Envelopes of Nonlinear Geometry

David Lutterkort

Ph.D. Dissertation

Department of Computer Science
Purdue University

December 1999

This manuscript is based on my Ph.D. dissertation at Purdue University. The typesetting has been changed from the official Purdue thesis style to AMS Book with Euler fonts.

The contents have been edited to correct mistakes and to add further material.

It was typeset on December 15, 1999.

Abstract

A general framework for comparing objects commonly used to represent nonlinear geometry with simpler, related objects, most notably their control polygon, is provided. The framework enables the efficient computation of bounds on the distance between the nonlinear geometry and the simpler objects and the computation of envelopes of nonlinear geometry.

The framework is used to compute envelopes for univariate splines, the four point subdivision scheme, tensor product polynomials and bivariate Bernstein polynomials.

The envelopes are used to approximate solutions to continuously constrained optimization problems.

Acknowledgments

After years of graduate school, I feel like I have every right to copy Olin Shivers' acknowledgments for his SCSH Manual. But after a moment's clear thought it becomes obvious that there are quite a few people who contributed, knowingly or unknowingly, to this dissertation.

First of all, of course, my advisor, Jörg Peters, who encouraged me relentlessly to keep working on this topic. The financial support I got through him was of tremendous help, too. Contrary to the old proverb, studying *is* easier with a full stomach.

Carlos Gonzalez Ochoa gave me a wider view on the world of computer graphics. Some of the unknowing contributors at Purdue were Mark Webster and Chenghua Chen. Susan Deno and William Gorman took away much of the pain that is usually caused by administrative red tape.

I want to thank Georg Umlauf for his patient reading of several versions of this manuscript. There would be even more mitsakes in it without him.

I thank Tamara Troy for being there and supporting me even in those days in a graduate student's life when the light at the end of the tunnel seems to clearly be the headlight of an approaching train.

Most of all, I thank my parents for starting it all and making it possible. They had no idea.

Contents

List of Figures

List of Symbols

$\mathcal{B}$	Linear space of functions.
$\mathcal{H}$	Linear space of simplifications for functions in $\mathcal{B}$.
$\mathcal{U} = \mathcal{B} + \mathcal{H}$	Linear space containing both $\mathcal{B}$ and $\mathcal{H}$.
B	Set of basis functions for $\mathcal{B}$.
B_i	The i-th basis function in B.
H	Set of basis functions for $\mathcal{H}$.
$E : \mathcal{B} \mapsto \mathcal{U}$	Linear map embedding $\mathcal{B}$ in $\mathcal{U}$.
$L : \mathcal{B} \mapsto \mathcal{H}$	Linear map mapping a $b \in \mathcal{B}$ to its simplification $Lb \in \mathcal{H}$.
$D : \mathcal{B} \mapsto \mathbb{R}^s$	Difference operator, considered as a linear map or a set of functionals.
$D_i : \mathcal{B} \mapsto \mathbb{R}$	Functional mapping b to the i-th component of Db.
$K : \mathbb{R}^s \mapsto \mathcal{B}$	Antidifference basis, a basis of $(E - L)\mathcal{B}$.
K_k	The k-th function in K.
$K_{k,i}$	The i-th coefficient of K_k with respect to the basis B.
$\lfloor K \rfloor , \lceil K \rceil : \mathbb{R}^s \mapsto \mathcal{B}$	Lower and upper estimates of $(E - L)K$.
$\underline{b} , \overline{b} \in \mathcal{U}$	Lower and upper envelope of a $b \in \mathcal{B}$ with $\underline{b} \leq b \leq \overline{b}$.
$b(t)$	Value of $b \in \mathcal{B}$ at t.
b_i	The i-th coefficient of b with respect to a fixed basis B of $\mathcal{B}$.
$[P]$	For a boolean expression P, $[P]$ is 1 if P is true and 0 otherwise.

CHAPTER 1

Introduction

In computer graphics and computer aided geometric design, geometry is routinely parametrized by nonlinear elements such as Bernstein polynomials or B–splines. In choosing between a piecewise linear and a nonlinear representation of geometry, there is a trade–off. On one hand, nonlinear representations allow the construction of smooth surfaces, the higher approximation order of nonlinear representations can be used to represent geometry more compactly, and their explicit parametrization is important for applications such as texture–mapping. On the other hand, nonlinear elements require more sophisticated treatment: they have to be linearized for forward problems such as rendering, analysis problems such as curve intersections require the solution of nonlinear equations, and inverse problems such as fitting a smooth spline inside a polygonal channel need either be reformulated as linear problems or be solved by some nonlinear method.

For many applications it is enough to linearize the geometry by some method that guarantees convergence of the linearization to the nonlinear geometry. For example, B-splines are usually rendered by rendering a suitably subdivided control polygon. The subdivision needs to avoid both a linearization that is too coarse, leading to visual artifacts, or too fine, leading to unnecessary computational effort. A measure of the accuracy of the linearization is therefore needed, and its computation should be a small fraction of the cost of computing the linearization itself.

In other applications, it is imperative to have a bound on the error between linearized and nonlinear geometry. A prime example is ray–tracing: while in theory ray–surface intersection tests can be performed purely on the nonlinear geometry, for example by Newton's method, this is too inefficient in practice and often unstable. Usually, the nonlinear geometry is approximated by a hierarchy of linear envelopes and ray–surface intersections are performed on the linear envelopes to minimize the number of expensive nonlinear intersection computations. The linear envelopes are constructed from a linear approximation of the surface and a bound on the error between surface and linear approximation.

In fact, most analysis and inverse problems can be solved approximately by the application of linear envelopes: finding the zeros of a polynomial in Bernstein–Bézier form or testing a B–spline curve for self–intersection are solved more easily and efficiently by repeated adaptive refinement of linear envelopes than by employing nonlinear algorithms [**15**].

1.1. The thesis

It is possible to construct envelopes, in particular piecewise linear ones, for the commonly used repesentations of nonlinear geometry both effectively and efficiently. The construction requires neither that the representation fulfill a convex–hull property nor that the representation is of a given smoothness.

The construction can exploit the refinability of a given representation through the subdivide–and–bound procedure explained in Section 3.7.1, even for representations that only have a procedural evaluation algorithm or that lack the convex hull property.

The construction is efficient since the costly steps in computing envelopes are moved to a preprocessing phase that only depends on the representation used, but not on individual functions. The computation of envelopes for individual functions requires only the lookup of precomputed values and a few scalar products.

1.2. Prior work

Prior bounds in the literature can be divided into *qualitative* bounds and *quantitative* bounds. In general, qualitative bounds are too large or too impractical to compute to be used for the construction of envelopes.

1.2.1. Qualitative bounds. Qualitative bounds arise typically in the proof of convergence properties of nonlinear representations, for example in the proof of the convergence of the B-spline control polygon to a spline curve under repeated knot insertion. The bounds that can be derived from these proofs either contain unknown constants or are too large for practical purposes.

DeBoor [**6**] proves that the B–spline control polygon a of a spline b converges quadratically to the spline under subdivision (knot insertion) by showing that

$$\sup_t |b(t) - a(t)| \leq C\,|\mathbf{t}|^2 \sup_t |b''(t)|\,, \tag{1.2.1}$$

where $|\mathbf{t}|$ is the maximal distance between two consecutive knots in the knot sequence $\mathbf{t}$. The constant C is not known explicitly.

Prautzsch and Kobbelt [**23**] and Dahmen [**3**] give two different proofs of the quadratic convergence of the control polygon of a Bézier–curve to the curve under subdivision. The estimates on the distance between control polygon and curve implied by these proofs are larger than a simple min–max bound.

1.2.2. Quantitative bounds. Dozens of quantitative bounds have been developed for the more common geometry representations. Some of them, for example [**17**] and [**12**], require sophisticated preprocessing steps such as finding all points with horizontal or vertical tangents on a curve. A good survey of bounds for curve and surface intersections can be found in [**15**].

Bounds that are easy to implement and yield rectangular envelopes are often based on the *min–max criterion*, which is a consequence of the convex–hull property. For example, for a polynomial b in Bernstein–Bézier form with control points b_i, the min–max criterion states that the minimal (maximal) value of b lies between the minimal (maximal) control point, and that therefore

$$\min_i b_i \leq b(t) \leq \max_i b_i \qquad t \in [0,1], \tag{1.2.2}$$

which is often used because it is so simple to compute. The simplicity comes at the price of large min–max envelopes: they are in general larger than the convex hull.

For bases B that have the convex hull property, the convex hull of the control polygon b can be used to enclose a curve or surface. Convex hulls have two shortcomings: the computation of the convex hull is expensive, particularly in more than three dimensions, and, for adaptive subdivision, a stopping criterion based on the convex hull is equivalent to testing the flatness of the control polygon: for a convex or concave control polygon, the convex hull consists of the control polygon and a line connecting the first and last control point. The size of the convex hull is directly related to the flatness of the control polygon. But under repeated subdivision, almost all control polygons encountered will be convex or concave, since a polynomial or rational curve has only a finite number of inflection points. Nevertheless, the convex hull is arguably the most popular bounding criterion for bases B that possess the convex hull property. For bases that do not possess this property, Goldman and DeRose [**14**] show how a similar bound can be computed by inflating the convex hull of the control points by a factor that depends on the minimal (negative) values of the basis functions.

In [**13**], bounds on the distances between curves and surfaces and *secants* (lines connecting two points on a curve or triangles connecting three points on a surface) are derived from Taylor–expansions. Bounding boxes are then constructed by enlarging the bounding boxes of the secants by the bounds. The bounding boxes can only be refined by computing more points on the surface, which reduces the advantages that the geometric meaning of the coefficients of the Bernstein–Bézier representation offer.

Kobbelt [**16**] uses a secant based approach to construct envelopes for subdivision curves or surfaces. This paper is remarkable in that it contains the first effort to envelope general subdivision surfaces for which no closed–form basis is known. Since he finds that axis–aligned bounding boxes constructed with his method are too large, he uses *oriented bounding boxes*, boxes that are parallel to the control polygon pieces. His bound is piecewise constant and uses a min–max criterion similar to (1.2.2). The construction is based on secants. The computation of envelopes requires therefore the evaluation of points on the curve or surface and its cost is dominated by the cost of the evaluation.

Farin [**10**] shows that for rational Bézier–curves, the convex hull property can be tightened to the convex hull of the first and the last control point and the *weight* points, which are points on the legs of the control polygon associated with the weights of the rational Bézier representation.

Sederberg [**27**] encloses a curve $\mathbf{p}$ between two concentric circular arcs by estimating the maximum and minimum distance d of the points $\mathbf{p}(t)$ from a circle through a point $\mathbf{x}$,

$$d^2(t) = (\mathbf{p}(t) - \mathbf{x})^2.$$

The point $\mathbf{x}$ is chosen as the midpoint of the circle through $\mathbf{p}(0)$, $\mathbf{p}(1/2)$ and $\mathbf{p}(1)$.

In extension of the results in [**19**], Reif [**25**] describes a method for deriving bounds of the form

$$\|b - Lb\|_p \leq \|b^* - Lb^*\|_p \|\Delta_2 b\|_\infty$$

on the L^p norm $\|\cdot\|_p$ of the difference between a spline b and its control polygon Lb. The spline b^* only depends on the basis B, so that $\|b^* - Lb^*\|_p$ can be precomputed. This approach is applied to uni– and bivariate Bernstein–polynomials, splines and tensor products of Bernstein polynomials and splines. The resulting bounds are closely related to the Hölder bounds introduced in Theorem 3 on page 10. Most remarkably, he shows that for splines of degree d the difference between a spline b and its control polygon Lb is bounded as

$$\|b - Lb\|_\infty \leq \frac{\sup_j \sigma_j^2}{2d} \|\Delta^2 b\|_\infty,$$

where σ_j^2 is the variance of d consecutive knots. He uses this bound to derive an alternative proof of (1.2.1) with an explicit constant C.

1.3. Notational conventions

We use script letters $\mathcal{B}$, $\mathcal{H}$, etc. to denote linear spaces. Capital roman letters denote sets of basis functions B, H, etc., with individual basis functions denoted as B_i or H_i. We also use capital roman letters to denote linear operators D, K, etc.

We use the same symbols for a linear operator K, the set of functions K_k defined by applying it to a basis and the i-th coefficient $K_{k,i}$ of the k-th function K_k with respect to a fixed basis.

For functions $b \in \mathcal{B}$, we use $b(t)$ to denote the value of b at t and b_i to denote the i-th coefficient of b with respect to a fixed basis B of $\mathcal{B}$. We write $\underline{b}$ and $\overline{b}$ for functions that enclose b by $\underline{b} \leq b \leq \overline{b}$.

For boolean expressions P, we define $[P]$ to be

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true,} \\ 0 & \text{if } P \text{ is false.} \end{cases}$$

This gives a compact notation not only for $[i = j]$, which could also be written with the usual Kronecker δ, but also for more general expressions such as $[i > j]$.

CHAPTER 2

Bounds on nonlinear geometry

Quantitative bounds on nonlinearly parametrized *geometry* are based on bounds on nonlinear functions. This chapter will lay the groundwork for deriving bounds on nonlinear functions on which Chapters 3 and 4 build.

Bounds are computed by bounding the difference between a nonlinear function b and a simpler, often piecewise linear, function h. This in turn yields an *envelope* of b which is guaranteed to enclose it. The simplifications h are usually intimately linked to the nonlinear representation, such as the B-spline control polygon of a spline. The approach in this chapter does not pose any restrictions on what the simpler objects are, although most of the envelopes computed and used in the following chapters are control polygons, simply because they are the description of b that is used for computations anyway.

Computing envelopes based on our approach in practice consists of two steps: estimating the range of a small number of basis functions and forming linear combinations of the estimates. Only the second step, which is very simple to implement, depends on an individual b, while the first only depends on the basis in which we represent the b and a basis for the simpler functions h. This feature is very important in practice: usually, the basis in which the b are represented, for example uniform cubic splines, stays fixed, while the coefficients of b are modified. Splitting the computation into these two steps provides envelopes that require a precomputation that can be costly. But even a high precomputation cost is justified since it need only be carried out once for any basis. The precomputation yields values that are tabulated once and for all. The computation of envelopes for specific functions b is then very efficient.

2.1. Nonlinear geometry

We are interested in bounding the difference between a geometric object and a second, simpler object. We assume that the geometric objects are elements of a vector space $\mathcal{B}$ for which the functions B_i form a basis and that the simpler objects are elements of a second vector space $\mathcal{H}$ with basis H_i. Both $\mathcal{B}$ and $\mathcal{H}$ might be finite- or infinitedimensional. We assume that $\mathcal{B}$ and $\mathcal{H}$ are contained in a bigger vector space $\mathcal{U} = \mathcal{B} + \mathcal{H}$ and call the subspace common to $\mathcal{B}$ and $\mathcal{H}$, a subspace of $\mathcal{U}$, $\mathcal{W} = \mathcal{B} \cap \mathcal{H}$.

Example. We will use the polynomials b of degree at most d over the interval $[0, 1]$ as a running example and clarify the following definitions by considering bounds on the difference between a polynomial and its Bernstein control polygon.

The vector space $\mathcal{B}$ for this example is the space of all polynomials of degree less or equal to d, and we use the Bernstein polynomials $B_i^d, i = 0, \dots, d$ as its basis, so that any b is written as

$$b = \sum_{i=0}^{d} b_i B_i^d.$$

Since we are interested in the difference between a polynomial and its Bernstein control polygon, the space $\mathcal{H}$ of simpler objects is the space containing all piecewise linear functions with breaks at the Greville abscissae i/d, $i = 0, \dots, d$. The piecewise linear "hat functions" H_i^d with $H_i^d(j/d) = [i = j]$, $j = 0, \dots, d$, form a basis of $\mathcal{H}$.

The space $\mathcal{W} = \mathcal{B} \cap \mathcal{H}$ is the space of all functions that are piecewise linear and, simultaneously, polynomial of degree at most d, i.e., the linear functions. Therefore, $\dim \mathcal{W} = 2$ and

$$\dim \mathcal{U} = \dim \mathcal{B} + \dim \mathcal{H} - \dim \mathcal{W} = 2d.$$

△

We use two linear operators to model the difference between objects in $\mathcal{B}$ and $\mathcal{H}$: the map $E : \mathcal{B} \mapsto \mathcal{U}$ which embeds $\mathcal{B}$ in $\mathcal{U}$ and the map $L : \mathcal{B} \mapsto \mathcal{H}$ which associates one simpler object with each geometry object. We require L to leave $\mathcal{W}$ pointwise fixed. Typically, we choose the operator L based on some important property of the representation B and take $\mathcal{H} = L\mathcal{B}$ so that L is surjective.

The difference between a $b \in \mathcal{B}$ and its simplification Lb is now $(E - L)b$. The linear map $E - L$ maps exactly the $b \in \mathcal{B} \cap \mathcal{H} = \mathcal{W}$ to 0, i.e., $\ker(E - L) = \mathcal{W}$. Therefore, if $\dim \mathcal{B} < \infty$, $\operatorname{rank}(E - L) = \dim \mathcal{B} - \dim \mathcal{W}$.

Example (*continued*). For polynomials in Bernstein form, L maps a polynomial b to its control polygon, the piecewise linear interpolant of its control points at the Greville abscissae i/d,

$$Lb = \sum_{i=0}^{d} b_i H_i^d.$$

By the linear precision of the Bernstein polynomials, $Lb = b$ for all $b \in \mathcal{W}$ and L does indeed leave $\mathcal{W}$ pointwise fixed.

The map $E - L$ maps a polynomial b to the difference between it and the control polygon, a piecewise polynomial with breaks at the Greville abscissae. The rank of $E - L$ is $(d+1) - 2 = d - 1$. △

2.2. Representing the difference

We split the difference $(E - L)b$ into two parts by rerepresenting b in a basis K of $(E - L)\mathcal{B}$.

Theorem 1. *Let* $s = \operatorname{rank}(E - L)$ *if* $\dim \mathcal{B} < \infty$ *and* $s = \infty$ *if* $\dim \mathcal{B} = \infty$. *Let the linear operators* $D : \mathcal{B} \mapsto \mathbb{R}^s$ *and* $K : \mathbb{R}^s \mapsto \mathcal{B}$ *be such that* $\ker D = \mathcal{W}$ *and*

$$DK = I, \qquad I \textit{ the identity on } \mathbb{R}^s. \tag{2.2.3}$$

Then

$$(E-L)b = (E-L)K\,Db \qquad \textit{for any } b \in \mathcal{B}. \tag{2.2.4}$$

We call the linear operator D the *difference operator* and the vector Db the *differences* of $b \in \mathcal{B}$. We call the basis K_i of $(E-L)\mathcal{B}$ the *antidifference basis*.

PROOF. For brevity, let $K^* = (E-L)K$. We extend D to a map D^* on all of $\mathcal{U}$ by defining $D^*h = 0$ for any $h \in \mathcal{H}$. This means that $D = D^*E$ and that $\ker D^* = \mathcal{H}$. Therefore, $D^*(E-L) = D$ which implies that $D^*(E-L)b = Db$ and that

$$D^*K^*Db = D^*(E-L)KDb = DKDb = Db.$$

Therefore,

$$Db - D^*K^*Db = 0,$$
$$D^*(Eb - K^*Db) = 0,$$

and, since $\ker D^* = \mathcal{H}$,

$$Eb - K^*Db \in \mathcal{H},$$

which means, since $Lb \in \mathcal{H}$, that

$$(E-L)b - K^*Db = (E-L)(b - KDb) \in \mathcal{H}.$$

But if for a $c \in \mathcal{B}$, $(E-L)c \in \mathcal{H}$, then $Ec = c \in \mathcal{H}$ and $c \in \mathcal{B} \cap \mathcal{H} = \ker(E-L)$, and therefore $(E-L)(b-KDb) = 0$ and $(E-L)b = (E-L)KDb$ as claimed. □

REMARK 1. For a given D, the solution K to equation (2.2.3) is not uniquely determined. Solutions to (2.2.3) differ only by elements of $\ker\ D = \mathcal{W}$, though. In the following chapters, we choose a unique solution to (2.2.3) by enforcing additional constraints on the functions K, for example by making their control polygons symmetric.

REMARK 2. For $\dim \mathcal{B} < \infty$, the requirement $\ker D = \mathcal{W}$ implies that $\operatorname{rank} D = \dim \mathcal{B} - \dim \mathcal{W} = \operatorname{rank}(E-L)$. The choice $s = \operatorname{rank}(E-L)$ is therefore the smallest possible. A theorem similar to Theorem 1 can be proven for values $s > \operatorname{rank}(E-L)$. This amounts to constructing a K that merely spans $(E-L)\mathcal{B}$ instead of forming a basis of $(E-L)\mathcal{B}$. The additional degrees of freedom in choosing D with an $s > \operatorname{rank}(E-L)$ can be used to construct an antidifference basis K with additional properties. For the bounds derived in the following chapters, these additional degrees of freedom have never proven to be useful so that we will always use the minimal choice for s.

REMARK 3. The support of the function $(E-L)K_i$ does not need to be finite, even if the basis functions B_j are finitely supported. The support of $(E-L)K_i$ will only be finite if L is chosen so that $b = Lb$ outside of a finite interval for all $b \in \mathcal{B}$.

EXAMPLE (*continued*). For Bernstein polynomials, we choose the second differences of the control points of b as D so that the vector Db is defined as

$$D_{j-1}b = b_{j-1} - 2b_j + b_{j+1} \qquad j = 1, \dots, d-1.$$

Clearly, $Dh = 0$ for any $h \in \mathcal{B} \cap \mathcal{H}$.

Since $\dim \mathcal{B} < \infty$, Equation (2.2.3) is an equation between matrices K and D. The map K is represented as a $d+1 \times d-1$ matrix, while D is a $d-1 \times d+1$ matrix. A solution to the equation $DK = I$ consists of polynomials K_i with

$$K_i = \sum_{j=0}^{d} K_{i,j}\, B_j^d \qquad K_{i,j} = \frac{1}{2}\,|i - j + 1|\,, \quad 0 \le i \le d-2, 0 \le j \le d.$$

Note that the functions K_i are convex because their control polygons are convex. △

2.3. Range estimation

We are now in the position to derive bounds on $(E - L)b$ by using the change of basis from Theorem 1 and an estimate on the range of the operator K on $\mathbb{R}^s$ by estimating the terms in the scalar product between $(E - L)K$ and Db.

Define the (nonlinear) operators $\lfloor\!\lfloor \cdot \rfloor\!\rfloor$ and $\lceil\!\lceil \cdot \rceil\!\rceil$ on $\mathbb{R}^s$ such that for a $v \in \mathbb{R}^s$ the vector $\lfloor\!\lfloor v \rfloor\!\rfloor \in \mathbb{R}^s$ ($\lceil\!\lceil v \rceil\!\rceil \in \mathbb{R}^s$) is obtained from v by replacing all positive (negative) entries in v by 0. Therefore, $\lfloor\!\lfloor v \rfloor\!\rfloor$ contains only entries which are nonpositive and $\lceil\!\lceil v \rceil\!\rceil$ contains only entries which are nonnegative.

THEOREM 2. *Let s and the linear operators K and D be defined as in Theorem 1. Let $\lfloor K \rfloor, \lceil K \rceil : \mathbb{R}^s \mapsto \mathcal{B}$ be linear operators such that the following inequalities hold pointwise:*

$$\lfloor K \rfloor\, e^i \le (E - L)Ke^i = (E - L)K_i \le \lceil K \rceil\, e^i, \tag{2.3.5}$$

where the e^i are the canonical basis vectors of $\mathbb{R}^s$, i.e., $e^i_j = [i = j]$, $0 \le i, j < s$. The difference $(E - L)b$ between any $b \in \mathcal{B}$ and Lb is bounded by

$$\lfloor K \rfloor\, \lceil\!\lceil Db \rceil\!\rceil + \lceil K \rceil\, \lfloor\!\lfloor Db \rfloor\!\rfloor \le (E - L)KDb \le \lfloor K \rfloor\, \lfloor\!\lfloor Db \rfloor\!\rfloor + \lceil K \rceil\, \lceil\!\lceil Db \rceil\!\rceil \tag{M}$$

PROOF. We only show the second inequality. For a $b \in \mathcal{B}$, let $v = Db$, $\underline{v} = \lfloor\!\lfloor Db \rfloor\!\rfloor$, $\overline{v} = \lceil\!\lceil Db \rceil\!\rceil$ in $\mathbb{R}^s$. By Theorem 1, $(E - L)b = (E - L)Kv$. The scalar product between $(E - L)K$ and v is bounded by

$$\begin{aligned}
(E - L)Kv &= \sum_{i=0}^{s} (E - L)K\, v_i e^i = \sum_{i=0}^{s} (E - L)K\, (\underline{v}_i + \overline{v}_i) e^i \\
&= \sum_{i=0}^{s} \underline{v}_i\, (E - L)Ke^i + \sum_{i=0}^{s} \overline{v}_i\, (E - L)Ke^i \\
&\le \sum_{i=0}^{s} \underline{v}_i\, \lfloor K \rfloor\, e^i + \sum_{i=0}^{s} \overline{v}_i\, \lceil K \rceil\, e^i = \lfloor K \rfloor\, \underline{v} + \lceil K \rceil\, \overline{v}.
\end{aligned}$$

The proof of the first inequality is analogous. □

Remark 4. The *bound* on $(E - L)b$ from Equation (M) is turned into an *envelope*, an offset of Lb, by adding Lb to the inequalities (M). This yields lower and upper envelopes of b,

$$Lb + \lfloor K \rfloor \lceil\!\lceil Db \rceil\!\rceil + \lceil K \rceil \lfloor\!\lfloor Db \rfloor\!\rfloor \leq b \leq Lb + \lfloor K \rfloor \lfloor\!\lfloor Db \rfloor\!\rfloor + \lceil K \rceil \lceil\!\lceil Db \rceil\!\rceil.$$

We often abbreviate the expressions on the left and on the right to $\underline{b}$ and $\overline{b}$ if D, L and K are understood from the context and write $\underline{b} \leq b \leq \overline{b}$.

