm, respectively, are treated. Both constrained ℓ_∞-approximation and constrained ℓ_1-approximation can simply be rewritten as linear programming (LP) problems. Dependent on the application at hand, the different ℓ_p-norms all have their advantages and drawbacks, which is illustrated by a simple univariate example using convex cubic B-splines.

Section 3 presents a new class of constrained approximation methods. These methods are constructed by considering the constrained ℓ_2-approximation problem by minimising the ℓ_2-residuals in the ℓ_p-norm, $p \geq 1$. We refer to those norms as ℓ_p'. When $p = 1$ or $p = \infty$, this again leads to optimisation problems that can straightforwardly be rewritten to an LP-problem. The advantage of this approach is that these ℓ_p' algorithms turn out to be attractive especially in case of a large number of data: the complexity of the LP-problem becomes much lower for ℓ_p' than for ℓ_p-approximation. In addition, numerical experiments show that the constrained ℓ_p'-methods generate solutions that are much closer to the solution of the

constrained least-squares (ℓ_2) method than the constrained ℓ_p-approximation solutions for $p = 1$ or $p = \infty$ are.
The reasons why we focus on methods that can be written as LP-problems are the following. Firstly, various implementations of linear programming are available, even public domain. These methods are often simplex-based, relatively cheap, fast, robust and accurate, especially when the number of constraints is high. Secondly, the matrix containing the constraints is sparse, *i.e.,* it contains a lot of zeroes. For large-scale computations, solvers exist that are based on a sparse treatment of the objective function and the constraints. These commercial LP-solvers can even handle millions of constraints. In section 5 it is shown that an accurate linearisation of the bivariate convexity constraints for tensor-product B-splines results in a large number of linear constraints.

2. The second part of the report deals with the linearisation of the constraints. We discuss sufficient linear conditions for shape preservation, for example when dealing with approximating data with B-splines. First, in section 4, the construction of sufficient linear constraints for shape preservation of univariate B-splines is briefly discussed: sufficient conditions for positivity, monotonicity and convexity are presented, and they are formulated as linear inequalities in the unknowns.
For bivariate spline functions, sufficient conditions for preservation of positivity or monotonicity are presented as linear inequalities in the unknowns. In contrast with positivity and monotonicity, the conditions for convexity of a bivariate *function* are nonlinear in second derivatives of that function. Recently, in [CFP97], a linearisation of convexity conditions was proposed that is much better than that based on diagonal dominance of the Hessian matrix. Generalisations to weaker conditions, *i.e.,* conditions that are sufficient and 'almost' necessary, are given in [Jüt97b, Jüt97a]. A simple interpretation of these weaker conditions is given in section 5.3.

3. Finally, the third part of this report provides a methodology to obtain a shape preserving approximation that almost interpolates the given data, *i.e.,* a method that provides a solution satisfying an arbitrarily small error tolerance. A sequence of approximations using tensor-product B-splines is constructed. Each approximation is obtained by calculating the solution of a linear programming problem on a given knot set. The knot set of the B-spline is refined after each iteration in a specific way, if the approximation does not satisfy the required error tolerance. After a number of iterations, this sequence of approximations generates a tensor-product B-spline approximant that satisfies the shape constraints and that approximates the given data sufficiently well. Some comments are made on how to generate this sequence of approximations, such that convergence is guaranteed.

2 Constrained ℓ_p-approximation methods

In this section, problem (1.2) is discussed for approximation in the ℓ_p-norm.
Several numerical experiments for constrained ℓ_p-approximation have been performed. A typical example is provided by data points drawn from the function $f(x) = |x|$ on $[-1, 1]$ and the approximant is chosen in the class of cubic B-splines. The spline approximant is required

to be convex, but the restriction to this type of constraint is not relevant for the comparison in this report.

Subsequently, the given data and the cubic B-spline approximant are graphically displayed for this illustrative example. In a separate figure, the difference of the spline approximant with the given function is shown. The position of the knots on which the approximations are based is indicated at the x-axis. Clearly the number of data points is much larger than the number of knots, *i.e.*, we consider the case $M \gg N$.

The ℓ_2, ℓ_∞ and the ℓ_1-approximation methods are presented and compared in the next sections.

2.1 Constrained ℓ_2-approximation

Least squares approximation ($p = 2$) is the most well-known and widely-used method for approximation. The reason for this is that *unconstrained* ℓ_2-approximation leads to the problem of solving an $N \times N$ system of linear equations:

$$A^T Ad - A^T f = 0. \tag{2.1}$$

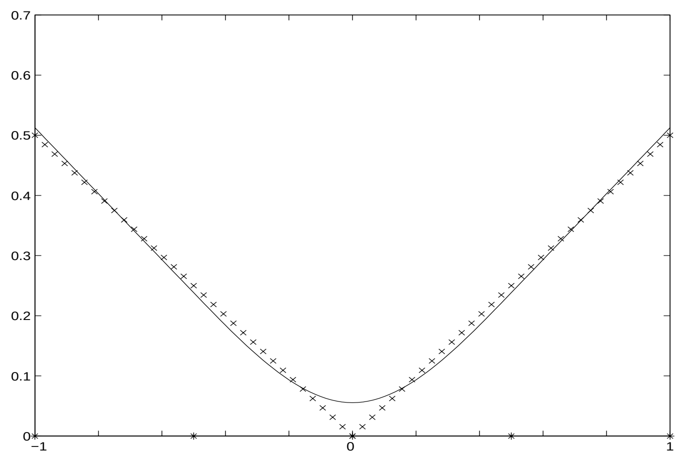

Figure 1: Convex ℓ_2-approximation of data from $|x|$.

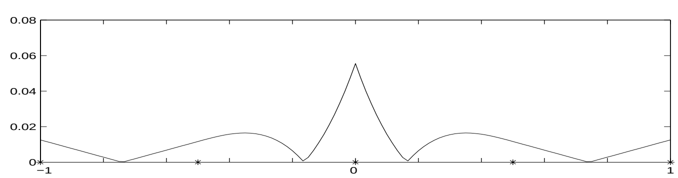

Figure 2: Difference between convex ℓ_2-approximation and $|x|$.

Efficient and accurate methods exist to solve this matrix equation, even in the case of large N. However, if the approximation problem is *constrained*, the solution of (2.1) in general is not feasible. The usual approach (in case of inequality conditions) is then to solve an optimisation problem consisting of a quadratic objective function subject to linear constraints. Efficient implementations of solution methods for constrained least squares exist. However, these quadratic programming (QP) methods are more complicated than linear programming (LP) solvers, especially for large problems.

