Volume xxx, (1999) number yyy pp. 000–000

Multiresolution Curve and Surface Representation: Reversing Subdivision Rules by Least-Squares Data Fitting

Faramarz F. Samavati† and Richard H. Bartels‡

†Sharif University of Technology
Mathematical Science Department
Tehran, Iran
‡University of Waterloo
Computer Science Department
Waterloo, Ontario N2L 3G1
Canada

Abstract

This work explores how three techniques for defining and representing curves and surfaces can be related efficiently. The techniques are subdivision, least-squares data fitting, *and* wavelets. *We show how least-squares data fitting can be used to "reverse" a subdivision rule, how this reversal is related to wavelets, how this relationship can provide a multilevel representation, and how the decomposition/reconstruction process can be carried out in linear time and space through the use of a matrix factorization.*
Some insights that this work brings forth are that the inner product used in a multiresolution analysis influences the support of a wavelet, that wavelets can be constructed by straightforward matrix observations, and that matrix partitioning and factorization can provide alternatives to inverses or duals for building efficient decomposition and reconstruction processes. We illustrate our findings using an example curve, grey-scale image, and tensor-product surface.

Keywords: Subdivision; Multiresolution; Curves; Surfaces; Wavelets; Least squares; Data fitting;

1. Introduction

Subdivision curves and surfaces begin with a polygonal network of points. In the case of curves, the network merely encodes a point-to-point sequence. In the case of surfaces, the network provides the edges and faces of a polyhedral topology. In conventional usage a subdivision curve or surface begins with a coarse network structure consisting of relatively few points. To this is applied a set of rules that replace the coarse network by a finer network. The set of rules could again be applied to this finer network. In the limit of repeated application, the rules yield curves or surfaces of some provable continuity (excluding exceptional points). In practice, of course, the rules are repeated only a finite number of times to yield a network containing a large number of points that represent a fine approximation of the limit.

Suppose we begin with some fine sampling of a curve or surface and we wish to represent this data approximately in some more compact way as if it had been generated by a certain subdivision rule. Ignoring the subdivision for a moment, the traditional approach for doing data approximation on a finite set of data points is through *least-squares data fitting*. The sampling will be the m items of a data vector, f, which serves as the right-hand side for an $m \times n$ ($n < m$) overdetermined set of equations, $Pc = f$ that is to be solved in the sense that the Euclidian norm of the residual vector, $\|Pc - f\|_2$, is to be minimized. (Sometimes this system is specified only implicitly by the $n \times n$ normal equations $P^T Px = P^T f$, whose solution, c, will

be the minimizer for the residual vector.) The $m \times n$ matrix P provides the "design" of the approximation, generally by specifying a sense in which some given functions ϕ_j approximate the data $\sum_j c_j \phi_j \approx f_i$. The particular specification is characterized by the nature in which the functions ϕ_i contribute values for the entries p_{ij} of P, or alternatively for the entries $\tilde{p}_{ij}$ of the matrix $P^T P$ and $\tilde{f}_i$ of the vector $P^T f$.

In our geometric context, f represents "fine" points, and c represents "coarse" points whose subdivision, through the rule given in the matrix P, approximately produce the f. The specification of interest to us, which characterizes least-squares data-fitting, gives the entries of P explicitly in the form p_{ij} and defines the minimization to be over the residuals

$$\sum_j c_j p_{ij} - f_i \tag{1}$$

The functions ϕ underlying the p_{ij} do not have to be known to solve this problem as given.

Another typical specification, which characterizes *least-squares approximation*, only specifies the data $P^T f$ in terms of an approximating inner product and a function g; e.g. $\tilde{f}_i = \langle \phi_i, g \rangle$ for some function g. In this case, the normal equations are be solved:

$$\sum_j c_j \tilde{p}_{ij} = \tilde{f}_i \tag{2}$$

where $\tilde{p}_{ij} = \langle \phi_i, \phi_j \rangle$. The need to specify a function g underlying the data is one of the distinguishing characteristics of the approximation problem that contrasts it with the data fitting problem.

In this paper we explore the possibilities for reversing a subdivision process from the residuals in (1). We begin with a fine network that did not necessarily come from a set of subdivision rules, and we wish to find a coarser network that could have produced the finer network, in an approximate sense, by the application of given subdivision rules. We specify the matrix P as the one which is defined by the subdivision.