Remark 5. If $\lfloor K \rfloor \leq 0 \leq \lceil K \rceil$, as is often the case, the lower and upper envelopes $\underline{b}$ and $\overline{b}$ of a function b enclose both the function b and its simplification Lb. The width of the envelope, $\overline{b} - \underline{b}$ is then an upper bound on the distance between b and Lb,

$$|b - Lb| \leq \overline{b} - \underline{b}.$$

This is no longer the case if either $\lfloor K \rfloor$ takes on positive values or $\lceil K \rceil$ takes on negative values.

For practical computations, the estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ are precomputed and can usually be represented by a few constants. The map K depends on $\mathcal{B}$, $\mathcal{H}$ and the difference operator D which are fixed for a specific application.

Since the functions Ke^i are elements of $\mathcal{B}$, we seem to have come full circle: to estimate the difference between functions b in $\mathcal{B}$ and their simplification Lb in $\mathcal{H}$, we need to bound certain functions in $\mathcal{B}$ by functions in $\mathcal{H}$. We made progress, however, for two reasons: firstly, we can bound $(E - L)b$ for any function in $\mathcal{B}$ if we can find bounds for a small number of functions in $\mathcal{B}$ and secondly, since, with an appropriate choice of D, we can choose functions Ke^i which are simple to bound by functions from $\mathcal{H}$.

Theorem 2 is valid for *any* $\lfloor K \rfloor$ and $\lceil K \rceil$ that fulfill Equation (2.3.5); the practical computation of bounds (M) is usually easier if $\lfloor K \rfloor$ and $\lceil K \rceil$ are in $\mathcal{H}$. Therefore, we usually compute $\lfloor K_i \rfloor$ by constructing a function $h \in \mathcal{H}$ with $h \leq K_i$ and set $\lfloor K_i \rfloor = h - LK_i \leq (E - L)K_i$.

Example (continued). Since the functions K_i for polynomials of degree d computed on the preceding page are convex and because of the convex hull property of the Bernstein representation, K_i lies above its control polygon LK_i and below its piecewise linear interpolant at the Greville abscissae j/d. Piecewise linear estimates $\lfloor K_i \rfloor$ and $\lceil K_i \rceil$ on $(E - L)K_i$ are therefore obtained by subtracting the control polygon LK_i from these estimates,

$$\lfloor K_i \rfloor = 0 \quad \text{and} \quad \lceil K_i \rceil = \sum_{j=0}^{d} (K_i(j/d) - K_{i,j})\, H_j^d.$$

$\triangle$

2.4. Hölder bounds

We can use Hölder's inequality to obtain bounds that are in general coarser, but simpler to compute than the ones from Theorem 2. This yields a pointwise bound on $(E - L)b$ in terms of a norm of the vector of differences of b and a norm on the vector of functions K_i, which is a function in $\mathcal{U}$. Further applying a norm on $\mathcal{U}$ to this bound gives an estimate of the norm of $(E-L)b$ as a function.

Theorem 3. *Let* s *and the linear operators* D *and* K *be defined as in Theorem 1. Let* p *and* q *be such that* $1/p+1/q=1$ *and* $\|\cdot\|_p$ *and* $\|\cdot\|_q$ *be vector norms. Then*

$$|(E-L)b| \leq \|(E-L)K\|_p \, \|Db\|_q. \tag{H}$$

Proof. By Theorem 1, $(E-L)b=(E-L)KDb$. Hölder's inequality gives the estimate

$$|(E-L)b(t)| = |(E-L)K(t)\,Db| \leq \|(E-L)K(t)\|_p \, \|Db\|_q$$

where we identify $(E-L)K(t)$ with the vector with entries $(E-L)K_i(t)$. □

In general, $\|(E-L)K\|_p$ is a function in $\mathcal{U}$. The above bound (H) becomes particularly useful if we estimate $\|(E-L)K\|_p$ by a function in $\mathcal{H}$ as will be the case in Chapter 3.

Using a norm on $\mathcal{U}$ gives the promised estimate on the norm of $(E-L)b$ as a function:

Corollary 4. *With the assumptions of Theorem 3 and* $\|\cdot\|_{\mathcal{U}} : \mathcal{U} \mapsto \mathbb{R}$ *a norm,*

$$\|(E-L)b\|_{\mathcal{U}} \leq \|\|(E-L)K\|_p\|_{\mathcal{U}} \, \|Db\|_q. \tag{2.4.6}$$

Example (continued). We use $p=1$ and $q=\infty$. Since

$$z(t) = \sum_{i=0}^{d-2} K_i(t) = \frac{d-1}{2} t^2 + t + 2d - 1,$$

as is shown in Section 3.4 on page 20, $\|(E-L)K\|_1 = (E-L)z$. Note that z is quadratic regardless of the degree d of the polynomials in $\mathcal{B}$. Theorem 3 now asserts that

$$|(E-L)b(t)| \leq z(t) \max_{i=0}^{d-2} |D_i b|.$$

Since z is a quadratic polynomial with positive leading coefficients, we can estimate it by its piecewise linear interpolant at the Greville abscissae, giving us the piecewise linear bound

$$|(E-L)b(t)| \leq \sum_{j=0}^{d} (E-L)z(j/d) \max_{i=0}^{d-2} |D_i b|.$$

Section 3.4 also shows that $(E-L)z \leq d/8$. With the norm

$$\|f\|_{\mathcal{U}} = \max_{t \in [0,1]} |f(t)| \qquad \text{for } f \in \mathcal{U},$$

the Corollary 4 asserts that

$$\|(E-L)b\|_{\mathcal{U}} \leq \frac{d}{8} \max_{i=0}^{d-2} |D_i b|.$$

△

2.5. Summary of the bounding process

The framework for deriving bounds in this chapter can be summarized in the following steps. These steps are *preprocessing* steps, depending only on the space $\mathcal{B}$ and therefore need only be carried out once for each basis.

(1) Choose a space $\mathcal{H}$ and an operator $L : \mathcal{B} \mapsto \mathcal{H}$ that leaves $\mathcal{W} = \mathcal{B} \cap \mathcal{H}$ pointwise fixed.
(2) Choose a difference operator $D : \mathcal{B} \mapsto \mathbb{R}^s$ with $\ker D = \mathcal{W}$.
(3) Find $K : \mathbb{R}^s \mapsto \mathcal{B}$ such that DK is the identity on $\mathbb{R}^s$, $s = \operatorname{rank}(E-L) = \dim \mathcal{B} - \dim \mathcal{W}$.
(4) For bounds of type (M), determine $\lfloor K \rfloor$ and $\lceil K \rceil$. For bounds of type (H), determine (an upper bound of) $\|K\|_p$.

With the knowledge of $\lfloor K \rfloor$, $\lceil K \rceil$ or $\|K\|_p$, the implementation of bounds of type (M) or (H) requires only the computation of $\llbracket \lfloor Db \rfloor \rrbracket$, $\llbracket \lceil Db \rceil \rrbracket$ or $\|Db\|_q$ and a few scalar products.

2.6. Refinement of the basis

Almost all bases for $\mathcal{B}$ used in practice have a refinement property, namely that, for a function $b \in \mathcal{B}$ over the domain $\mathbb{U}$, its restriction to a subdomain $\mathbb{S} \subset \mathbb{U}$ can be efficiently represented as a function $\hat{b} \in \mathcal{B}$ over $\mathbb{U}$ after reparametrization. Changing the representation from b to $\hat{b}$ is usually called subdivision referring to the fact that the *domain* is subdivided into smaller domains.

Subdivision allows us to use the estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ of some K to compute envelopes for functions over $\mathbb{S}$ in the same manner as for functions over $\mathbb{U}$ without incurring additional approximation errors.

In fact, many representations, for example B-splines, have the property that certain differences Db of the control polygon of a b tend rapidly to zero as the size of $\mathbb{S}$ goes to zero. If the factor by which Db shrinks under subdivision is known, envelopes of b computed with the difference operator D shrink by at least that factor; this yields *a priori* estimates on the maximal size of $\mathbb{S}$ for which the width of the envelope of $\hat{b}$ is smaller than a prespecified tolerance. Section 3.6 on page 24 explains this for uniform splines.

Subdivision can also be used to improve estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ on K as is explained in Section 3.7.1 and to approximate solutions to one– and two–sided approximation problems as is shown in Section 5.4 on page 58.

CHAPTER 3

Univariate functions

We demonstrate the framework laid out in Chapter 2 for two cases: splines and the cubic interpolatory subdivision scheme. Piecewise linear envelopes for splines are of great practical importance because they are routinely used as approximating functions and for representing geometry for numerical calculations. The efficiency of many applications, for example rendering or intersection testing, hinges on a tight quantitative estimate of the maximal distance of the spline from its control polygon.

We consider cubic interpolatory subdivision because the basis for this scheme lacks most of the useful features of the B–spline basis. We can still derive piecewise linear envelopes for cubic interpolatory subdivision functions solely based on their subdividabilty and on their smoothness.

3.1. Splines in B–spline form

A piecewise polynomial b of degree d is in B–Spline form if

$$b(t) = \sum_{k\in\mathbb{Z}} b_k N_k^d(t)$$

where the *control points* b_k are real numbers and the B–Spline basis functions N_k^d are defined recursively based on a nondecreasing sequence of real numbers, the *knot sequence* (t_i) [**6**]. The following is valid both for finite knot sequences and for infinite knot sequences (t_i).

We may assume that b is at least continuous. Otherwise, we can treat b as two separate splines. This implies that any knot can appear with multiplicity at most d, except for the first and last knot of a finite knot sequence (t_i) which can have multiplicity $d+1$.

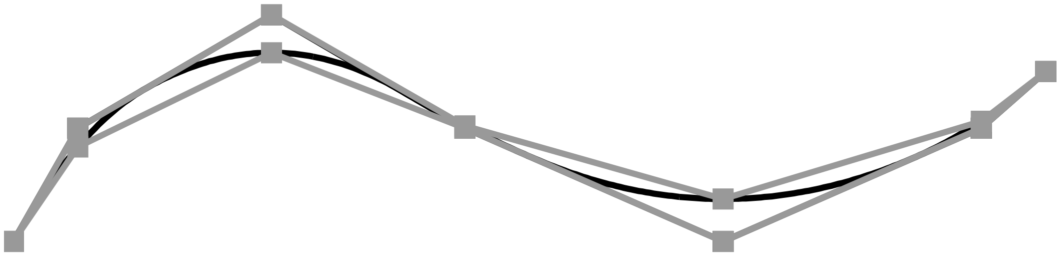


FIGURE 3.1. A quadratic spline over the knot sequence $(0,0,0,1,3,4,7,8,8,8)$ with control points $(0,2,4,2,0,2,3)$. The spline is shown in black. The envelope according to Theorem 5 on page 16 is shown in grey. The control polygon is covered by the envelope.

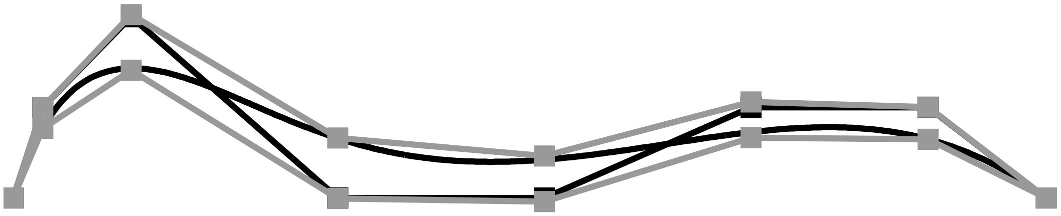


FIGURE 3.2. A quintic spline and its envelope according to Theorem 5. The knots are at $(0, 1, 3, 7)$ and the control points are $(0, 1, 2, 0, 0, 0, 1, 1, 0)$. The first and last knot have multiplicity 6.

Therefore, the *Greville abscissae* t_k^*,

$$t_k^* := \frac{1}{d} \sum_{i=k+1}^{k+d} t_i,$$

are distinct.

The *control polygon* Lb,

$$(Lb)(t) = \sum_{k \in \mathbb{Z}} b_k H_k(t)$$

of b is the piecewise linear interpolant to the control points b_k at the Greville abscissae t_k^*. The "hat functions" H_k are the piecewise linear functions with

$$H_k(t) = \begin{cases} \dfrac{t - t_{k-1}^*}{t_k^* - t_{k-1}^*} & t \in [t_{k-1}^*, t_k^*] \\ \dfrac{t_{k+1}^* - t}{t_{k+1}^* - t_k^*} & t \in [t_k^*, t_{k+1}^*] \\ 0 & \text{otherwise.} \end{cases}$$

The first and second *divided* differences of b are

$$b_i' = d\frac{b_i - b_{i-1}}{t_{i+d} - t_i} \qquad b_i'' = (d-1)\frac{b_i' - b_{i-1}'}{t_{i+d-1} - t_i}.$$

The b_i' are the control points of the first derivative b' of b in the B-spline basis N_i^{d-1}.

3.2. Envelopes for splines

We construct piecewise linear envelopes for splines over a given knot sequence (t_i) by comparing a spline b to its control polygon Lb, yielding envelopes that are offsets of the control polygon Lb. In the notation developed in Chapter 2, we have $\mathcal{B} = \text{span}\{N_k^d\}$ and $\mathcal{H} = \text{span}\{H_k\}$. By the linear precision of the B-spline representation, the space $\mathcal{B} \cap \mathcal{H}$ consists of all linear functions and the operator L defined in the previous section leaves $\mathcal{H} \cap \mathcal{B}$ pointwise fixed.

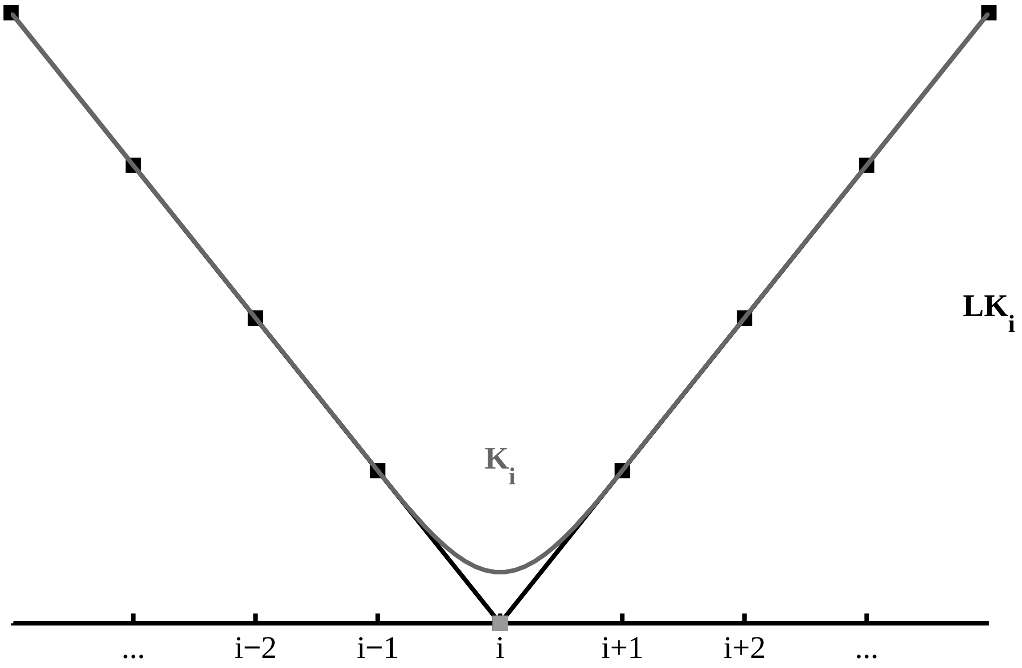


FIGURE 3.3. A function K_i for uniform cubic splines over the knot sequence $t_i = i$. Outside of the interval $[i-1, i+1]$, K_i coincides with its control polygon LK_i.

The difference operators D need to annihilate all linear functions. We define D as the *centered* second differences of b,

$$D_i b = b'_{i+1} - b'_i.$$

Only the control points b_{i-1}, b_i, and b_{i+1} influence D_i. The functions K that fulfill $DK = I$ have the B-spline representation $K_i = \sum_{j\in\mathbb{Z}} K_{i,j} N^d_j$ with the B-spline coefficients

$$K_{i,j} = \frac{1}{2}\left|t^*_j - t^*_i\right|. \tag{3.2.7}$$

The functions K_i are nonnegative since their control polygons are nonnegative and convex since their control polygons are convex.

With these definitions, Theorem 1 on page 6 applies and the difference between a spline b and its control polygon Lb can be written as

$$(E - L)b = (E - L)KDb \qquad \text{for all } b \in \operatorname{span}\{N^d_i\}$$

Piecewise linear estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ on $(E - L)K$ can be easily obtained from the nonnegativity and convexity of the functions K: from below, $(E - L)K$ is bounded by zero, and from above by the piecewise linear interpolant of the values $(E - L)K(t^*_k)$, i.e.,

$$\lfloor K_i \rfloor = 0 \qquad \lceil K_i \rceil = \sum_{k\in\mathbb{Z}} \nu_{i,k} H_k$$

where $\nu_{i,k} = (E - L)K_i(t_k^*)$ are constants that only depend on the knot sequence (t_i) and the degree d of the B-spline basis. With these definitions and by Theorem 2 on page 8, we have the following theorem.

THEOREM 5. *Let* b *be a spline of degree* d *over the knot sequence* (t_i), D *the centered second differences, and* $\nu_{i,k} = (E - L)K_i(t_k^*)$. *The distance between* b *and its control polygon* Lb *is bounded by*

$$\llbracket Db \rrbracket \nu H \le (E - L)b \le \ulcorner\!\!\ulcorner Db \urcorner\!\!\urcorner \nu H \tag{3.2.8}$$

or, more detailed,

$$\sum_{i\in\mathbb{Z}} \llbracket D_i b \rrbracket \sum_{k\in\mathbb{Z}} \nu_{i,k} H_k \le (E - L)b \le \sum_{i\in\mathbb{Z}} \ulcorner\!\!\ulcorner D_i b \urcorner\!\!\urcorner \sum_{k\in\mathbb{Z}} \nu_{i,k} H_k. \tag{3.2.9}$$

3.2.1. Implementation. Figure 3.3 suggests that, for any fixed i, K_i and LK_i are identical outside a small intervall around t_i^*. This is true because of the linear precision of the B-spline representation and because of the local support of B-splines. For a practical implementation, it is imperative to determine exactly the intervals over which the $(E - L)K_i$ are nonzero and which constants $\nu_{i,k}$ are therefore nonzero.

In evaluating the sums in equation (3.2.8) it suffices to take the inner sums over all $k \in \mathcal{I}(i)$ with

$$\mathcal{I}(i) = \{\, k : (E - L)K_i(t_k^*) > 0 \,\}. \tag{3.2.10}$$

Since, for a fixed t_k^*, $K_i(t_k^*)$ and $LK_i(t_k^*)$ coincide whenever the control points $K_{i,j}$ that influence K_i at t_k^* lie on a straight line, $(E - L)K_i(t_k^*)$ can only be nonzero if t_k^* is in the intersections of the support of N_{i-1}^d and N_{i+1}^d. Since a B-spline basis function N_j^d is nonzero only on the interval (t_j, t_{j+d+1}), the constants $\nu_{i,k}$ can therefore be only nonzero for $t_k^* \in (t_{i+1}, t_{i+d})$. The set $\mathcal{I}(i)$ is therefore

$$\mathcal{I}(i) = \{\, k : t_{i+1} < t_k^* < t_{i+d} \,\}.$$

Conversely, the sets $\mathcal{I}^*(k)$,

$$\mathcal{I}^*(k) = \{\, i : k \in \mathcal{I}(i) \,\} = \{\, i : t_{i+1} < t_k^* < t_{i+d} \,\},$$

contains all indices i for which $D_i b$ influences the bound from Theorem 5 at t_k^*. $\mathcal{I}^*(k)$ contains at most $d - 1$ elements. We can therefore compute the bound from Theorem 5 as

$$\sum_{i\in\mathbb{Z}} \llbracket D_i b \rrbracket \sum_{k\in\mathcal{I}(i)} \nu_{i,k} H_k \le (E - L)b \le \sum_{i\in\mathbb{Z}} \ulcorner\!\!\ulcorner D_i b \urcorner\!\!\urcorner \sum_{k\in\mathcal{I}(i)} \nu_{i,k} H_k$$

or as

$$\sum_{k\in\mathbb{Z}} H_k \sum_{i\in\mathcal{I}^*(k)} \llbracket D_i b \rrbracket \nu_{i,k} \le (E - L)b \le \sum_{k\in\mathbb{Z}} H_k \sum_{i\in\mathcal{I}^*(k)} \ulcorner\!\!\ulcorner D_i b \urcorner\!\!\urcorner \nu_{i,k}. \tag{3.2.11}$$

For an efficient implementation, we use the second formulation and precompute the $\mathcal{I}^*(\cdot)$ and the $\nu_{i,k}$ for $i \in \mathcal{I}^*(k)$. Computing the lower and upper envelope from Theorem 5 for any b over the fixed knot sequence (t_i) then only requires computing one second difference and one scalar product with $d - 1$ terms for each Greville abscissa t_k^*.

EXAMPLE. For Bernstein polynomials of degree d, the knots are $t_i = [i > d]$, $i = 0, \dots, 2d-1$. The Greville abscissae are $t_k^* = k/d$, so that

$$\mathcal{I}(i) = \begin{cases} \{1, \dots, d-1\} & 1 \leq i \leq d-1 \\ \emptyset & \text{otherwise.} \end{cases}$$

△

3.2.2. Sharpness and convergence.

The envelope from Theorem 5 is sharp in the sense that the upper (lower) part of the envelope *equals* b at t_k^* if $D_i b \geq 0$ ($D_i b \leq 0$) for all $i \in \mathcal{I}^*(k)$ since the $\lceil K_i \rceil$ are piecewise linear interpolants of $(E-L)K_i$. This is the case if the whole polynomial piece of b on which $b(t_k^*)$ lies is convex (concave). In particular, the envelope is sharp for splines of degree 2 since $\mathcal{I}^*(k) = \{k\}$ contains only one element.

Sharpness implies that, under subdivison, the width of the envelope shrinks quadratically in the distance of the knots: since the envelope consists of linear pieces, the maximum width is attained at the break points. Since a spline has only finitely many inflection points almost all pieces generated by subdivision are either concave or convex for which the width of the envelope equals the distance of the spline to the control polygon which is known to decrease quadratically [**2, 3, 25**]. This ensures that the envelopes are suitable for use in root finding algorithms that refine root enclosures through adaptive refinement of the control polygon.

3.3. Hölder envelopes

The envelope in Theorem 5 requires the evaluation of $d-1$ values $\nu_{i,k}$ at each Greville abscissa t_k^* in the precomputation phase. We can reduce the computational cost to just one evaluation per Greville abscissa by subsuming the values of the $\nu_{:,k}$ into one spline z and estimating its values. The resulting envelopes, though very simple to compute, are in general considerably larger than the ones from Theorem 5.

LEMMA 6. *Let* $t \in [t_l, t_{l+1})$. *Then*

$$\sum_i (E-L)K_i(t) = (E-L)z^m(t)$$

with

$$z^m(t) = \sum_j \left(j t_j^* + \sum_{i=j}^{m} t_i^* \right) N_j^d \qquad \text{for any } m \geq l-1. \tag{3.3.12}$$

z^m *is strictly convex and* $(E-L)z^m \geq 0$.

PROOF. For brevity, we suppress the argument t. K_i can only be nonzero at t if $l-d < i < l$. Therefore, by (3.2.7),

$$\sum_i (E-L)K_i = (E-L) \sum_{i=l-d+1}^{l-1} K_i = \frac{1}{2}(E-L) \sum_{i=l-d+1}^{l-1} \sum_{j=l-d}^{l} \left| t_i^* - t_j^* \right| N_j^d,$$

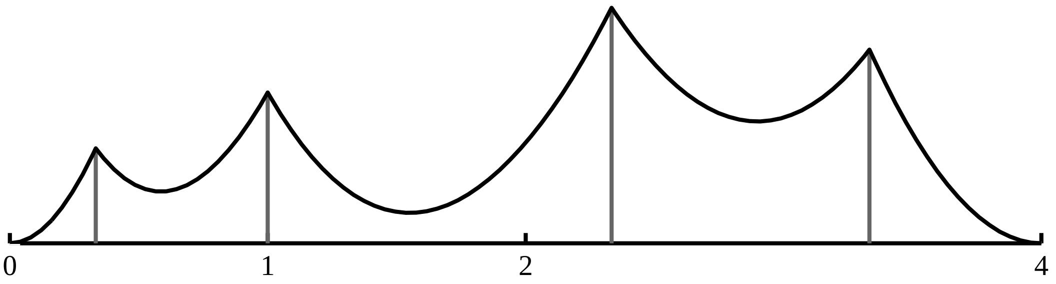


FIGURE 3.4. The function z defined in Equation 3.3.12 on the preceding page for cubic splines over the knot sequence $(0,0,0,0,1,2,4,4,4,4)$. The grey lines indicate the position of the Greville abscissae. Knots are shown as ticks on the x–axis. The values of z range from 0 to 0.26.

since $(E-L)b = 0$ for any linear function b, this equals

$$= \frac{1}{2}(E-L)\sum_{j=l-d}^{l}\left[\sum_{i=l-d+1}^{l-1}\left|t_i^* - t_j^*\right| + \sum_{i=l-d+1}^{l-1}(t_i^* - t_j^*)\right]N_j^d$$

$$= \frac{1}{2}(E-L)\sum_{j=l-d}^{l}\left[\sum_{i=j}^{l-1}2(t_i^* - t_j^*)\right]N_j^d$$

$$= (E-L)\sum_{j=l-d}^{l}\left[(j-l)t_j^* + \sum_{i=j}^{l-1}t_i^*\right]N_j^d,$$

which equals, again because $E-L$ annihilates linear functions,

$$= (E-L)\sum_{j=l-d}^{l}\left[jt_j^* + \sum_{i=j}^{l-1}t_i^*\right]N_j^d.$$

By the locality of the B-spline-basis N_j^d, we can extend the range of summation in the outer sum to all j. The upper limit $l-1$ in the inner summation can be raised to an arbitrary integer $m \geq l-1$ since $E-L$ annihilates linear functions.