2.2 Constrained ℓ_∞-approximation

In case of the max-norm ($p = \infty$), also called minimax-approximation, problem (1.2) can be transformed to a linear programming problem, in contrast with a QP-problem for constrained ℓ_2-approximation. It is a well-known fact that these LP-problems can be solved much faster and more efficient and accurate than QP-problems, especially for large problems.
The way to arrive at the LP-problem is as follows: in case of ℓ_∞-approximation, the maximum error is minimised, and the way to incorporate this in a LP-problem is to define the maximum error to be the scalar r, and add this single variable r as an additional unknown in the approximation problem:

$$\begin{cases} \min r, \\ \text{s.t. } Cd \geq b, \\ -r \cdot 1 \leq Ad - f \leq r \cdot 1, \end{cases} \tag{2.2}$$

where 1 is an M-vector in which all elements are equal to one.

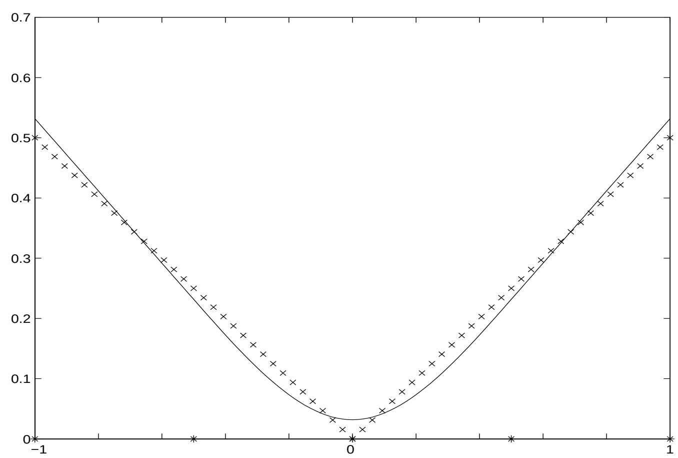

Figure 3: Convex ℓ_∞-approximation of data from $|x|$.

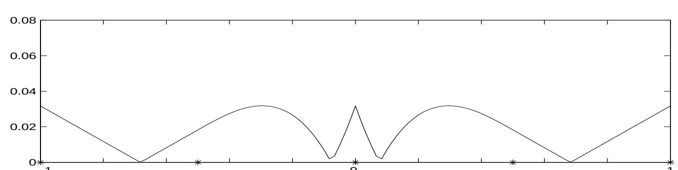

Figure 4: Difference between convex ℓ_∞-approximation and $|x|$.

This yields that the constraint matrix consists of $L+2M$ linear inequalities in $N+1$ unknowns.

2.3 Constrained ℓ_1-approximation

Another method is obtained by taking $p = 1$, and it is shown that ℓ_1-approximation can also be transformed to an LP-problem. Instead of introducing one single slack-variable (the unknown r in section 2.2), two slack vectors v and w of dimension M are introduced, satisfying:

$$v - w = Ad - f, \quad \text{with} \;\; v \geq 0,\; w \geq 0, \tag{2.3}$$

where v represent the positive part of $Ad - f$ and w the negative part. According to ℓ_1-norm approximation,

$$\sum_{i=1}^{M} |v_i - w_i|\,, \tag{2.4}$$

has to be minimised.

A simple observation shows that it is always possible to take (2.3) with the condition that either $v_i = 0$ or $w_i = 0$, $\forall i$: the case $v_i = a > 0$ and $w_i = b > 0$ cannot be a solution, as $v_i = a - \min\{a, b\}$ and $w_i = b - \min\{a, b\}$ then also solves the LP-problem with a lower objective.

Under these conditions the objective function in (2.4) simplifies and the LP-problem for constrained ℓ_1-approximation becomes:

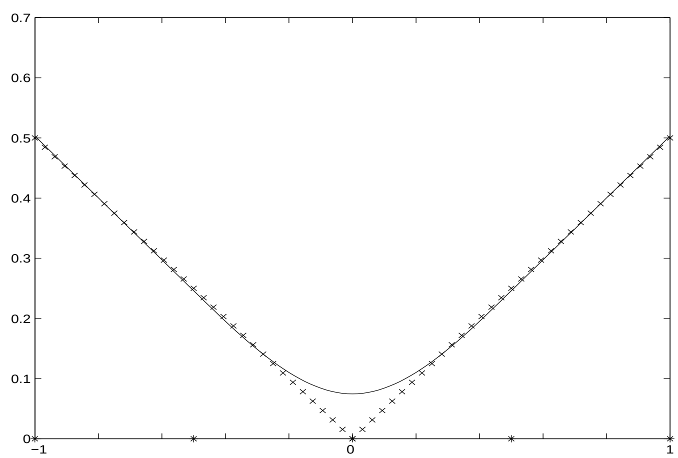

Figure 5: Convex ℓ_1-approximation of data from $|x|$.

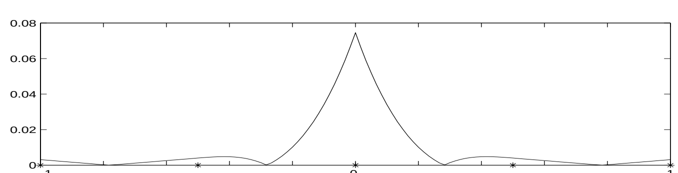

Figure 6: Difference between convex ℓ_1-approximation and $|x|$.

$$
\begin{cases}
\min \sum_{i=1}^{M} (v_i + w_i) , \\
\text{s.t. } Cd \geq b, \\
Ad - v + w = f, \\
v \geq 0, \ w \geq 0.
\end{cases}
\tag{2.5}
$$

This yields that the number of variables in this LP-problem is $N + 2M$ and the number of linear constraints $L + 3M$.

2.4 Comparison of constrained ℓ_p-approximation methods

Three methods for constrained ℓ_p-approximation have briefly been discussed in the previous sections: the cases $p = 1, 2, \infty$. We briefly summarise their advantages and drawbacks:

ℓ_2-approximation is probably the most well-known method. The *constrained* approximation problem however, cannot be solved by solving a system of linear equations, but generates a linearly constrained quadratic programming (QP) problem. This increases the computational costs and leads to inaccuracies, especially in case of many data points or a high dimensional spline space.

ℓ_∞-approximation is the method that minimises the maximum error that occurs in the approximation, and from a mathematical point of view, this is 'the best'. Max-norm approximation can be written as linear programming, but the method gives rise to the well-known minimax-behaviour: the maximal error occurs with both positive and negative sign at several locations of the approximation. For our later purposes, it turns out that this behaviour is disadvantageous, see section 6.

Also ℓ_1-approximation can be transformed to a linear programming problem. Approximation in the ℓ_1-norm does not suffer from the drawbacks of ℓ_2 (QP) or ℓ_∞ (minimax behaviour).

