Generalized Lane–Riesenfeld algorithms

Thomas J. Cashman[a], Kai Hormann[a], Ulrich Reif[b]

[a] *Università della Svizzera italiana, Via Giuseppe Buffi 13, CH-6904 Lugano*
[b] *TU Darmstadt, Schlossgartenstrasse 7, D-64289 Darmstadt*

Abstract

The Lane–Riesenfeld algorithm for generating uniform B-splines provides a prototype for subdivision algorithms that use a refine and smooth factorization to gain arbitrarily high smoothness through efficient local rules. In this paper we generalize this algorithm by maintaining the key property that the same operator is used to define the refine and each smoothing stage. For the Lane–Riesenfeld algorithm this operator samples a linear polynomial, and therefore the algorithm preserves only linear polynomials in the functional setting, and straight lines in the geometric setting. We present two new families of schemes that extend this set of invariants: one which preserves cubic polynomials, and another which preserves circles. For both generalizations, as for the Lane–Riesenfeld algorithm, a greater number of smoothing stages gives smoother curves, and only local rules are required for an implementation.

1. Introduction

In the functional setting, the Lane–Riesenfeld algorithm (Lane and Riesenfeld, 1980) is an efficient method for subdividing uniform B-splines. It acts on an input sequence $f = (f_i)_{i\in\mathbb{Z}}$ by using a refine stage $\mathbf{R}$ followed by k smoothing stages $\mathbf{S}$. A subdivision step $\mathbf{T}$ is therefore

$$\mathbf{T} = \mathbf{S}^k\mathbf{R},$$

where $\mathbf{R} \coloneqq \mathbf{R}_L$ is defined as

$$(\mathbf{R}_L f)_{2i} = f_i, \tag{1}$$

$$(\mathbf{R}_L f)_{2i+1} = \frac{1}{2}f_i + \frac{1}{2}f_{i+1}, \tag{2}$$

and $\mathbf{S} \coloneqq \mathbf{S}_L$ is the smoothing operator

$$(\mathbf{S}_L f)_i = \frac{1}{2}f_i + \frac{1}{2}f_{i+1}. \tag{3}$$

If f provides B-spline coefficients for a spline s of degree $k+1$ with parameter values in $\mathbb{Z}$, then the Lane–Riesenfeld algorithm gives the coefficients $\mathbf{T}f$ of s on the refined grid

$$\mathbf{T}\mathbb{Z} = \begin{cases} \frac{1}{2}\mathbb{Z}, & \text{if } k \text{ is even,} \\ \frac{1}{2}\mathbb{Z} + \frac{1}{4}, & \text{if } k \text{ is odd.} \end{cases}$$

Iterating this process gives a piecewise linear function that interpolates $\mathbf{T}^\ell f$ at the abscissae $\mathbf{T}^\ell\mathbb{Z}$ for any level $\ell \in \mathbb{N}$. The sequence of these functions converges to s as $\ell \to \infty$ (Lane and Riesenfeld, 1980), and the Lane–Riesenfeld algorithm is therefore a way of *generating* uniform arbitrary-degree B-splines through subdivision, while using the efficient, highly-local operations (1–3). Increasing k gives a spline of higher degree, and hence a limit function with greater smoothness.

Email addresses: `tom.cashman@cantab.net` (Thomas J. Cashman), `kai.hormann@usi.ch` (Kai Hormann), `reif@mathematik.tu-darmstadt.de` (Ulrich Reif)

In this paper, we observe that the Lane–Riesenfeld refine and smooth stages both sample a local linear interpolant in (2) and (3); the only difference is that the refine stage $\mathbf{R}_L$ retains the existing sequence in (1), whereas each smoothing stage $\mathbf{S}_L$ discards it. We can therefore consider generalized Lane–Riesenfeld algorithms by replacing both instances of this linear sampling rule with an alternative smoothing operator (see Section 2). This idea makes it easy to design schemes that have a given set of *invariants*: if the chosen smoothing operator preserves a class of shapes, such as cubic polynomials or circles, then each member of the class is also preserved in the limit. Like the original Lane–Riesenfeld algorithm, this gives rise to a whole family of subdivision schemes with the same set of invariants, by varying the number of smoothing stages in each subdivision step.

As examples of this concept, we present a linear algorithm based on sampling cubics in Section 4, and a geometric variant based on sampling circles in Section 5. Both generalizations share the property with the Lane–Riesenfeld algorithm that using more smoothing stages increases the smoothness of the limit curve. The same behaviour is observed by Schaefer et al. (2008), who suggest replacing the linear smoothing operator by a non-linear averaging rule. For example, taking the geometric mean of f_i and f_{i+1} in (2) and (3) instead of the arithmetic mean gives a subdivision scheme that preserves exponential functions. The analysis of these non-linear Lane–Riesenfeld algorithms can be carried out for a wide class of averaging rules (Goldman et al., 2009; Dyn and Goldman, 2011), but depends crucially on the fact that the averaging uses a two-point rule. By contrast, in this paper we consider four-point rules, and in general our approach does not assume any restrictions on the support of the smoothing operator.

2. Generalizing the Lane–Riesenfeld algorithm

We start by establishing the framework used to generalize the Lane–Riesenfeld algorithm in the rest of the paper. We employ a notation that avoids the extensive use of indices, by writing the data for a subdivision scheme as a pair of sequences F. In the functional setting, $F = (u, f) \in \mathbb{R}^{\mathbb{Z}} \times \mathbb{R}^{\mathbb{Z}}$ for some uniform grid $u = (u_i)_{i\in\mathbb{Z}}$, and linear operators distribute over the pair, as in $\mathbf{T}F = (\mathbf{T}u, \mathbf{T}f)$. Given that F associates each data value f_i with its abscissa u_i, we are ready to define the *invariants* of a subdivision scheme.

Definition 1. Given an operator $\mathbf{A}$, a polynomial p is in the operator's set of *invariants* $\mathcal{I}[\mathbf{A}]$ if p interpolates $\mathbf{A}F$ whenever p interpolates F. That is, $p \in \mathcal{I}[\mathbf{A}]$ whenever $f = p(u)$ implies $p(\mathbf{A}u) = \mathbf{A}f$ for any uniform grid u.

In generalizing the Lane–Riesenfeld algorithm, we want to guarantee that invariants of the smoothing operator are also invariants of the whole subdivision step. We therefore define the refine stage in terms of the smoothing stage by making use of a *zipper* operator $\mathbf{Z}(f, g)$, which interleaves two sequences such that

$$\begin{aligned}(\mathbf{Z}(f,g))_{2i} &= f_i,\\ (\mathbf{Z}(f,g))_{2i+1} &= g_i.\end{aligned}$$

The main contribution of this paper is that by defining a refine stage to introduce points in the same way as each smoothing stage, we obtain a family of subdivision schemes which preserve the invariants of the smoothing operator.