Because least-squares systems are not necessarily solved exactly, the coarse network may not return the fine network under the use of the subdivision rules. However, we can reconstruct the fine network with appropriate corrections derived from error terms. This will structure our approach along lines familiar to the users of wavelets. A fine set of data will be decomposed into a coarse approximation and error information. The fine data can subsequently be reconstructed from the coarse data and the error. As a byproduct, some degree of compression may be achieved by suppressing less important elements of the error.

Although we shall draw a comparison with wavelets and multiresolution analysis, we don't assume that we know what scale functions underlie our data in order to work out the decomposition and reconstruction algorithms. Furthermore, even if the scale functions were known, we would not necessarily want to find the conventional wavelets that are associated with them. Wavelets are usually defined in the context of L_2 function spaces; that is, by (2) in terms of an inner product that is *continuously* dependent on the scale functions ($\langle \phi_i, \phi_j \rangle = \int \phi_i \bar{\phi}_j$). We shall work only with the coefficients of the scale functions, which we identify with the data points, the coefficients of the wavelet functions, which we associate with the errors, and the conventional discrete (Euclidian) vector inner product, which measures the errors directly on the data f_i rather than over some function g. As such, we shall be working in finite-dimensional ℓ_2 spaces.

In Section 2 we relate what we are doing with some previous work. In Section 3 we shall review basic matrix notation for subdivision rules. In Section 4 we introduce the least-squares problem of interest, and in Section 5 we review the normal equations. In Section 6 we outline an approach to the construction of orthogonal complements for subdivision matrices. In Section 7 we use the special form of these orthogonal complements to solve for the error coefficients. Remarks on error are presented in Section 8. Several curve subdivision rules are presented in Section 9. Periodic rules are covered in Section 10. Tensor-product surfaces are covered in Section 11. In Section 12 we present our approach applied to an example curve, grey-scale image, and tensor-product surface. In Section 13 we shall review the connection of these matrices to wavelets. We close in Section 14 with some pointers to work in progress and future work.

We concentrate here on the mechanics for reversing subdivision rules using least-squares, data-fitting techniques. The end product of our work will be a multiresolution decomposition of given data into a sequence of error (detail) information and a base set of coarse data. We are presenting here only a technique that starts from suitable subdivision rules, is based upon least-squares data fitting, handles finite data sets, costs linear time, and achieves the decomposition. Explorations of the use of the technique for compression, multilevel interactive design, and multilevel rendering are outside the intended scope of this work.

2. Previous Work

Most of our discussion will use curve-subdivision rules that are, in fact, derived from uniform knot insertion acting on a basis of B-splines on a closed, bounded interval. The B-splines are taken as uniform, except

for those near the end points of the interval, which accommodate the boundary of the interval by having either their leftmost or rightmost knots be suitably multiple, depending on which boundary they are near, and how near they are. Such an arrangement yields nested spaces with the n^{th} space having polynomial pieces with breakpoints at $\frac{b-a}{n}$ on an interval $[a, b]$. Conventional B-spline wavelets of minimal support for essentially this situation were investigated by Chui and Quack [5] and Quack and Weyrich [16]. (Even more general B-spline wavelets, subsuming the ones just mentioned, can be found in a paper by Lyche and Mørken [14].) In contrast to the methods they use, our method develops wavelets purely by matrix considerations starting from the subdivision matrix (i.e. from the 2-scale relations represented by the knot-insertion refinement).

Since we work directly from the 2-scale relations in matrix form, we do not need to restrict ourselves to B-splines (e.g. we can use τ-scale relations, or the Daubechies [7, 8] 2-scale relation given a brief mention in Section 10). Also, in contrast to their methods, we work on finite subspaces of the space ℓ_2 rather than L_2.

The method we use corresponds to that used by Finklestein and Salesin [11] for cubic B-splines alone; namely, the wavelets come from establishing a matrix, Q, whose columns are orthogonal to the columns of the subdivision matrix, P, and whose run of column-nonzeros is of minimal length. The difference in our wavelets for the cubic B-spline case (and the simplicity of their defining coefficients, which are the entries in Q) comes purely from the difference in the inner product we use vs. the one used by Finklestein and Salesin. (They, like Chui, Quack, and Weyrich, use the inner product on L_2.) However, whereas Finklestein and Salesin restricted themselves to the cubic B-spline case and merely present their wavelets without development, apparently as the result of sessions with a symbolic algebra system, we show how these wavelets can be provided in general by a simple construction.