Both linearly constrained ℓ_1 and ℓ_∞-approximation can be rewritten to linear programming problems. However, the computational complexity increases in case of a large number of data, *i.e.,* if M becomes large. Then, the number of constraints increases significantly, and the LP-problem leads to large-scale problems.

In the next section, the data are approximated by so-called ℓ_p'-methods. These approximation methods can also be transformed to LP-problems and they are especially suited for large data sets.

Remark 2.1 (Sparse matrices) *In case of* e.g., *spline approximation, each spline segment is only governed by a relatively small number of coefficients, and the same holds for the linear shape constraints. Although this implies that the constraint matrix contains many zeroes, most standard implementations do not use this sparsity. However, when only nonzero elements are stored, this gives the opportunity of dealing with a large number of constraints. For this purpose, there exist large-scale sparse LP-solvers, which can even handle millions of constraints.*

3 Linear methods for constrained 'least squares'

In this section two methods for constrained approximation are introduced that lead to solutions that are closer to the constrained least squares solution. This turns out to be attractive as it gives an indication how to determine where knots have to be inserted. The methods can be solved by LP-problems, and the complexity is much lower than ℓ_1-approximation if the number of data is large.

3.1 Constrained ℓ_p'-approximation

In case of *unconstrained* ℓ_2-approximation the $N \times N$ system of linear equations (2.1) has to be solved for the unknown d. For constrained least squares approximation however, the solution of this linear system does not satisfy the constraints in general. The idea of the construction in this section is to minimise the ℓ_2-residuals in the ℓ_p-norm. The method, we call it ℓ_p'-approximation, is defined as follows:

$$\begin{cases} \min\limits_d \left\| A^T A d - A^T f \right\|_p, \quad p \geq 1, \\ \text{s.t. } Cd \geq b, \end{cases} \tag{3.1}$$

It is now straightforward to pose the linear programming problems that arise from ℓ_p'-approximation for the cases $p = \infty$ and $p = 1$.

The constrained ℓ_∞'-approximation problem is solved by the following LP-problem:

$$\begin{cases} \min r, \\ \text{s.t. } Cd \geq b, \\ -r \cdot 1 \leq A^T A d - A^T f \leq r \cdot 1, \end{cases} \tag{3.2}$$

and its computational complexity is determined by the $L + 2N$ linear constraints and $N + 1$ unknowns.

The LP-problem that solves constrained ℓ_1'-approximation becomes:

$$\begin{cases} \min \sum\limits_{i=1}^{M} (v_i + w_i), \\ \text{s.t. } Cd \geq b, \\ A^T A d - v + w = A^T f, \\ v \geq 0, \; w \geq 0. \end{cases} \tag{3.3}$$

The number of constraints is $L + 3N$, and the number of unknowns is $3N$.
Observe that the complexity of the ℓ_1' and ℓ_∞' method does not depend on the number of data M. These methods are therefore especially suited for approximation of large data sets and functions.

The results obtained by the ℓ_p'-approximation methods are compared with the results in the ℓ_p-norm in the next section.

3.2 Comparison of constrained ℓ_p'- and ℓ_p-approximation

In this section, the performance of the ℓ_p'-approximation methods is compared to results obtained by using the ℓ_p-approximation.

An important advantage of the LP-problems resulting from ℓ_p'-approximation is that their complexity is much lower than ℓ_p-approximation in case of large data sets, *i.e.*, if M is large. For example, the number of constraints for ℓ_∞ is $L + 2M$, where ℓ_∞'-approximation results in $L + 2N$ constraints. As a result, the constrained ℓ_p'-approximation methods are better for approximating functions from the point of complexity.

Secondly, ℓ_p'-approximation is attractive, because various numerical experiments clearly display the resemblance of ℓ_p' solutions to the ℓ_2-solution. Again, we consider the typical example from section 2. The difference between the spline approximants with the given function is shown for respectively the norms ℓ_∞, ℓ_∞', ℓ_2, ℓ_1', and ℓ_1. It is clearly seen in figure 7 that the approximations ℓ_∞' and ℓ_1' more resemble least squares than ℓ_∞ or ℓ_1 do, which is natural from the construction.

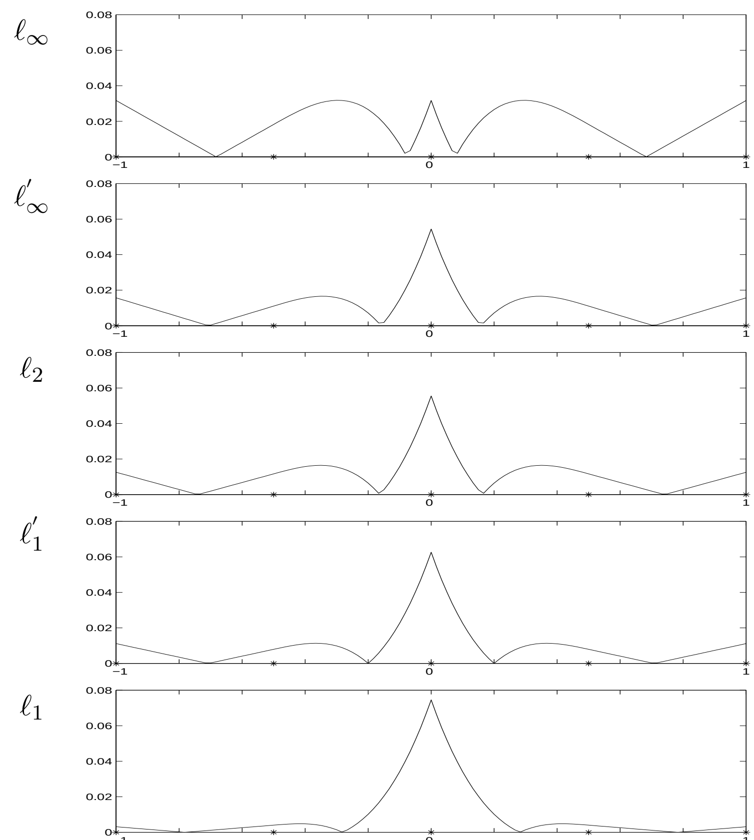

Figure 7: Error of convex approximation of dense data from $|x|$ on $[-1, 1]$.

4 Linear constraints for shape preservation of univariate splines

In this section we discuss the construction of sufficient linear constraints for shape preservation in case of univariate B-splines. The sufficient conditions can be formulated as linear inequalities in the B-spline control points. Subsequently, sufficient linear constraints for preservation of positivity, monotonicity and convexity are discussed.