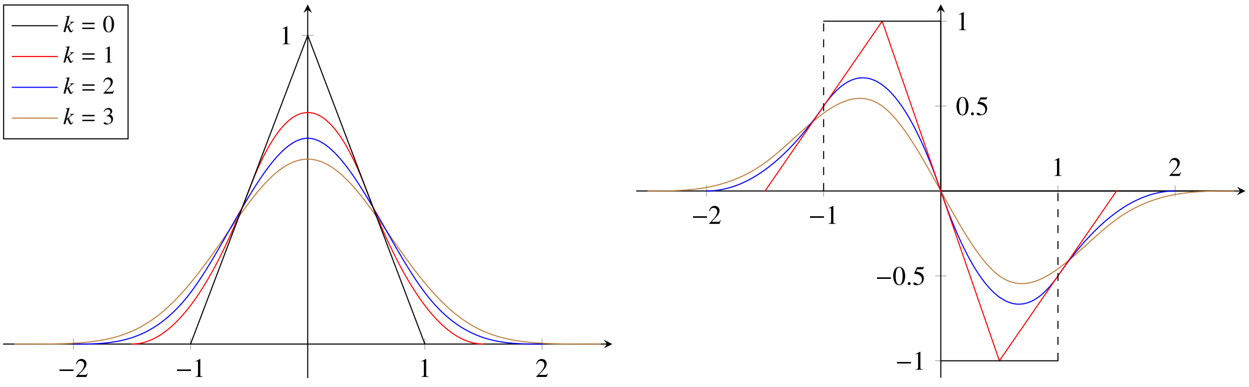
Theorem 2. *For any smoothing operator* $\mathbf{S}$*, using the refine stage* $\mathbf{R} = \mathbf{Z}(\,\cdot\,, \mathbf{S}\,\cdot\,)$ *in the subdivision step* $\mathbf{T} = \mathbf{S}^k\mathbf{R}$ *implies* $\mathcal{I}[\mathbf{S}] \subseteq \mathcal{I}[\mathbf{T}]$.

Proof. Given some $p \in \mathcal{I}[\mathbf{S}]$ and any uniform grid u, we can set $f = p(u)$ and thus have $\mathbf{S}f = p(\mathbf{S}u)$ by Definition 1. That is, p interpolates both (u, f) and $(\mathbf{S}u, \mathbf{S}f)$. Moreover, as

$$\mathbf{R}F = (\mathbf{R}u, \mathbf{R}f) = (\mathbf{Z}(u, \mathbf{S}u), \mathbf{Z}(f, \mathbf{S}f)),$$

p also interpolates $\mathbf{R}F$ and so $p \in \mathcal{I}[\mathbf{R}]$. Since $\mathbf{T} = \mathbf{S}^k\mathbf{R}$, it follows that $p \in \mathcal{I}[\mathbf{T}]$. □

For convenience, we identify $\mathbf{T}^\ell F$ with the piecewise linear function that interpolates the data $\mathbf{T}^\ell f$ at the abscissae $\mathbf{T}^\ell u$. Then, since $p \in \mathcal{I}[\mathbf{T}]$ is interpolated after each subdivision step, the limit $\mathbf{T}^\infty F = \lim_{\ell\to\infty} \mathbf{T}^\ell F$ must be equal to p if the subdivision scheme converges. The set of invariants $\mathcal{I}[\mathbf{T}]$ is therefore the set of polynomials that are *reproduced* by the scheme: if the initial data F is constructed by sampling values f from a polynomial in $\mathcal{I}[\mathbf{T}]$ at u, then the subdivision algorithm recreates that same polynomial in the limit.



(a) Basis function value $\phi(x)$.

(b) Basis function derivative $\mathrm{d}\phi(x)/\mathrm{d}x$.

Figure 1: Plots of basis functions for the uniform B-splines generated by the L-schemes with $k = 0, 1, 2$ and 3.

In the rest of the paper, we make use of Theorem 2 by choosing three different smoothing operators $\mathbf{S}$, and then setting $\mathbf{R}F = \mathbf{Z}(F, \mathbf{S}F)$ to refine data using the same operator. A subdivision step acting on data F is then

$$\mathbf{T}F = \mathbf{S}^k\mathbf{Z}(F, \mathbf{S}F). \tag{4}$$

Notice that the Lane–Riesenfeld algorithm follows this pattern, as $\mathbf{R}_L$ and $\mathbf{S}_L$ use the same rule to introduce new points in (2) and (3), and so $\mathbf{T}_L F = \mathbf{S}_L^k \mathbf{R}_L F = \mathbf{S}_L^k \mathbf{Z}(F, \mathbf{S}_L F)$. This means that the Lane–Riesenfeld algorithm can be specified using just the smoothing operator $\mathbf{S}_L$, and since $\mathbf{S}_L$ is based on local *linear* interpolation, we refer to the subdivision schemes created by the Lane–Riesenfeld algorithm as the *L-schemes*.

3. The *L*-schemes

Before generalizing the Lane–Riesenfeld algorithm by introducing new smoothing operators in (4), we recall the main properties of the L-schemes for comparison to later sections.

3.1. Invariants

Let π_d be the set of polynomials of degree at most d. Then it is well known that the linear polynomials π_1 are reproduced by the L-schemes. Our framework confirms this because linear polynomials are invariant under $\mathbf{S}_L$ and therefore $\pi_1 \subseteq \mathcal{I}[\mathbf{T}_L]$ by Theorem 2. However, $\mathcal{I}[\mathbf{T}_L]$ includes no polynomials of higher order (Dyn et al., 2008) and so $\mathcal{I}[\mathbf{T}_L] = \pi_1$.

3.2. Support

For a uniform linear subdivision step $\mathbf{T}$, such as (4) with a linear operator $\mathbf{S}$, the limit $\mathbf{T}^\infty F$ is a weighted sum of translates of a single *basis function*. The basis function ϕ is defined as the limit $\mathbf{T}^\infty\Delta$, where $\Delta = (\mathbb{Z}, (\delta_{i,0})_{i\in\mathbb{Z}})$ is the Kronecker delta sequence on $\mathbb{Z}$.

A uniform B-spline created by the L-scheme with k smoothing stages has a basis function with support width $k + 2$. Each additional smoothing stage increases the support width by 1. This increase is minimal for a non-trivial smoothing stage, and results from the fact that each Lane–Riesenfeld smoothing stage (3) uses just two old values to determine a new smoothed value.

Figure 1 plots basis functions for the L-schemes using 0, 1, 2 and 3 smoothing stages in each subdivision step. A large number of smoothing stages results in a basis function with an influence which is concentrated around a small part of the domain. For a designer editing a closed curve by moving control points, the visible influence of a point is governed by this smaller ‘effective support’, which is significantly less than the full support width $k + 2$. We can

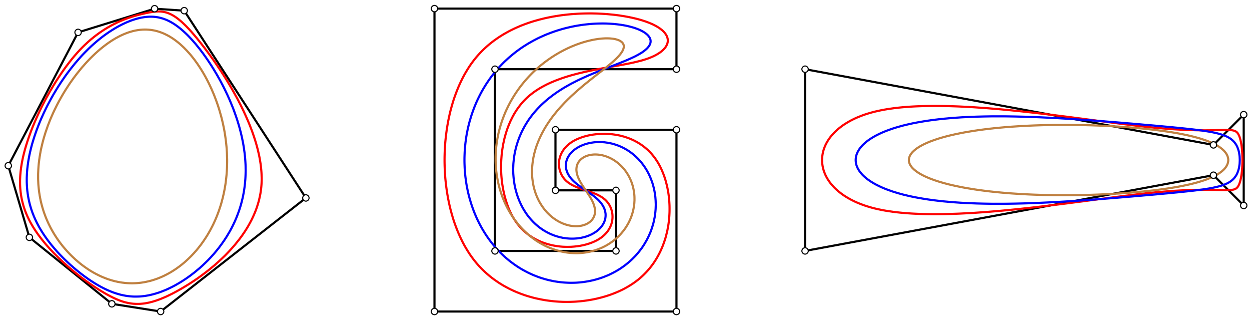


Figure 2: Example limit curves generated by the L-schemes, using 0 (black), 2 (red), 4 (blue) and 8 (brown) smoothing stages per subdivision step.

measure the effective support by taking into account the value of a basis function over its domain. For a threshold τ, we let $\sigma_L(\tau)$ be the minimal value such that

$$|\phi(u)| \leq \tau \quad \text{for all } |u| > \sigma_L(\tau)/2.$$

When $\tau = 0$, this gives the usual measure of support width $\sigma_L(0) = k + 2$. By choosing slightly higher values for τ, however, we can evaluate the region of influence with more sensitivity, ignoring any areas of the domain where a basis function has near-zero value. Figure 4a shows $\sigma_L(\tau)$ for the uniform B-splines at four different thresholds τ.