Our use of a least-squares inner product (and consequently a Euclidian norm) that measures error on the points c and f rather than with respect to underlying scale functions ϕ bears a relationship to decisions made by Lyche and Mørken [15] for a norm to measure deviations under knot removal in splines. When a subdivision rule is given by a B-spline scale relation, what we are proposing here has the appearance of removing every second knot.

Focusing on matrix techniques for the construction of wavelets is not new with Finkelstein and Salesin nor here. Papers by Warren [20] and Dahmen and Micchelli [6] give other examples. These achieve sparse decomposition and reconstruction matrices (P, and Q as well as A and B), but in a context quite distinct from least-squares data fitting.

Unser, et. al. [1], have also looked at the ℓ_2 case and least-squares fitting. They, however, concentrate on the infinite line and Schoenberg's cardinal B-splines, which is suitable for the treatment of a countably infinite number of regularly-sampled data points. Once again, our approach treats data of finite extent (in the B-spline case, on an interval domain), and is not restricted to the use of B-spline scaling functions.

The wavelet literature began by considering functions on the entire real line rather than on a closed bounded interval. In the infinite case all 2-scale relations are taken as identical, and further regularity is provided when every basis (scale) function at a given level is a translate of a single (mother) function. The subdivision matrix reduces to an endless repetition of a single column whose nonzero entries are displaced downward by 2 positions with each advance in column index (an infinite, 2-slanted matrix). The easiest finite matrix equivalent to this situation is provided by regular, periodic subdivision rules. The subdivision matrix is 2-slanted, and entries falling off the bottom of the matrix are "wrapped around" to the top of the matrix. Our techniques apply to this situation, as we outline in Section 10. We prefer to center the discussion around the case in which boundary wavelets must be produced, however, both because it forces us to confront a more general situation and because the factorization of the normal equations is a little easier to explain for matrices with a nonperiodic format.

B-spline wavelets for the infinite-domain case are covered in the book by Chui [4]. We shall observe that the matrix method we have used yields the same wavelets, if the inner product used by Chui is employed.

3. Subdivisions and Matrix Notation

We begin with *subdivision curves*. These start as a vector of *fine points*, f_j, that have been provided as data. To economize on notation, we shall think of these as the $k+1^{\text{st}}$ instance of a collection of points: c_j^{k+1}. If these points had come from a *subdivision rule*, we would have the relationship

$$c_i^{k+1} = \sum_j p_{i,j}^{k+1} c_j^k \tag{3}$$

From a matrix point of view:

$$\left[P^{k+1}\right]\left[C^k\right] = \left[C^{k+1}\right] \tag{4}$$

where the matrix P^{k+1} is $m \times n$ with $m > n$.

If we knew the matrix and the right hand side of

equation (4), we could consider finding the vector of *coarse points* C^k by solving the equation in the least-squares sense. The result would satisfy the equation only if the fine points, C^{k+1}, really were the result of subdividing the coarse points by the rules P^{k+1}. If the subdivision rules did not produce the fine points, the equation could be corrected as follows:

$$\begin{aligned}\begin{bmatrix} P^{k+1} & Q^{k+1} \end{bmatrix} \begin{bmatrix} C^k \\ D^k \end{bmatrix} &= \begin{bmatrix} P^{k+1} \end{bmatrix} \begin{bmatrix} C^k \end{bmatrix} \\ &+ \begin{bmatrix} Q^{k+1} \end{bmatrix} \begin{bmatrix} D^k \end{bmatrix} \\ &= \begin{bmatrix} C^{k+1} \end{bmatrix}\end{aligned} \tag{5}$$

where the columns of the $m \times m-n$ matrix Q^{k+1} form an *extension* of the column vectors of P^{k+1} to a basis for the m-dimensional vector space. Many extensions are possible; we need only adjoin linearly independent columns. However, the extension corresponding to the least-squares approximation requires the columns of Q^{k+1} to be orthogonal to the columns of P^{k+1}; that is, to be a basis for the null space of the transpose of P^{k+1}. For any extension Q^{k+1}, the vector $Q^{k+1}D^k$ will express the *error correction* $C^{k+1} - P^{k+1}C^k$. If Q^{k+1} is an orthogonal extension, the error correction expresses the least-squares error.