The generation of shape constraints is relatively simple in the univariate case. It is necessary and sufficient for k-convexity of a univariate spline is that its k-th derivative is nonnegative, see report 2. In case of B-splines of degree n, this k-th derivative can be written as a B-spline of degree $n-k$ [dB78, Sch82, Far90]:

$$s^{(k)}(x) = \sum_{i=-n+k}^{N_\xi - 1} d_i^{(k)} N_i^{n-k}(x), \tag{4.1}$$

where the coefficients $d_i^{(k)}$ are the B-spline coefficients of the k-th derivative of s, which depend on d_i in a known way.

A well-known result for B-splines is that non-negativity of the B-spline coefficients yields a nonnegative B-spline, which straightforwardly follows from the convex-hull property of B-splines. The requirement that $d_i^{(k)} \geq 0$ is sufficient for k-convexity of the spline $s(x)$. This generates a number of linear inequality constraints in the B-spline coefficients d_i. The *Bézier polygon* of a spline is known to be closer to the spline than the B-spline control polygon, however. Non-negativity of the Bézier net is also a sufficient condition for non-negativity of a Bézier-Bernstein polynomial, and this conditions on the Bézier points are therefore less restrictive for preservation of k-convexity than the conditions on the B-spline control points $d_i^{(k)}$. Hence, the B-spline $s^{(k)}$ is converted into its Bézier-Bernstein formulation, and on every segment $[\xi_i, \xi_{i+1}]$, the spline can be written as a polynomial $s_i^{(k)}$ that is based on the Bézier points say b_i, $i = 0, \dots, n-k$. It is known that the spline segment $s_i^{(k)}$ is nonnegative if the Bézier points b_i are nonnegative. These conditions can straightforwardly be rewritten to linear inequalities in the coefficients d_i of the B-spline u.

Necessity of the constraints. The restriction of the sufficient linear constraints can be relaxed by degree elevation. According to the degree-raising formulas, instead of $n-k+1$ Bézier points b_i, a spline segment $s_i^{(k)}$ can be represented by $n-k+2$ Bézier points $b_i^{(1)}$. The Bézier polygon $\{b_i^{(1)}\}_i$ then lies in the convex hull of $\{b_i\}_i$. A condition for k-convexity of $s_i^{(k)}$ is then that $b_i^{(1)} \geq 0$, $\forall i$. After repeated degree raising, the Bézier net of $s_i^{(k)}$ is known to converge to the spline $s_i^{(k)}$ [dB78, Sch82, Far90]. As a result, the sufficient conditions converge to the necessary and sufficient shape conditions, when the degree of the representation of the spline tends to infinity. The degree of the spline u itself of course does not change, but the number of linear inequalities in the B-spline control points increases significantly.

The degree of the B-spline has not to be raised globally: the degree of the spline can be raised only in regions where the approximation is expected to have difficulties to satisfy the constraints. In those regions, the linear inequalities in the LP-problem will become active, and additional degrees of freedom have to be introduced, *e.g.*, by knot insertion.

In the bivariate case, the construction of linear constraints becomes much more involved. This is discussed in the next section.

5 Linear constraints for shape preservation of bivariate splines

In this section, the construction of linearised shape constraints for *bivariate* functions is discussed. Subsequently, positivity, monotonicity and convexity conditions are discussed, and the conditions are applied to tensor-product B-splines.
Again, as in the univariate case, the conditions for shape preservation are only sufficient and not necessary in general. However, the linear constraints become less restrictive after degree elevation, as the Bézier polygon is known to converge to the Bézier curve. When the degree of the spline is raised to a sufficiently high degree, the restriction becomes weaker and (almost) necessary.

5.1 Bivariate positivity constraints

In case of the construction of positivity constraints for bivariate tensor-product B-splines $s(x, y)$ it is sufficient to convert the B-spline to the Bézier-Bernstein form. Sufficient positivity constraints are then obtained by requiring that Bézier points $b_{i,j}$ are nonnegative.

5.2 Bivariate monotonicity constraints

In this section, we discuss the construction of linear conditions which are sufficient for monotonicity preservation.
The most frequently used condition for monotonicity reads

$$f_x \geq 0 \quad \text{and} \quad f_y \geq 0, \; \forall x, y, \tag{5.1}$$

and this condition can straightforwardly be applied to tensor-product B-splines by determining the B-spline derivatives s_x and s_y. As has been done for the univariate shape conditions, sufficient linear constraints are obtained by converting the B-spline expressions for both $s_x(x, y)$ and $s_y(x, y)$ to their Bézier-Bernstein form and requiring that the corresponding Bézier nets of s_x and s_y are both non-negative.
More general conditions for monotonicity of a multivariate function can be given. However, the definition of monotonicity is not unique and more complicated, especially in the multivariate case: it requires a suitable definition of ordering of data points. In the univariate case, the natural ordering (using $<$ and $\leq$) is unique (up to a sign). In the multivariate case, the definition makes use of a symbol $\prec$ (and $\preceq$), defined as follows:

Definition 5.1 (Ordering) *Let $\gamma_j \in \mathbb{R}^d$, $j = 1, \ldots, d$, be d linearly independent vectors. Any two points $x_1, x_2 \in \Omega \subset \mathbb{R}^d$ can be written as*

$$x_2 - x_1 = \sum_{j=1}^{d} \alpha_j \gamma_j, \quad \alpha_j \in \mathbb{R}.$$

The ordering notations $\succ$, $\prec$, $\succeq$, *and* $\preceq$ *are defined as*

$$\begin{aligned} x_2 \succ x_1 &\iff \alpha_j > 0,\ j = 1, \ldots, d, \\ x_2 \prec x_1 &\iff \alpha_j < 0,\ j = 1, \ldots, d, \\ x_2 \succeq x_1 &\iff \alpha_j \geq 0,\ j = 1, \ldots, d, \\ x_2 \preceq x_1 &\iff \alpha_j \leq 0,\ j = 1, \ldots, d. \end{aligned}$$

Note that $x_1 \prec x_2 \iff x_2 \succ x_1$.
Using the notion of ordering, suitable definitions for monotone functions and monotone data can be given:

Definition 5.2 (Monotone functions) *A function* $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ *is said to be* monotone *for a given ordering if for any two points* $x_1, x_2 \in \Omega$*:* $x_1 \preceq x_2 \quad \Rightarrow \quad f(x_1) \leq f(x_2)$.

Definition 5.3 (Monotone data) *A data set* $\{(x_i, f_i), x_i \in \Omega \subset \mathbb{R}^d, f_i \in \mathbb{R}\}_i$ *is said to be* monotone*, if for all* $x_1, x_2 \in \Omega \subset \mathbb{R}^d$*:* $x_1 \preceq x_2 \quad \Longrightarrow \quad f_1 \leq f_2$.