It is easy to check that the B-spline coefficients z_j'' of the second derivative of z are positive. This implies, by the convex hull property of the B-spline representation, that $(E-L)z^m(t) \geq 0$. □

THEOREM 7. *The difference between the spline* b *and its B–spline control polygon* Lb *over the interval* $[t_k^*, t_{k+1}^*]$ *is bounded by*

$$|(E-L)b| \leq \mu\,(E-L)z^{k+d}. \tag{3.3.13}$$

where $\mu(t) = \max\{\,|D_i b| : l-d < i < l$ *and* $t \in [t_l, t_{l+1})\,\}$.

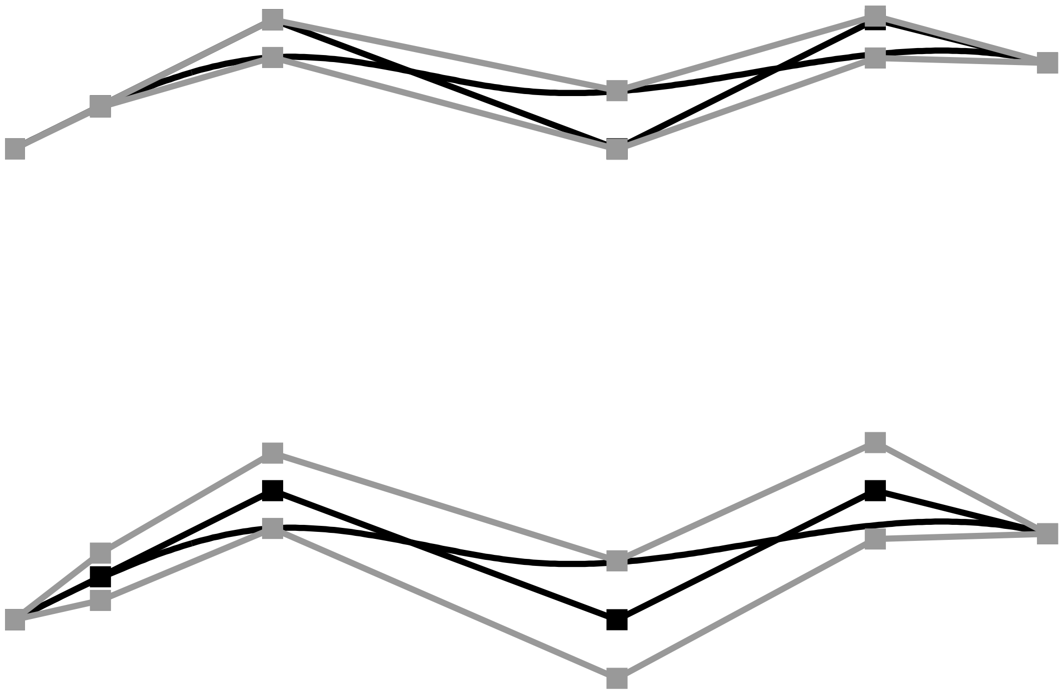


FIGURE 3.5. A cubic spline and its envelope according to Theorem 5 *(top)* and Corollary 8 *(bottom)* over the knot sequence $(0,0,0,0,1,2,4,4,4,4)$ with control points $(0,1,3,0,3,2)$.

PROOF. Using Theorem 3 on page 10 with $p = 1$ and $q = \infty$, we have

$$|(E-L)b| \leq \|(E-L)K\|_1 \, \|Db\|_\infty .$$

According to Lemma 6, $\|(E-L)K(t)\|_1 = z^{k+d}(t)$ for any $t \in [t_k^*, t_{k+1}^*]$. The computation of $\|Db\|_\infty$ need only extend over the elements $D_i b$ of Db for which $(E-L)K_i \neq 0$. □

COROLLARY 8. *The difference between* b *and its control polygon* Lb *is bounded by*

$$|(E-L)b| \leq \sum_k \mu(t_k^*)\, \zeta_k H_k$$

where $\zeta_k = (E-L)z^{k+d}(t_k^*)$.

PROOF. This is an easy consequence of Theorem 7 and the fact that z^m is convex. □

3.3.1. Bounding with a quadratic spline. The same considerations as in Theorem 7 but with divided instead of centered second differences yield a quadratic bounding spline.

COROLLARY 9. *The difference between the spline* b *and its control polygon* Lb *over the interval* $[t_k^*, t_{k+1}^*]$ *is bounded by*

$$|(E-L)b| \leq \overline{\mu}(t)\bar{z}^{k+d}$$

where $\overline{\mu}(t) = \max\{\,|b_i''| : l-d < i < l$ *and* $t \in [t_l, t_{l+1})\,\}$. *The function* $\bar{z} = \sum \bar{z}_j N_j^d$ *is a quadratic spline with* $\bar{z}'' = 1$.

Proof. Using divided second differences $\overline{D}$,

$$\overline{D}_i b = b_i'' = \frac{d-1}{t_{i+d-1} - t_i} D_{i-1},$$

a solution $\overline{K}$ to $\overline{D}\,\overline{K} = I$ is given by the functions $\overline{K}_i$ with

$$\overline{K}_i = \frac{t_{i+d-1} - t_i}{d-1} K_{i-1} = \sum_j \frac{1}{2} \frac{t_{i+d-1} - t_i}{d-1} \left|t_{i-1}^* - t_j^*\right| N_j^d.$$

In analogy to the proof of Theorem 7 we find that the function $\bar{z}^m = \sum_i (E - L)\overline{K}_i$ is given by

$$\bar{z}^m = \sum_j \sum_{i=j}^{m} \frac{t_{i+d} - t_{i+1}}{d-1} \left|t_i^* - t_j^*\right| N_j^d.$$

It is now easy to verify that the B-spline coefficients of the second derivative of $\bar{z}^m$ are equal to 1 so that $\bar{z}'' = 1$. □

3.3.2. Implementation. In practice, the knot sequence (t_i) is finite, $(t_i) = (t_0, \ldots, t_{m+d})$. The envelope from Corollary 8 requires $\mu(t_k^*)$ and $z(t_k^*)$ for the Greville abscissae $t_0^*, \ldots, t_m^*$.

For finite knot sequences, we only need to compute the spline z^m for envelopes over all the $[t_k^*, t_{k+1}^*]$. The B-spline coefficients z_j^m can be computed from the iteration

$$\begin{aligned} z_j^m &= (m+1)\, t_m^* \\ z_j^m &= z_{j+1}^m - \frac{j+1}{d}(t_{j+d+1} - t_{j+1}) \quad \text{for } j = m-1 \ldots 0. \end{aligned}$$

While $\mu(t_k^*)$ depends on the control points b_i, the underlying index sets do not and need only be updated when the knot sequence is changed.

An efficient implementation of the bound of Corollary 8 stores the values t_k^*, $z(t_k^*)$, and the index sets used to compute $\mu(t_k^*)$ together with the knot sequence, so that only second differences and their maximum $\mu(t_k^*)$ need to be recomputed whenever the control points of a spline change.

3.3.3. Sharpness. The Hölder inequality, which is the only inequality used to proof Theorem 7 on page 18, becomes an equality when all the second differences D_i used to compute $\mu(t)$ are equal. The splines with this property are the linear polynomials and z. Similarly, the bound from Corollary 9 is sharp for the linear polynomials and $\bar{z}$, and therefore for all quadratic polynomials.

3.4. A bound for Bernstein polynomials

The B–splines N_j^d of degree d over the knot sequence (t_i) with $t_i = [i > d]$, $i = 0, \ldots, 2d-1$ are called the Bernstein polynomials B_j^d of degree d. An explicit bound for the Bernstein polynomials was already given in [**19**, Theorem 3.1]; we give an alternate proof for this theorem, showing how it can be derived from the more general exposition in this chapter.

For Bernstein polynomials, the Greville abscissae are $t_i^* = i/d$ and the coefficients z_j^m, $m = 2d-1$ are

$$z_j^m = \frac{j(j+1)}{2d} + 2d - 1 = \frac{1}{d}\binom{j}{2} + \frac{j}{d} + 2d - 1.$$

Thus,

$$z^m(t) = \frac{d-1}{2}t^2 + t + 2d - 1$$

and

$$(E-L)z^m(t_k^*) = \frac{k(d-k)}{2d^2}.$$

The maximal value $(E-L)z^m(t_k^*)$ over all k is taken on for $k = \lfloor d/2 \rfloor$ and the main theorem from [**19**] follows as a special case of Theorem 7:

THEOREM 10 ([**19**, Theorem 3.1]). *Let* $b = \sum_{i=0}^{d} b_i B_i^d$ *be a polynomial of degree* d *in Bernstein–Bézier form. The distance between* b *and its control polygon* Lb *is uniformly bounded by*

$$|(E-L)b| \leq \frac{\lfloor d/2 \rfloor \lceil d/2 \rceil}{2d} \max_{i=1}^{d-1} |b_{i-1} - 2b_i + b_{i+1}|.$$

This bound decreases by a factor of 4 under subdivision at $1/2$ [**19**, Lemma 6.1]. Figures 3.6 on the next page and 3.7 on page 23 compare the convex hull, the min–max envelope and various Hölder bounds and show the effect of subdivision and refinement on the envelopes.

3.5. A bound for uniform splines

A spline is *uniform* if the knots are equidistant. Without loss of generality, we choose $t_k = k$. The uniform B–spline basis functions N_j^d are shifts of one another and we define $N^d = N_0^d$ so that $N_j^d(t) = N^d(t-j)$. The corresponding Greville abscissae are

$$t_k^* = \frac{1}{d}\sum_{i=k+1}^{k+d} i = k + \frac{d+1}{2}$$

and z^m has the B–spline coefficients

$$z_j^m = j(j + \frac{d+1}{2}) + \sum_{i=j}^{m}\left(i + \frac{d+1}{2}\right) = (m+1)(j + \frac{d+1}{2}) + \binom{j-m}{2}.$$

Equation (3.3.12) simplifies for uniform splines to

$$z^m = \sum_j \left[(m+1)(j + \frac{d+1}{2}) + \binom{j-m}{2}\right] N_j^d,$$

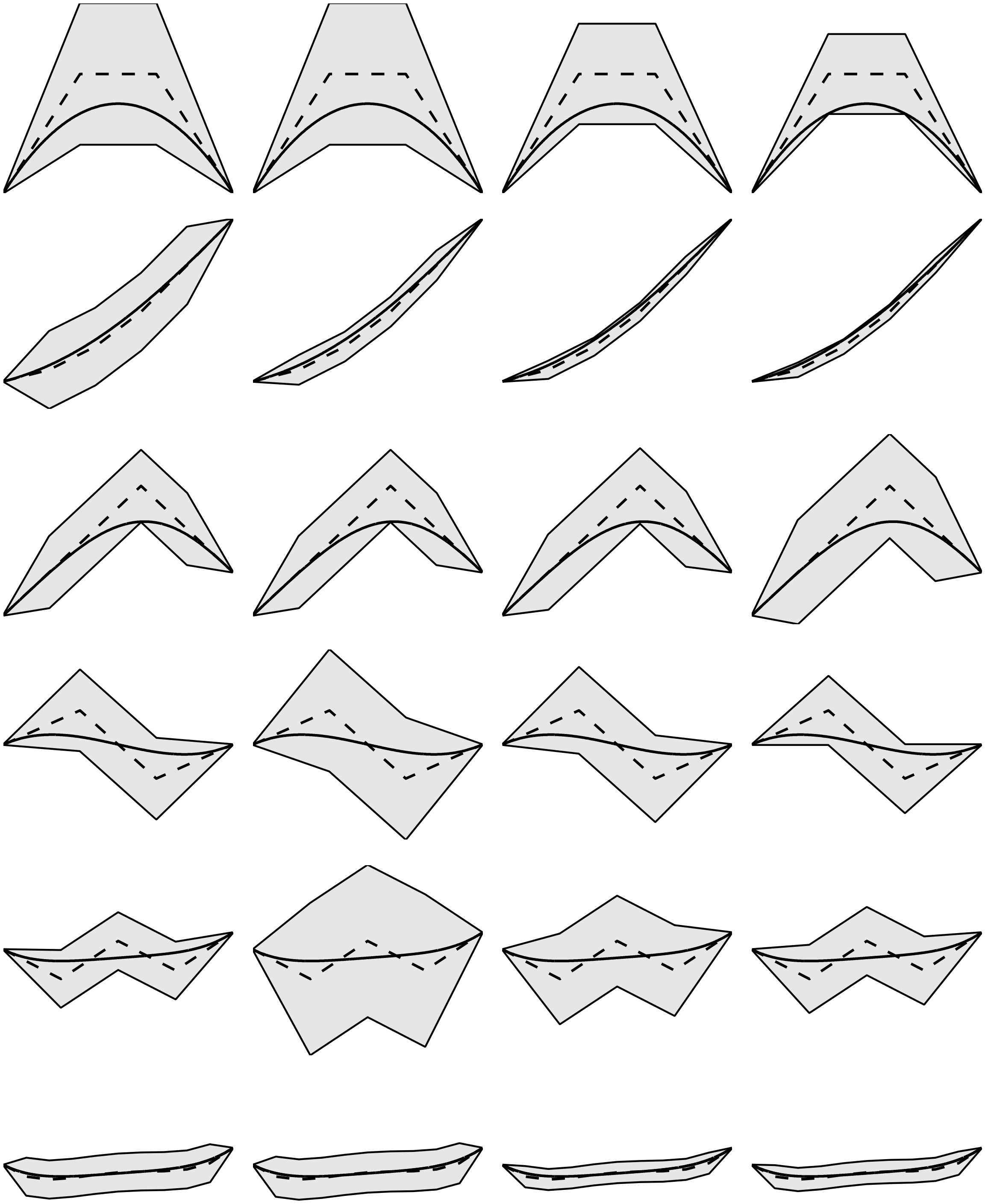


FIGURE 3.6. Bernstein polynomials and envelopes: the control polygon is dashed, bounding regions are shaded. Control polygons from top to bottom: (1) [0 1 1 0], (2) [0 1 3 6 10 14], (3) [0 1 2 3 2 1], (4) [0 1 − 1 0], (5) [0 − 7 2 − 5 4], (6) the same polynomial raised to degree 10. Bounds from left to right: all bounds compare to the control polygon, with D second differences, except for (a), which uses first differences. Types of bounds: (a) (H_1), (b) (H_∞), (c) (H_2), (d) (H_1).

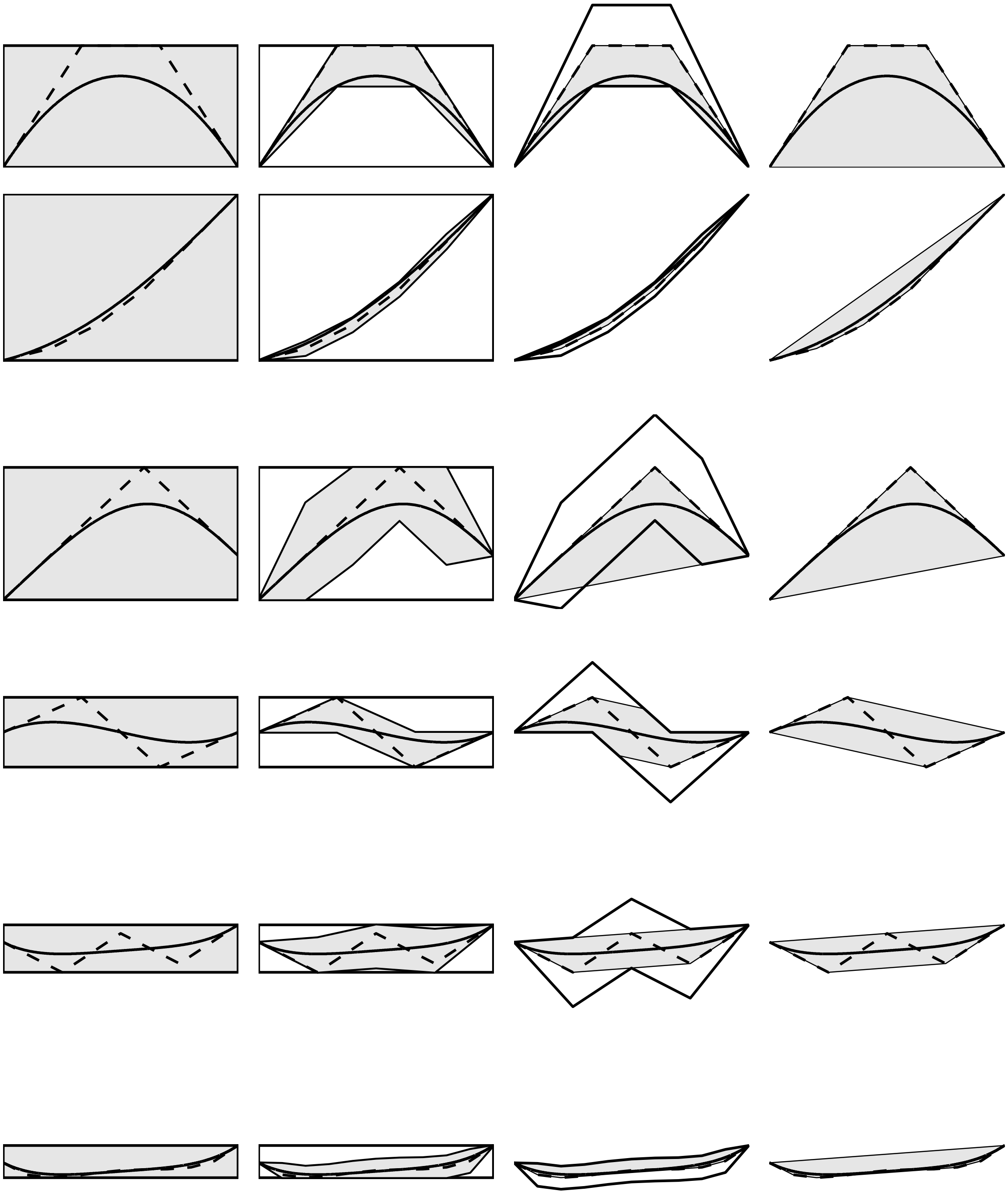


FIGURE 3.7. Bernstein polynomials and envelopes: the control polygon is dashed, bounding regions are shaded. Comparison of min–max envelope and convex hull to the (H_1) bound for D second differences of the control points. From top to bottom: functions as in Figure 3.6. From left to right: (a) Min–max bound on the Bernstein basis functions, (b) the (H_1) bound clipped against (a), (c) the (H_1) bound clipped against (d), (d) the convex hull of the control points.

so that z^m is a quadratic spline, regardless of the degree of b. Therefore, $(E-L)z^{k+d}$ is a quadratic polynomial over the interval $[t_k^*, t_{k+1}^*]$ with

$$z^{k+d}(t) = \frac{1}{2}\left(t^2 - (t_k^* + t_{k+1}^*)t + \left(\frac{t_k^* + t_{k+1}^*}{2}\right)^2 + \frac{d-2}{12}\right).$$

Since z is a positive and convex function, it attains its maximum over $[t_k^*, t_{k+1}^*]$ at one of the endpoints of the interval. Its values there are

$$z(t_k^*) = z(t_{k+1}^*) = \frac{d+1}{24}.$$

This proves the following simplified version of Corollary 8, which yields a particularly simple Hölder envelope for uniform spline:

Corollary 11. *Let* b *be a uniform spline over the knot sequence* $t_k = k$. *The difference between* b *and its control polygon* Lb *is bounded by*

$$|(E-L)b| \le \frac{d+1}{2} \sum_k \mu(t_k^*)\, H_k.$$

For uniform splines, the computation of $\mu(t_k^*)$ simplifies to

$$\mu(t_k^*) = \max\{\, D_i b : k - \lfloor d/2 \rfloor < i < k + \lfloor d/2 \rfloor \,\}.$$

Since $z(t_k^*) = (d+1)/24$ and $\mathcal{I}^*(k) = \{k\}$ for $d=2$ or $d=3$, the bound from Theorem 5 in the form of Equation 3.2.11 reduces for quadratic and cubic uniform splines to

$$\frac{d+1}{24}\sum_{k\in\mathbb{Z}} \llbracket D_k b \rrbracket\, H_k \le (E-L)b \le \frac{d+1}{24}\sum_{k\in\mathbb{Z}} \lceil\!\lceil D_k b \rceil\!\rceil\, H_k. \tag{3.5.14}$$

This implies, in particular, that the above envelope for quadratic and cubic uniform splines is sharp so that the breaks of the envelope are the control points and the values of the spline at the Greville abscissae.

3.6. Uniform refinement

An important operation on splines is the refinement of the knot sequence, or knot insertion. Knot insertion changes the representation of the piecewise polynomial p over the original knot sequence to one over an enlarged knot sequence.

For uniform splines, we consider the refinement of the knot sequence $t_k = k$ to the sequence $\hat{t}_k = k/2$. We now have two representations for b,

$$b(t) = \sum_k b_k N^d(t-k) = \sum_k \hat{b}_k N^d(2t-k),$$

where the new control points $\hat{b}_k$ are linear combinations of the old control points b_k, given by

$$\hat{b}_{2i} = 2^{-d} \sum_{j=0}^{\lceil d/2 \rceil} \binom{d+1}{2j} b_{i-j}, \tag{3.6.15}$$

$$\hat{b}_{2i+1} = 2^{-d} \sum_{j=0}^{\lceil d/2 \rceil} \binom{d+1}{2j+1} b_{i-j}. \tag{3.6.16}$$

Since $b_i'' = D_{i-1}b$, we can use the B–spline representation of b'' together with equation (3.6.15) to relate the centered second differences of the new control polygon to those of the old control polygon as

$$2^d D_{2i}\hat{b} = \sum_j \binom{d-1}{2j-1} D_{i-j}b \qquad 2^d D_{2i+1}\hat{b} = \sum_j \binom{d-1}{2j} D_{i-j}b.$$

The equality $\sum_j \binom{d-1}{j} = 2^{d-1}$ and the symmetry of the binomial coefficients imply

$$\sum_j \binom{d-1}{2j} = \sum_j \binom{d-1}{2j-1} = 2^{d-2},$$

so that

$$\max_i |D_i\hat{b}| \leq \frac{1}{4} \max_i |D_i b|.$$

This ensures that the bound from Corollary 11 converges quadratically to zero under repeated uniform refinement. Figure 3.8 illustrates this for a cubic uniform spline.

EXAMPLE. For quadratic splines, uniform refinement is called Chaikin's algorithm and

$$\hat{b}_{2i} = 2^{-2}(3b_{i-1} + b_i), \qquad \hat{b}_{2i+1} = 2^{-2}(b_{i-1} + 3b_i).$$

This yields

$$D_{2i}\hat{b} = D_{2i-1}\hat{b} = \frac{1}{4} D_{i-1}b.$$

Since every second difference decreases by a factor of four, subsequent envelopes are contained in one another.

Similarly, for cubic splines we have

$$\hat{b}_{2i} = 2^{-3}(b_{i-2} + 6b_{i-1} + b_i), \qquad \hat{b}_{2i+1} = 2^{-3}(4b_{i-1} + 4b_i).$$

and therefore

$$D_{2i}\hat{b} = \frac{1}{4} D_{i-1}b, \qquad D_{2i+1}\hat{b} = \frac{1}{8}(D_{i-1}b + D_i b).$$

△

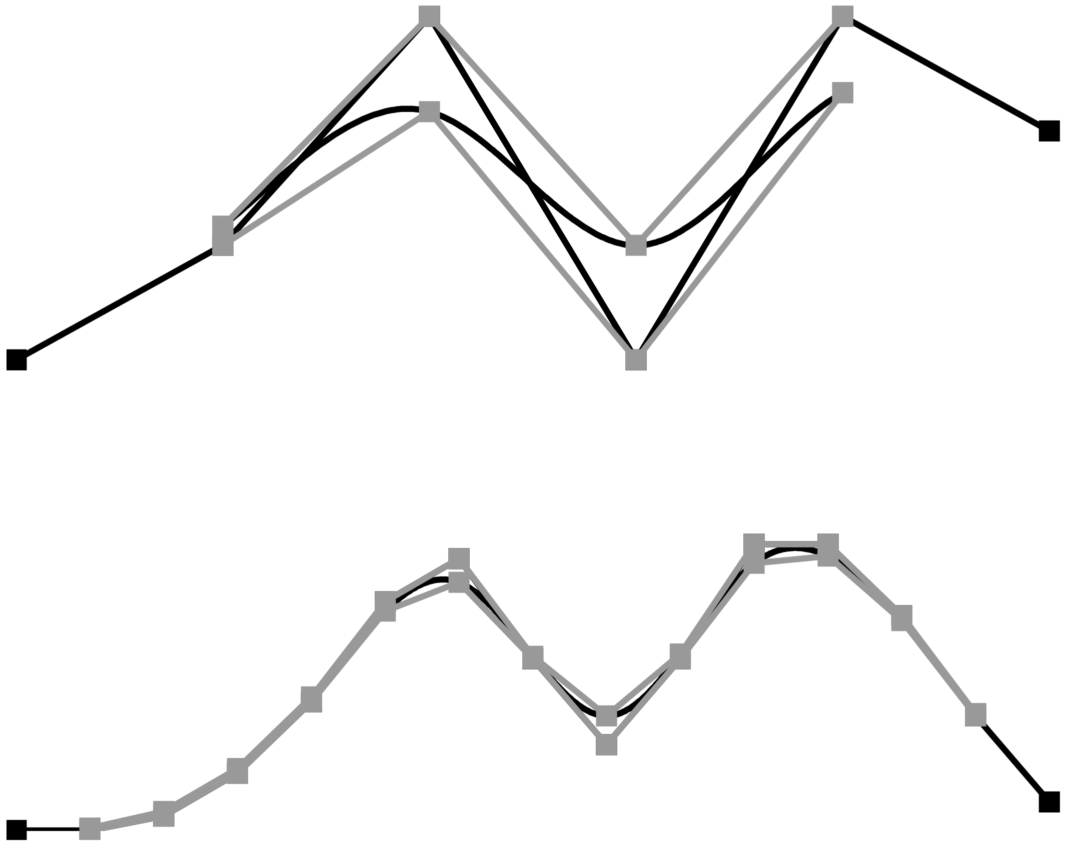


FIGURE 3.8. A uniform cubic spline with control points $(0, 1, 3, 0, 3, 2)$. Shown are the envelope according to equation (3.5.14) *(top)* and the envelope after one step of uniform refinement *(bottom)*.