If the process given by equation (5) is carried out a number of times, a *multiresolution representation* of the original data results:

$$C^j, D^j, D^{j+1}, \ldots, D^k \tag{6}$$

whereby:

$$c_i^{\lambda+1} \leftarrow \sum_j p_{i,j}^{\lambda+1} c_j^{\lambda} + \sum_j q_{i,j}^{\lambda+1} d_j^{\lambda} \tag{7}$$

for $\lambda = j, \ldots, k$.

Tangentially, some further observations can be made about subdivision schemes from this matrix view. The matrix P^{k+1} defines the subdivision rules as transformations on the points C^k. In order to be geometrically meaningful [12], these transformations must produce *affine combinations* of the C^k, which implies that all row sums of P^{k+1} are 1. That restriction does not apply to the matrix Q^{k+1}. The D^k represent vectors, not points, since the second term in the sum appearing in equation (5) constitutes a displacement of the points represented by the first term.

Finally, note that the main requirements on Q^{k+1} is that it have full column rank and satisfy orthogonality, ${Q^{k+1}}^T P^{k+1} = 0$. These requirements can only specify Q^{k+1} up to an arbitrary nonsingular matrix $Q^{k+1}M$. In particular, each column of Q^{k+1} can be scaled individually as we choose. Such scaling influences the magnitude of the error coefficients, D^k. This, in turn, influences our judgment on whether to suppress an error coefficient for the purposes of compression. We shall come back to this point in Section 8.

4. Least Squares

Assume that a subdivision rule, P^{k+1}, and a set of fine points (not necessarily generated by that rule), C^{k+1}, have been specified. Then the following hold:

1. C^k provides the least-squares solution to

$$\begin{bmatrix} P^{k+1} \end{bmatrix} \begin{bmatrix} C^k \end{bmatrix} = \begin{bmatrix} C^{k+1} \end{bmatrix} \tag{8}$$

if and only if C^k satisfies the *normal equations*:

$$\begin{bmatrix} {P^{k+1}}^T P^{k+1} \end{bmatrix} \begin{bmatrix} C^k \end{bmatrix} = \begin{bmatrix} {P^{k+1}}^T \end{bmatrix} \begin{bmatrix} C^{k+1} \end{bmatrix} \tag{9}$$

2. If C^k is produced as in equation (9) and if the columns of Q^{k+1} form a basis for the nullspace of ${P^{k+1}}^T$, then the equations

$$\begin{bmatrix} Q^{k+1} \end{bmatrix} \begin{bmatrix} D^k \end{bmatrix} = \begin{bmatrix} C^{k+1} \end{bmatrix} - \begin{bmatrix} P^{k+1} \end{bmatrix} \begin{bmatrix} C^k \end{bmatrix} \tag{10}$$

are compatible.

3. The overdetermined system (10) may be solved in the least-squares sense via the normal equations

$$\begin{aligned}&\begin{bmatrix} Q^{k+1} \end{bmatrix}^T \begin{bmatrix} Q^{k+1} \end{bmatrix} \begin{bmatrix} D^k \end{bmatrix} \\ &= \begin{bmatrix} Q^{k+1} \end{bmatrix}^T \left(\begin{bmatrix} C^{k+1} \end{bmatrix} - \begin{bmatrix} P^{k+1} \end{bmatrix} \begin{bmatrix} C^k \end{bmatrix} \right)\end{aligned} \tag{11}$$

But this is equivalent to

$$\begin{bmatrix} Q^{k+1} \end{bmatrix}^T \begin{bmatrix} Q^{k+1} \end{bmatrix} \begin{bmatrix} D^k \end{bmatrix} = \begin{bmatrix} Q^{k+1} \end{bmatrix}^T \begin{bmatrix} C^{k+1} \end{bmatrix} \tag{12}$$

since $\begin{bmatrix} Q^{k+1} \end{bmatrix}^T \begin{bmatrix} P^{k+1} \end{bmatrix} = 0$. In view of the compatibility of (10), however, the solution will exactly satisfy the equation system, and this solution can be obtained equivalently from any subselection of the rows of Q^{k+1} that forms a nonsingular matrix, $\hat{Q}^{k+1}$, with a corresponding selection of the elements of the residual vector $C^{k+1} - P^{k+1}C^k$.