For *monotonicity in the univariate case* ($d = 1$), the ordering is usually based on the vector $\gamma_1 = e_1 = 1$, and then the notation $\preceq$ (which equals $\leq$) means monotone increasing, and $\preceq$ corresponds to $\geq$ and stands for monotone decreasing. Without loss of generality, in this , we only examine the first case which is usually referred to as *monotone*, *i.e.*, $\gamma_1 = 1$.
Again, if the function is univariate and continuously differentiable, monotonicity means that the first derivative has to be nonnegative everywhere. In the multivariate case, the notion of monotonicity for a given ordering is determined by $\gamma_1, \ldots, \gamma_d$ and for a continuously differentiable function, it becomes:

$$\gamma_j \cdot \nabla f \geq 0, \quad j = 1, \ldots, d. \tag{5.2}$$

An often-used ordering for monotonicity is the following:

Example 5.4 *A special case of ordering is obtained in the multivariate case by choosing for the vectors* γ_j *the unit vectors* e_j*, i.e.,* $\gamma_1 = (1, 0, \ldots)$*, etc. Then, the condition* $x_1 \preceq x_2$ *means that all components of* x_1 *and* x_2 *satisfy* $x_{1,j} \leq x_{2,j}$*,* $j = 1, \ldots, d$*. If case of a multivariate continuously differentiable function, this means that all partial derivatives are nonnegative everywhere.*

The the vectors γ_j determined the directions for which monotonicity is required. According to this definition, the condition for a bivariate continuously differentiable function becomes $\gamma_j \cdot \nabla f \geq 0$, $\forall j$, see (5.2), *e.g.*, in the bivariate case

$$\gamma_{1,1} f_x + \gamma_{1,2} f_y \geq 0 \quad \text{and} \quad \gamma_{2,1} f_x + \gamma_{2,2} f_y \geq 0. \tag{5.3}$$

Condition (5.1) is a special case of this condition, *i.e.*, $\gamma_j = e_j$, $j = 1, 2$.
The sufficient monotonicity conditions (5.3) for a bivariate *function* f are applied to tensor-product B-splines $s(x, y)$ of degree $n_x \times n_y$. The derivatives s_x and s_y are converted into the Bézier-Bernstein form, and linear combinations of s_x and s_y have to be calculated. To be

able to *algebraically* calculate the sum of the Bézier-nets of s_x and s_y, both must be of the same degree. Since s_x is of degree $(n_x - 1) \times n_y$, and s_y of degree $n_x \times (n_y - 1)$, the degree of s_x as well as s_y have to be raised to (at least) $n_x \times n_y$. Then, the linear combinations in the inequalities (5.3) can be determined, and sufficient conditions for monotonicity are obtained by requiring that the Bézier nets of both inequalities are nonnegative. Again, the resulting conditions are easily converted into linear inequalities in the B-spline control points.

5.3 Bivariate convexity constraints

The conditions for convexity of multivariate functions and data are well-known and defined in *e.g.*, [dC76] and [DM88]. The following three conditions are necessary and sufficient for convexity of a twice differentiable bivariate function $f(x, y)$:

$$f_{xx}(x,y) \geq 0, \quad f_{yy}(x,y) \geq 0 \quad \text{and} \quad f_{xx}(x,y)f_{yy}(x,y) - f_{xy}^2(x,y) \geq 0. \tag{5.4}$$

However, the third condition is nonlinear in the second derivatives of the function f, and therefore when applied to a spline these conditions are also nonlinear in the control points:

$$f_{xx}(x,y)f_{yy}(x,y) - f_{xy}^2(x,y) \geq 0. \tag{5.5}$$

In this section, this convexity condition is linearised, and it is shown that the linearised constraints on f are sufficient for convexity of f. The conditions are then applied to tensor-product splines.

A simple condition on f that is sufficient for convexity is [Dah91]:

$$f_{xx}(x,y) \geq 0, \quad f_{yy}(x,y) \geq 0, \quad f_{xy}(x,y) = 0, \ \forall x, y. \tag{5.6}$$

It is easily verified that (5.6) indeed implies the three conditions in 5.4. However, the class of surfaces that satisfy the condition $f_{xy} \equiv 0$ (a so-called translational function) is known to be too restrictive for real applications: on a rectangular domain for example, such a surface is fully described by two adjoining boundaries.

The following linearisation of the convexity conditions has turned out to be useful for practical purposes, and the conditions are less restrictive than (5.6). These conditions can be interpreted as diagonal dominance of the Hessian matrix, which elements are defined as

$$H_{i,j}(x) = \frac{\partial^2 f}{\partial x_i \partial x_j}(x), \quad i, j = 1, \ldots, d. \tag{5.7}$$

Proposition 5.5 (Linearisation of convexity conditions) *A two times continuously differentiable function $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ is convex if*

$$f_{xx}(x,y) \geq |f_{xy}(x,y)| \quad \textit{and} \quad f_{yy}(x,y) \geq |f_{xy}(x,y)|, \quad \forall (x,y) \in \Omega. \tag{5.8}$$

Proof. Since $|f_{xy}(x,y)| \geq 0$, the first and second condition in (5.4) are satisfied. The inequality

$$f_{xx}(x,y)f_{yy}(x,y) - f_{xy}^2(x,y) \geq |f_{xy}(x,y)|^2 - f_{xy}^2(x,y) = 0$$

makes that the third condition is also satisfied. □

The linear conditions (5.8) can be written into the following four inequalities in f and its derivatives:

$$f_{xx} + f_{xy} \geq 0, \quad f_{xx} - f_{xy} \geq 0, \quad f_{yy} + f_{xy} \geq 0 \;\text{ and }\; f_{yy} - f_{xy} \geq 0. \tag{5.9}$$

Application to splines. The sufficient convexity conditions (5.9) for a bivariate function $f(x, y)$ are applied to tensor-product splines $s(x, y)$. The degree of the bivariate spline is assumed to be $n_x \times n_y$, and the relevant second derivative functions are s_{xx}, s_{yy}, and s_{xy}. As an example, consider the first inequality from (5.9): $s_{xx} + s_{xy} \geq 0$. Again, as in the univariate case, both B-splines s_{xx} and s_{xy} are converted to the Bézier-Bernstein formulation. The degree of s_{xx} as well as s_{xy} has to be raised to at least $(n_x - 1) \times n_y$, to be able to algebraically calculate their sum. It is sufficient for non-negativity of this inequality that its Bézier net is nonnegative, which is a weaker condition for shape preservation than non-negativity of the B-spline control net. The conditions on the Bézier points are transformed back into linear inequalities in the B-spline coefficients. The other inequalities from (5.9) are treated similarly.