3.3. Smoothness

A convenient way to analyse the smoothness of a subdivision scheme is by using its *symbol*. The symbol corresponding to the subdivision operator $\mathbf{T}$ is the Laurent polynomial

$$t(z) = \sum_{i \in \mathbb{Z}} t_i z^i$$

with coefficients $(t_i)_{i\in\mathbb{Z}} = \mathbf{T}(\delta_{j,0})_{j\in\mathbb{Z}}$. According to Rioul (1992) and Dyn and Levin (2002), if $m \in \mathbb{N}$ is maximal such that

$$t(z) = \left(\frac{1+z}{2}\right)^m r(z)$$

for some Laurent polynomial $r(z)$, then the Hölder regularity of the limit functions generated by the scheme is bounded from below by

$$m - \log_2 \|r\|\,.$$

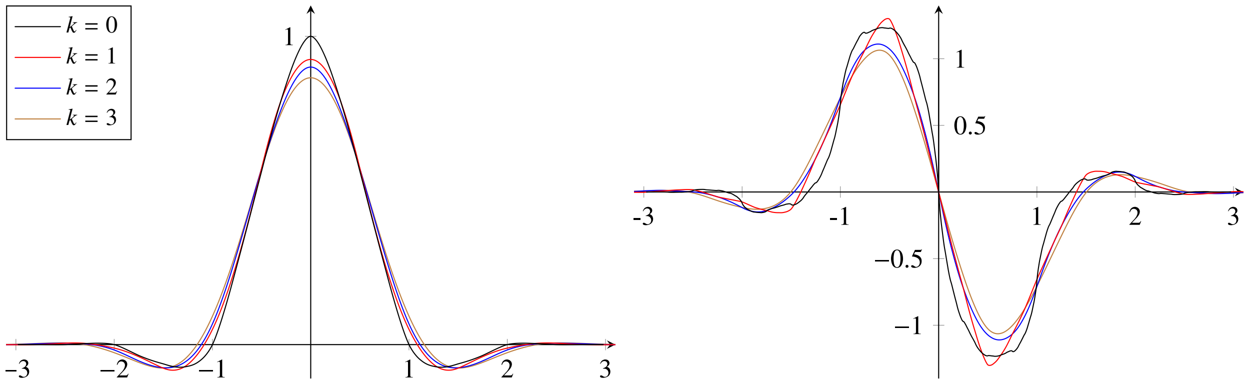
Here the norm of $r(z) = \sum_{i\in\mathbb{Z}} r_i z^i$ is defined as

$$\|r\| = \max\left\{\sum_{i\in\mathbb{Z}} |r_{2i}|\,, \sum_{i\in\mathbb{Z}} |r_{2i+1}|\right\}.$$

As the symbol of the L-schemes is

$$2\left(\frac{1+z}{2}\right)^{k+2},$$

the Hölder regularity of the generated limit functions is $k + 1$. This means that they are in $C^{k+1-\epsilon}$, which agrees with the fact that the L-scheme with k smoothing stages generates B-splines of degree $k + 1$.



(a) Basis function value $\phi(x)$.

(b) Basis function derivative $\mathrm{d}\phi(x)/\mathrm{d}x$.

Figure 3: Plots of basis functions for the C-schemes with $k = 0, 1, 2$ and 3.

3.4. Shrinkage

Although we introduced the Lane–Riesenfeld algorithm in the functional setting, it is also commonly used to create uniform B-spline *curves*. In this geometric setting, we apply the subdivision operator $\mathbf{T}$ to the x- and y-coordinates of a closed polygon independently. Each subdivision step doubles the number of points, and the initial polygon is known as the *control polygon*.

Figure 2 shows a few example limit curves produced by the L-schemes. We observe that the final limit curve has a greater distance from the control polygon as the number of smoothing stages increases. The refine stage does not contribute to this 'shrinkage' effect, which we can therefore analyse by examining just repeated smoothing stages. For a closed polygon with n points, a matrix representation of the Lane–Riesenfeld smoothing stage (3) is

$$S_L = \frac{1}{2}\begin{pmatrix} 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 1 & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ 1 & 0 & 0 & 0 & \dots & 0 & 1 \end{pmatrix} \in \mathbb{R}^{n\times n}.$$

As S_L is a circulant matrix, it directly follows (Gray, 2006) that its eigenvalues are

$$\lambda_j = \frac{1}{2}(1 + \exp(2\pi \mathrm{i} j/n)), \quad j = 0, 1, \dots, n-1.$$

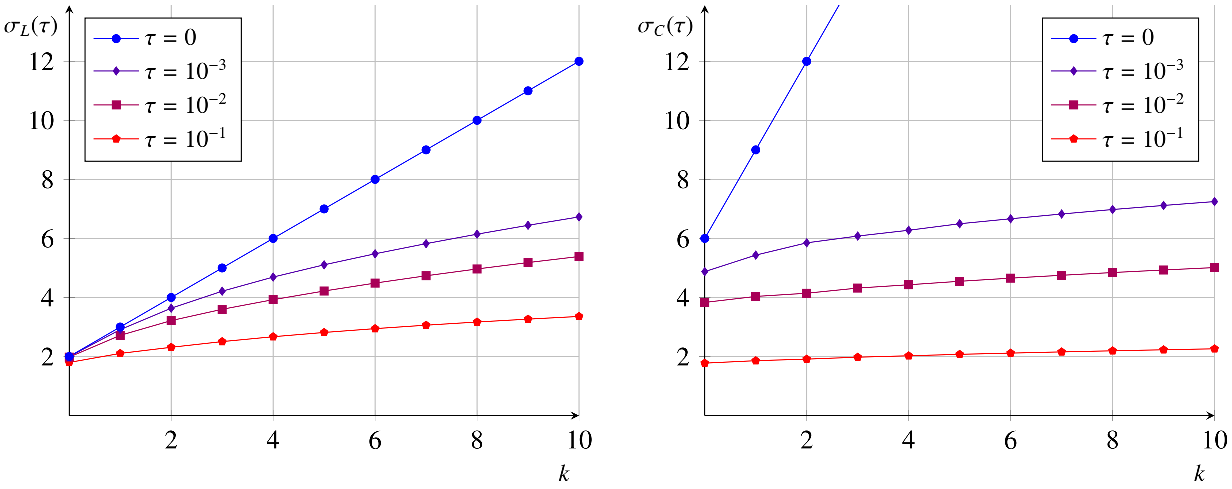
This shows a dominant eigenvalue $\lambda_0 = 1$, and since the rows of S_L have unit sum, it follows that $S_L\mathbf{1} = \mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^{n\times 1}$ is the column vector filled with ones. Therefore $\mathbf{1}$ is the eigenvector associated with the dominant eigenvalue, and this shows that the dominant behaviour, if we use an infinite number of smoothing stages, is for the polygon to collapse to a single point at its barycentre.

The subdominant eigenvalues λ_1 and λ_{n-1} are complex conjugate and they encode the shrinkage of the polygon towards its barycentre as well as a phase shift. Ignoring this rotation, the rate of shrinkage for each smoothing stage is $|\lambda_1| = |\lambda_{n-1}|$. We plot this rate against k in Figure 6.