Another observation is also important:

4. Subdivision matrices P^{k+1} are usually banded and characterized by columns that are shifted versions of a single column (except for a few initial and final columns).

cubic B-spline subdivision gives a good example [11]. Except for the first and last 3 rows, the odd rows contain only two adjacent nonzero entries, $\frac{1}{2}, \frac{1}{2}$, and the even rows contain only three adjacent nonzero entries $\frac{1}{8}, \frac{3}{4}, \frac{1}{8}$, and the shifted locations provide that each column except the first 3 and last 3 contains only $\frac{1}{8}, \frac{1}{2}, \frac{3}{4}, \frac{1}{2}, \frac{1}{8}$. Each such column's nonzero entries

is shifted from its predecessor's by 2 positions:

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & \frac{3}{4} & \frac{1}{4} & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & \frac{3}{16} & \frac{11}{16} & \frac{1}{8} & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \cdots & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{8} & \frac{11}{16} & \frac{3}{16} & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & \frac{1}{4} & \frac{3}{4} & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \tag{13}$$

5. Normal Equations

The linear independence of the columns of P^{k+1} implies that the *normal-equation matrix* ${P^{k+1}}^T P^{k+1}$ has an LDL^T factorization [13]. By means of this factorization the normal equations can be solved with an amount of work that is linear in the size of C^{k+1}. Again, cubic B-spline subdivision provides a convenient example for a normal-equation matrix:

$$\begin{bmatrix}
\frac{5}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
\frac{1}{4} & \frac{217}{256} & \frac{81}{256} & \frac{3}{128} & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & \frac{81}{256} & \frac{205}{256} & \frac{55}{128} & \frac{1}{64} & 0 & 0 & \cdots & 0 & 0 \\
0 & \frac{3}{128} & \frac{55}{128} & \frac{35}{32} & \frac{7}{16} & \frac{1}{64} & 0 & \cdots & 0 & 0 \\
0 & 0 & \frac{1}{64} & \frac{7}{16} & \frac{35}{32} & \frac{7}{16} & \frac{1}{64} & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & \frac{1}{64} & \frac{7}{16} & \frac{35}{32} & \frac{7}{16} & \frac{1}{64} & 0 & 0 \\
0 & 0 & \cdots & 0 & \frac{1}{64} & \frac{7}{16} & \frac{35}{32} & \frac{55}{128} & \frac{3}{128} & 0 \\
0 & 0 & \cdots & 0 & 0 & \frac{1}{64} & \frac{55}{128} & \frac{205}{256} & \frac{81}{256} & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 & \frac{3}{128} & \frac{81}{256} & \frac{217}{256} & \frac{1}{4} \\
0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{5}{4}
\end{bmatrix} \tag{14}$$

More generally, a subdivision rule will have most columns in a standard form containing μ nonzero elements that begin at row λ for the first occurrence of these standard columns and are shifted by τ rows for each successive column. Since the (i, j) element of the normal-equation matrix is given by the inner product of columns i and j of the matrix this means that, except for a certain number of initial and final rows, each row of the normal-equation matrix will consist of exactly the same $2\left\lfloor \frac{\mu-1}{\tau} \right\rfloor + 1$ numbers.

All entries of the normal-equation matrix can be obtained in advance for the subdivision rules under consideration, so we do not have to count their cost. The process of computing C^k will require applying ${P^{k+1}}^T$ to C^{k+1} to produce an intermediate vector G^{k+1}. This process looks like the application of a finite filter and involves μ multiplications and additions for each element of G^{k+1}. The total effort will be $O(2m\mu)$ elementary floating point operations. The use of the LDL^T factorization to solve a banded system, with right-hand side given by G^{k+1} and band width given by $\left\lfloor \frac{\mu-1}{\tau} \right\rfloor$, will cost $O\left(n(\left\lfloor \frac{\mu-1}{\tau} \right\rfloor^2 + 8\left\lfloor \frac{\mu-1}{\tau} \right\rfloor + 1)\right)$ elementary floating point operations [13]. Thus, overall, finding C^k will incur a cost that is linear in m; that is, linear in the size of C^{k+1}. (This counts both the cost of producing the factors L and D as well as the forward- and back-solution processes. One could save a considerable portion of this, at the cost of $n \times \left\lfloor \frac{\mu-1}{\tau} \right\rfloor$ storage, by computing the factors in advance.)

6. Orthogonal Complements

The second part of the process is to find the elements of D^k whereby the error $C^{k+1} - P^{k+1}C^k$ may be expressed. For this we must extend the matrix P^{k+1} by adding linearly-independent columns Q^{k+1} such that ${Q^{k+1}}^T P^{k+1} = 0$. As will be seen, there is an additional bonus in constructing the columns of Q^{k+1} to have a regular pattern of shifted columns with as few nonzero entries as possible, imitating the structure of P^{k+1}.