Complexity. An example shows that in general a large number of constraints is generated by this approach for the construction of sufficient linear convexity constraints. For a $n_x \times n_y$-degree B-spline based on the knot partitions $\xi_0, \ldots, \xi_{N_\xi}$ and $\psi_0, \ldots, \psi_{N_\psi}$, the number of inequalities becomes at least

$$L = N_\xi \times N_\psi \times (2(n_x - 1)n_y + 2n_x(n_y - 1)) = 2N_\xi N_\psi(2n_x n_y - n_x - n_y).$$

In case of a relatively simple example of cubic tensor-product B-splines based on $N_\xi = N_\psi = 20$, the number of inequalities becomes $L = 2 \cdot 20 \cdot 20(2 \cdot 3 \cdot 3 - 3 - 3) = 9600$. Note that the complexity increases when the approximation is based on a denser knot set and the number of inequalities increases even more when the degree of the spline becomes higher.

The following example shows that the sufficient linearised convexity conditions (5.8) are not necessary for convexity preservation:

Example 5.6 *Consider the two times differentiable bivariate function*

$$f(x, y) = 2x^2 + xy + \frac{1}{5}y^2.$$

This function is convex, since the (nonlinear) condition (5.5) holds:

$$4 \geq 0, \quad \frac{2}{5} \geq 0, \quad 4 \cdot \frac{2}{5} - 1^2 = \frac{3}{5} > 0.$$

The linearised conditions (5.8) however yield:

$$f_{xx} = 4 \geq 1 = |f_{xy}| \quad \textit{and} \quad f_{yy} = \frac{2}{5} \not\geq 1 = |f_{xy}|,$$

We discuss a method to improve the sufficient convexity conditions given by (5.8). This improvement is based on example 5.6 and given in the following proposition:

Proposition 5.7 (Linearisation of convexity conditions) *A two times continuously differentiable function* $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ *is convex if there exists a* $0 < L < \infty$*, such that*

$$f_{xx}(x,y) \geq L\,|f_{xy}(x,y)| \quad \text{and} \quad f_{yy}(x,y) \geq \frac{1}{L}\,|f_{xy}(x,y)|\,, \quad \forall (x,y) \in \Omega. \tag{5.10}$$

Proof. Since $L > 0$, we obtain that the first and second condition in (5.4) are satisfied. The inequality

$$f_{xx}(x,y)f_{yy}(x,y) - f_{xy}^2(x,y) \geq L\,|f_{xy}(x,y)|\,\frac{1}{L}\,|f_{xy}(x,y)| - f_{xy}^2(x,y) = 0,$$

makes that the third condition is also satisfied. □

The case $L = 1$ in proposition 5.7 corresponds to the conditions in proposition 5.5. In general, this choice is not optimal and the question arises how to determine a better choice for L, which is required to satisfy (5.10):

$$\frac{|f_{xy}(x,y)|}{f_{yy}(x,y)} \leq L \leq \frac{f_{xx}(x,y)}{|f_{xy}(x,y)|}.$$

Note that L in proposition 5.7 can depend on (x,y). A local estimate for L can be determined for each subdomain of the spline: $\Omega_{i,j} = \{(x,y)|\ \xi_i \leq x \leq \xi_{i+1},\ \psi_i \leq y \leq \psi_{i+1}\}$, by calculating numbers $a_{i,j}$, $b_{i,j}$, and $c_{i,j}$ that approximate f_{xx}, f_{yy} and $|f_{xy}|$ on $\Omega_{i,j}$ respectively by examining the data near $\Omega_{i,j}$. A local estimate for L is then for example obtained by the arithmetic mean or geometric mean, *i.e.*, :

$$L_{i,j} = \frac{1}{2}\left(\frac{c_{i,j}}{b_{i,j}} + \frac{a_{i,j}}{c_{i,j}}\right) \quad \text{or} \quad L_{i,j} = \frac{2}{\frac{c_{i,j}}{a_{i,j}} + \frac{b_{i,j}}{c_{i,j}}}.$$

Weak linear convexity conditions. The construction of linear convexity constraints is further simplified. The approach starts from the general convexity condition for a bivariate function, and it also gives a simple graphical interpretation of the general class of linear constraints that is presented in [Jüt97b].

Define λ_1 and λ_2 by

$$\lambda_1 := \frac{f_{xx}}{|f_{xy}|}, \quad \lambda_2 := \frac{f_{yy}}{|f_{xy}|}.$$

The condition that the Hessian matrix is positive semi-definite, see (5.5), becomes in terms of λ_1 and λ_2:

$$\lambda_1\lambda_2 \geq 1, \quad \text{with} \quad \lambda_1 \geq 0,\ \lambda_2 \geq 0. \tag{5.11}$$

The set of λ_1 and λ_2 that satisfy (5.11) is denoted by Λ^*, *i.e.,*

$$\Lambda^* := \left\{(\lambda_1, \lambda_2) \in \mathbb{R}^2 \mid\ \lambda_1\lambda_2 \geq 1,\ \lambda_1 \geq 0,\ \lambda_2 \geq 0\right\}.$$

The following condition is sufficient for (5.11):

$$\lambda_1 \geq 1 \quad \text{and} \quad \lambda_2 \geq 1. \tag{5.12}$$

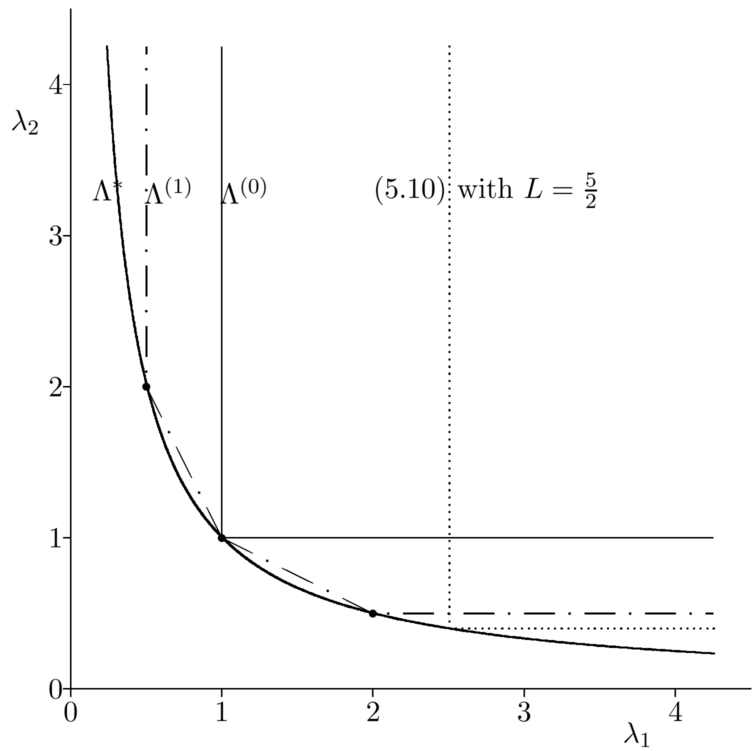

Figure 8: Graphical interpretation of linear convexity conditions.