3.7. Cubic interpolatory subdivision

In cubic interpolatory subdivision [**9**], functions b are defined as the limit of a subdivision process. One subdivision step refines a control polygon b with breaks at all $i \in \mathbb{Z}$, to one with twice as many control points $\widehat{b}$ with breaks at $i \in \mathbb{Z}/2$. It is interpolatory, since old control points are copied through the rule $\widehat{b}_{2i} = b_i$ and cubic since the remaining new control points $\widehat{b}_{2i+1}$ are obtained by evaluating the cubic interpolant q_i of $b_{i-1}, b_i, b_{i+1}, b_{i+2}$ with ordinates $-3, -1, 1, 3$ at 0, $\widehat{b}_{2i+1} = q_i(0)$. The limit function obtained by this process is $\mathcal{C}^1$.

This scheme differs significantly from the polynomial spline bases since the basis functions C_j for this scheme have no known closed–form representation. Figure 3.9 shows the basis function C_0 obtained from subdividing the control polygon b with $b_i = [i = 0]$. C_0 is nonzero only on the interval $[-3, 3]$. Because of the uniformity of the subdivision scheme, all basis functions C_i are shifts of each other, $C_i(t) = C_0(t - i)$. The space $\mathcal{B}$ is the space of all functions obtainable by cubic interpolatory subdivision; it is spanned by the functions C_i which also form a basis of $\mathcal{B}$.

The scheme possesses *cubic precision* by construction, i.e., if the input polygon lies on a cubic polynomial p, the limit function is that cubic polynomial p.

Cubic interpolatory subdivision does not possess the convex hull property since the basis functions take on negative values (cf. Figure 3.9). Therefore, there is no convex hull bound. The only published bounds that lend themselves easily to constructing envelopes for this scheme are the piecewise constant bounds in [**16**].

3.7.1. Linear envelopes and the subdivide–and–bound procedure. For a function $b \in \mathcal{B}$, its control polygon Lb is the piecewise linear interpolant at the integers,

$$Lb = \sum_j b_j H_j \qquad \text{for } b = \sum_j b_j C_j.$$

The H_j are the piecewise linear functions with breaks at the integers for which $H_j(i) = [i = j]$ for $i \in \mathbb{Z}$.

A bound on the difference between b and its control polygon Lb is obtained by choosing D as the second difference operator. The i–th component of Db is $D_i b = b_{i-1} - 2b_i + b_{i+1}$. The control polygon of the antidifference K_i corresponding to this difference is

$$K_{i,j} = |j - i|/2, \qquad K_i = \sum_j K_{i,j} C_j.$$

Clearly, $DK = I$ where I is the identity on $\mathbb{R}^\infty$. Because cubic interpolatory subdivision reproduces linear functions and because of the size of the support of C_0, $(E - L)K_0(t) = 0$ for all $t \notin [-2, 2]$.

We compute piecewise linear bounds $\lfloor K \rfloor$ and $\lceil K \rceil$ for K using subdivision to iteratively refine a rough estimate to tighter bounds. Because of the uniformity of cubic interpolatory subdivision, it suffices to compute bounds on K_0 on $[-2, 2]$. Bounds on K_i are obtained as integer shifts of the bounds on K_0.

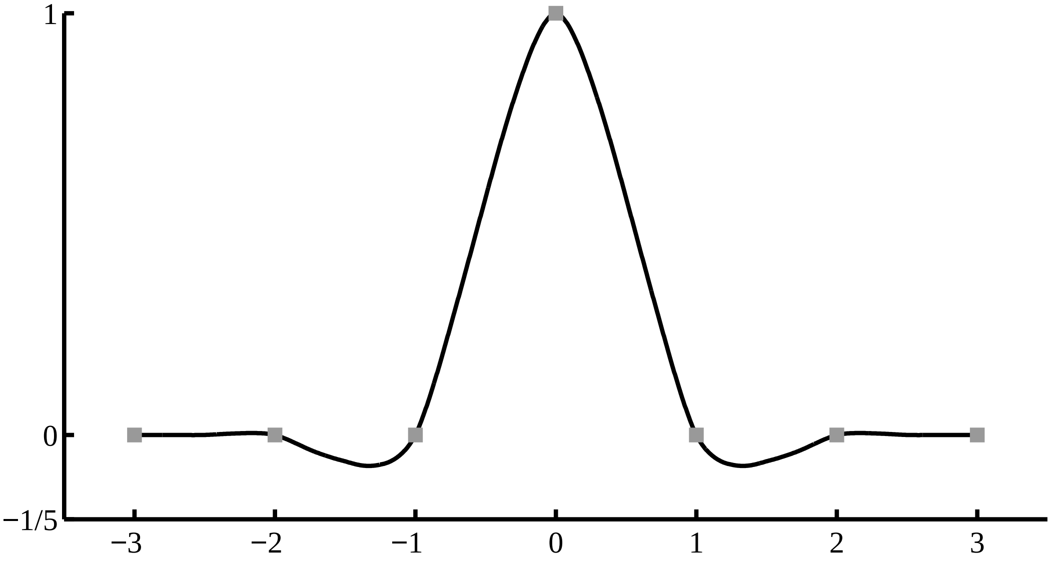


FIGURE 3.9. The basis function C_0 of cubic interpolatory subdivision.

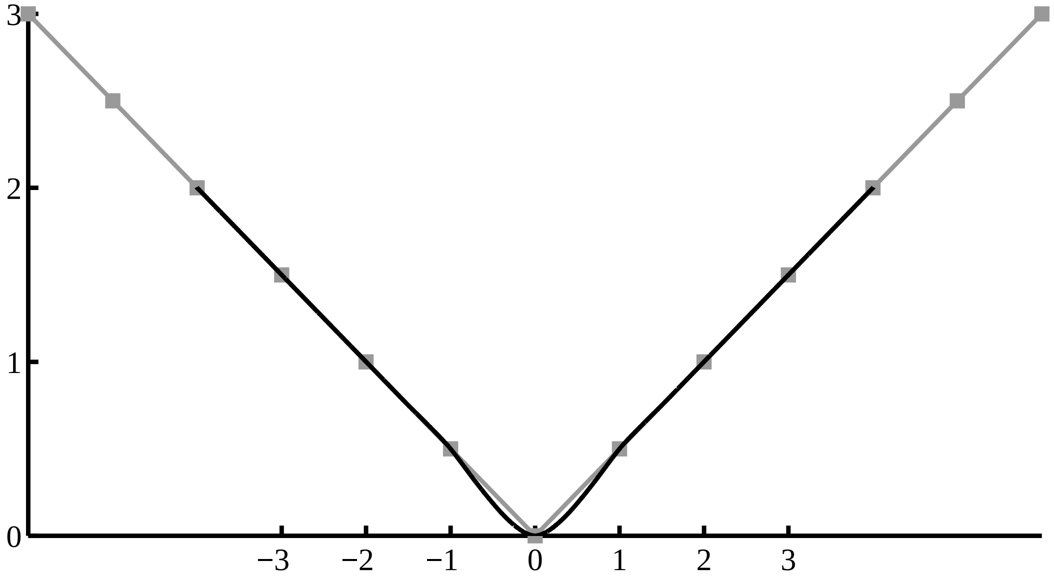


FIGURE 3.10. The antidifference K_0 with respect to the centered second differences D_i centered at 0.

Let $r > 0$ be a fixed integer and $\underline{e}_0$ and $\overline{e}_0$ be a piecewise linear envelope of $(E-L)K_0$ over $[-2, 2]$ with breaks at $\{-2, -1, 0, 1, 2\}$,

$$\underline{e}_0 = \sum_{j=-2}^{2} \underline{e}_0(j)\, H_j, \qquad \overline{e}_0 = \sum_{j=-2}^{2} \overline{e}_0(j)\, H_j.$$

We use this initial envelope to construct a piecewise linear envelope of the r times subdivided control polygon c_r of K_0 on $[-2, 2]$. The envelope consists of two piecewise linear functions, $\underline{e}_r$ and $\overline{e}_r$, with $\underline{e}_r \le (E-L)K_0 \le \overline{e}_r$. The polylines $\underline{e}_r$ and $\overline{e}_r$ have their breaks exactly at the breaks $u_i = i/2^r$, $i = -2^{r+1}, \dots, 2^{r+1}$ of c_r. We construct an improved upper envelope $\overline{e}_0'$ from $\overline{e}_r$ by solving a linear program with constraints

$$\overline{e}_r(u_i) \le \overline{e}_0'(u_i) \quad \text{for each } u_i$$

minimizing the error $\max_i\{\overline{e}_0'(u_i) - \overline{e}_r(u_i)\}$. The variables of the linear program are the five values $\overline{e}_0'(j)$, $j = -2, \dots, 2$. Similarly, an improved lower envelope $\underline{e}_0'$ is computed from $\underline{e}_r$. This process is iterated with $\overline{e}_0'$ and $\underline{e}_0'$ as initial envelopes for K_0 until initial and improved envelopes in an iteration step are within a prespecified tolerance. The envelopes of K_0 can be further improved by repeating the whole iteration for a larger subdivision level r.

TABLE 3.1. Piecewise linear upper and lower envelopes of $(E-L)K_0$ over $[-2, 2]$. The values are directionally truncated to 4 digits precision.

i	-2	-1	0	1	2
$\underline{e}(i)$	0.0	−0.0055	−0.1277	−0.0055	0.0
$\overline{e}(i)$	0.0	0.0056	0.0001	0.0056	0.0

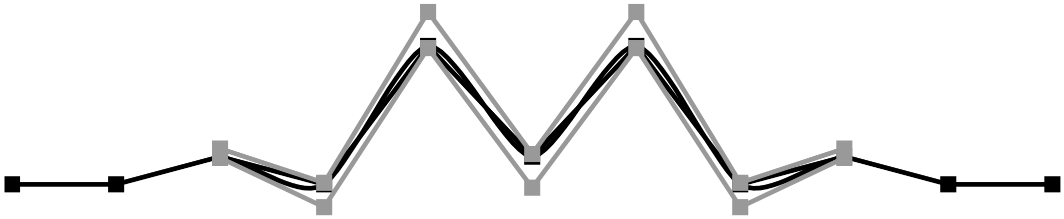


FIGURE 3.11. A cubic interpolatory subdivision function *(black)* with its control polygon *(black squares)* and the envelope according to Theorem 2 on page 8.

Applying the above iteration for $r = 9$ yields the values given in Table 3.1. The envelope for a function b with control polygon b_i is the piecewise linear interpolant of

$$b_i + \sum_{j=-1}^{1} \underline{e}(i)\lceil\!\lceil D_{i+j}b \rceil\!\rceil + \overline{e}(i)\lfloor\!\lfloor D_{i+j}b \rfloor\!\rfloor$$

for the lower envelope and

$$b_i + \sum_{j=-1}^{1} \overline{e}(i)\lceil\!\lceil D_{i+j}b \rceil\!\rceil + \underline{e}(i)\lfloor\!\lfloor D_{i+j}b \rfloor\!\rfloor$$

for the upper envelope.

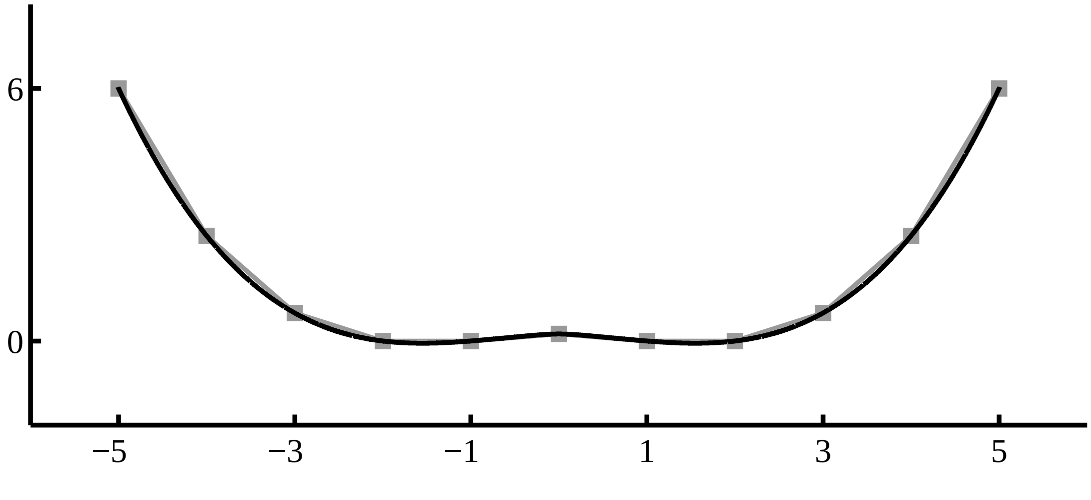


FIGURE 3.12. The antidifference K_0 for the fourth differences discussed in Section 3.7.2.

3.7.2. Cubic envelopes. We can also envelope functions in $\mathcal{B}$ by cubic splines. This lets us represent one cubic interpolatory subdivision function by two cubic splines and demonstrates that the approach from Chapter 2 is not restricted to piecewise linear envelopes.

Let $\mathcal{H} = \operatorname{span}\{N_j^3\}$ be a cubic spline space with N_j^3 the $\mathcal{C}^1$ cubic B-splines over the knot sequence $(t_i) = (\dots, -1, -1, 0, 0, 1, 1, \dots)$. The functions that can be represented both by cubic interpolatory subdivision and as cubic splines in $\mathcal{H}$ are exactly the cubic polynomials.

We therefore need to choose a difference operator D that annihilates cubic polynomials. We use fourth centered differences for D, so that the i–th component of Db is given by

$$D_i b = b_{i-2} - 3b_{i-1} + 6b_i - 3b_{i+1} + b_{i+2}.$$

The antidifferences K are determined up to cubic polynomials. We use these four degrees of freedom to obtain symmetric antidifferences. Figure 3.12 shows a finite part of the antidifference K_0 corresponding to the difference operator D_0; all of K_0 is obtained by extending the control polygon of K_0 cubically, i.e., so that the fourth differences involving control points not depicted in Figure 3.12 are 0. The antidifferences K_i are shifts $K_i(\cdot) = K_0(\cdot - i)$ of K_0 so that it suffices to compute enclosures $\lfloor K_0 \rfloor$ and $\lceil K_0 \rceil$.

To fulfill the requirements of Theorem 2 on page 8, the operator L that maps a cubic interpolatory subdivision function $b \in \mathcal{B}$ to a cubic spline $h \in \mathcal{H}$ only needs to leave $\mathcal{W} = \mathcal{B} \cap \mathcal{H}$ pointwise fixed. But L has also an influence on the size of the support of $(E - L)K$ and therefore on the size of the support of $\lfloor K \rfloor$ and $\lceil K \rceil$.

By the cubic precision of cubic interpolatory subdivision, K is a cubic polynomial outside of the interval $[-2, 2]$. We define Lb as the first order Hermite interpolant to b at the integers,

$$Lb(i) = b(i), \quad (Lb)'(i) = b'(i), \qquad \text{for all } i \in \mathbb{Z}.$$

This ensures that Lb and b coincide over the interval $[i, i+1]$ if b is cubic over this interval which is the case if and only if the 6 control points $b_{i-2}, \ldots, b_{i+3}$ influencing b on $[i, i+1]$ lie on one cubic. As a consequence, $(E - L)K_0 = 0$ outside of $[-2, 2]$ so that we only need to compute estimates $\lfloor K_0 \rfloor$ and $\lceil K_0 \rceil$ over this interval.

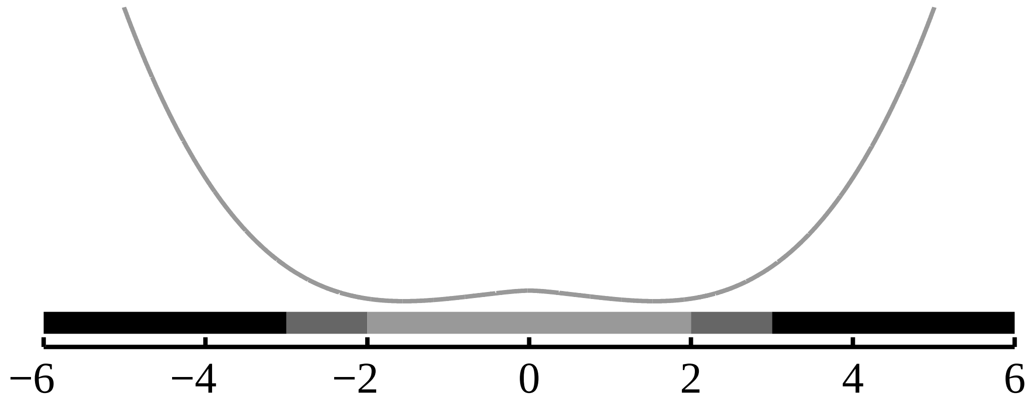


FIGURE 3.13. Constraints for estimating $(E - L)K_0$: the upper estimate $\lceil K_0 \rceil$ is constrained to be equal to $(E - L)K_0$ *(black bars)*, greater or equal than $(E - L)K_0$ by the choice of the spline coefficients *(dark grey bars)*, and greater or equal than the piecewise linear envelope of $(E - L)K_0$ *(light grey bars)*.

We construct the estimates $\lfloor K_0 \rfloor$ and $\lceil K_0 \rceil$ of $(E - L)K_0$ as cubic splines in $\mathcal{H}$. The estimates are computed with the help of the piecewise linear envelopes for cubic interpolatory subdivision functions and envelopes for splines constructed in the previous sections. To

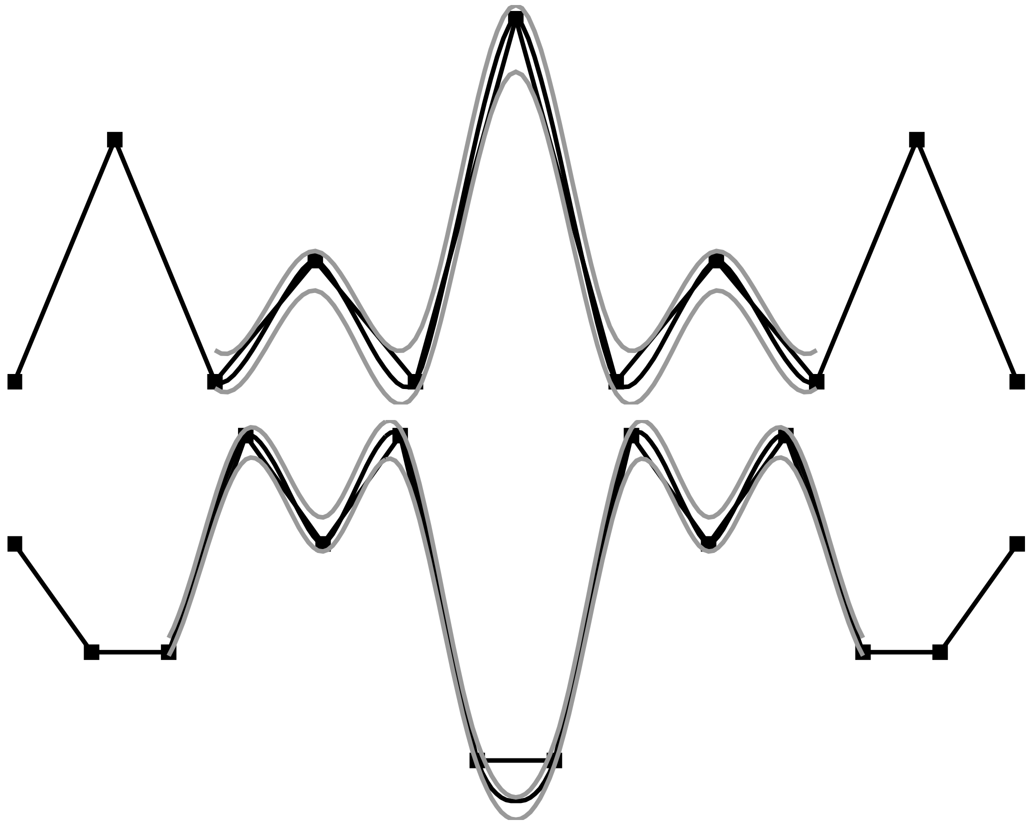


Figure 3.14. Two cubic interpolatory subdivision functions *(black)*, their control points *(black squares)* and their cubic envelopes *(grey)*.

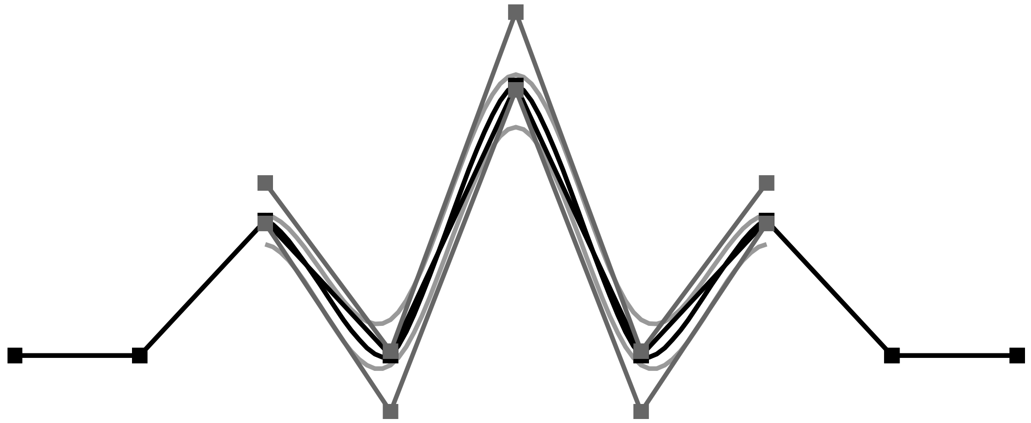


Figure 3.15. The linear *(dark grey)* and the cubic envelope *(light grey)* of a cubic interpolatory subdivision function *(black)*

bound $(E - L)K_0$ from above, we compute its piecewise linear envelope and construct a spline $h \in \mathcal{H}$ above the piecewise linear envelope using the algorithm Support introduced in Chapter 5. For our purposes, we can consider Support to be a black box that takes a knot sequence and a piecewise linear function as input and outputs a spline over that knot sequence that stays above the input polygon and minimizes the L_1 error between input polygon and spline.

Unfortunately, a spline h computed in this manner has infinite support since we sandwich the upper piecewise linear envelope of $(E-L)K_0$ between $(E-L)K_0$ and h. This can be remedied by solving SUPPORT only over the interval $[-2,2]$ and imposing additional constraints on h: since $(E-L)K_0 = 0$ outside of $[-2,2]$, we construct h such that $h = 0$ outside of $[-3,3]$. We use the interval $[-3,-2] \cup [2,3]$ as a transition zone be constraining the *B-spline coefficients* that influence h over this interval, but not outside $[-3,3]$, to be nonnegative, which implies $h \geq 0 = (E-L)K_0$ on $[-3,-2] \cup [2,3]$. This gives us enough flexibility to use the SUPPORT construction on top of the piecewise linear envelope of K_0 over $[-2,2]$.

The estimates $\lceil K_0 \rceil$ and $\lfloor K_0 \rfloor$ are further improved by using the symbolic refinement version of SUPPORT explained in Section 5.3 on page 57. With this technique, the estimates $\lceil K_0 \rceil$ and $\lfloor K_0 \rfloor$ can be brought arbitrarily close to the one–sided L_1 approximands of $(E-L)K_0$ in the spline space $\mathcal{H}$.

Figure 3.14 shows the cubic envelopes for two cubic interpolatory subdivision functions. Figure 3.15 shows how the cubic envelope smoothes the linear envelope of a cubic interpolatory subdivision function. The estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ were computed with the symbolic refinement version of SUPPORT over the knot sequence $(\hat{t}_i)$ with $\hat{t}_i = t_i/8$.

3.8. Envelopes of parametric curves

It is only a small step to go from the piecewise linear envelopes of univariate functions to piecewise linear envelopes of parametric curves: enveloping each coordinate function of a curve seperately yields piecewise linear envelopes for the curve.

Let $\mathbf{b}$ be a parametric curve, given in a representation

$$\mathbf{b} = \sum_j \mathbf{b}_j B_j, \qquad \mathbf{b}_j = (\mathbf{b}_j{}^1, \ldots, \mathbf{b}_j{}^n) \in \mathbb{R}^n,$$

for which we have piecewise linear envelopes for functions, for example uniform cubic splines or Bernstein polynomials.