Matrices Q satisfying $P^T Q = 0$ are not uniquely defined. At the very least, scale is not determined, since $P^T(\sigma Q) = 0$, for any $\sigma \neq 0$, if $P^T Q = 0$. Of more interest to us is the fact that the sparsity of Q may vary widely, since $P^T(QM) = 0$, for any matrix M of suitable dimensions, if $P^T Q = 0$. (We would, of course, wish the rank of (QM) to be the same as the rank of Q.) This leads us to explore the kinds of conditions we might impose on Q to maximize its sparsity (that is, the number of contiguous nonzero elements in any column). The following simple example shows how that exploration proceeds.

Suppose a sequence of columns of P (2-slanted) each consists of the same 4 nonzero entries; that is, each

column has the form:

$$\begin{bmatrix} \vdots \\ 0 \\ a \\ b \\ c \\ d \\ 0 \\ \vdots \end{bmatrix} \tag{15}$$

Then the condition that three inner products simultaneously yield a zero value determines whether a single column of Q might exist with only 4 nonzero entries:

$$\left.\begin{matrix} a & b & c & d \\ t & u & & \end{matrix}\right\} \rightarrow 0$$
$$\left.\begin{matrix} a & b & c & d \\ r & s & t & u \end{matrix}\right\} \rightarrow 0 \tag{16}$$
$$\left.\begin{matrix} a & b & c & d \\ & & r & s \end{matrix}\right\} \rightarrow 0$$

Here r, s, t, and u are the assumed elements of the column of Q. The three inner products correspond to three successive columns of P, and the nonzero locations for the column of Q are assumed to be in the same row positions as those of the second of the columns of P in question. In matrix form this becomes:

$$\begin{bmatrix} c & d & 0 & 0 \\ a & b & c & d \\ 0 & 0 & a & b \end{bmatrix} \begin{bmatrix} r \\ s \\ t \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{17}$$

This reveals that any vector in the nullspace of the left-hand matrix in equation (17) will provide a column for Q (assuming no other columns of P have nonzeros in positions that correspond to the nonzeros of this column of Q). A nullspace is assured, since the left-hand matrix has more columns than rows. Indeed, by having one more column than rows, we are predicting that the nullspace will be one dimensional; that is, will consist of all multiples of a single, nonzero vector; namely, the vector consisting of the four entries $[r, s, t, u]$. This is the most "economical" situation to investigate, since it yields the shortest sequence of nonzero entries for the column of Q. (Assuming only one nonzero in the column of Q can be seen to yield a 1×1 left-hand matrix; assuming two nonzeros yields a 2×2 left-hand matrix; assuming three nonzeros yields a 3×3. The assumption of four nonzeros is the first assumption that yields a left-hand matrix that must, from its structure, have a nullspace, which further justifies the claim of minimal support.)

A similar nullspace problem can be posed for the special columns on the right and left of P. For example, for the subdivision based on cubic B-splines, the following is one of the nullspace problems we would investigate:

$$\begin{bmatrix} 1 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{3}{4} & \frac{3}{16} & 0 \\ 0 & 0 & \frac{1}{4} & \frac{11}{16} & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{8} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} r \\ s \\ t \\ u \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{18}$$

The general approach to finding a vector satisfying a nullspace equation such as (17) or (18) is as follows:

1. Reorder the columns of the left-hand matrix, if necessary, so that its leftmost columns form a nonsingular matrix. (If this cannot be done, the nullspace equation can be reduced at least by one row and column.)
2. The nullspace equation can be partitioned as follows:

$$\begin{bmatrix} B & M \end{bmatrix} \begin{bmatrix} v_B \\ v_M \end{bmatrix} = 0 \tag{19}$$

   where B is nonsingular. v_B is the portion of the (possibly reordered) nullspace vector corresponding to B, and v_M corresponds to M.
3. Choose v_M arbitrarily (nonzero), and set

$$v_B = -B^{-1} M v_M \tag{20}$$

For equation (18), for example, $v_M = v$ can be set to -1, and the result will be

$$v_B = \begin{bmatrix} r \\ s \\ t \\ u \end{bmatrix} = \begin{bmatrix} -6 \\ 12 \\ -9 \\ 4 \end{bmatrix} \tag{21}$$

The vector $[v_B, v_M]^T = [-6, 12, -9, 4, -1]^T$ divided by 12 shows up as the first column of the matrix in (24) below.