4. Generalization based on sampling cubics: the *C*-schemes

We can view the Lane–Riesenfeld smoothing operator $\mathbf{S}_L$ in (3) as sampling a local linear interpolant at the midpoint between every two adjacent values. The four-point scheme (Dubuc, 1986) works in a similar way, but samples a local *cubic* interpolant to every four adjacent values instead. This gives the smoothing operator

$$(\mathbf{S}_C f)_i = \frac{1}{16}(-f_{i-1} + 9f_i + 9f_{i+1} - f_{i+2}), \tag{5}$$



(a) Effective support for the L-schemes.

(b) Effective support for the C-schemes.

Figure 4: Effective support for the L- and C-schemes at several different thresholds τ.

which we can substitute for $\mathbf{S}$ in (4) to create a generalized Lane–Riesenfeld algorithm. The new algorithm uses the refine stage $\mathbf{R}_C = \mathbf{Z}(\,\cdot\,, \mathbf{S}_C\,\cdot\,)$, that is,

$$\begin{aligned} (\mathbf{R}_C f)_{2i} &= f_i, \\ (\mathbf{R}_C f)_{2i+1} &= \frac{1}{16}(-f_{i-1} + 9f_i + 9f_{i+1} - f_{i+2}). \end{aligned} \tag{6}$$

This refine stage is followed by k smoothing stages $\mathbf{S}_C$, hence the subdivision operator is $\mathbf{T}_C = \mathbf{S}_C^k\mathbf{Z}(\,\cdot\,, \mathbf{S}_C\,\cdot\,)$. We call this new family of generalized Lane–Riesenfeld algorithms the C-schemes, and the family member with $k = 0$ smoothing stages is the original four-point scheme. Figure 3 shows the basis functions of the first few C-schemes, and Table 1 summarizes their properties.

4.1. Invariants

By construction, $\mathcal{I}[\mathbf{S}_C]$ includes all cubic polynomials π_3, which implies $\pi_3 \subseteq \mathcal{I}[\mathbf{T}_C]$ as a result of Theorem 2. As with the L-schemes, no higher-degree polynomials are reproduced by these schemes (Dyn et al., 2008) and so $\mathcal{I}[\mathbf{T}_C] = \pi_3$.

4.2. Support

The cubic-sampling rule (5) uses four adjacent values to construct a new one, whereas the original Lane–Riesenfeld rule (3) uses only two. The result is that the support width, which is $3k + 6$ for k smoothing stages, grows much faster for the C-schemes than for the L-schemes. This is shown by the plot for $\tau = 0$ in Figure 4b compared to Figure 4a. However, Figure 4 also shows that our measure of effective support is very similar for both the L-schemes and the C-schemes, where we define σ_C in the same way as σ_L except using the C-scheme basis functions ϕ. In fact, the range where the basis function value is greater than $\tau = 0.1$ is even more compact for the C-scheme generalization than for the B-splines, that is $\sigma_C(0.1) < \sigma_L(0.1)$ for all k.

4.3. Smoothness

Table 1 shows that additional smoothing stages $\mathbf{S}_C$ do not increase the smoothness of the limit curves by as much as $\mathbf{S}_L$, which increases the level of derivative continuity by 1. Nevertheless, there is a lower bound on the smoothness of the curves produced by the C-schemes which grows steadily with k. The bound relies on the fact that the symbols of the C-schemes have factors whose coefficients alternate in sign.

Definition 3. We say that the symbol $p(z)$ is *alternating*, if its coefficients have alternating signs, that is

$$p(z) = \sum_{i\in\mathbb{Z}} a_i z^{2i} - \sum_{i\in\mathbb{Z}} b_i z^{2i+1} \tag{7}$$

with $a_i, b_i \geq 0$ for all $i \in \mathbb{Z}$.

In other words, the symbol $p(z) = \sum_{i\in\mathbb{Z}} p_i z^i$ is alternating if and only if its *even* coefficients p_{2i} are non-negative and its *odd* coefficients p_{2i+1} are non-positive.

Lemma 4. *The product of two alternating symbols is alternating, too.*

Proof. Let the two symbols

$$p(z) = \sum_{i\in\mathbb{Z}} a_i z^{2i} - \sum_{i\in\mathbb{Z}} b_i z^{2i+1} \qquad \text{and} \qquad q(z) = \sum_{i\in\mathbb{Z}} c_i z^{2i} - \sum_{i\in\mathbb{Z}} d_i z^{2i+1}$$

be alternating, so that all coefficients a_i, b_i, c_i, d_i are non-negative. It is then clear that the product of both symbols

$$\begin{aligned} r(z) = p(z)q(z) &= \sum_{i,j\in\mathbb{Z}} a_i c_j z^{2(i+j)} - \sum_{i,j\in\mathbb{Z}} a_i d_j z^{2(i+j)+1} - \sum_{i,j\in\mathbb{Z}} b_i c_j z^{2(i+j)+1} + \sum_{i,j\in\mathbb{Z}} b_i d_j z^{2(i+j)+2} \\ &= \sum_{j\in\mathbb{Z}} \alpha_j z^{2j} - \sum_{j\in\mathbb{Z}} \beta_j z^{2j+1} \end{aligned}$$

with non-negative coefficients

$$\alpha_j = \sum_{i\in\mathbb{Z}} (a_i c_{j-i} + b_i d_{j-i-1}) \qquad \text{and} \qquad \beta_j = \sum_{i\in\mathbb{Z}} (a_i d_{j-i} + b_i c_{j-i}) \tag{8}$$

is alternating as well. □

While the expressions for the even and odd coefficients of the product of two alternating symbols in (8) are rather complicated, their respective sums satisfy a remarkably simple relation.

Definition 5. Given a symbol $p(z) = \sum_{i\in\mathbb{Z}} p_i z^i$, we denote the sum of its even and its odd coefficients by

$$p^* = \sum_{i\in\mathbb{Z}} p_{2i} \qquad \text{and} \qquad p_* = \sum_{i\in\mathbb{Z}} p_{2i+1},$$

respectively.

For an alternating symbol $p(z)$ as in (7) we have

$$p^* = \sum_{i\in\mathbb{Z}} a_i = \frac{p(1) + p(-1)}{2} \qquad \text{and} \qquad p_* = -\sum_{i\in\mathbb{Z}} b_i = \frac{p(1) - p(-1)}{2}. \tag{9}$$

Lemma 6. *Let $p(z)$ and $q(z)$ be two alternating symbols and $r(z) = p(z)q(z)$. Then,*

$$r^* = p^* q^* + p_* q_* \qquad \textit{and} \qquad r_* = p^* q_* + p_* q^*. \tag{10}$$

Proof. To prove the first statement, we first recall from Lemma 4 that $r(z)$ is an alternating symbol, and then use (9) for all three symbols $p(z)$, $q(z)$, and $r(z)$ to get

$$\begin{aligned} 4(p^* q^* + p_* q_*) &= [p(1) + p(-1)][q(1) + q(-1)] + [p(1) - p(-1)][q(1) - q(-1)] \\ &= 2p(1)q(1) + 2p(-1)q(-1) = 2r(1) + 2r(-1) = 4r^*. \end{aligned}$$

The second statement follows similarly. □

		Hölder regularity			
k	support width	lower bound (Theorem 7)	numerical lower bound	numerical upper bound	integer continuity class
0	6	2	2	2	C^1
1	9	2.541	2.622	2.625	C^2
2	12	3.046	3.232	3.239	C^3
3	15	3.524	3.833	3.844	C^3
4	18	3.983	4.425	4.441	C^4

Table 1: Smoothness and support properties of the C-schemes.