These constraints provide a linearisation of the general convexity condition (Λ^*): the conditions generate the four linear convexity constraints that are presented in proposition 5.5.

Observe that every subset $\Lambda \subset \Lambda^*$ defines a collection of sufficient convexity conditions on f. Furthermore, Λ defines a set of *linear* sufficient convexity conditions, if the boundaries $\partial\Lambda$ of Λ are a set of straight lines.

We now set up a sequence $\Lambda^{(k)}$, $k \in \mathbb{N}$, of sets of sufficient linear convexity conditions with the following property $\Lambda^{(k)} \subset \Lambda^*$, $\forall k$. If, in addition, $\Lambda^{(k)}$ gets closer to Λ^*, the set of sufficient convexity conditions becomes weaker.

These observations make it easy to set up a suitable construction for generating linear convexity constraints. The function satisfying $\lambda_1\lambda_2 = 1$ simply has to be divided in a number of linear segments. This is done in the following example:

Example 5.8 (Weak sufficient convexity conditions) *Approximate the curve $\lambda_1\lambda_2 = 1$ with N_k linear segments, where $N_k = 2(k+1)$. Define the set of knots t_i as a partition of $[0,1]$: $0 = t_0 < t_1 < \ldots < t_k < 1$. For example, this can be done by $t_i = i/(1+k)$, $i = 0,\ldots,k$. Define $y_i = 1 - t_i$ and $x_i = 1/y_i$, then*

$$y_i = 1 - t_i = 1 - \frac{i}{1+k} = \frac{1+k-i}{1+k}, \quad \textit{and} \quad x_i = \frac{1}{y_i} = \frac{1+k}{1+k-i}.$$

Then define the following constraints:

$$c_i := (\lambda_1 - x_i)(y_{i-1} - y_i) + (\lambda_2 - y_i)(x_i - x_{i-1}) \geq 0, \; i = 1, \ldots, k, \quad \textit{and}$$
$$d_i := (\lambda_1 - y_i)(x_i - x_{i-1}) + (\lambda_2 - x_i)(y_{i-1} - y_i) \geq 0, \; i = 1, \ldots, k.$$

and

$$c_{k+1} = \lambda_1 \geq y_k = \frac{1}{1+k} = t_1 \quad \textit{and} \quad d_{k+1} = \lambda_2 \geq y_k = \frac{1}{1+k} = t_1.$$

Only if $t_0 > 0$, the additional constraint $\lambda_1 + \lambda_2 - (x_0 + y_0) \geq 0$ is necessary. The sequence $\Lambda^{(k)}$ that is constructed in this way is nested as $\Lambda^{(2k)} \subset \Lambda^{(k)}$ and approximates Λ^* better as k tends to infinity.
In the case $k = 0$ results in the conditions (5.12). The conditions proposed in [CFP97] are

$$L\lambda_1 \geq 1, \quad L\lambda_2 \geq 1, \quad L\lambda_1 + \lambda_2 \geq L + 1, \quad \lambda_1 + L\lambda_2 \geq L + 1, \quad L \geq 1, \tag{5.13}$$

and the case $L = 2$ corresponds with $k = 1$, i.e., $\Lambda^{(1)}$ in the construction above.
Furthermore, the case $k = 2$ results in the conditions $\Lambda^{(2)}$:

$$3\lambda_1 \geq 1, \; 3\lambda_2 \geq 1, \; 3\lambda_1 + 2\lambda_2 \geq 5, \; 2\lambda_1 + 3\lambda_2 \geq 5, \; 9\lambda_1 + 2\lambda_2 \geq 9, \; 2\lambda_1 + 9\lambda_2 \geq 9.$$

The examples are graphically shown in figure 8.

As before, these linear constraints on second derivatives of the function f can be applied to tensor-product B-splines, which generates a number of linear inequalities in the B-spline coefficients.

6 An iterative algorithm for shape preserving approximation

In this section, we propose an algorithm for constrained B-spline approximation. Consider the bivariate gridded data set $\{(x_i, y_j, f_{i,j}) \in \mathbb{R}^3\}$, but note that the restriction to gridded data is not essential. Next, a tensor-product B-spline function $s(x, y)$ is defined as

$$s(x, y) = \sum_{i=-n_x}^{N_\xi - 1} \sum_{j=-n_y}^{N_\eta - 1} d_{i,j} N_i^{n_x}(x) N_j^{n_y}(y). \tag{6.14}$$

which is based on the knot set $\{\xi_j\}_{j=0}^{N_\xi}$ in the x-direction and $\{\psi_j\}_{j=0}^{N_\psi}$ in the y-direction. These knot distributions divide the domain in $N_\xi \times N_\psi$ segments. The functions N_i^n are the B-spline basis functions, and the degree of the B-spline is assumed to be n_x in the x-direction, and n_y in the y-direction.

6.1 The algorithm

Once some ℓ_p-norm (or ℓ_p'-norm) is chosen, the objective function in (1.2) is defined. In addition, a linearisation of certain shape constraints for univariate B-splines (section 4) or

bivariate tensor-product B-splines (section 5) is applied. This properly defines the approximation problem (1.2), which in fact results in a linear programming (LP) problem, provided $p = 1, \infty$.
Usually a solution of this constrained approximation problem does not satisfy the prescribed error tolerance. The common way to deal with this problem is to introduce additional degrees of freedom. In this report, the dimension of the spline space is increased by inserting additional knots (this in contrast with increasing the degree of the spline). Refinement of the knot vectors results in more freedom for the approximant to satisfy the shape constraints as well as the required error tolerance. The idea behind the algorithm presented here is to insert extra knots where the approximation is not good enough, and then calculating an improved approximant. The algorithm distinguishes between 'method A' and 'method B'; the differences between the two are discussed later.
These observations lead to the following constrained spline approximation algorithm based on iterative knot refinement.

Algorithm 6.1 *Given a bivariate data set* $\{(x_i, y_j, f_{i,j})\}$.

