Notes on uniform spline subdivision

Michael S. Floater

September 16, 2009

Abstract

These notes provide an introduction to the subdivision rules for uniform splines, including the Chaikin algorithm. We also explain the Lane-Reisenfeld algorithm.

1 Introduction

One way of defining uniform B-splines is recursively as follows. The B-spline N^0 is the function

$$N^0(x) = \begin{cases} 1 & 0 \le x < 1; \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

and for $d \ge 1$, the B-spline N^d is defined as

$$N^d(x) = \int_0^1 N^{d-1}(x-t)\,dt. \tag{2}$$

We see that N^0 is non-negative, piecewise-constant, with support $[0, 1]$. For general d, one can show by induction on d that N^d is a non-negative, piecewise polynomial of degree d, of smoothness C^{d-1} at the breakpoints (called 'knots') $0, 1, \ldots, d+1$, and has support $[0, d+1]$. One can also show by induction that

$$\int_{-\infty}^{\infty} N^d(x)\,dx = 1,$$

and

$$\sum_{i \in \mathbb{Z}} N^d(x-i) = 1.$$

The B-splines of degree 1 and 2 are

$$N^1(x) = \begin{cases} x & 0 \le x < 1; \\ 2-x & 1 \le x < 2; \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

and

$$N^2(x) = \begin{cases} \frac{x^2}{2} & 0 \le x < 1; \\ -\frac{3}{2} + 3x - x^2 & 1 \le x < 2; \\ \frac{1}{2}(-3+x)^2 & 2 \le x < 3; \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Another way of expressing (2) is clearly as

$$N^d(x) = \int_{-\infty}^{\infty} N^0(t) N^{d-1}(x-t)\, dt.$$

Thus, if we recall that the *convolution* $p \otimes q$ of two functions p and q is defined as

$$(p \otimes q)(x) = \int_{\infty}^{\infty} p(t) q(x-t)\, dt,$$

we can express (2) simply as

$$N^d = N^0 \otimes N^{d-1}. \tag{5}$$

Thus N^d is the d-fold convolution of N^0 with itself:

$$N^d = \underbrace{N^0 \otimes N^0 \otimes \cdots \otimes N^0}_{d+1}.$$

2 Subdivision

A *uniform spline* is any linear combination of integer translates of a B-spline of a certain degree. Thus,

$$s(x) = \sum_{i \in \mathbb{Z}} c_i N^d(x-i) \tag{6}$$

is a spline, which is clearly a piecewise polynomial of degree d, with smoothness C^{d-1}. The breakpoints, or knots, of s are the integers bause the translated B-spline $N^d(x-i)$ has knots at the integers in its support, $[i, i+d+1]$.

Notice that for a fixed degree d, the spline s is completely determined by its coefficient vector

$$\mathbf{c} = (\ldots, c_{-1}, c_0, c_1, \ldots)^T.$$

The idea of subdivision is to represent the spline s in terms of the scaled B-splines $N^d(2x-i)$ whose knots are at the half-integers. The support of $N^d(2x-i)$ is $[i/2, (i+d+1)/2]$. We would like to find the coefficients b_i such that

$$s(x) = \sum_{i \in \mathbb{Z}} b_i N^d(2x-i). \tag{7}$$

To do this we will establish the *refinement relation*

$$N^d(x) = \sum_{i \in \mathbb{Z}} s_i^d N^d(2x-i). \tag{8}$$

In fact, by considering the supports of the B-splines in this equation it is clear that we must have $s_i^d = 0$ for $i < 0$ and $i > d+1$, and so if (8) holds we must have

$$N^d(x) = \sum_{i=0}^{d+1} s_i^d N^d(2x - i).$$

Assuming for the time being that (8) holds, let us see how we can use it to find the coefficients b_i from the coefficients c_i. Starting from (6) we have

$$\begin{aligned} s(x) = \sum_j c_j N^d(x - j) &= \sum_j c_j \sum_i s_i^d N^d(2(x - j) - i) \\ &= \sum_j c_j \sum_i s_{i-2j}^d N^d(2x - i) \\ &= \sum_i \sum_j s_{i-2j}^d c_j N^d(2x - i) \end{aligned}$$

and equating this with (7), and using the fact that the B-splines $N(2x - i)$ are linearly independent, we can equate coefficients, giving