For curves $\mathbf{b}$, the functional bound in the i–th component is denoted by

$$\underline{\mathbf{b}}^i \leq (E-L)\mathbf{b}^i \leq \overline{\mathbf{b}}^i.$$

The functional bounds are applied componentwise to parametric curves. Then any control point $\mathbf{b}_k$ and the curve point $\mathbf{b}(t_k^*)$ corresponding to $\mathbf{b}_k$ lie in a box whose width in the ith component is the bound in the ith component. Therefore, $\mathbf{b}(t_k^*)$ is located in the axis–aligned box S_k,

$$S_k = \left\{ \mathbf{x} : \underline{\mathbf{b}}^i(t_k^*) \leq \mathbf{x}^i - \mathbf{b}_k{}^i \leq \mathbf{x}^i(t_k^*) \text{ for } i = 1, \ldots, n \right\}.$$

Each point of the curve segment $\mathbf{b}(t)$, $t \in [t_k^*, t_{k+1}^*]$ lies in a box $S(t)$ that by the linearity of the functional envelopes is a convex combination of S_k and S_{k+1}:

$$S(t) = \frac{t_{k+1}^* - t}{t_{k+1}^* - t_k^*} S_k + \frac{t - t_k^*}{t_{k+1}^* - t_k^*} S_{k+1}.$$

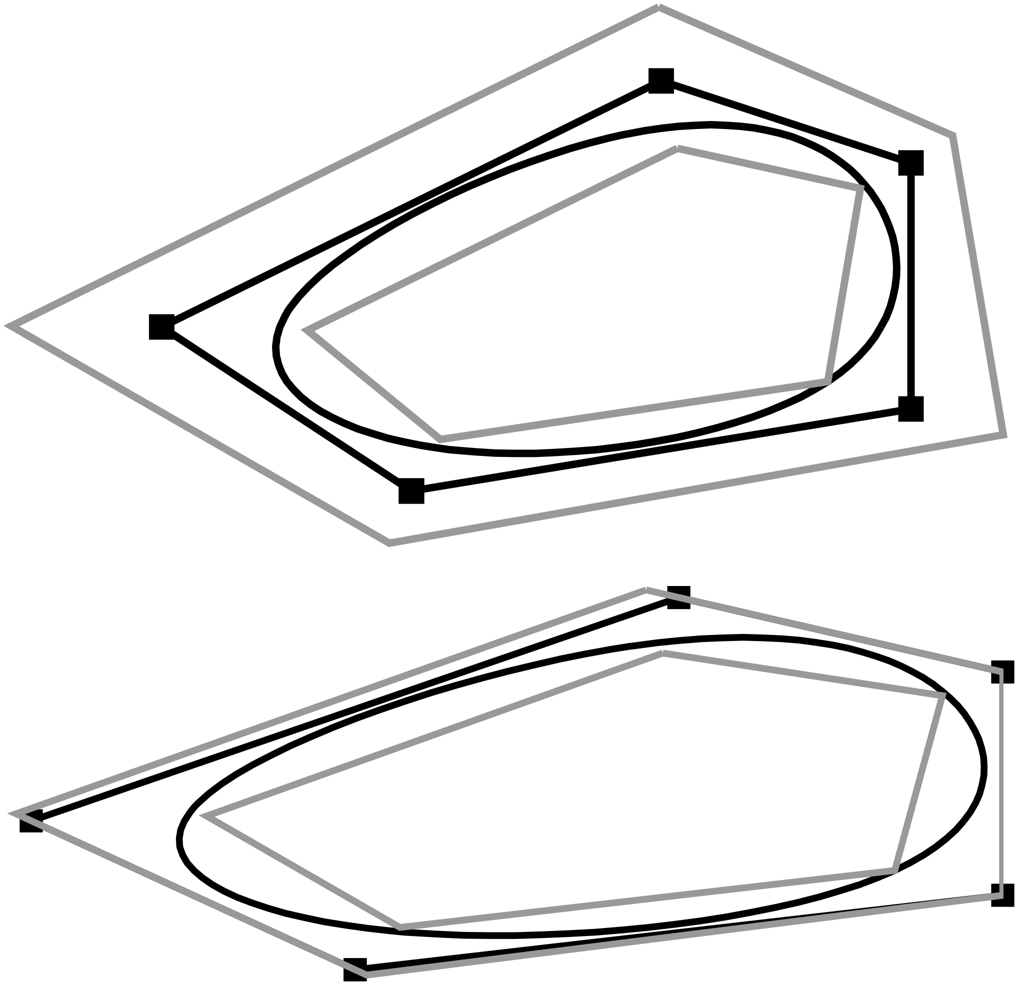


FIGURE 3.16. A uniform cubic spline curve *(black)* and its control points *(black squares)*. On top, the envelope *(grey)* is constructed with the Hölder bound from Equation (3.5.14), on the bottom with the bound from Theorem 5 on page 16

The curve segment is therefore contained in the union of all $S(t)$, $t \in [t^*_k, t^*_{k+1}]$, which is the convex hull H^k of the corners of S_k and S_{k+1}. To be specific, we discuss the case of planar curves.

3.8.1. Enveloping planar curves. Let $\mathbf{w}_{i,k}$, $i = 1, \dots, 4$, be the line segments connecting corresponding corners of S_k and S_{k+1}; that means $\mathbf{w}_{1,k}$ connects the lower left corner of S_k to the lower left corner of S_{k+1}, $\mathbf{w}_{2,k}$ connects the lower right corner of S_k to the lower right corner of S_{k+1} etc. as in Figure 3.17.

H_k consists of parts of the boundaries of S_k and S_{k+1} and exactly two additional line segments $\mathbf{u}_{1,k}$ and $\mathbf{u}_{2,k}$ chosen from the $\mathbf{w}_{i,k}$. Since $\mathbf{u}_{1,k}$ and $\mathbf{u}_{2,k}$ are part of the convex hull H_k, they do not intersect the interiors of S_k and S_{k+1}. We do not need to actually compute

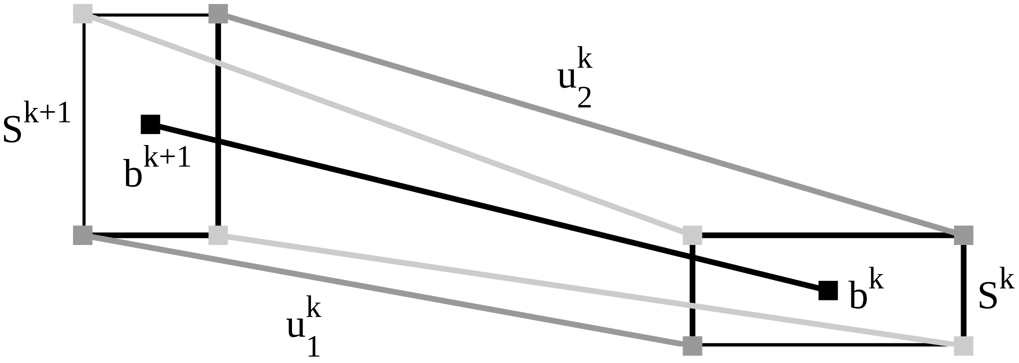


FIGURE 3.17. Constructing the envelope of a curve from the bounding rectangles S_k and S_{k+1}: only the outer line segments $\mathbf{u}_{1,k}$ and $\mathbf{u}_{2,k}$ are part of the convex hull of $S_k \cup S_{k+1}$ and therefore of the envelope

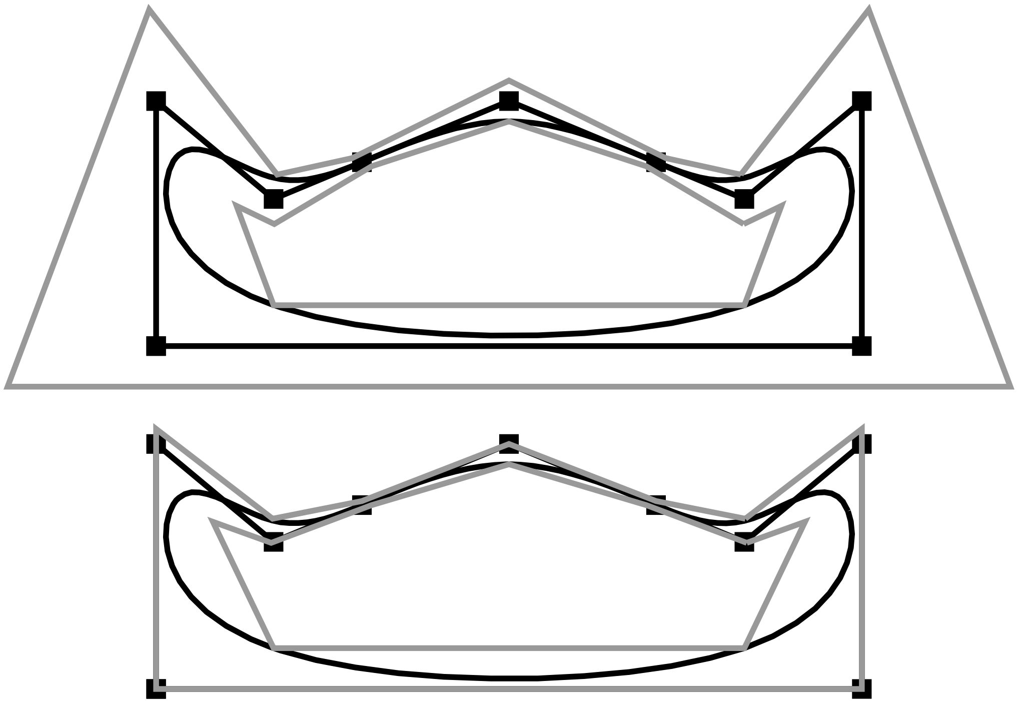


FIGURE 3.18. A uniform cubic spline curve *(black)* and its control points *(black squares)*. On top, the envelope *(grey)* is constructed with the Hölder bound from Equation (3.5.14), on the bottom with the bound from Theorem 5 on page 16

intersections of the $\mathbf{w}_{i,k}$ and S_k, S_{k+1} to select $\mathbf{u}_{1,k}$ and $\mathbf{u}_{2,k}$: since S_k and S_{k+1} are axis–aligned it suffices to look at the signs of the slopes of the $\mathbf{w}_{i,k}$. The $\mathbf{u}_{i,k}$ are separated by the line from $\mathbf{b}^k$ to $\mathbf{b}^{k+1}$; we call the one lying to the left of this line $\mathbf{u}_{1,k}$ and the one lying to the right of this line $\mathbf{u}_{2,k}$.

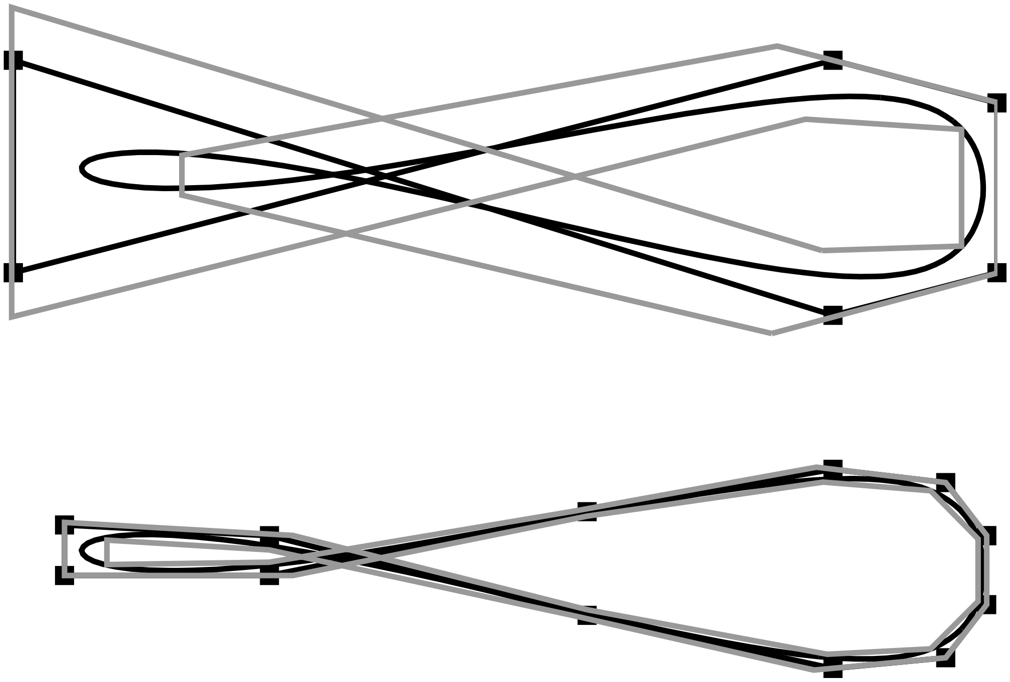


FIGURE 3.19. A self–intersecting quartic curve *(black)* and its control points *(black squares)*. The envelopes *(grey)* are constructed with the bound from Theorem 5 on page 16. The envelope converges rapidly to the curve as the comparison of the original envelope, top, and the envelope after one step of uniform refinement, bottom, shows

The sets $U_i = \{\mathbf{u}_{i,k}\}$ are not yet polylines: consecutive line segments $\mathbf{u}_{i,k}$ and $\mathbf{u}_{i,k+1}$ may intersect or not touch at all. But note that the line extending $\mathbf{u}_{i,k}$ always intersects the one extending $\mathbf{u}_{i,k+1}$. We obtain a proper polyline W_i with exactly one line segment for each control point of $\mathbf{b}$ by taking this intersection as starting point and the intersection with the line through $\mathbf{u}_{i,k+1}$ as the end point of W_i. The polylines W_1 and W_2 then form a *local* envelope of $\mathbf{b}$: the curve–piece $\mathbf{b}([t_k^*, t_{k+1}^*])$ lies in the quadrangle spanned by the k–th pieces of W_1 and W_2.

Figures 3.16 and 3.18 show global envelopes based on the Hölder and on the range estimation envelopes. Figure 3.19 shows the behavior of the global envelope under subdivision.

CHAPTER 4

Bivariate functions

We now turn to envelopes of bivariate functions and show how the framework from Chapter 2 can be applied to tensor product and bivariate (total–degree) Bernstein polynomials. The main difference between the envelopes for univariate and for bivariate Bernstein polynomials is that basing the envelopes on convex basis functions K and use cheap estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ that only exploit the convexity of the K is not sufficient to yield satisfactory envelopes. The estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ have to be obtained through the subdivide–and–bound procedure introduced in Section 3.7.1 on page 27; therefore, convexity, and in a sense simplicity, of the K_i is not important anymore and can be traded for simplicity of the difference operators.

This chapter is only concerned with envelopes of functions. Envelopes for parametric surfaces can be obtained from functional envelopes in a way similar to the construction for parametric curves in Section 3.8: applying the functional envelopes to the coordinate functions of a parametric surface at the control points yields axis aligned boxes enclosing the point on the surface corresponding to the control point. The local linear or bilinear hulls of these boxes provide an envelope for a part of the surface. The local hulls can then be further united to yield a global hull.

4.1. Tensor product Bernstein polynomials

A bivariate polynomial b is in tensor product Bernstein form of bidegree $d_1 \times d_2$ if it can be written as

$$b(u,v) = \sum_{i=0}^{d_1} \sum_{j=0}^{d_2} b_{i,j} B_{i,j}^{d_1,d_2}$$
$$\text{with } B_{i,j}^{d_1,d_2}(u,v) = B_i^{d_1}(u) B_j^{d_2}(v).$$

The tensor product Bernstein polynomials form a basis of $\mathcal{B} = \operatorname{span}\{B_{i,j}^{d_1,d_2}\}$. We are only concerned with the polynomial over the domain $\mathbb{U} = [0,1] \times [0,1]$. We use $n_1 \times n_2$, $n_i = d_i + 1$, to denote the *order* of a tensor product polynomial of bidigree $d_1 \times d_2$.

The control net of b has $n_1 n_2$ control points. We arrange the control points in a regular quadrilateral $n_1 \times n_2$ mesh over the xy–plane by assigning the *Greville abscissa* $t_{ij}^* = (i/d_1, j/d_2)$ to the control point $b_{i,j}$. We call control points $b_{i,j}$ with $(i,j) \in \{0, d_1\} \times \{0, d_2\}$ *corner control points* and all other control points *non–corner control points*. A control point $b_{i,j}$ is called an *interior control point* if $i \notin \{0, d_1\}$ and $j \notin \{0, d_2\}$, and a *boundary control point* otherwise.

Let $\mathcal{H} = \text{span}\{H_{i,j}^{d_1,d_2}\}$ be the space of bilinear functions with breaks at the Greville abscissae t_{ij}^* with $H_{i,j} = H_{i,j}^{d_1,d_2}$ defined as

$$H_{i,j}^{d_1,d_2} = H_i^{d_1} H_j^{d_2}$$

so that $H_{i,j}(t_{kl}^*) = [i = k]\,[l = j]$. The *control polygon* Lb of b is the piecewise bilinear interpolant of the control points,

$$Lb = \sum_{i=0}^{d_1} \sum_{j=0}^{d_2} b_{i,j}\, H_{i,j}.$$

The space $\mathcal{W} = \mathcal{B} \cap \mathcal{H}$ consists of all bilinear functions and has therefore dimension 4. A difference operator D must therefore annihilate all bilinear functions. The space of such difference operators is spanned by the univariate second differences D^{uni} depicted in Figure 4.1. There are $n_1(n_2-2)+(n_1-2)n_2$ univariate second differences.

Since $\dim \mathcal{W} = 4$, the map $E - L$ has rank $s = n_1 n_2 - 4$, which is also the rank of D^{uni}. There are of course many possible choices for a difference operator $D : \mathcal{B} \mapsto \mathbb{R}^s$ of full rank, and therefore many different envelopes for tensor product Bernstein polynomials. We concentrate on two particular operators, one for which the antidifference basis K is directionally convex and easily approximated by bilinear functions and one where the difference operator D is particularly simple.

In both cases, we associate one difference operator with each control point $b_{i,j}$ except for the corner control points. To formalise this association, let τ be a map defined on the indices of all non–corner control points that maps bijectively into $\{0, \ldots, n_1 n_2 - 5\}$. The k–th difference operator D_k is now associated with the control point $b_{i,j}$ for which $k = \tau(i,j)$. With this association, we call a difference operator D_k a boundary operator if $b_{i,j}$ is a boundary control point and an interior difference operator if $b_{i,j}$ is in the interior.

4.1.1. A directionally convex basis K.

A tensor product Bernstein polynomial b is directionally convex if every isoparameter curve $b(u,\cdot)$ and $b(\cdot,v)$ is convex [**26**]. This is the case exactly when the control polygon is convex in both parameter directions and therefore if $D^{\text{uni}} b \geq 0$. Such polynomials can be easily bounded from above by their piecewise bilinear interpolant at the Greville abscissae and from below by their control polygon.

We define difference operators $D = D^{\text{cnv}}$ using the masks in Figure 4.1. Each second difference D_k^{cnv}, $k = \tau(i,j)$, corresponds to exactly one non–corner control point $b_{i,j}$. For $k = \tau(i,j)$, a difference operator D_k^{cnv} on the boundary is the univariate second difference along that boundary centered at $b_{i,j}$. The interior difference operator D_k^{cnv} is the tensor product of the two univariate differences centered at $b_{i,j}$ as depicted on the right in Figure 4.1.

Rather than showing directly that D^{cnv} has full rank, we first describe a matrix K with $D^{\text{cnv}} K = I$, which implies that both D and K have full rank s.

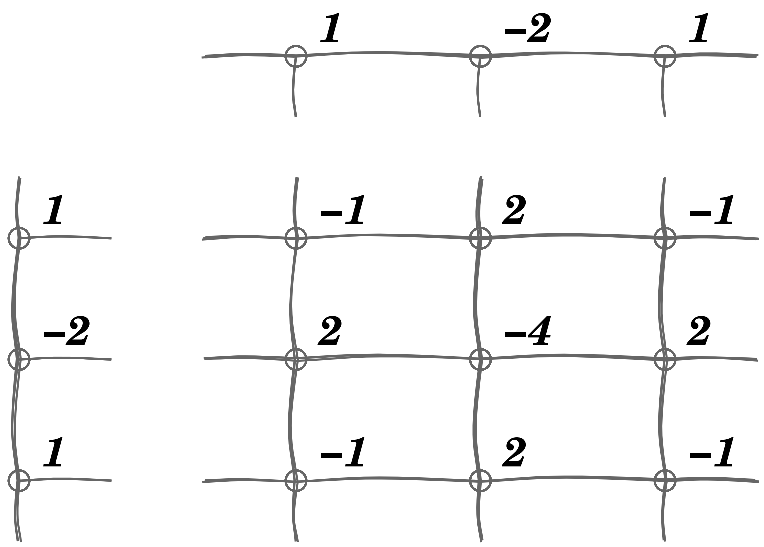


FIGURE 4.1. Second difference masks for tensor products: the univariate second differences in one parameter direction *(top)* and in the second parameter direction *(bottom left)* and the tensor product mask *(bottom right)* used for the convex operator D^{cnv}.

Let the function f_i^d be defined as

$$f_i^d(i_0) = \begin{cases} \dfrac{i_0(d-i)}{d} & i_0 \le i \\ \dfrac{i(d-i_0)}{d} & \text{otherwise.} \end{cases}$$

For the boundary operator D_k, $k = \tau(i,j)$, with $j = 0$ or $j = d_2$, K_k is defined as

$$K_{k,i_0,j_0} = \begin{cases} \dfrac{d_2 - j_0}{d_2} f_i^{d_1}(i_0) & j = 0 \\ \dfrac{j_0}{d_2} f_i^{d_1}(i_0) & j = d_2. \end{cases}$$

The entries $K_{k,:,:}$ corresponding to a boundary operator D_k with $i = 0$ or $i = d_1$ are defined analogously.

For an interior operator D_k, we define K_k as

$$K_{k,i_0,j_0} = f_i^{d_1}(i_0)\, f_j^{d_2}(j_0)$$

The K_k defined in this manner are linear in at least one direction except at $K_{\tau(i,j),i,j}$. This means that applying D_{k_0} to K_k with $k \ne k_0$ yields 0. It is straightforward to check that the application of D_k to K_k yields 1. Therefore $DK = I$ so that $D = D^{\mathrm{cnv}}$ is of full rank and $\ker D^{\mathrm{cnv}} = \mathcal{W}$. This means that, by Theorem 1,

$$(E - L)b = (E - L)K\, D^{\mathrm{cnv}} b \quad \text{for any } b \in \mathcal{B}.$$

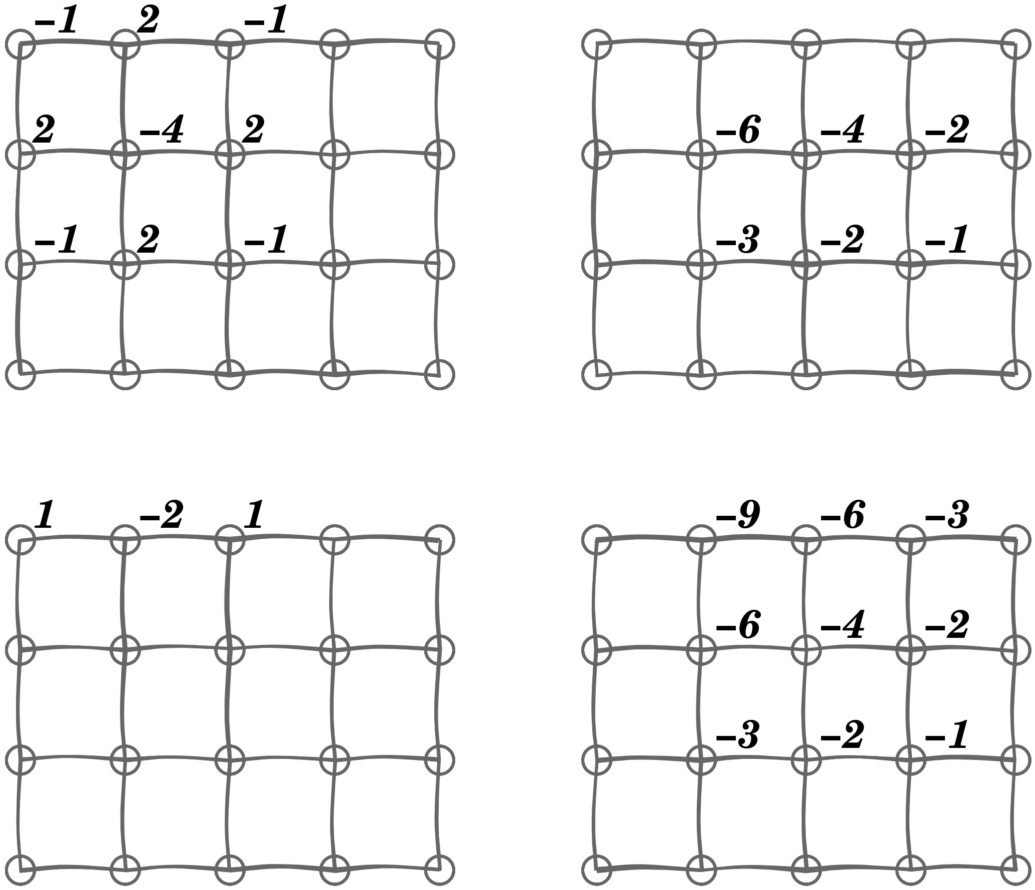


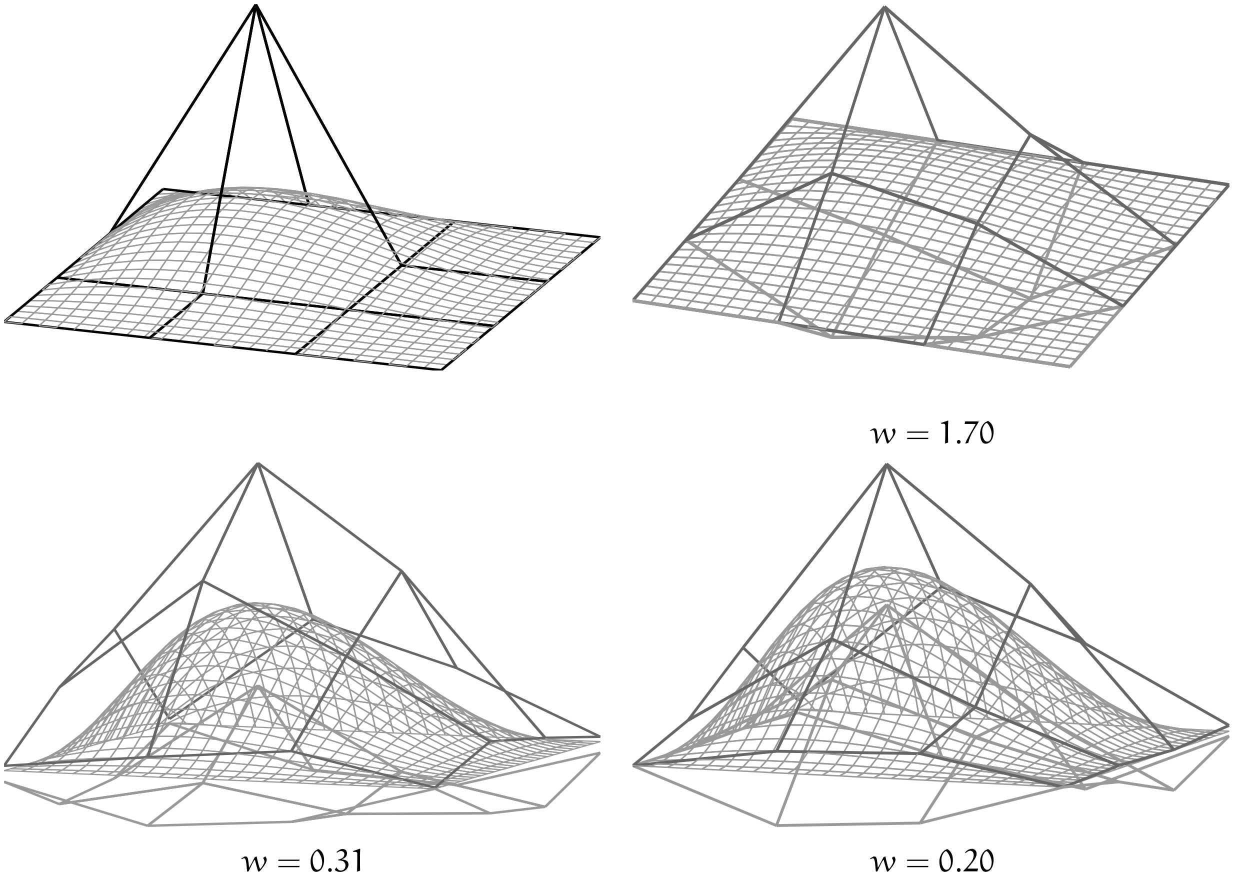
FIGURE 4.2. Difference masks D_k^{cnv} *(left)* and the corresponding control nets K_k *(right)* for tensor product polynomials of bidegree 3×4. The control nets K_k are scaled by 12.