1. *Define suitable initial knot sets* $\Xi = \{\xi_j\}_{j=0}^{N_\xi}$, *and* $\Psi = \{\psi_j\}_{j=0}^{N_\psi}$.
2. *Calculate initial constrained approximant using method A.*
3. *Continue loop until data are well approximated:*
   (a) *Determine the location of the point* (x_i, y_j) *for which* $|s(x_i, y_j) - f_{i,j}|$ *is maximal.*
   (b) *Add degrees of freedom using knot insertion: If* $x_i \notin \Xi$, *then add a knot* $\bar{\xi}_1 = x_i$, *i.e.,* $\Xi := \Xi \cup \{\bar{\xi}_1\}$,
   Else, add two additional knots $\bar{\xi}_1 = (\xi_{j-1} + x_i)/2$ *and* $\bar{\xi}_2 = (x_i + \xi_{j+1})/2$, *i.e.,* $\Xi := \Xi \cup \{\bar{\xi}_1, \bar{\xi}_2\}$.
   Similarly, Ψ *is extended by inserting one or two knots near* y_j.
   (c) *Optional: calculate constrained spline approximant using method B. The algorithm is terminated if the error is smaller than the given tolerance.*
   (d) *Calculate a new constrained spline approximant based on the extended knot sets* Ξ *and* Ψ *using method A.*
   The loop is terminated when the error satisfies the prescribed tolerance.
4. *Optional: calculate constrained spline approximant using method B.*

In the following section, the choice of the norm in the approximation method is discussed, *i.e.*, which method is suited for 'method A' and 'method B'. Next, some comments are made on knot insertion.

Remark 6.2 *The restriction to B-splines in this approach is not necessary. Instead of B-splines, one can use standard splines and add smoothness constraints, which are linear in the spline coefficients. Another possibility is to use splines defined on a triangulation, and a suitable initial triangulation is the convex triangulation. Iterative triangulation refinement* *is based on halving edges where the error is large. Auxiliary function values, only used to determine the new convex triangulation, are assigned to these new points using convexity preserving subdivision.*

6.2 Choice of the norm

In this section we discuss which methods A and B are suitable for application in algorithm 6.1.

'Method A' has to provide appropriate locations for introducing additional knots. As shown in the algorithm, this has to be done at locations where the error is large. Therefore ℓ_∞-approximation is not suited for the purpose of knot insertion, as it suffers from the alternating error behaviour of minimax-approximation, see figure 2.2. Least-squares approximation leads to a QP-problem, and therefore ℓ_1-approximation seems to be the best method. The ℓ_1-method is dual to ℓ_∞, which is also clearly displayed in the error distributions in figure 7. However, when rewriting ℓ_p-approximation, $p = 1, \infty$, to an LP-problem, the number of linear constraints increases significantly in case of a large number of data. The ℓ_p' methods from section 3 do not have this drawback. The illustrative example of approximating data from the function $|x|$, see figure 7, shows that the ℓ_p'-methods for $p = 1$ or $p = \infty$ are suited for the purpose of knot insertion. The error distribution strongly suggests to add a knot at (or near) $x = 0$. ${\ell_\infty}'$-approximation is a little more attractive since the complexity of the LP-problem is lower.

'Method B' is not used for knot-insertion, so any ℓ_p or ℓ_p' approximation method with $p = 1$ or $p = \infty$ can be used for this method. It is clear that in general the maximum difference between the spline approximation and the given data is to get smaller when ℓ_∞-approximation is used after application of any ℓ_p-norm or ℓ_p'-norm.

6.3 Location of the knots

The problem of choosing a suitable knot set is known to be difficult. Some comments are made in this section.

First of all, an initial knot set has to be chosen. For *unconstrained interpolation* problems, it is well-known that the dimension of the spline space that is necessary to interpolate the data is at most equal to the number of data points. In the constrained case however, it is not simple to determine the lowest number of knots that is necessary for interpolation. This number is generally not bounded by the number of data points, but depends on the data set. For example for data drawn from $f(x, y) = |x|$ on $[-1, 1]^2$, the sketched algorithm will never be able to interpolate in a convexity preserving way. Although in many situations it will be possible to use fewer knots, a conservative and suitable choice in general is to place a knot at every data point for the initial knot set.

According to algorithm 6.1, one or more knots are added in every iteration. This choice of knot refinement has turned out to be useful in various numerical experiments, *e.g.*, the univariate example of approximating data from $f(x) = |x|$ which is used as an illustration in sections 4 and 5. Nevertheless, it is remarked that more effort can be done to improve the treatment discussed in this . One extension is to use the following knot refinement method:

$$\bar{\xi}_1 = t\ \xi_{j-1} + (1-t)\ x_i, \quad \bar{\xi}_2 = (1-t)\ x_i + t\ \xi_{i+1},$$

where $0 < t < 1$. The choice of small values for t turned out to be more efficient in the example. The conservative and convenient choice $t = \frac{1}{2}$ has been chosen in the algorithm. For this value of t a homogeneous refinement near the data values is obtained. In general, an optimal value for t depends on the behaviour of the data.

For tensor-product B-splines, the insertion of new knots means that knot lines parallel to the coordinate axes have to be inserted. It is then simple to construct examples of difficult data for which any knot placement method requires a huge number of knots to satisfy small error tolerances in general. To overcome this problem of tensor-product B-splines, it is more natural to apply a method based on triangulations. The convex triangulation is suitable for this purpose. Nevertheless, constrained tensor-product B-spline approximation methods are very useful for many practical applications, and no triangular method is known to work as efficient as this tensor-product B-spline method as an approximation tool. However, methods based on triangulations could be more efficient, see remark 6.2.

References

[CFP97] J. M. Carnicer, M. S. Floater, and J. M. Peña. Linear convexity conditions for rectangular and triangular Bernstein-Bézier surfaces. *Comput. Aided Geom. Design*, 15:27–38, 1997.

[Dah91] W. Dahmen. Convexity and Bernstein-Bézier polynomials. In P.-J. Laurent, A. Le Méhauté, and L. L. Schumaker, editors, *Curves and Surfaces*, pages 107–134, Boston, 1991. Academic Press.

[dB78] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York-Heidelberg-Berlin, 1978.

[dC76] M. P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[DM88] W. Dahmen and C.A. Micchelli. Convexity of multivariate Bernstein polynomials and box spline surfaces. *Studia Sci. Math. Hungar.*, 23:265–287, 1988.

[Far90] G.E. Farin. *Curves and Surfaces for Computer Aided Geometric Design - A practical guide.* Academic Press, San Diego, second edition, 1990.

[Jüt97a] B. Jüttler. Arbitrarily weak linear convexity conditions for multivariate polynomials. Technical report, University of Technology, Department of Mathematics, Darmstadt, Germany, 1997.

[Jüt97b] B. Jüttler. Surface fitting using convex tensor-product splines. *J. Comput. Appl. Math.*, 84:23–44, 1997.

[Sch82] L.L. Schumaker. *Spline Functions, Basic Theory.* John Wiley, 1982.

Frans Kuijt and Ruud van Damme
University of Twente, Faculty of Applied Mathematics,
P.O. Box 217, NL-7500 AE Enschede, The Netherlands.
`http://www.math.utwente.nl/~kuijt`
`http://www.math.utwente.nl/~vandamme`
`f.kuijt@math.utwente.nl`
`vandamme@math.utwente.nl`