$$b_i = \sum_j s_{i-2j}^d c_j. \tag{9}$$

This formula tells us how to convert the coarse representation of s in (6) to the finer representation in (7). If, like the coarse coefficients we arrange the fine coefficients in a column vector

$$\mathbf{b} = (\ldots, b_{-1}, b_0, b_1, \ldots)^T,$$

we can express (9) in vector and matrix notation as

$$\mathbf{b} = S^d \mathbf{c}.$$

The matrix

$$S^d = (s_{i-2j}^d)_{ij},$$

which is infinite in both dimensions, is known as the *subdivision matrix*. The subdivision scheme (9) can be split into two parts, for coefficients b_i with even and odd indices. We find

$$b_{2i} = \sum_j s_{2(i-j)}^d c_j = \sum_j s_{-2j}^d c_{j+i} = \sum_j s_{2j}^d c_{i-j}, \tag{10}$$

and

$$b_{2i+1} = \sum_j s_{2(i-j)+1}^d c_j = \sum_j s_{-2j+1}^d c_{j+i} = \sum_j s_{2j+1}^d c_{i-j}. \tag{11}$$

So

$$b_{2i} = s_0^d c_i + s_2^d c_{i-1} + \cdots, \tag{12}$$

$$b_{2i+1} = s_1^d c_i + s_3^d c_{i-1} + \cdots. \tag{13}$$

3 The refinement relation

It is easy to see from (1) that

$$N^0(x) = N^0(2x) + N^0(2x-1), \tag{14}$$

and using (3) a simple calculation shows that

$$N^1(x) = \frac{1}{2}N^1(2x) + N^1(2x-1) + \frac{1}{2}N^1(2x-2). \tag{15}$$

Thus $s_0^0 = s_1^0 = 1$ and $s_0^1 = 1/2$, $s_1^1 = 1$, and $s_2^1 = 1/2$. We will derive the general formula for s_i^d using the recurrence relation (2). We do this by first showing how the coefficients of degree d relate to those of degree $d-1$.

Lemma 1 *If the refinement relation (8) holds for degree $d-1$ with coefficients s_i^{d-1} then it also holds for degree d and the coefficients are*

$$s_i^d = \frac{1}{2}(s_i^{d-1} + s_{i-1}^{d-1}).$$

Proof. Using (8), we have

$$\begin{aligned} N^d(x) &= \int_0^1 \sum_i s_i^{d-1} N^{d-1}(2(x-t)-i)\,dt \\ &= \sum_i s_i^{d-1} \int_0^1 N^{d-1}(2(x-t)-i)\,dt. \end{aligned}$$

But

$$\begin{aligned} &\int_0^1 N^{d-1}(2(x-t)-i)\,dt \\ &= \frac{1}{2}\int_0^2 N^{d-1}(2x-u-i)\,du \\ &= \frac{1}{2}\left(\int_0^1 N^{d-1}(2x-u-i)\,du + \int_1^2 N^{d-1}(2x-u-i)\,du\right) \\ &= \frac{1}{2}\int_0^1 \left(N^{d-1}(2x-u-i)\,du + N^{d-1}(2x-u-i-1)\right)\,du \\ &= \frac{1}{2}\left(N^d(2x-i) + N^d(2x-i-1)\right), \end{aligned}$$

and so

$$\begin{aligned} N^d(x) &= \frac{1}{2}\sum_i s_i^{d-1}\left(N^d(2x-i) + N^d(2x-i-1)\right) \\ &= \frac{1}{2}\sum_i (s_i^{d-1} + s_{i-1}^{d-1}) N^d(2x-i). \end{aligned}$$

□

Iterating the formula of Lemma 1 from $s^0_0 = s^0_1 = 1$ immediately gives

Theorem 1 *The refinement relation (8) holds with coefficients*

$$s^d_i = \frac{1}{2^d}\binom{d+1}{i}, \qquad 0 \le i \le d+1.$$

The first few examples, with $\mathbf{s}^d = (s^d_i)_i$ are

$$\begin{aligned}
\mathbf{s}^0 &= (1,1)\,, \\
\mathbf{s}^1 &= \left(\frac{1}{2}, 1, \frac{1}{2}\right), \\
\mathbf{s}^2 &= \left(\frac{1}{4}, \frac{3}{4}, \frac{3}{4}, \frac{1}{4}\right), \\
\mathbf{s}^3 &= \left(\frac{1}{8}, \frac{1}{2}, \frac{3}{4}, \frac{1}{2}, \frac{1}{8}\right).
\end{aligned}$$