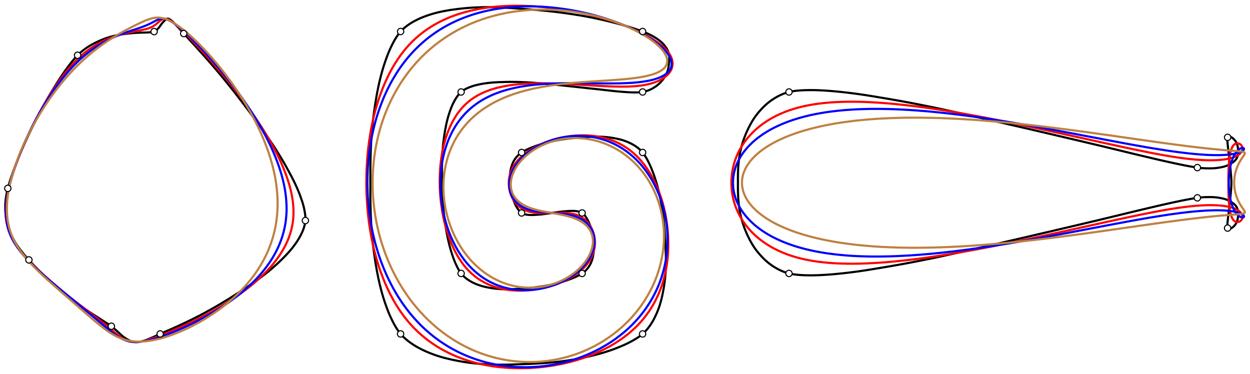


Figure 5: Example limit curves generated with the C-schemes, using 0 (black), 2 (red), 4 (blue) and 8 (brown) smoothing stages per subdivision step.

Note that the two identities in (10) can also be written compactly in matrix-vector form as

$$\begin{pmatrix} r^* \\ r_* \end{pmatrix} = \begin{pmatrix} p^* & p_* \\ p_* & p^* \end{pmatrix} \begin{pmatrix} q^* \\ q_* \end{pmatrix}. \tag{11}$$

Theorem 7. *The Hölder regularity of the limit curves generated by the C-scheme with k smoothing steps is bounded from below by*

$$k + 4 - \log_2(3(3/2)^k + 1).$$

Proof. If we write the symbols of the cubic smoothing operator $\mathbf{S}_C$ and the cubic refine stage $\mathbf{R}_C$ in their factored forms,

$$\frac{1+z}{2} \cdot \frac{-z^{-1} + 10 - z}{8} \qquad \text{and} \qquad \left(\frac{1+z}{2}\right)^4 (-z^{-1} + 4 - z),$$

then it is clear that the symbol of the C-scheme with k smoothing steps is

$$\left(\frac{1+z}{2}\right)^{k+4} r(z),$$

with

$$r(z) = p(z)^k q(z), \qquad p(z) = \frac{-z^{-1} + 10 - z}{8}, \qquad q(z) = -z^{-1} + 4 - z. \tag{12}$$

Therefore, the Hölder regularity of the limit curves is bounded from below by

$$k + 4 - \log_2 \|r\|.$$

By Lemma 4 we know that $r(z)$ in (12) is an alternating symbol, and so

$$\|r\| = \max(r^*, -r_*).$$

Using Lemma 6 and (11), we further have

$$\begin{aligned}
\begin{pmatrix} r^* \\ r_* \end{pmatrix} &= \begin{pmatrix} p^* & p_* \\ p_* & p^* \end{pmatrix}^k \begin{pmatrix} q^* \\ q_* \end{pmatrix} \\
&= \begin{pmatrix} 5/4 & -1/4 \\ -1/4 & 5/4 \end{pmatrix}^k \begin{pmatrix} 4 \\ -2 \end{pmatrix} \\
&= \frac{1}{2} \begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 3/2 & 0 \\ 0 & 1 \end{pmatrix}^k \begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 4 \\ -2 \end{pmatrix} \\
&= \begin{pmatrix} 3(3/2)^k + 1 \\ 1 - 3(3/2)^k \end{pmatrix},
\end{aligned}$$

so that

$$\|r\| = 3(3/2)^k + 1,$$

which completes the proof. □

Table 1 lists the lower bounds given by Theorem 7 for small k. We also give numerical approximations of the Hölder regularity, which we found by joint spectral radius analysis (Rioul, 1992). Notice that our theoretical lower bounds are some distance from the actual values.

4.4. Shrinkage

Figure 5 shows example curves created by the C-schemes using the same control polygons as in Figure 2. It is clear from these figures that the C-scheme limit curves pass much closer to the starting sequences f than the L-scheme limit curves shown in Figure 2. This is indicated by the fact that the value $\phi(0)$ of a basis function evaluated at the origin is closer to 1 for the C-schemes (see Figure 3a) than for the L-schemes (see Figure 1a).

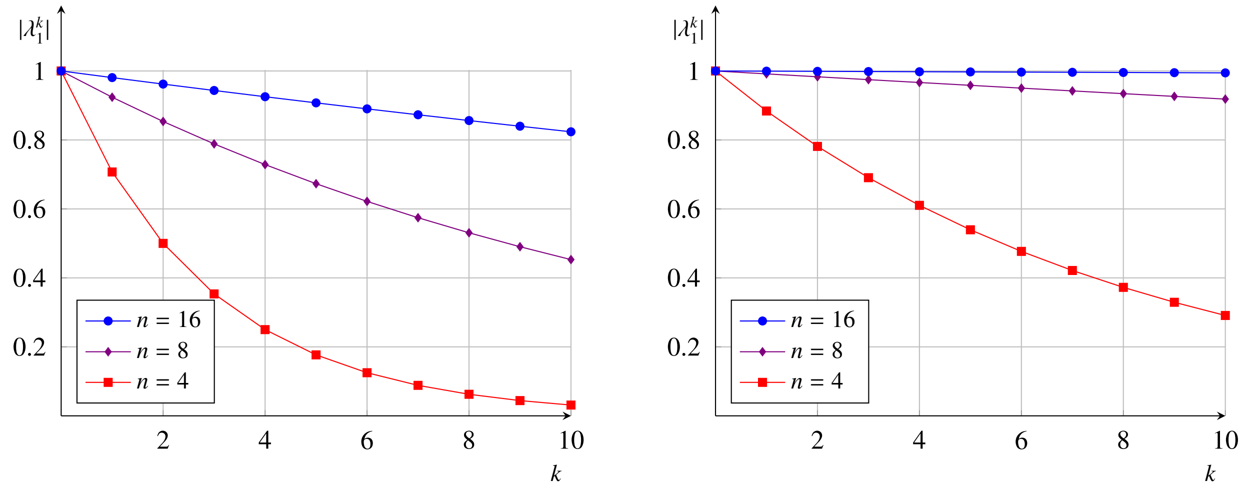
To quantify this shrinkage, we can examine a single smoothing stage as in Section 3.4. Writing the smoothing stage $\mathbf{S}_C$ for a closed n-point polygon as the matrix

$$S_C = \frac{1}{16} \begin{pmatrix} 9 & 9 & -1 & 0 & \dots & 0 & 0 & -1 \\ -1 & 9 & 9 & -1 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \ddots & & \vdots \\ -1 & 0 & 0 & 0 & \dots & -1 & 9 & 9 \\ 9 & -1 & 0 & 0 & \dots & 0 & -1 & 9 \end{pmatrix} \in \mathbb{R}^{n \times n},$$

we can again use the fact that S_C is a circulant matrix to conclude that its eigenvalues are

$$\lambda_j = \frac{1}{16}(-\exp(-2\pi \mathrm{i} j/n) + 9 + 9\exp(2\pi \mathrm{i} j/n) - \exp(4\pi \mathrm{i} j/n)), \quad j = 0, 1, \dots, n-1.$$

A dominant unit eigenvalue shows that, as for the L-schemes, letting $k \to \infty$ collapses the whole configuration to a single constant point. The corresponding left eigenvector is $\mathbf{1}^T$, and so again this limit is the barycentre of the original polygon (Sabin, 2010). However, plotting the magnitude of the subdominant eigenvalue in Figure 6 shows slower shrinkage for the C-schemes, particularly for high values of k. When $n = 8$ the C-schemes have $1 - |\lambda_1| < 0.01$, accounting for the limit curves which are extremely close to the control points in Figure 5 (left).