It is also straightforward to check that $D^{uni}K \geq 0$ so that the functions K induced by D^{cnv} are directionally convex. This means that the functions $\lfloor K_k \rfloor$ and $\lceil K_k \rceil$ defined by

$$\lfloor K_k \rfloor = 0 \qquad \lceil K_k \rceil = \sum_{i=0}^{d_1} \sum_{j=0}^{d_2} (K_k(t_{ij}^*) - K_{k,i,j}) H_{i,j}$$

are piecewise bilinear estimates of $(E-L)K_k$. Therefore, $\lfloor K \rfloor \leq (E-L)K \leq \lceil K \rceil$ and Theorem 2 on page 8 holds.

The Figures 4.3, 4.4 and 4.5 show examples of envelopes resulting from these estimates at the top right. The envelopes computed with these estimates are reasonable in many cases. There are some examples though for which the envelopes are unacceptably bad: the envelope of the function in Figure 4.3 based on these estimates of K is considerably larger than the convex hull of the control polygon. The bounds based on D^{cnv} can be improved considerably through the subdivide–and–bound procedure that was used to compute bounds for cubic interpolatory subdivision functions in Section 3.7.1 on page 27.


$w = 1.70$

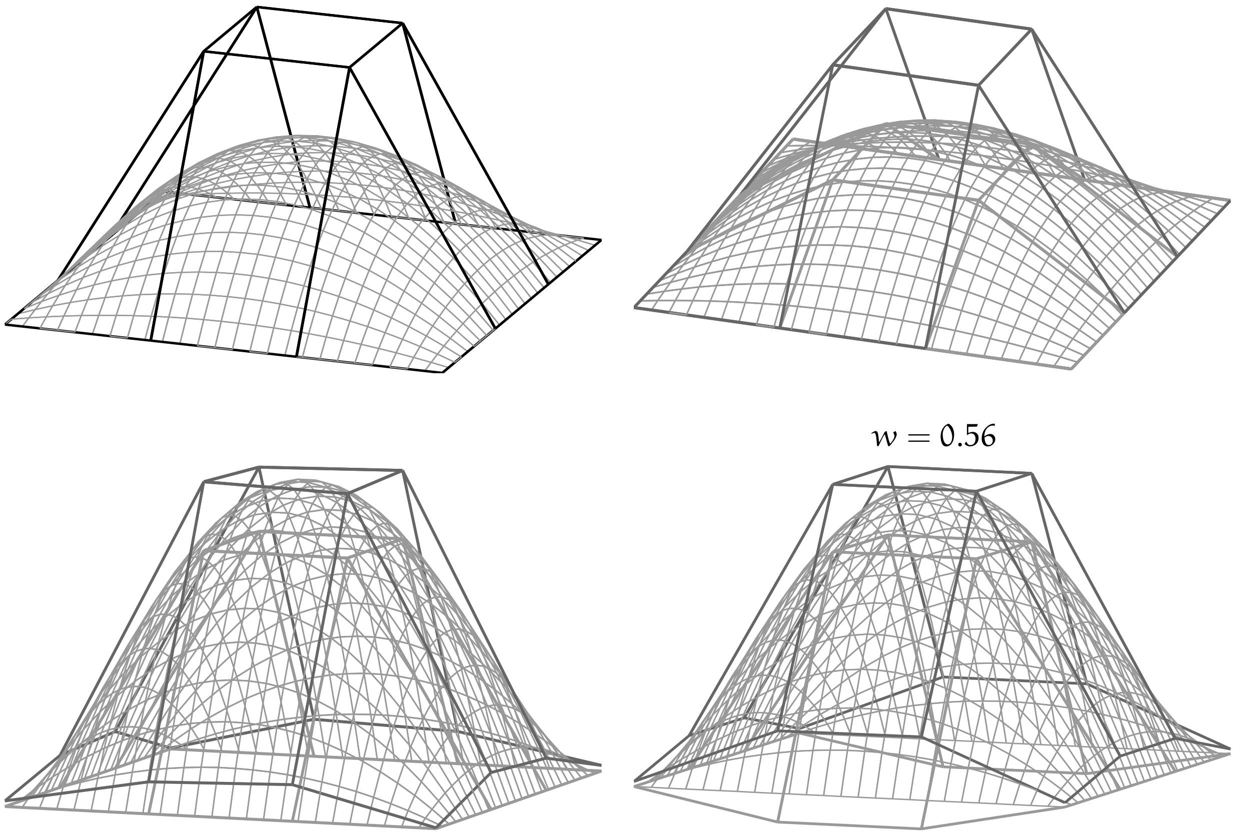
$w = 0.31$

$w = 0.20$

FIGURE 4.3. The bicubic basis function $B^{3,3}_{1,1}$ and some envelopes. Shown are the function and its control polygon *(top left)*, the envelope based on D^{cnv} with simple estimates of K *(top right)*, the envelope based on D^{cnv} with improved estimates *(bottom left)*, and the envelope based on D^{cmb} *(bottom right)*. The value w gives the maximal width of the respective envelope.

The estimate $\lceil K \rceil$ is already optimal since it interpolates $(E-L)K$ at the Greville abscissae. We can therefore only improve the estimate $\lfloor K \rfloor$; for this, we use the bounds $\lfloor K \rfloor$ and $\lceil K \rceil$ as the basis for the initial envelopes $\underline{e}_0$ and $\overline{e}_0$ of a function K_k. Splitting K_k uniformly into a certain number of polynomials, for example $d_1 \times d_2$, yields a refined lower envelope $\underline{e}_r$. An improved bilinear lower estimate $\underline{e}'_0$ for $(E-L)K_k$ is obtained by solving a linear program. The constraints for this program are linear because we compute bilinear estimates. Applying this procedure to each K_k in turn produces improved estimates on the range of $(E-L)K$. Examples of the improved envelopes can be found in Figures 4.3, 4.4 and 4.5 at the bottom left. It is clear from these figures that good estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ improve the envelope greatly.

REMARK 6. Since $\lfloor K \rfloor$ now takes on positive values, the control polygon Lb does not have to lie between the lower envelope $\underline{b}$ and $\overline{b}$ anymore. As a consequence, $\overline{b}$ and $\underline{b}$ approximate b with *less* error than Lb. In applications such as rendering, Lb can no longer be used as a



$w = 0.56$

$w = 0.13$

$w = 0.18$

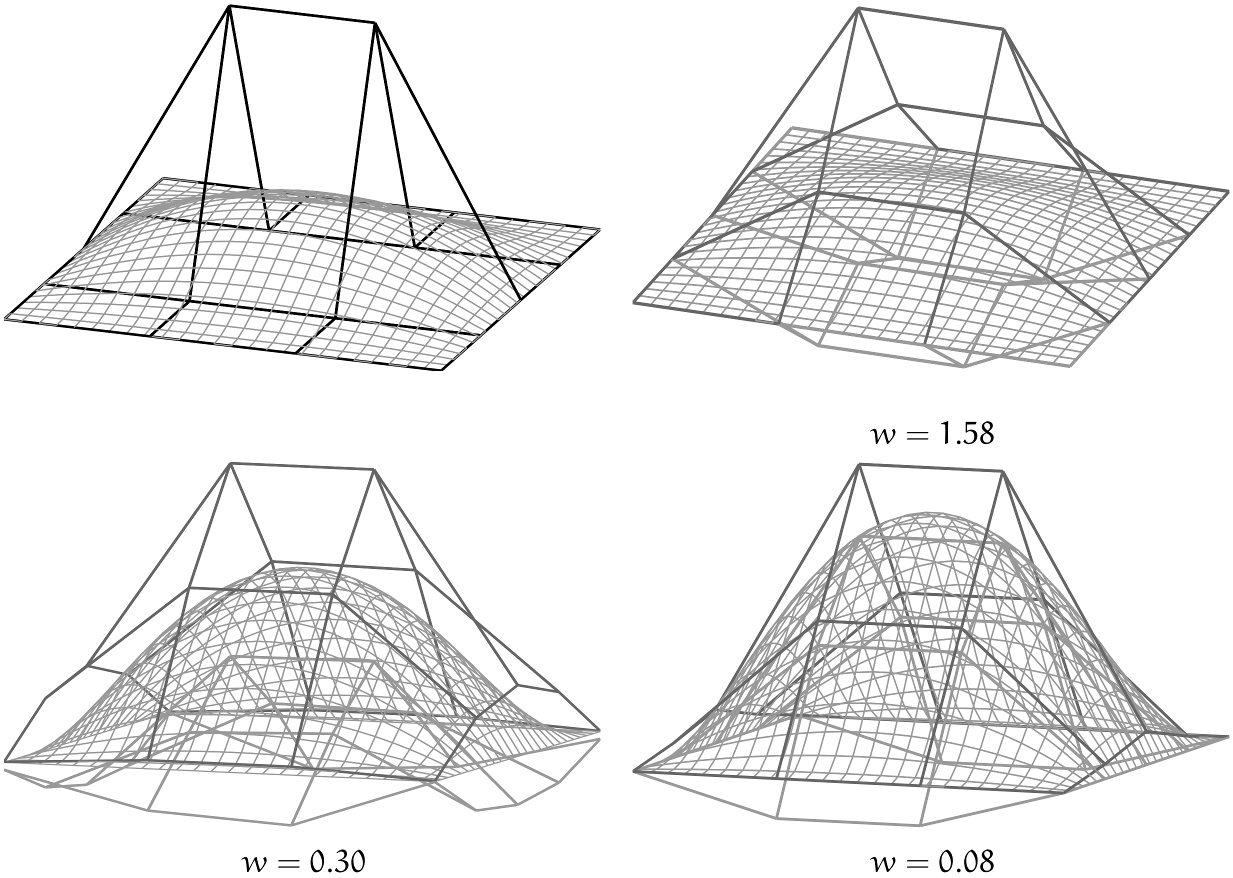
FIGURE 4.4. A bicubic Bernstein polynomial. The boundary coefficients are 0 and the 4 center coefficients are 1. Shown are the same envelopes as in Figure 4.4. The upper envelope at the top right coincides with the control polygon, while the lower envelope interpolates the function at the Greville abscissae.

standin for b with $\overline{b} - \underline{b}$ as an error bound. This is easily remedied by rendering either $\overline{b}$, $\underline{b}$ or $Lb + (\overline{b} - \underline{b})/2$.

4.1.2. Using univariate second differences. In computing envelopes in practice, the main effort goes into computing differences of the control net. It is therefore desirable to compute envelopes based on differences with small masks. We can construct a simple difference operator $D = D^{\mathrm{cmb}}$ by choosing an independent subset of the operators D^{uni}; the operators we pick are arranged in a comb–pattern. For each control point $b_{i,j}$ with $0 < i < d_1$, we pick the univariate second difference in the i–direction, i.e., the one mapping $b_{i,j}$ to $b_{i-1,j} - 2b_{i,j} + b_{i+1,j}$. For control points $b_{i,j}$ on the boundary of the control net with $0 < j < d_2$ and $i \in \{0, d_1\}$, we pick the univariate second difference in the j–direction.

We define a matrix K in terms of D^{cmb} in the following manner: for an operator D_k with $k = \tau(i,j)$ and $0 < i < d_1$,

$$A_{k,i_0,j_0} = [j = j_0]\, f_i^{d_1}(i_0).$$



$w = 1.58$

$w = 0.30$

$w = 0.08$

FIGURE 4.5. A tensor product polynomial of bidegree 3×4. The coefficients are either 0 or 1. Shown are the same envelopes as in Figure 4.4. The envelope on the top right is considerably larger than the control polygon.

For a k associated with a control point $b_{i,j}$ with $0 < j < d_2$, $i \in \{0, d_1\}$ we set K_k to

$$K_{k,i_0,j_0} = f_i^{d_1}(i_0).$$

It is easily checked that now $D^{\mathrm{cmb}} K = I$. The antidifference basis D induced by D^{cmb} is not directionally convex, though, and we need to employ the subdivide–and–bound procedure from Section 3.7.1 on page 27 to compute estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ of $(E - L)K$ that produce usable envelopes. The initial estimates required for the subdivide–and–bound procedure can be easily computed from the smallest and largest control point of each K_k. Figures 4.3, 4.4 and 4.5 show examples of envelopes based on D^{cmb} at the bottom right.

4.1.3. Comparing envelopes. Having two different ways to compute envelopes for tensor product polynomials, one based on D^{cnv} and one based on D^{uni}, raises of course the question which of the envelopes is "better" for a suitable definition of "better" and whether using one or the other difference operator can be guaranteed to always yield better envelopes.

The most important feature of envelopes is that they enclose a function b, and therefore specify its location. The smaller the width of an envelope at each (u, v), the better we know the location of b. It is therefore reasonable to call one envelope of a function b better than another envelope of b if the maximal width of the first envelope is smaller than the maximal width of the second.

In general, envelopes based on different difference operators D and Δ are incomparable with respect to the above criterion, i.e., there will always be two polynomials p and q so that the envelope of p based on D is *smaller* than the one based on Δ while the envelope of q based on D is *larger* than its envelope based on Δ.

4.1.4. Tensor product B-splines. Piecewise bilinear envelopes for tensor product B–splines can be constructed in complete analogy to the construction for tensor product Bernstein polynomials.

The difference operators need to take the non–equidistant spacing of the Greville abscissae into account in a fashion similar to the univariate case described in Section 3.2.

For finite knot sequences, operators similar to D^{cnv} and D^{uni} can be constructed. If the knot sequences are infinite though, the analog of D^{cnv} consists only of operators using the 3×3 tensor product mask shown in Figure 4.2, bottom right. As was pointed out by Ulrich Reif [**24**], these operators annihilate not only bivariate functions and can therefore not be used to derive a bound on the difference between a tensor product B-spline and its control polygon.

The estimates $\lfloor K \rfloor$ and $\lceil K \rceil$ have to be obtained by the subdivide–and–bound procedure from Section 3.7.1 on page 27. There are two possible strategies one can follow: to get the best possible estimates, the control nets of the K need to be subdivided as finely as possible. The computation of $\lfloor K \rfloor$ and $\lceil K \rceil$ can take considerable time, though. It is also possible to leverage (existing) estimates for tensor product Bernstein polynomials of the same bidegree as the B-spline basis in question by converting the K into tensor product Bernstein polynomials, one for each mesh cell of the B-spline control polygon.

4.2. Bivariate Bernstein polynomials

The bivariate Bernstein polynomials $B_{\mathbf{i}}^d$ are the $\binom{d+2}{2}$ summands of the multinomial equation

$$1 = (u_0 + u_1 + u_2)^d = \sum_{|\mathbf{i}|=d} \binom{d}{\mathbf{i}} \mathbf{u}^{\mathbf{i}} = \sum_{|\mathbf{i}|=d} \binom{d}{i_0} \binom{d-i_0}{i_1} {u_0}^{i_0} {u_1}^{i_1} {u_2}^{i_2},$$

where $\mathbf{u} = (u_0, u_1, u_2)$ with $u_0 + u_1 + u_2 = 1$, $\mathbf{i} = (i_0, i_1, i_2) \in \mathbb{N}_0^3$, and $|\mathbf{i}| = i_0 + i_1 + i_2$. Their properties, similar to the univariate Bernstein polynomials, include the convex hull property and a recursion that serves as the basis for deCasteljau's algorithm and their subdividability. See [**11**, **5**] for an in–depth introduction to bivariate Bernstein polynomials.

The bivariate Bernstein form of a bivariate polynomial b of degree d is

$$b(\mathbf{u}) = \sum_{|\mathbf{i}|=d} b_{\mathbf{i}} B_{\mathbf{i}}^d.$$

The $b_{\mathbf{i}}$ are called the Bernstein control points of b. The bivariate Bernstein polynomials are a basis of the space $\mathcal{B}$ of all bivariate polynomials of total degree d. Usually, one considers

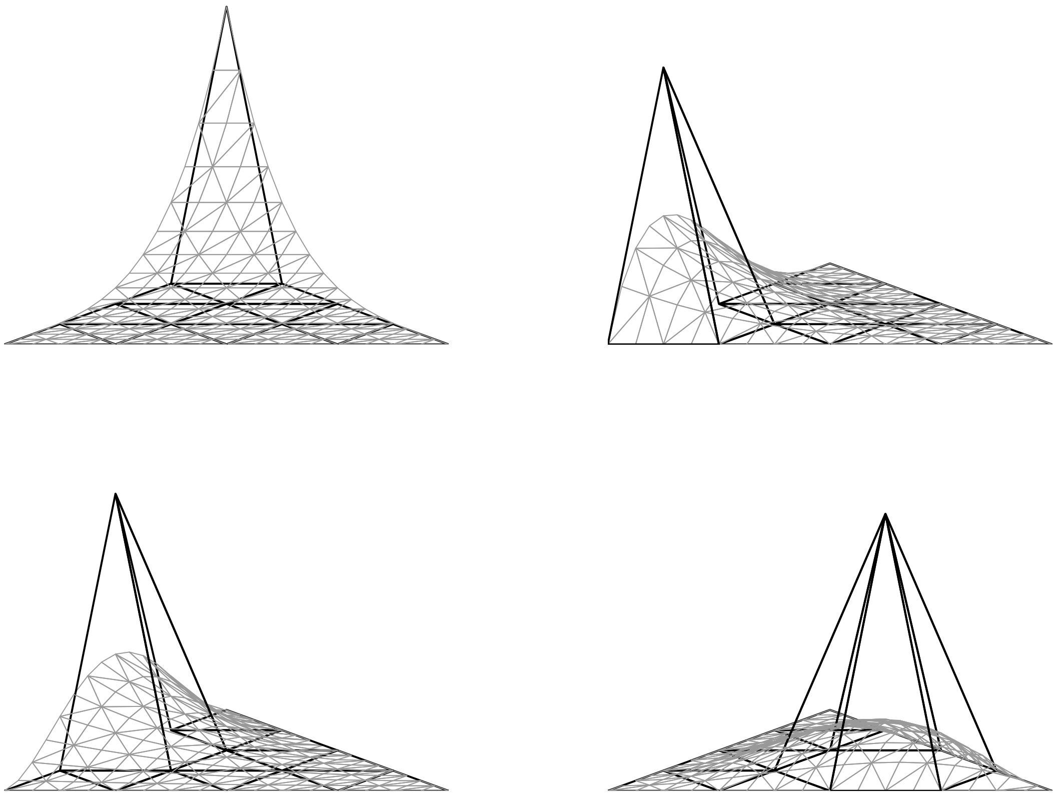


FIGURE 4.6. Some bivariate Bernstein polynomials of degree 4 and their control polygons. The remaining basis functions follow from symmetry.

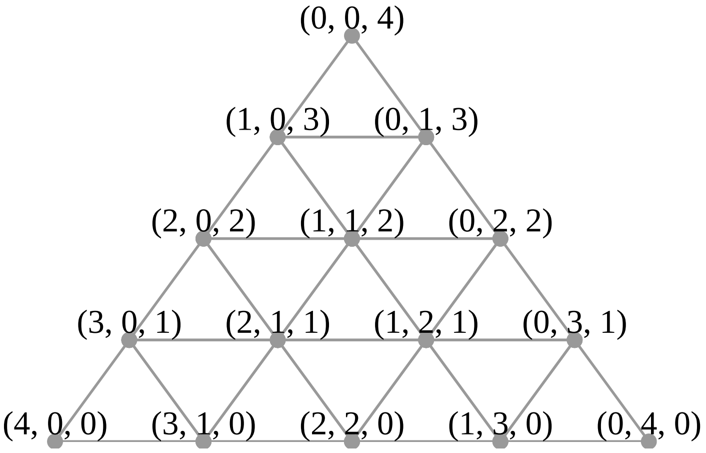


FIGURE 4.7. The control net of a cubic Bézier triangle with the multiindices of the control points.

these polynomials only over the triangular domain

$$\mathbb{U} = \{ (u_0, u_1) : u_0, u_1 \geq 0 \text{ and } u_0 + u_1 \leq 1 \}.$$

The Bernstein control polygon of b is the piecewise linear interpolant Lb of the control points b_i at the Greville abscissae t_i^*,

$$t_i^* = (i_0/d, i_1/d) \in \mathbb{U}, \qquad i_0 + i_1 + i_2 = d.$$

A basis for the space $\mathcal{H}$ of control polygons is formed by the piecewise linear hat functions H_i with $H_i(\mathbf{j}) = [\mathbf{i} = \mathbf{j}]$.

The space $\mathcal{H} \cap \mathcal{B}$ is the space of bivariate linear functions, which has dimension 3, so that $\operatorname{rank}(E - L) = n - 3$ where $n = \binom{d+2}{2}$ is the dimension of $\mathcal{B}$. In analogy to the tensor product case, we associate one difference operator with each control point except for the corner control points, the ones with indices $(d, 0, 0)$, $(0, d, 0)$, and $(0, 0, d)$. For the remaining, non–corner, control points, let τ map their indices bijectively into $\{0, \ldots, n - 4\}$.

The space of difference operators that annihilate linear functions is spanned by the $3\binom{n}{2}$ linearly dependent *mixed differences* D_k^{mix}. Each D_k^{mix} corresponds to one interior edge of the control polygon and involves only the two control points on that edge and the two control points directly across the edge as shown in Figure 4.8. The operators D^{mix} measure the "kink" between triangles in the control mesh. A bivariate Bernstein polynomial b is convex if $D^{\text{mix}} b \geq 0$, i.e., if the control mesh curls upwards across each interior edge. Reif [**25**] computes Hölder bounds based on these operators.

4.2.1. A directionally convex basis K. The operators D^{cnv} shown in Figure 4.9 yield a directionally convex antidifference basis K. Experimentation with a computer algebra system shows that these operators have the smallest support possible.

The control nets of the functions K do not have a simple closed–form representation.

Because of their convexity, the functions K can be estimated from above by their piecewise linear interpolant and from below by their control net. Similar to the tensor product case, the

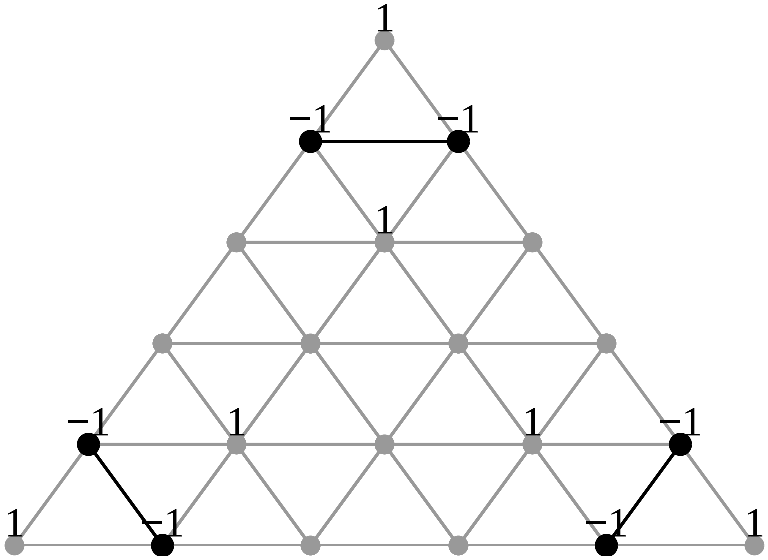


FIGURE 4.8. Some mixed differences D^{mix}. The three operators are the ones corresponding to the three edges next to the corner control points.

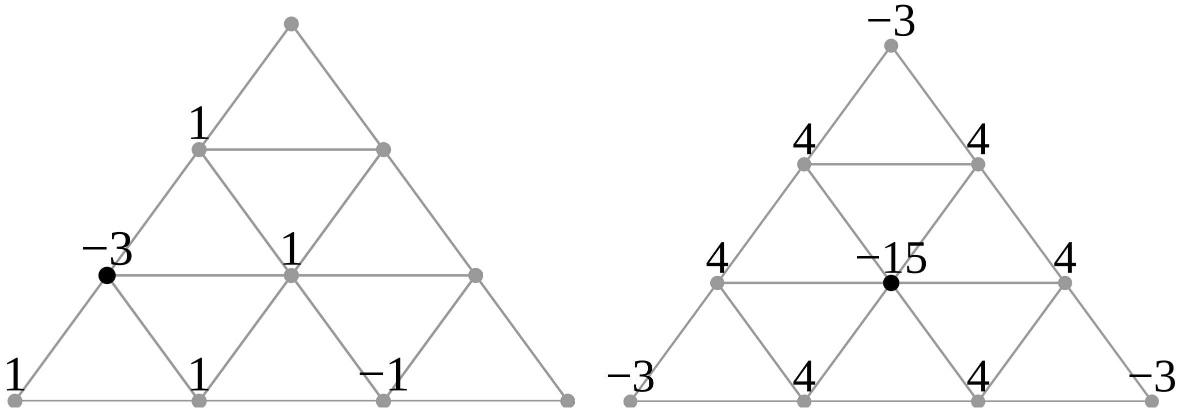


FIGURE 4.9. The operators $\mathrm{D}^{\mathrm{cnv}}$ yielding a convex antidifference basis K. Shown are an operator on the boundary *(left)* and an operator in the interior *(right)* and their corresponding control points *(black circles)*.

resulting bounds are poor and need to be improved by the subdivide–and–bound procedure from Section 3.7.1 on page 27. The second row of Figure 4.11 shows some functions and their envelopes based on $\mathrm{D}^{\mathrm{cnv}}$.