Corresponding to these, the first few subdivision matrices are

$$S^0 = \begin{pmatrix}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & 0 & 0 & 0 & \cdot \\
\cdot & 1 & 0 & 0 & 0 & \cdot \\
\cdot & 0 & 1 & 0 & 0 & \cdot \\
\cdot & 0 & 1 & 0 & 0 & \cdot \\
\cdot & 0 & 0 & 1 & 0 & \cdot \\
\cdot & 0 & 0 & 1 & 0 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{pmatrix}, \qquad
S^1 = \begin{pmatrix}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdot \\
\cdot & 0 & 1 & 0 & 0 & \cdot \\
\cdot & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdot \\
\cdot & 0 & 0 & 1 & 0 & \cdot \\
\cdot & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{pmatrix},$$

$$S^2 = \begin{pmatrix}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \frac{3}{4} & \frac{1}{4} & 0 & 0 & \cdot \\
\cdot & \frac{1}{4} & \frac{3}{4} & 0 & 0 & \cdot \\
\cdot & 0 & \frac{3}{4} & \frac{1}{4} & 0 & \cdot \\
\cdot & 0 & \frac{1}{4} & \frac{3}{4} & 0 & \cdot \\
\cdot & 0 & 0 & \frac{3}{4} & \frac{1}{4} & \cdot \\
\cdot & 0 & 0 & \frac{1}{4} & \frac{3}{4} & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{pmatrix}, \qquad
S^3 = \begin{pmatrix}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & \cdot \\
\cdot & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdot \\
\cdot & 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \cdot \\
\cdot & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{pmatrix}$$

The Lane-Riesenfeld algorithm is an elegant way of implementing the subdivision scheme and follows from Lemma 1. In this algorithm we initially set

$$b^0_{2i} = b^0_{2i+1} = c_i,$$

and then, for $k = 1, \ldots, d$, we let

$$b^k_i = (b^{k-1}_i + b^{k-1}_{i-1})/2.$$

Then $b_i = b^d_i$ is the required coefficient.

We can also view this algorithm in terms of matrices. The subdivision matrix can be expressed as

$$S^d = \underbrace{AA\cdots A}_{d} S^0,$$

where A is the 'averaging' matrix

$$A = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdot \\ \cdot & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdot \\ \cdot & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix},$$

and we can view S^0 as a 'doubling' matrix. Thus to compute the new coefficients $\mathbf{b}$ from the old, $\mathbf{c}$, one first applies S^0 to $\mathbf{c}$, which has the effect of 'doubling' the coefficients in $\mathbf{c}$, and one then applies the matrix A, which replaces all points by their mid-points, d times.

4 Convergence

Suppose now that starting from a spline

$$s(x) = \sum_i c_i^0 N^d(x - i),$$

we apply several steps of subdivision. If we subdivide s once, we obtain the finer representation

$$s(x) = \sum_i c_i^1 N^d(2x - i),$$

where

$$c_i^1 = \sum_j s_{i-2j}^d c_j^0,$$

with s_i^d given by Theorem 1. We can continue in this way, subdividing again and again, so that in general

$$s(x) = \sum_i c_i^k N^d(2^k x - i),$$

where

$$c_i^k = \sum_j s_{i-2j}^d c_j^{k-1}.$$

At each level of subdivision, k, we can form a polygon p_k, a piecewise linear function with the value c_i^k at the point $2^{-k}i$. It can be shown that the sequence of polygons $(p_k)_k$ converges to s, i.e.,

$$s(x) = \lim_{k\to\infty} p_k(x), \qquad x \in \mathbb{R}.$$

This provides a way of plotting the spline s. After a few steps of subdivision, we simply plot the polygon p_k. If k is large enough, p_k will appear to be a smooth function.

References

[1] J. Warren and H. Weimer, *Subdivision methods in geometric design*, Morgan Kaufmann, 2002.

[2] H. Prautzsch, W. Boehm, and M. Paluszny, Bezier and B-spline techniques, Springer, 2002.