(a) Magnitude $|\lambda_1|$ of the subdominant eigenvalue for the L-scheme smoothing operator $\mathbf{S}_L$.

(b) Magnitude $|\lambda_1|$ of the subdominant eigenvalue for the C-scheme smoothing operator $\mathbf{S}_C$.

Figure 6: Smoothing operator shrinkage plotted for the L-schemes and the C-schemes against the number of smoothing stages k, for polygons containing 4, 8 and 16 points. The C-schemes suffer less shrinkage, and therefore better retain the shape of the control polygon as the number of smoothing stages increases.

5. Generalization based on sampling circles: the κ-schemes

Floater and Micchelli (1998) note that the rule (6) leads to the relation

$$\begin{aligned}\frac{(\mathbf{R}_C f)_{2i} - 2(\mathbf{R}_C f)_{2i+1} + (\mathbf{R}_C f)_{2i+2}}{1/4} &= \frac{f_i - \frac{1}{8}(-f_{i-1} + 9f_i + 9f_{i+1} - f_{i+2}) + f_{i+1}}{1/4} \\ &= 4f_i - \tfrac{1}{2}(-f_{i-1} + 9f_i + 9f_{i+1} - f_{i+2}) + 4f_{i+1} \\ &= \tfrac{1}{2}(f_{i-1} - f_i - f_{i+1} + f_{i+2}) \\ &= \tfrac{1}{2}(f_{i-1} - 2f_i + f_{i+1}) + \tfrac{1}{2}(f_i - 2f_{i+1} + f_{i+2}),\end{aligned} \tag{13}$$

that is, the refine stage $\mathbf{R}_C$ inserts points so that the new second divided difference at $(\mathbf{R}_C f)_{2i+1}$ is the mean of the old second divided differences at f_i and f_{i+1}.

Sabin and Dodgson (2004) use this observation to create a circle-preserving scheme by replacing second divided differences with curvature estimators. To describe their scheme, we adapt our notation to the geometric setting by redefining the initial data $F := (x, y) \in \mathbb{R}^{\mathbb{Z}} \times \mathbb{R}^{\mathbb{Z}}$ to be the x- and y-coordinates of the data instead of the abscissa-ordinate values we used in Section 2. Note that geometric schemes do not use linear operators, and so a subdivision step no longer distributes over the two components of F separately.

Now we can define the Sabin and Dodgson smoothing operator $\mathbf{S}_\kappa$, which positions a new point $(\mathbf{S}_\kappa F)_i$ by sampling a circle with the curvature which is the mean of the old curvatures at F_i and F_{i+1}. In other words, if $K(a, b, c)$ is the signed curvature of the circumcircle through the point triple (a, b, c), then

$$K(F_i, (\mathbf{S}_\kappa F)_i, F_{i+1}) = \tfrac{1}{2}(K(F_{i-1}, F_i, F_{i+1}) + K(F_i, F_{i+1}, F_{i+2})). \tag{14}$$

This gives a scheme which reproduces circles whenever the initial data F are positioned on a circle, even if they are unevenly spaced. Just as for the L-schemes and C-schemes, we can create a generalized Lane–Riesenfeld algorithm using this operator by substituting $\mathbf{S}_\kappa$ into (4). Naturally, we call the resulting family of schemes the κ-schemes. Figure 7 shows some examples of limit curves created by the κ-schemes.

Sabin and Dodgson (2004) use a special rule to even out the edge lengths in a polygon after subdivision, but we find that using multiple smoothing stages has the effect of smoothing the edge lengths anyway (see Figure 8), so it is not

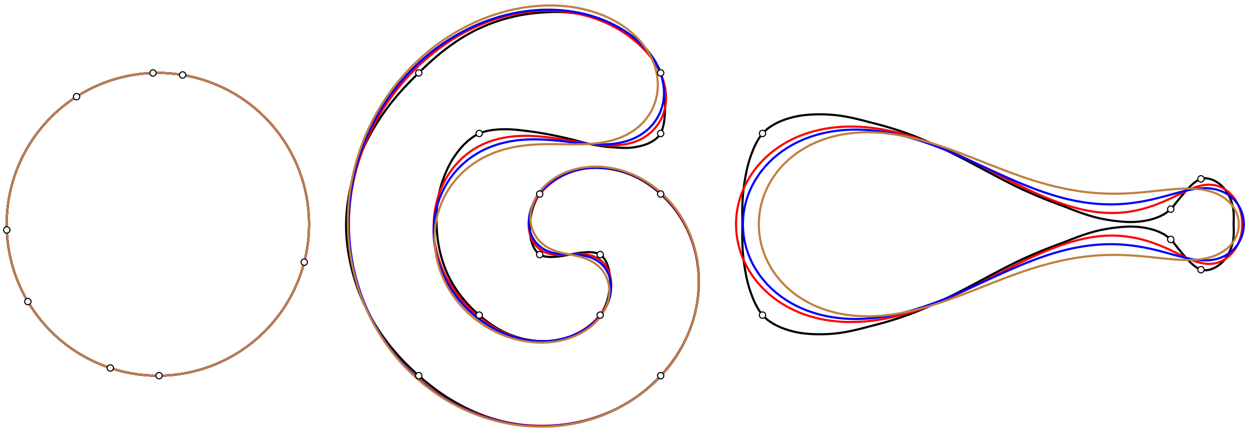


Figure 7: Example limit curves generated with the κ-schemes, using 0 (black), 2 (red), 4 (blue) and 8 (brown) smoothing stages per subdivision step.

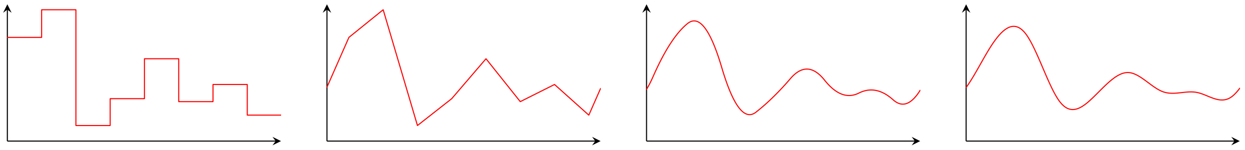


Figure 8: The edge lengths for the κ-schemes acting on the example shown in Figure 7 (left), plotted after six subdivision steps for $k = 0, 1, 2$ and 3 (from left to right). The edge lengths are smoothed out as k increases.

clear that this rule is necessary in our refine-and-smooth setting. We therefore define $(\mathbf{S}_\kappa F)_i$ by simply sampling at the midpoint of each circular arc between F_i and F_{i+1}. That is, we define $(\mathbf{S}_\kappa F)_i$ to be the unique point on the perpendicular bisector of the line $F_i F_{i+1}$ such that (14) holds, and therefore

$$(\mathbf{S}_\kappa F)_i = \frac{F_i + F_{i+1}}{2} + \frac{s}{1 + \sqrt{1 - s^2}} \left(\frac{F_{i+1} - F_i}{2} \right)^{\perp},$$

where $x^\perp$ denotes the vector x rotated counterclockwise by 90° and $s = (\sin\alpha + \sin\beta)/2$ is the arithmetic mean of the sines of the signed angles $\alpha = \sphericalangle(F_{i+1} - F_{i-1}, F_i - F_{i-1})$ and $\beta = \sphericalangle(F_{i+1} - F_{i+2}, F_i - F_{i+2})$.