4.2.2. Using simple difference operators. We can also compute envelopes using the operators depicted in Figure 4.10. These operators are particularly simple: boundary operators are univariate second differences along that boundary, while the interior operators are the “umbrella” or discrete Laplace operators.

The functions K with $\mathrm{D}^{\mathrm{lpl}}\mathrm{K} = \mathrm{I}$ are not convex and have to be estimated numerically using the subdivide–and–bound procedure from Section 3.7.1 on page 27. The min–max criterion applied to the control nets of the functions K yields an initial estimate on the range of the K for this procedure.

The third row of Figure 4.11 shows some functions and their envelopes based on these simple differences $\mathrm{D}^{\mathrm{lpl}}$.

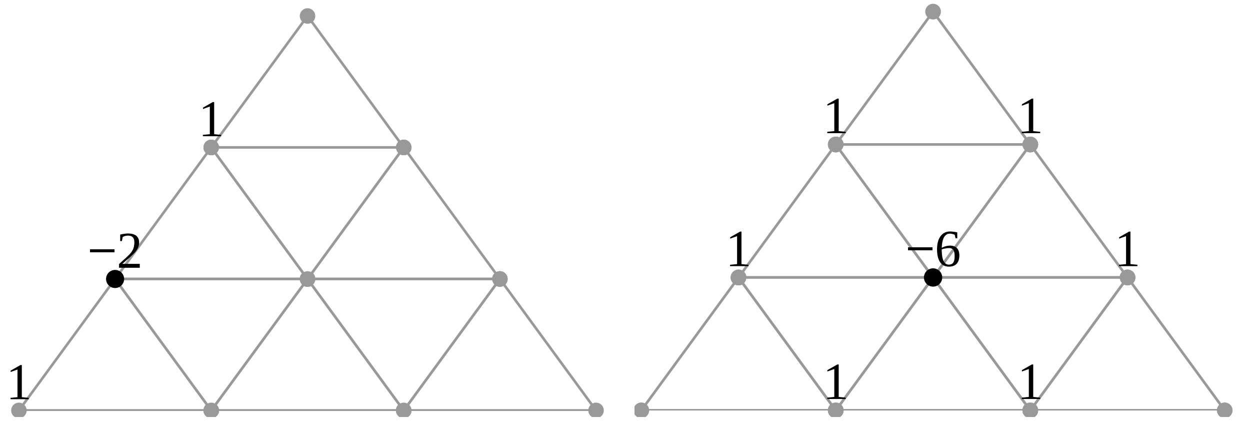
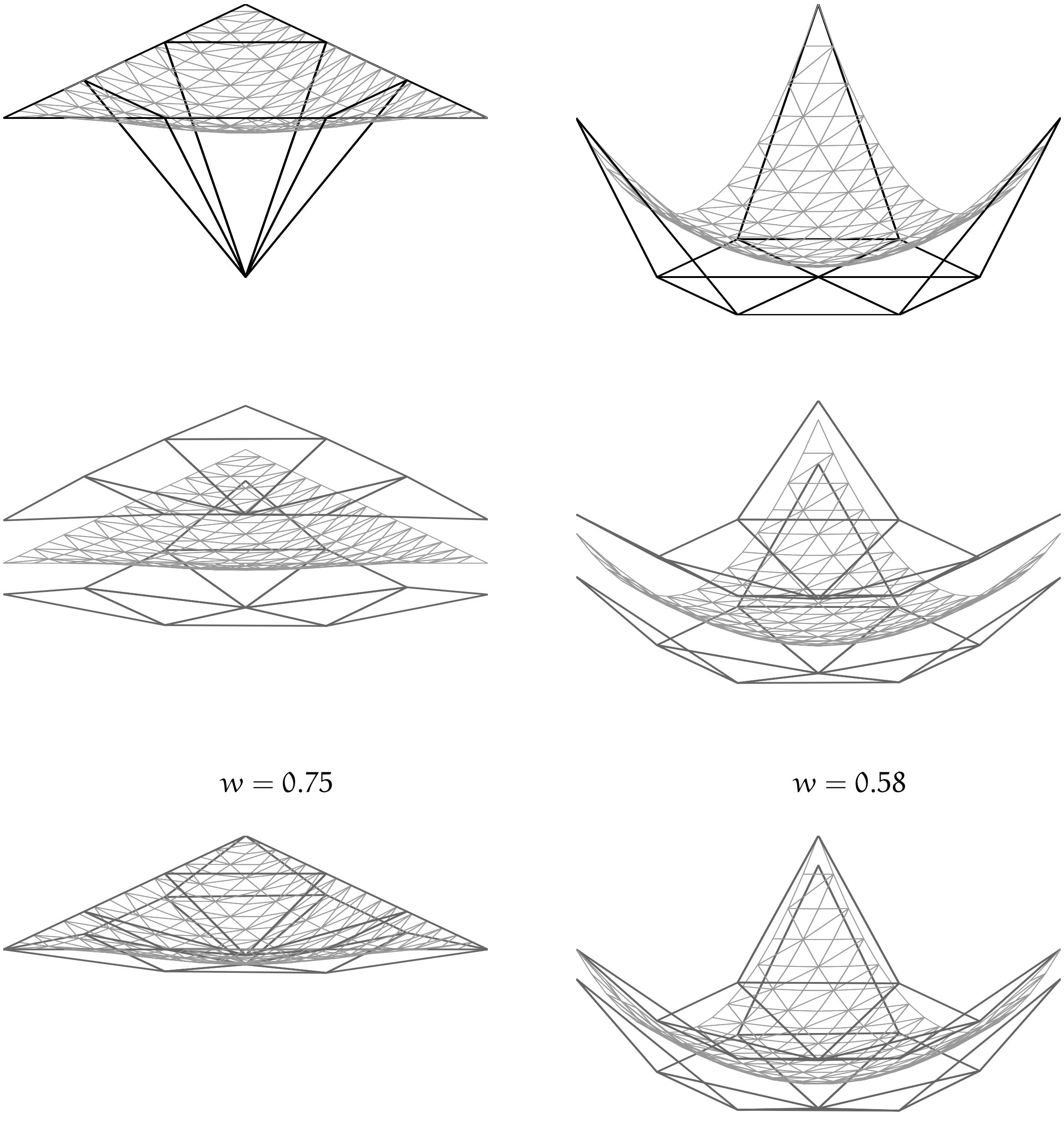


FIGURE 4.10. The operators $\mathrm{D}^{\mathrm{lpl}}$ on the boundary *(left)* and in the interior *(right)*.



$w = 0.75$

$w = 0.58$

$w = 0.12$

$w = 0.29$

FIGURE 4.11. The bivariate Bernstein polynomial $-B^3_{(1,1,1)}$ *(left column)* and the bivariate Bernstein polynomial $B^3_{(3,0,0)} + B^3_{(0,3,0)} + B^3_{(0,0,3)}$ *(right column)*. The first row shows the function and its control polygon, the second row shows the function and its envelope based on D^{cnv}, and the third row shows the function and its envelope based on D^{lpl}. The value w gives the maximal width of the respective envelope.

CHAPTER 5

One–sided constructions

Many industrial applications require the construction of smooth paths that avoid certain forbidden regions. A typical example is the path of a cutting tool which must stay away from the surrounding machine to avoid damage to tool and machine. Another is controlling a robot moving through a maze. A third is graph layout [8].

The rich toolkit of computational geometry delivers permissible piecewise linear paths in these situations (see e.g. the references in Chapter 13 of [4]). However, many physical applications demand a smoother path, for example a path that is tangent continuous or even has a continuously varying curvature. The (offset) path of the cutting tool, for example, must be parametrically curvature–continuous to avoid jumps in the acceleration that would strain or damage the driver motors.

5.1. The Support and Channel problems

We formulate and give practical algorithms for two basic problems, Support and Channel. Both problems are stated in terms of spline *functions* rather than planar parametric curves. For physically motivated path problems, the functional notion of continuity is more appropriate than the notion of geometric continuity. A parametric problem can be reduced to a functional problem by independently constructing functions for each coordinate or by recasting it in polar coordinates if the input polygon is star–shaped.

The term "close" in the following statement of the two basic problems is made precise in Section 5.2. For both problems, we assume that a knot sequence (t_i) is given and that the desired spline is to be constructed over this knot sequence.

Support: Given a piecewise linear function $\underline{a}$ (the *input polygon*) with vertices $(\underline{x}_i, \underline{a}_i)$ and $\underline{x}_0 < \underline{x}_1 < \cdots < \underline{x}_{\underline{n}}$, construct a spline function b of a given degree and smoothness that on the interval $[\underline{x}_0, \underline{x}_{\underline{n}}]$ stays close to and entirely above or on the input polygon.

Channel: Given two non–intersecting input polygons $\underline{a} < \overline{a}$ with vertices $(\underline{x}_i, \underline{a}_i)$ and $(\overline{x}_i, \overline{a}_i)$, construct a spline function b that stays between $\underline{a}$ and $\overline{a}$.

We write Support$[t]$ and Channel$[t]$ to indicate over which knot sequences the problems are to be solved.

The main difficulty in solving Support and Channel is that they are *continuous problems*: we have constraints of the form $\underline{a}(t) \leq b(t)$ respectively $\underline{a}(t) \leq b(t) \leq \overline{a}(t)$. We avoid solving these continuously constrained feasibility problems by recasting Support and Channel in terms of piecewise linear envelopes. To solve Support it suffices to find a spline b whose lower envelope $\underline{b}$ lies above the input polygon. Similarly, we can solve Channel by finding a spline whose envelope stays inside the channel formed by $\underline{a}$ and $\overline{a}$. This lets us

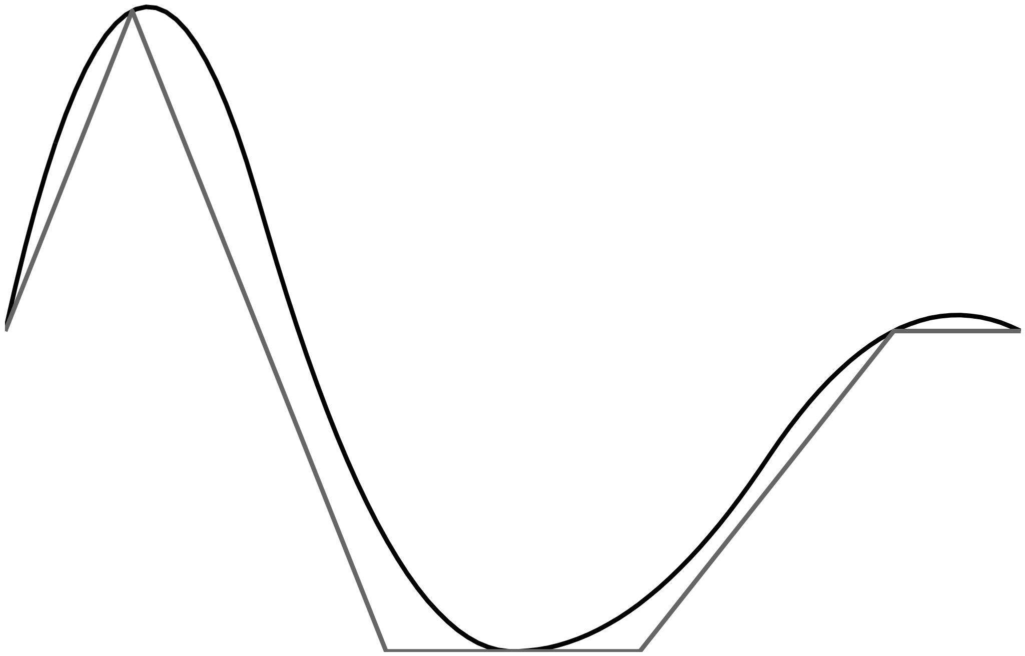


FIGURE 5.1. The SUPPORT problem: given an input polygon $\underline{a}$ *(grey)* find a spline *(black)* that stays above but close to $\underline{a}$.

solve SUPPORT and CHANNEL by linear, respectively quadratic programming depending on what additional properties we would like to enforce.

5.1.1. Prior work. Approximation theory has long considered the closely related problems of *one–sided approximation* and *two–sided approximation*. Based on Buck's seminal "Applications of Duality in Approximation Theory" [**1**], DeVore [**7**] established the close relation of one–sided approximation to quadrature formulas with nonnegative coefficients and gives a Remez-type algorithm for determining a (unique) solution. This, however, is based on the assumptions that the function to be one–sidedly approximated is differentiable, and that approximands form a strong Chebyshev space. Neither assumption holds for the problems of interest in this chapter. Mangasarian and Schumaker [**18**] formulate necessary and sufficient conditions for continuously constrained two–sided approximation problems that guarantee the existence and uniqueness of a solution. Thirty more years have seen a rich body of literature concerning one– and two–sided approximation.

The state of the art has been collected in the monograph [**22**] which also offers algorithms in its last chapter. The reader may check however, that the proposed procedures do not fall into the realm of practical computing in that they rely on a discretization refinement without a bound on the size of the subproblems. Termination is not guaranteed other than that a

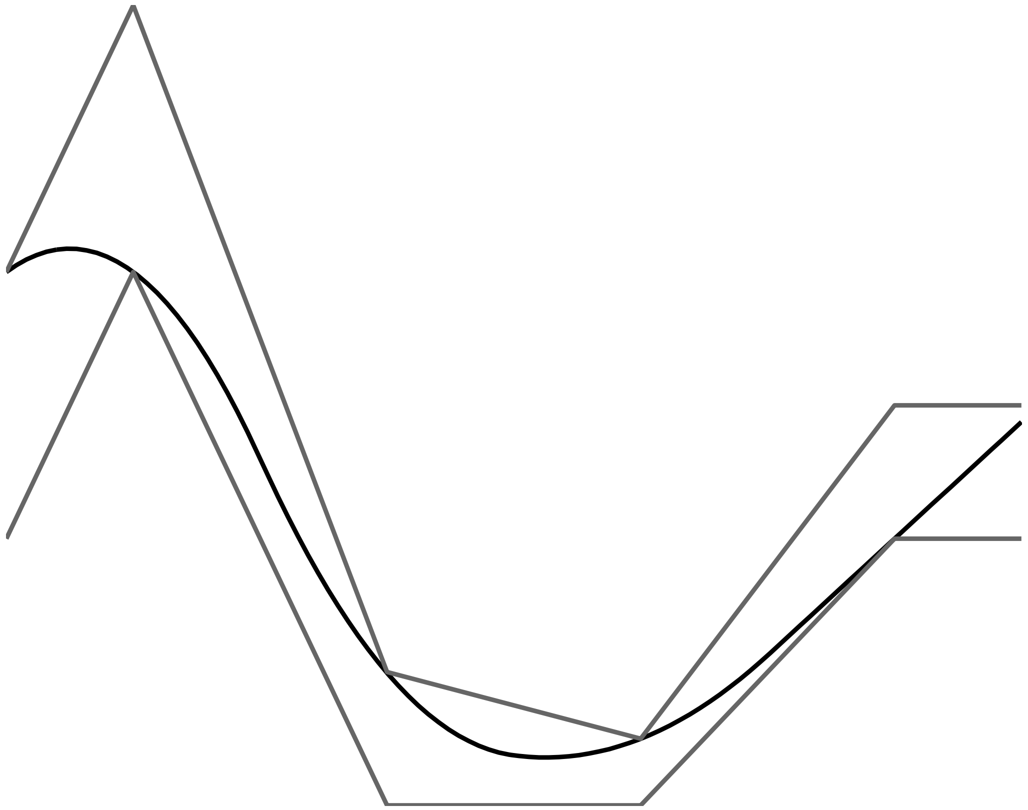


FIGURE 5.2. The CHANNEL problem: given two non–intersecting input polygons $\underline{a}$ and $\overline{a}$ *(grey)*, find a spline *(black)* that stays between $\underline{a}$ and $\overline{a}$.

subsequence must exist that converges. To quote the author [**22**], page 181, “Suffice to say that convergence is generally very slow, at least in theory”.

This chapter does not solve the hard problems of *one–sided approximation* as formulated by the approximation theorists — we are satisfied with an efficiently computable and tight approximation with a small, quantifiable error, while the approximation theorist looks to establish optimality and uniqueness. We hope to make this difference clear by calling the problems SUPPORT and CHANNEL rather than one–sided approximation and two–sided approximation.

In “A path router for graph drawing” [**8**], Dobkin, Gansner, Koutsofios and North derive interpolating piecewise cubic curves in Bézier form (Catmull-Rom splines) from the shortest path polyline to trace out a once differentiable path around obstacles. After construction, the path is checked for admissibility (non-intersection with the forbidden region) and recomputed with more curve pieces where it fails. In contrast, our approach below uses *a priori* bounds and supports the full gamut of spline representations.

Opfer and Oberle [**21**] have addressed a related problem: that of finding a natural cubic spline that interpolates given values and stays inside a polygonal channel. They derive necessary conditions for a solution through a theorem from control theory; based on this characterization, an iterative, Newton–type algorithm is constructed that inserts additional

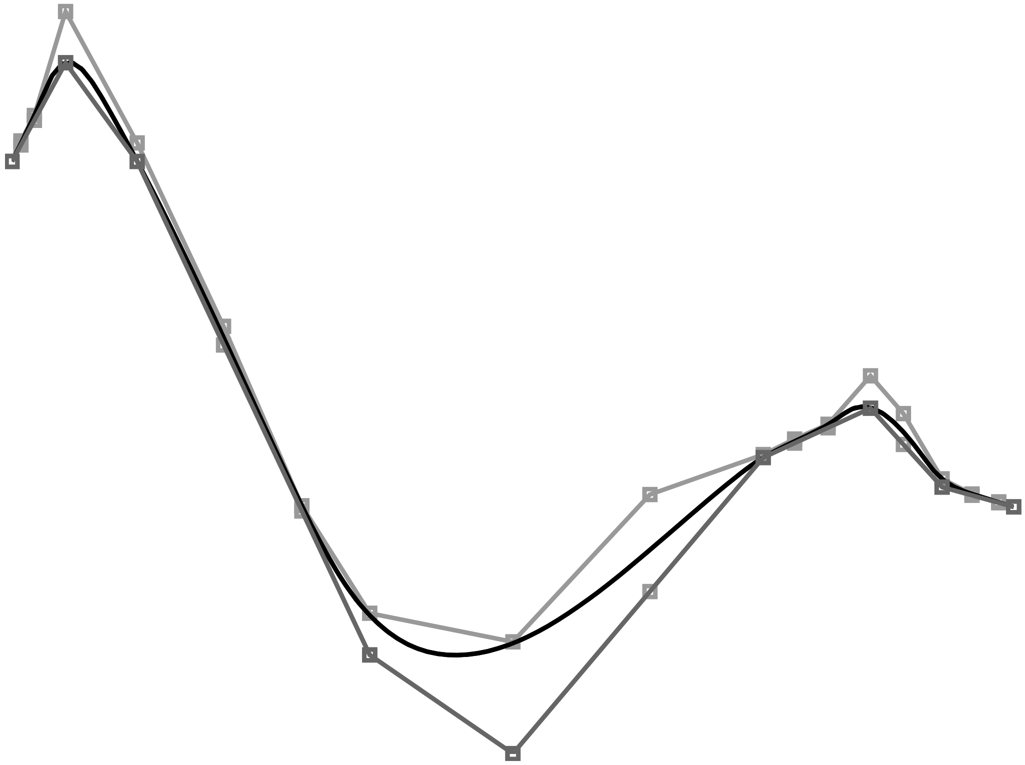


FIGURE 5.3. A solution to the SUPPORT problem. The constructed quadratic spline *(black)* lies above the input polygon *(dark grey)*. The lower part of the envelope *(light grey)* is not visible since it matches the input polygon exactly.

knots into an initial interpolant and optimizes the location of the additional knots according to the necessary conditions for the solution. Oberle and Opfer [**20**] extends these results to natural quintic splines.

5.2. Solving SUPPORT and CHANNEL

With the piecewise linear envelope for splines derived in Section 3.2 on page 14, SUPPORT and CHANNEL can be formulated as feasibility problems with linear constraints. Specifically, let a knot sequence $t_0 \leq t_1 \leq \cdots \leq t_{m+d}$ for a spline of degree d with $m+1$ control points be given and recall that $t_i^* = \sum_{i+1}^{i+d} t_i/d$ is the abscissa of the i–th control point (see Algorithm 1 below for a default choice). The constraints for SUPPORT ensure that the lower envelope of the spline stays above the input polygon. The constraints for CHANNEL ensure that the envelope of the spline stays between the input polygons.

5.2.1. Constraints.

The lower constraints $\underline{\mathcal{C}}$ on the $2m$ unknowns b_i and $\underline{w}_i$ are

(1) For $1 \leq i \leq m-1$ bound the variable $\underline{w}_i$, which is a placeholder for $\min\{0, D_i b\}$, by

$$\underline{w}_i \leq 0$$
$$\underline{w}_i \leq D_i b.$$

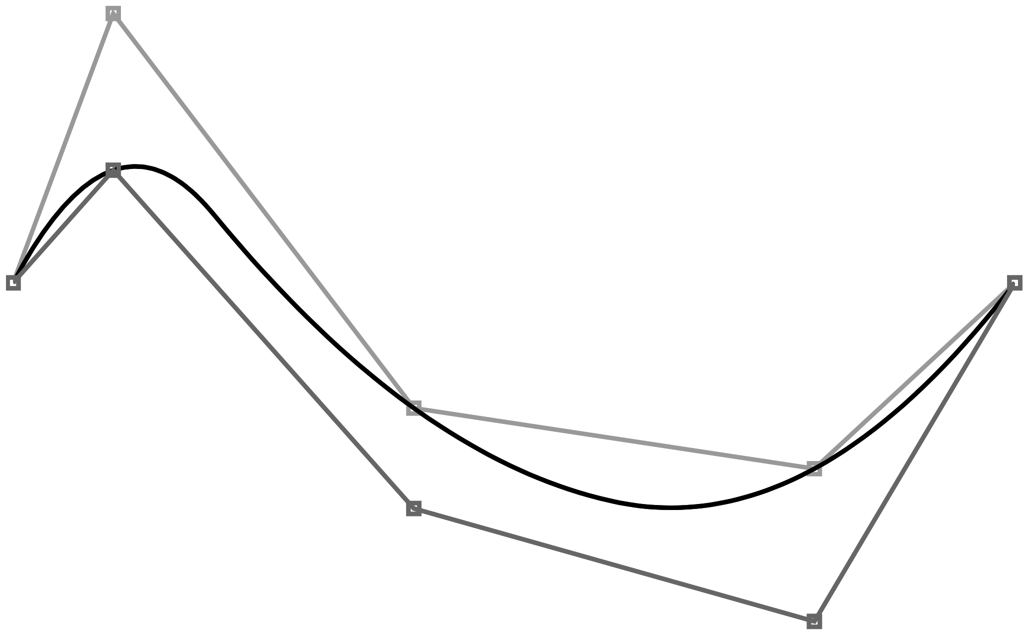


FIGURE 5.4. A solution to the SUPPORT problem. The constructed cubic spline *(black)* lies above the input polygon *(dark grey)*. The envelope *(light grey)* matches the input polygon in its lower part.

(2) For $0 \leq k \leq m$ ensure that all breakpoints of the lower envelope $\underline{b}$ of the spline stay above the lower polygon $\underline{a}$,

$$\underline{a}(t_k^*) \leq \underline{b}(t_k^*) = b_k + \sum_{i \in \mathcal{I}^* k} v_{i,k} \underline{w}_l.$$

(3) For each concave corner $\underline{a}_j$ of the input polygon ensure that the corner lies below the linear envelope,

$$\underline{a}_j \leq \underline{b}(\underline{x}_j).$$

The constraints $\overline{\mathcal{C}}$ are obtained from $\underline{\mathcal{C}}$ by replacing all inequality signs $\leq$ by $\geq$, all underlines by overlines, and concave corners by convex corners in (3).

5.2.2. Solving SUPPORT. Given the at most $3m - 1 + \underline{n}$ constraints $\underline{\mathcal{C}}$, the optimization goal is to keep the control polygon of b as close as possible to $\underline{a}$. The resulting cost function is

$$\sum_{i=0}^{m} (b_i - \underline{a}(t_i^*)) \to \min . \tag{5.2.17}$$

In this formulation, SUPPORT can be solved by a standard solver for linear programs. Figures 5.3 and 5.4 show two examples of SUPPORT and their solutions.

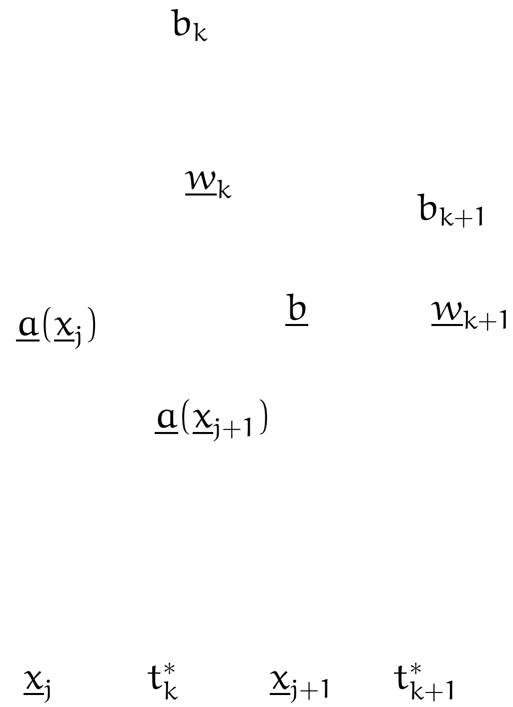


FIGURE 5.5. Variables of the constraint formulation. Input polygon *(thick)*, lower envelope *(dash-dotted)* and control polygon *(thin)*.