5.1. Invariants

The eight points in Figure 7 (left) are sampled from a circle, and so every limit curve is the same circle, regardless of the value of k, as circles are an invariant of the smoothing operator $\mathbf{S}_\kappa$. We can formalize this statement with an analogue of Definition 1 for the geometric setting.

Definition 8. Given an operator $\mathbf{A}$ acting on the polygon $F = (x, y)$, a geometric primitive g is in the operator's set of *invariants* $\mathcal{I}[\mathbf{A}]$ if g interpolates $\mathbf{A}F$ whenever g interpolates F.

Note that Theorem 2 still applies with this alternative definition. This allows us to say that all circles are in $\mathcal{I}[\mathbf{S}_\kappa]$, including straight lines as circles through infinity, whereas the L-schemes and the C-schemes reproduce only straight lines as geometric objects in the sense of Definition 8.

5.2. Region of influence

As the κ-schemes are non-linear, the limit curves are not a linear combination of basis functions and so there is no support width that we can measure. However, the smoothing operator $\mathbf{S}_\kappa$ uses the same number of points to define $(\mathbf{S}_\kappa F)_i$ as $\mathbf{S}_C$: in both cases the antecedents are the four values F_{i-1} to F_{i+2}. So we know that the number of points $(\mathbf{T}^\ell F)_i$ on level ℓ that are influenced by any control point F_j is the same for both the κ-schemes and the C-schemes. Although we cannot measure the effective support, we can still say that the position of any point on the limit curve is influenced by at most $3k + 6$ control points for k smoothing stages.

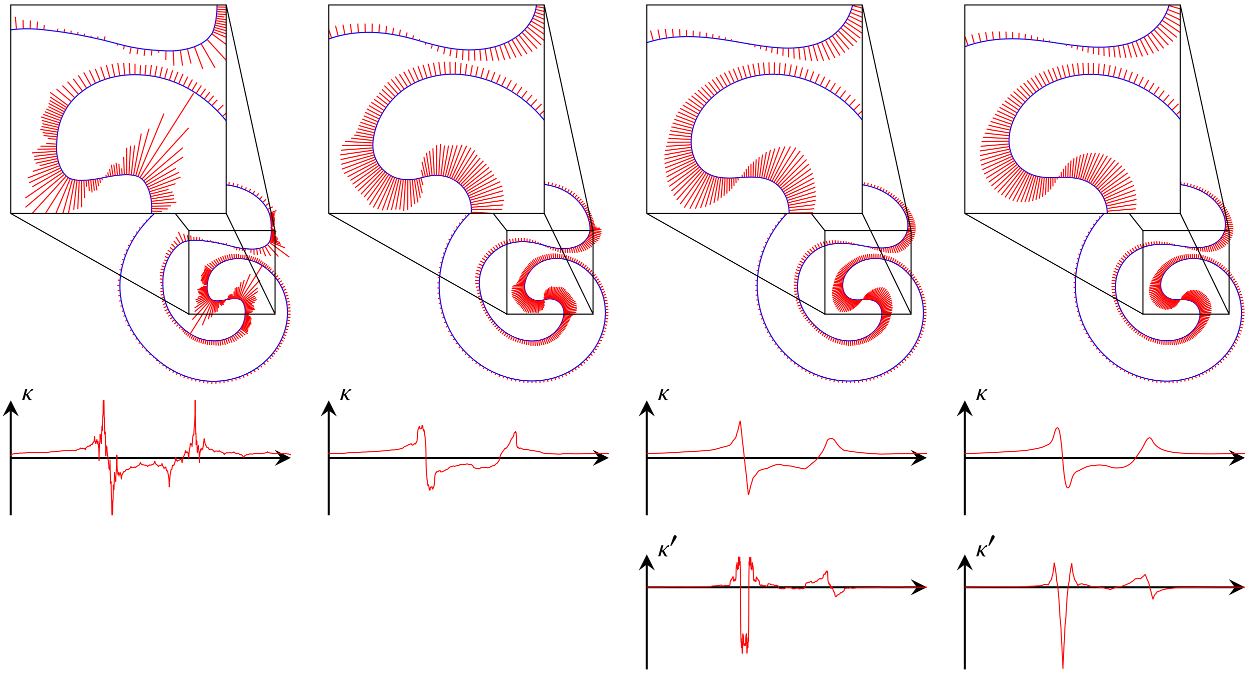


Figure 9: Curvature combs for the κ-schemes on the same starting data F, using $k = 0, 1, 2$ and 3 (from left to right). Underneath the combs we show signed curvature κ plotted against arc length and its derivative κ' (where it exists), after six subdivision steps. Note that curvature *diverges* for $k = 0$, since this scheme is a geometric analogue of the four-point scheme, which is only $C^{2-\epsilon}$. The first curvature plot is therefore only finite because we plot the value after a finite number of subdivision steps.

5.3. Smoothness

Figure 9 shows curvature combs for the κ-schemes with $k = 0, 1, 2$ and 3 on the same starting data. A curvature comb consists of a line drawn from each point F_i in the normal direction, with length equal to the curvature at that point. In addition, we show the signed curvature and its first and second derivative plotted against arc length. The plots show that the curvature has a fractal-like distribution when $k = 0$, indicating Sabin and Dodgson's result that this κ-scheme probably has the same Hölder continuity as the C-scheme when $k = 0$, namely $C^{2-\epsilon}$. However, the κ-scheme with $k = 1$ seems to have continuous curvature vectors, as we would expect from the corresponding C-scheme, and with more smoothing stages the combs become increasingly smooth.

5.4. Shrinkage

It is clear from comparing Figure 7 with Figures 2 and 5 that the κ-schemes have far less shrinkage than the L- or C-schemes. Although non-linear subdivision rules prevent us from analysing a single κ-scheme smoothing stage in the same way as in Sections 3.4 and 4.4, infinitely repeating the smoothing operator does not collapse a polygon to a single point, in general, as it does for the L- and C-schemes. Instead, many smoothing stages appear to move the data slowly towards a regular polygon, which we know is a fixed-point for the double smoothing stage $\mathbf{S}_\kappa^2$. Figure 10 demonstrates that even 100 smoothing stages may not be enough to reach a fixed configuration, as shown by the central example.

6. Conclusion

The key idea in this paper is that one same rule is used to both introduce new points in a refine stage and to perform each smoothing stage. Section 2 shows that the invariants of the smoothing rule are then invariants of the whole subdivision step too, and are hence reproduced by the subdivision scheme in the limit. Whereas the L-schemes reproduce linear polynomials, the C-schemes cubic polynomials and the κ-schemes circles, future generalizations might consider other function classes, such as rational or exponential functions, or other geometric objects, such as ellipses or hyperbolas.