5.2.3. Solving CHANNEL. There are at most $6m - 2 + \underline{n} + \overline{n}$ constraints $\underline{\mathcal{C}}$ and $\overline{\mathcal{C}}$, and we can choose from a variety of optimization goals. To keep b centered between $\underline{a}$ and $\overline{a}$, i.e. $\overline{a}_i - b_i = b_i - \underline{a}_i$ for each i, we use the cost function

$$\sum_{i=0}^{m} \overline{a}_i - 2b_i + \underline{a}_i \to \min . \tag{5.2.18}$$

In this formulation, CHANNEL can be solved by a standard solver for linear programs.

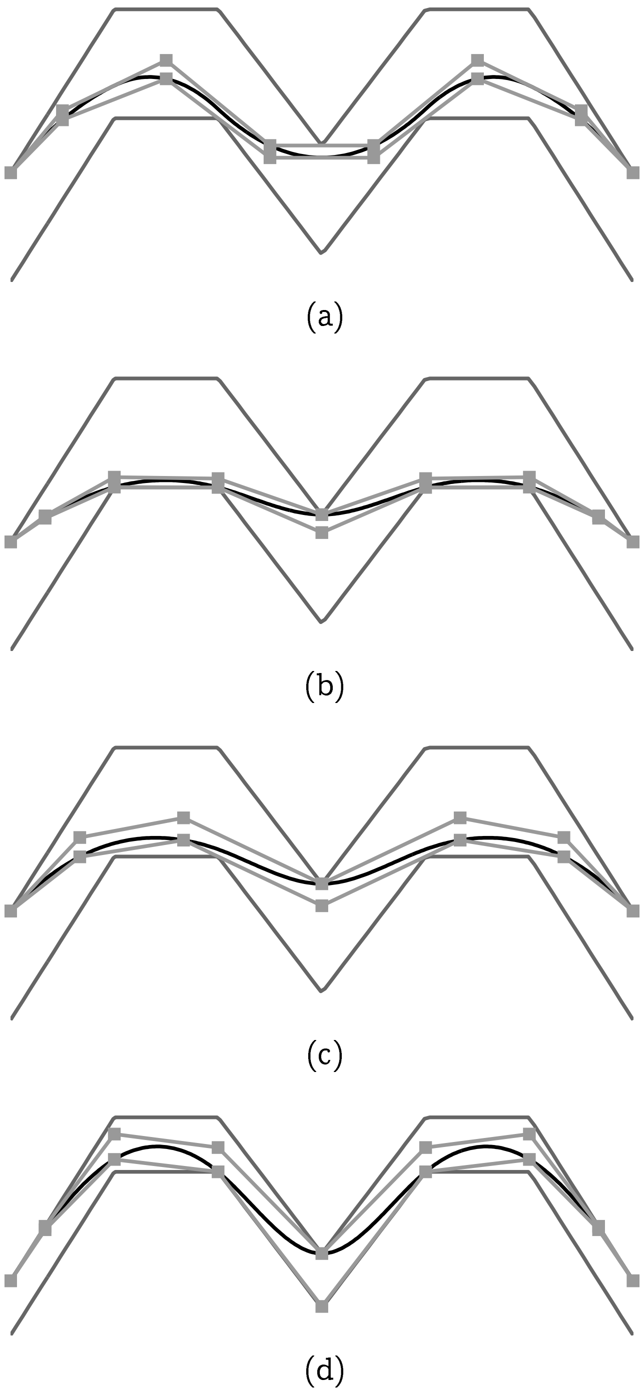
To keep b as close as possible to a straight line, for which $\Delta_2 b_j = 0$, we use the cost function

$$\sum_{i=1}^{m-1} (D_i b)^2 \to \min . \tag{5.2.19}$$

This results in a quadratic program that again can be solved by a standard solver.

5.2.4. Initial knot sequence. The user may choose the initial non-decreasing knot sequence freely, for example as the abscissae of the corners of $\underline{a}$. Algorithm 1 below generates a short knot sequence (t_i) such that the corresponding sequence of Greville abscissae include the breakpoints x_i of the input polygon(s). The idea is to introduce additional knots (and Greville abscissae) where the input sequence (x_i) contains gaps that are large relative to the surrounding intervals.

5.2.5. Knot insertion. While we are assured of a solution to SUPPORT, for example a straight line above $\underline{a}$, CHANNEL may not have a solution for the given requirements of



(a)

(b)

(c)

(d)

FIGURE 5.6. Solutions to a CHANNEL problem: the upper input polygon is a constant offset of the lower input polygon. The offset amount is 1 for (a)–(c) and 1/2 for (d). The solution is quadratic for (a) and cubic for (b)–(d). For (a) and (c), the breaks of the control polygon do *not* coincide with the breaks of the input polygons, while they do coincide for (b) and (d).

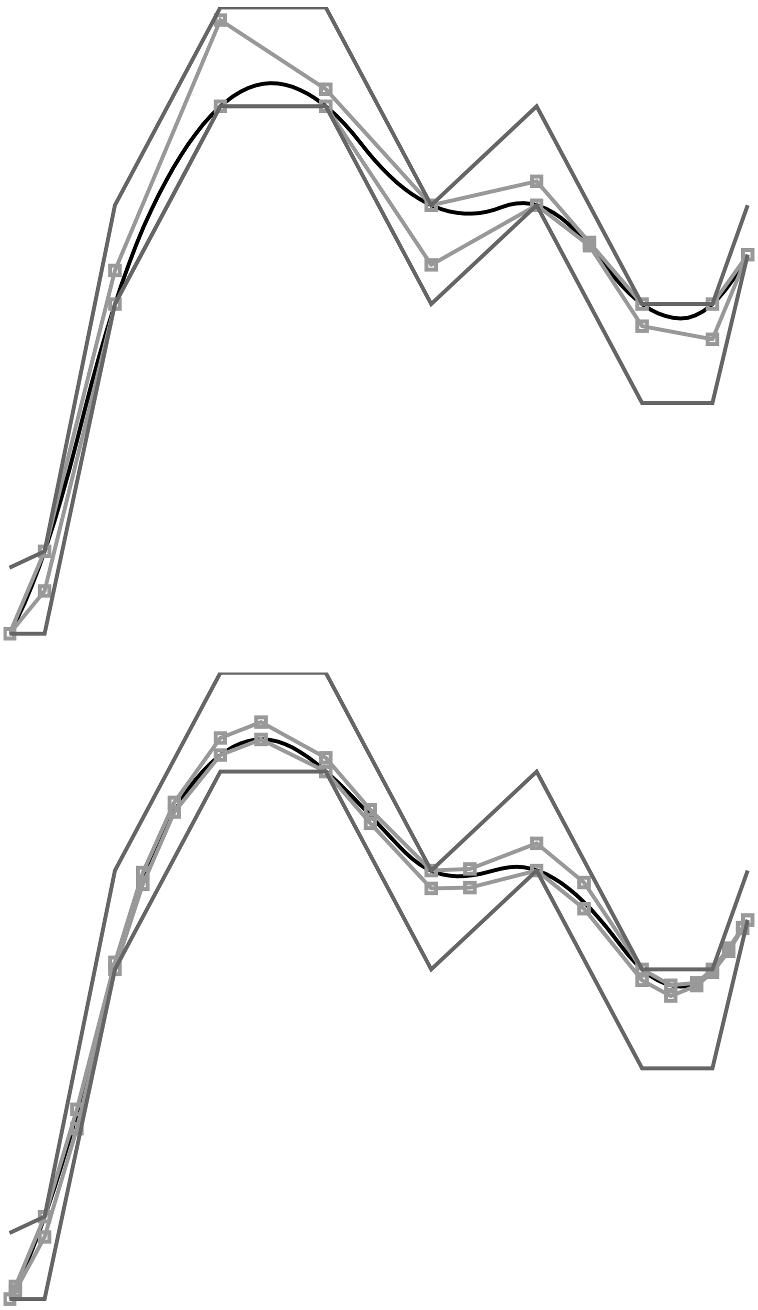


FIGURE 5.7. A tangent continuous quadratic *(top)* and a curvature continuous cubic *(bottom)* solution to a CHANNEL problem. The input polygons $\underline{a}$ and $\overline{a}$ *(dark grey)* enclose the splines envelope *(light grey)* and hence the spline *(black)*.

degree, smoothness and break points. This difficulty is overcome by inserting knots where the channel constraints are violated. Concretely, we augment the knot sequence (t_i) with additional breaks near the breaks of the input polygons where the envelope returned by the linear or quadratic program violates the CHANNEL conditions. This adaptive strategy is

Algorithm 1 Compute an interpolating knot sequence (t_i) for a B–spline basis of degree d such that the Greville abscissae contain the sequence $x_0 < x_1 < \cdots < x_m$.

```
Require: $x_0 < x_1 < \cdots < x_m$ and $d > 1$
  $t_0 = t_1 = \cdots = t_d \leftarrow x_0$
  $k \leftarrow d$
  for $l \leftarrow 1 \dots m-1$ do
    repeat
      $t_{k+1} \leftarrow d\,x_l - \sum_{j=k-d+2}^{k} t_j$
      if $t_{k+1} > x_{l+1}$ then        -- New knot too big, add intermediate Greville abscissae
        $t_{k+1} \leftarrow x_l - \min\{x_{l+1} - x_l, x_l - t_k\}/2$
      $k \leftarrow k+1$
    until $d\,x_l = \sum_{j=k-d+1}^{k} t_j$
  $t_{k+1}, \dots, t_{k+d} \leftarrow x_m$
Ensure: $(t_k)$ interpolating knot sequence
Ensure: For any $0 \le i \le m$ exists a $k \ge i$ with $x_i = \sum_{j=k+1}^{k+d} t_j$
```

guaranteed to yield a solution since the envelope of b converges to b under refinement of the knot sequence. A rigorous proof of finite termination inserts knots equidistantly so that the simple bound for uniform splines, $|b - Lb| \le (d+1)/24 \max |D_i b|$, applies which contracts by $1/4$ as the knot distance is halved.

Example. Figure 5.7 shows two solutions to a Channel problem. Both were computed using the quadratic cost function above and the knot sequence generated by Algorithm 1. For the quadratic spline, the control polygon has only one more break than the input polygon, while the cubic spline has twice as many control points as there are points in the input polygon. △

5.3. Symbolic refinement

The solutions to Support and Channel described in the last section are not the best possible: the linearization of the problem can causes errors in the approximation that depend on the distance between the spline and its envelope. The distance, and therefore the approximation error, is reduced by refining the input knot sequence (t_i) to a finer knot sequence $(\hat{t}_i)$. At the same time, the number of control points of the solution $\hat{b}$ over the refined knot sequence $(\hat{t}_i)$ increases with the length of $(\hat{t}_i)$.

This section describes how the approximation error between the solution b to the linearized problem and the solution $b^\star$ of a continuously constrained problem over the fixed knot sequence (t_i) can be reduced without increasing the number of control points. This is achieved by modifying the programs for Support$[\hat{t}]$ or Channel$[\hat{t}]$ with the subdivision rules that map a spline over the original knot sequence (t_i) to the refined knot sequence $(\hat{t}_i)$. As a consequence, the solutions to Support$[\hat{t}]$ and Channel$[\hat{t}]$ converge to the solution

$b^\star$ of a continuous problem as the knot spacing in $(\hat{t}_i)$ converges to zero as explained in Section 5.4.

5.3.1. Modifying the programs. Let S be the subdivision matrix that maps the control vector b of a spline over the coarse knot sequence (t_i) to the control vector $\hat{b}$ of the same spline over the refined knot sequence $(\hat{t}_i)$,

$$\hat{b} = Sb.$$

The programs for SUPPORT$[\hat{t}]$ or CHANNEL$[\hat{t}]$ are expressed in terms of the variables $\hat{y}$,

$$\hat{y} = \begin{bmatrix} \hat{b} \\ w \end{bmatrix} \qquad w = \begin{cases} \underline{w} & \text{for SUPPORT} \\ \begin{bmatrix} \underline{w}^T & \overline{w}^T \end{bmatrix}^T & \text{for CHANNEL.} \end{cases}$$

To obtain a program in terms of the control points b of a spline over the coarse knot sequence (t_i), we substitute the variables $\hat{y}$ by variables y with

$$\hat{y} = \begin{bmatrix} S & 0 \\ 0 & I \end{bmatrix} y \qquad \text{where } y = \begin{bmatrix} b \\ w \end{bmatrix}$$

and 0 and I are zero and identity matrices of appropriate sizes. Both y and $\hat{y}$ contain the same variables w, which are placeholders for the minimum of the second differences of the refined control points $\hat{b}$ and 0 and depend nonlinearly on the control points b. They can therefore not be replaced by a linear function of the analogous expressions in the control points b over the coarse knot sequence.

A linear program to solve SUPPORT$[\hat{t}]$ or CHANNEL$[\hat{t}]$ can be written as

$$c^T\hat{y} \to \min \qquad \text{subject to } A\,\hat{y} \le f.$$

for a suitable matrix A and some vector f. After substituting y for $\hat{y}$, the new linear program that yields a spline over (t_i) is

$$c^T \begin{bmatrix} S & 0 \\ 0 & I \end{bmatrix} y \to \min \qquad \text{subject to } A \begin{bmatrix} S & 0 \\ 0 & I \end{bmatrix} y \le f.$$

Quadratic programs can be rewritten analogously.

We refer to the solutions of the modified programs in y as SUPPORT$[t\,|\,\hat{t}]$ and CHANNEL$[t\,|\,\hat{t}]$. The solutions are splines over the knot sequence (t_i) which are optimal with respect to the linearized constraints over the refined knot sequence $(\hat{t}_i)$. In the limit, as the modified problems are solved for finer and finer knot sequences $(\hat{t}_i)$, the solutions b of one of these problems converge to the solution $b^\star$ of the continuously constrained problem.

5.4. Constrained approximation

The algorithms explained in the last section can be used to approximate solutions to continuously constrained approximation problems under certain conditions. Specifically, let $\mathcal{A}$ be a function space for which we have piecewise linear envelopes, for example the space of cubic interpolatory subdivision functions introduced in Section 3.7, and for which we can refine the envelopes $\underline{a}$ and $\overline{a}$ of a function $a \in \mathcal{A}$ so that for any $\varepsilon > 0$ there is a subdivided representation a' of a such that the envelopes $\underline{a'}$ and $\overline{a'}$ of a' lie within ε, $\overline{a'} - \underline{a'} \leq \varepsilon$.

We can now use $\textsc{Support}[\,t\,|\,\hat{t}\,]$ to approximate solutions to the one–sided L_1 approximation problem

$$\int_{t_0}^{t_{m+d}} b(t) - a(t)\,dt \to \min \qquad \text{subject to } b \geq a \tag{5.4.20}$$

where b ranges over all splines of degree d over the knot sequence (t_i). The idea is very simple: for the fixed function $a \in \mathcal{A}$, compute a suitably refined upper envelope $\overline{a'}$ and solve $\textsc{Support}[\,t\,|\,(t_i)\,]$ for a refinement $(\hat{t}_i)$ of the knot sequence (t_i) with the cost function

$$\int_{t_0}^{t_{m+d}} b(t)\,dt = \sum_{i=0}^{m} \frac{t_{i+d+1} - t_i}{d+1} b_i\,dt \to \min .$$

For a solution b to this problem let $\overline{\hat{b}}$ be the upper envelope of $\hat{b}$, the refinement of the control polygon of b over the knot sequence $(\hat{t}_i)$. The approximation error between b and a is bounded by the difference $\overline{\hat{b}} - \underline{a'}$ of the upper envelope $\overline{\hat{b}}$ and the lower envelope $\underline{a'}$ of the refinement a' of a.

The approximation error can be further reduced by repeating the above process for ever finer knot sequences $(\hat{t}_i)$ and ever finer refinements a' of a. Since both envelopes converge to the respective functions under repeated refinement, the solutions of the $\textsc{Support}$ problems converge to the solution $b^\star$ of the continuously constrained problem (5.4.20).

We can use $\textsc{Channel}$ in a similar fashion to construct functions b that lie between two non–intersecting input functions $a_1, a_2 \in \mathcal{A}$ with $a_1 < a_2$, for example the two–sided problem of finding a spline b of degree d over the knot sequence (t_i) with

$$\int_{t_0}^{t_{m+d}} (b''(t))^2\,dt \to \min \qquad \text{subject to } a_1 \leq b \leq a_2. \tag{5.4.21}$$

The cost function for this problem is quadratic and can be written as $b^T Q b \to \min$ where Q is the positive definite symmetric matrix with entries

$$Q_{i,j} = \int_{t_0}^{t_{m+d}} N_i^{d-2}(t) N_j^{d-2}(t)\,dt \qquad 0 \leq i,j \leq d-2.$$

The solutions to this continuously constrained approximation problem can again be approximated by solving $\textsc{Channel}[\,t\,|\,\hat{t}\,]$ for refinements $(\hat{t}_i)$ of (t_i) and with the upper envelope $\overline{a'_1}$ of a refinement a'_1 of a_1 and the lower envelope $\underline{a'_2}$ of a refinement of a_2 as input polygons. This requires solving the linearly constrained problem

$$b^T Q b \to \min \qquad \text{subject to } \overline{a'_1} \leq \underline{b}\,,\ \overline{b} \leq \underline{a'_2} \tag{5.4.22}$$

where $\underline{b} \leq b \leq \overline{b}$ are the refined envelopes of b.

Again, the solutions b of the linearized problem (5.4.22) converge to the solution $b^\star$ of the continuously constrained problem (5.4.21) as the knot sequence $(\hat{t}_i)$ and the refinements a'_1 and a'_2 of a_1 and a_2 are progressively refined because of the convergence of the piecewise linear envelopes to the functions they envelop.

CHAPTER 6

Conclusions

We have described a new approach to constructing envelopes of nonlinear geometry. The approach is used to construct piecewise linear envelopes for many practically important bases such as univariate B-splines and tensor product polynomials. These envelopes serve, although useful in their own right, as examples that demonstrate the versatility of our approach. Our approach yields envelopes that are, in general, tighter than previously known envelopes.

The general framework laid out in Chapter 2 is very useful in generating envelopes of nonlinear geometry that are efficiently computable in practice. The main computational burden is shifted to a precomputation phase that need only be carried out once for each basis used to represent nonlinear geometry.

The construction is very general in that it assumes little about the basis in which the nonlinear geometry is represented. In particular, the construction assumes neither that the basis fulfills the convex hull property nor that it is of a given smoothness class. The construction also requires only a loose coupling between a geometry object b and its simplification Lb and does not require, for example, that the simplification Lb interpolates b. The simplification operator L can therefore be chosen based on computational convenience, for example as a simpler object that is already used to represent b such as its control polygon.

We have shown in Section 3.7 on page 26 that the refinability of a basis can be exploited to compute tight estimates on an antidifference basis even in the absence of a convex hull property or an evaluation algorithm. We showed in Section 3.7.2 that Lb does not need to be a linearization of b, but can be of higher order, such as the piecewise cubic envelopes we computed for cubic interpolatory subdivision. This can be used, for example, to convert one function b given in one basis to two functions $\underline{b}$ and $\overline{b}$ in another basis that enclose b and gives a simple algorithm for basis conversions in a CAD system based on interval arithmetic: the interval associated with the conversion Lb of b evaluated at t is $[\underline{b}(t), \overline{b}(t)]$. Using higher–order envelopes instead of piecewise linear ones has the advantage that Lb will still be smooth.

An important feature of the piecewise linear and bilinear envelopes constructed in Chapters 3 and 4 is that they have the same structure as the underlying control polygons. This eases the construction of piecewise linear or bilinear envelopes for parametric curves or surfaces considerably. For adaptive subdivision of a curve or surface, it is enough to apply the bounds at the control points, enclosing them in axis aligned boxes, and basing the decision whether further subdivision is needed solely on the sizes of these boxes. This is considerably simpler than constructing local convex hulls.

The structural simplicity of the envelopes makes it possible to solve inverse problems as described in Chapter 5. The linear programs used to approximate continuously constrained optimization problems are sparse with a small bandwidth. Because of the similarity between the envelopes for bivariate functions and those for univariate functions, the constructions from Chapter 5 are easily extended to bivariate functions and can be used, for example, for the one–sided smoothing of bivariate piecewise linear functions such as height fields.

One important area of further investigation is that of computing tight enclosures $\lfloor \mathsf{K} \rfloor$ and $\lceil \mathsf{K} \rceil$ of the antidifference bases K, in particular for bivariate functions such as tensor product polynomials and bivariate Bernstein polynomials. The cost of the precomputation phase could be reduced greatly if a more direct algorithm for computing $\lfloor \mathsf{K} \rfloor$ and $\lceil \mathsf{K} \rceil$ than the subdivide–and–bound procedure were available.

A second area of further investigation are generalized subdivision surfaces, a representation of nonlinear geometry that has become very popular in recent years. Most of these representations lack envelopes and therefore satisfactory intersection algorithms.

Bibliography

[1] R. C. Buck. Applications of duality in approximation theory. In *Approximation of Functions (Proceedings of the Symposium on Approximation of Functions, General Motors Research Laboratories, Warren, Michigan, 1964. Ed. Henry L. Garabedian)*, pages 27–42. Elsevier Publ. Co., Amsterdam, 1965.

[2] Elaine Cohen and Larry L. Schumaker. Rates of convergence of control polygons. *Computer Aided Geometric Design*, 2(1-3):229–235, 1985. Surfaces in CAGD '84 (Oberwolfach, 1984).

[3] Wolfgang Dahmen. Subdivision algorithms converge quadratically. *Journal of Computational and Applied Mathematics*, 16(2):145–158, 1986.

[4] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf. *Computational geometry*. Springer-Verlag, Berlin, 1997.

[5] Carl de Boor. B-form basics. In Gerald E. Farin, editor, *Geometric modeling*, pages 131–148. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, 1987.

[6] Carl de Boor. B(asic)–spline basics. In *Fundamental Developments of Computer Aided Geometric Modeling*, pages 27–49. Academic Press, London, 1993. ed. Les Piegl.

[7] R. DeVore. One-sided approximation of functions. *Journal of Approximation Theory*, 1:11–25, 1968.

[8] D. Dobkin, E.R. Gansner, E. Koutsofios, and S.C. North. A path router for graph drawing. In *Proceedings of the 14th Annual Symposium on Computational Geometry*, pages 415–416. ACM Press, New York, 1998. June 1-10 1998, Minneapolis, Minnesota.

[9] Serge Dubuc. Interpolation through an iterative scheme. *Journal of Mathematical Analysis and Applications*, 114(1):185–204, 1986.

[10] Gerald Farin. Tighter convex hulls for rational Bézier curves. *Computer Aided Geometric Design*, 10:123–125, 1993.

[11] Gerald Farin. *Curves and surfaces for computer-aided geometric design*. Computer Science and Scientific Computing. Academic Press Inc., San Diego, California, fourth edition, 1997.

[12] Rida T. Farouki. Direct surface section evaluation. In *Geometric modeling*, pages 319–334. SIAM, Philadelphia, Pennsylvania, 1987.

[13] Daniel Filip, Robert Magedson, and Robert Markot. Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design*, 3(4):295–311, 1986.

[14] Ronald N. Goldman and Tony D. DeRose. Recursive subdivision without the convex hull property. *Computer Aided Geometric Design*, 3(4):247–265 (1987), 1986.

[15] Josef Hoschek and Dieter Lasser. *Computer Aided Geometric Design*. A.K. Peters, 1989.

[16] Leif P. Kobbelt, Katja Daubert, and Hans-Peter Seidel. Ray tracing of subdivision surfaces. In *Rendering Techniques '98 (Proceedings of the Eurographics Workshop)*, pages 69–80, New York, June 1998. Springer-Verlag.

[17] P. A. Koparkar and S. P. Mudur. A new class of algorithms for the processing of parametric curves. *Computer Aided Design*, 15:41–45, 1983.

[18] O. L. Mangasarian and L. L. Schumaker. Splines via optimal control. In *Approximations with Special Emphasis on Spline Functions (Proceedings of a Symposium Conducted by the Mathematics Research Center, United States Army, at the University of Wisconsin, Madison, Wisconsin, 1969)*, pages 119–156. Academic Press, New York, 1969.

[19] D. Nairn, J. Peters, and D. Lutterkort. Sharp, quantitative bounds on the distance between a polynomial piece and its Bézier control polygon. *Computer Aided Geometric Design*, 16(7):613–631, 1999.

[20] H. J. Oberle and G. Opfer. Nonnegative splines, in particular of degree five. *Numerische Mathematik*, 79(3):427–450, 1998.

[21] Gerhard Opfer and Hans Joachim Oberle. The derivation of cubic splines with obstacles by methods of optimization and optimal control. *Numerische Mathematik*, 52(1):17–31, 1988.

[22] Allan M. Pinkus. *On* L^1*-approximation*. Cambridge University Press, Cambridge, 1989.

[23] Hartmut Prautzsch and Leif Kobbelt. Convergence of subdivision and degree elevation. *Advances in Computational Mathematics*, 2(1):143–154, 1994.

[24] Ulrich Reif. Personal communication, November 1999.

[25] Ulrich Reif. Best bounds on the approximation of polynomials and splines by their control polygon. Technical report, Universität Stuttgart, February 1999.

[26] Thomas Sauer. Multivariate Bernstein polynomials and convexity. *Computer Aided Geometric Design*, 8(6):465–478, 1991.

[27] Thomas W. Sederberg, Scott C. White, and Alan K. Zundel. Fat arcs: A bounding region with cubic convergence. *Computer Aided Geometric Design*, 6:205–218, 1989.