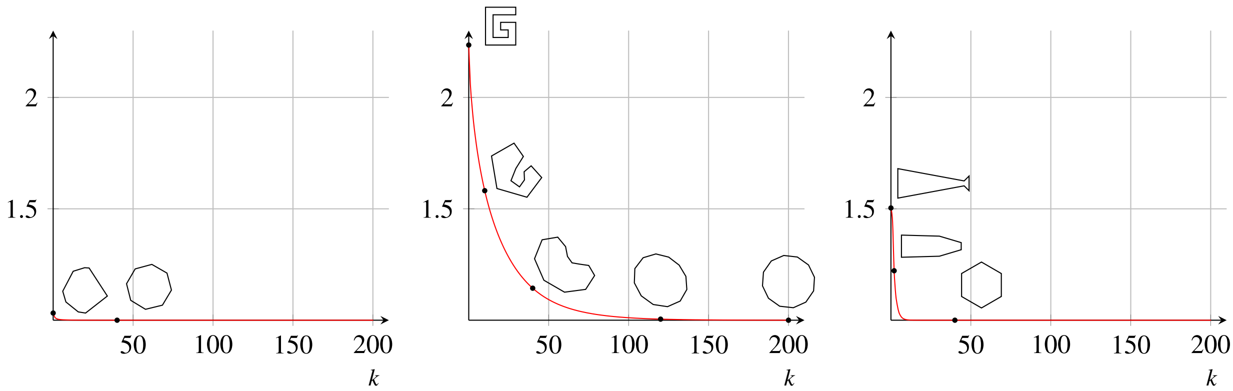


Figure 10: For the three initial polygons in Figure 7, we show the result of repeating a single smoothing stage without refinement by plotting $c/\sqrt{a}$, where c is the circumference of each polygon and a is the area, against the number of smoothing stages k. This value is minimal for a regular polygon, and each polygon is normalized such that the limiting value of $c/\sqrt{a}$ is 1.

Thus, while the C-schemes and κ-schemes are two examples of generalized Lane–Riesenfeld algorithms, there are many more possibilities for smoothing operators that we could introduce in (4). For example, the Dubuc–Deslauriers $2n$-point schemes (Deslauriers and Dubuc, 1989) allow us to create generalized Lane–Riesenfeld algorithms that reproduce polynomials of arbitrarily high order. In fact, any interpolatory subdivision scheme provides a smoothing operator that can be used in a generalized Lane–Riesenfeld algorithm. Such a scheme is recovered exactly when $k = 0$, which we can always use to guarantee interpolation of the input data, but greater values of k may give schemes of arbitrarily high smoothness. Our numerical results indicate that for the generalized Dubuc–Deslauriers schemes, the smoothness of the limit curves increases with the number of smoothing stages for any n, although the tools developed in Section 4.3 are insufficient to prove this hypothesis. Nevertheless, in the context of a Dubuc–Deslauriers scheme, this paper provides a way of *decoupling* the required smoothness from the order of the smoothing operator, for at least a wide range of target Hölder continuity values. Smoothing operators of lower order mean that the schemes can benefit from subdivision rules of lower complexity, but without sacrificing smoothness.

When choosing or designing a smoothing operator **S** for a generalized Lane–Riesenfeld algorithm, the following considerations should be taken into account. For efficiency reasons, the support of **S** should not be too large. In this paper we focus on four-point operators, while non-linear two-point operators are discussed in depth by Schaefer et al. (2008), Goldman et al. (2009) and Dyn and Goldman (2011). Moreover, as we usually want the resulting subdivision scheme to be symmetric, then **S** should also be symmetric. In the linear setting this implies that the corresponding symbol $s(z) = \sum_{i\in\mathbb{Z}} s_i z^i$ must have an even number of coefficients with $s_i = s_{1-i}$, $i \in \mathbb{Z}$, so that the symbol can be written as $s(z) = \frac{1+z}{2} p(z)$. We also need **S** to be affine, that is, its coefficients must add up to one. Then $s(1) = 1$ and the symbol $t(z) = s(z)^k r(z)$ of the overall subdivision scheme, where $r(z) = 1 + s(z^2)/z$ is the symbol of the refinement operator, satisfies $t(1) = 2$ and $t(-1) = 0$, which are the necessary conditions for convergence (Dyn and Levin, 2002). Note that $s(1) = 1$ is equivalent to the condition $p(1) = 1$, and our experiments suggest that the norm of p should satisfy $\|p\| < 2$ to ensure that increasing the number of smoothing stages k also increases the smoothness of the limit curves. We do not have a proof for this observation, however.

In principle, it is also possible to use different smoothing operators in the refine and each smoothing stage, and as long as these operators share a certain set of invariants, the subdivision scheme reproduces these invariants in the limit. However, it is beyond the scope of this paper to discuss and analyse such schemes.

Finally, the same idea could be applied to subdivision *surfaces*, where refine and smooth factorizations have already proved valuable (Oswald and Schröder, 2003). In this context, such a factorization serves a dual purpose: not only does it allow greater efficiency, but local rules are also necessary to extend a subdivision scheme to arbitrary meshes including extraordinary points (Cashman et al., 2009). A very useful extension of this work would be an extension of the geometric κ-schemes to create subdivision surfaces that preserve natural quadrics. However, such a geometric scheme appears out of reach at present.

References

Cashman, T. J., Augsdörfer, U. H., Dodgson, N. A., Sabin, M. A., 2009. NURBS with extraordinary points: High-degree, non-uniform, rational subdivision schemes. ACM Transactions on Graphics 28 (3), #46:1–9.

Deslauriers, G., Dubuc, S., 1989. Symmetric iterative interpolation processes. Constructive Approximation 5 (1), 49–68.

Dubuc, S., 1986. Interpolation through an iterative scheme. Journal of Mathematical Analysis and Applications 114 (1), 185–204.

Dyn, N., Goldman, R., 2011. Convergence and smoothness of nonlinear Lane-Riesenfeld algorithms in the functional setting. Foundations of Computational Mathematics 11 (1), 79–94.

Dyn, N., Hormann, K., Sabin, M. A., Shen, Z., 2008. Polynomial reproduction by symmetric subdivision schemes. Journal of Approximation Theory 155 (1), 28–42.

Dyn, N., Levin, D., 2002. Subdivision schemes in geometric modelling. Acta Numerica 11, 73–144.

Floater, M. S., Micchelli, C. A., 1998. Nonlinear stationary subdivision. In: Govil, N. K., Mohapatra, R. N., Nashed, Z., Sharma, A., Szabados, J. (Eds.), Approximation Theory: in Memory of A. K. Varma. Marcel Dekker, New York, pp. 209–224.

Goldman, R., Vouga, E., Schaefer, S., 2009. On the smoothness of real-valued functions generated by subdivision schemes using nonlinear binary averaging. Computer Aided Geometric Design 26 (2), 231–242.

Gray, R. M., 2006. Toeplitz and circulant matrices: A review. Foundations and Trends in Communications and Information Theory 2 (3), 155–239.

Lane, J. M., Riesenfeld, R. F., 1980. A theoretical development for the computer generation and display of piecewise polynomial surfaces. IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (1), 35–46.

Oswald, P., Schröder, P., 2003. Composite primal/dual $\sqrt{3}$-subdivision schemes. Computer Aided Geometric Design 20 (3), 135–164.

Rioul, O., 1992. Simple regularity criteria for subdivision schemes. SIAM Journal on Mathematical Analysis 23 (6), 1544–1576.

Sabin, M., 2010. Analysis and Design of Univariate Subdivision Schemes. Vol. 6 of Geometry and Computing. Springer.

Sabin, M. A., Dodgson, N. A., 2004. A circle-preserving variant of the four-point subdivision scheme. In: Dæhlen, M., Mørken, K., Schumaker, L. L. (Eds.), Mathematical Methods for Curves and Surfaces: Tromsø 2004. Nashboro Press, Brentwood, TN, pp. 1–12.

Schaefer, S., Vouga, E., Goldman, R., 2008. Nonlinear subdivision through nonlinear averaging. Computer Aided Geometric Design 25 (3), 162–180.