There is a short cut for the general, shifted column of P^{k+1}, corresponding to a nullspace equation of the form (17) (of whatever size is appropriate for the number of nonzeros in the general column of P^{k+1}). The solution to the nullspace equation is given by a column of Q^{k+1} that contains the nonzeros of the general column of P^{k+1}, reversed in order, alternated in sign, and shifted in row position so as to overlap the nonzeros of P^{k+1} in an even number of rows. For equation (17) specifically, this would mean the following column of

Q^{k+1}:

$$\begin{bmatrix} \vdots \\ 0 \\ -d \\ c \\ -b \\ a \\ 0 \\ \vdots \end{bmatrix} \tag{22}$$

where the nonzeros appear in the same rows as they do in the colum shown in (15). For an odd number of nonzeros, the row positions would be shifted in Q^{k+1} by one upward or downward by those in P^{k+1}. The cancellation pattern that results can easily be seen from the following example:

$$\begin{array}{cc} \underline{P^{k+1}} & \underline{Q^{k+1}} \\ \vdots & \vdots \\ \vdots & 0 \\ 0 & e \\ a & -d \\ b & c \\ c & -b \\ d & a \\ e & 0 \\ 0 & 0 \\ \vdots & \vdots \end{array} \tag{23}$$

To quote from page 88 of Stollnitz, et. al. [18]: "This recipe for creating a wavelet sequence from a scaling function sequence is common to many wavelet constructions on the infinite real line; such sequences are known as *quadrature mirror filters*."

Using such observations, we arrive at the following extension for the cubic B-spline subdivision matrix of equation (13):

$$\begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ -\frac{3}{4} & \frac{1}{8} & 0 & 0 & 0 & \cdots & 0 & 0 \\ \frac{1}{3} & -\frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{12} & \frac{3}{4} & \frac{1}{8} & 0 & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & \cdots & 0 & 0 & \frac{1}{8} & \frac{3}{4} & -\frac{1}{12} \\ 0 & 0 & \cdots & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{3} \\ 0 & 0 & \cdots & 0 & 0 & 0 & \frac{1}{8} & -\frac{3}{4} \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \tag{24}$$

7. Solving for the Error Coefficients

The system (10) is an overdetermined, yet consistent, system of equations. This means that any selection of the rows of system (10) that produces a nonsingular submatrix of Q^{k+1} may be solved for D^k. The result will not depend on which selection of rows is taken.

The subdivision rules of primary interest to us are those for which the matrix Q^{k+1} has a selection of rows that yield a triangular submatrix, for then the vector D^k can be obtained without any significant further matrix factorization. So far, this has been true of *all* subdivision rules we have looked at.

The selection of rows of Q^{k+1}, just as the construction of Q^{k+1} and the construction of ${P^{k+1}}^T P^{k+1}$, can be made in advance of having any sets of data, and so accounts for no computational cost.

An appropriate row-selection from the matrix of (24), for example, would be the matrix $\hat{Q}^{k+1}$ given

by:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{3} \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{25}$$

This matrix represents the selection of the second row of Q^{k+1}, followed by rows $6, 8, 10, \ldots$ and ending with the next to last row. Correspondingly, elements $2, 6, 8, 10, \ldots, m-1$ of the residual vector must be chosen to constitute the equations that must be solved for D^k.

Solving for the vector D^k can be done in linear time, and the back-solution computation looks essentially like the application of a finite filter of length 2 to the vector subselected from $C^{k+1} - P^{k+1}C^k$. This is a fitting place to remark that, while both the decomposition and reconstruction processes we are describing have linear cost, the decomposition process is the more expensive. Reconstruction involves only the processing of the vector C^k by the rows of P^{k+1}; that is, by applying a filter of at most length 3 in our example, and processing the vector D^k by the rows of Q^{k+1}, again a filter of at most length 3 in our example. In common applications (e.g. compression, multilevel rendering), fast reconstruction is desirable.

8. Remarks on Error

A frequent use of multiresolution representations is for the purpose of compression. In this use, the original data C^{k+1} is represented as a telescoping series

$$\begin{aligned} C^{k+1} &= P^{k+1}C^k + Q^{k+1}D^k \\ &= P^{k+1}\left(P^kC^{k-1} + Q^kD^{k-1}\right) + Q^{k+1}D^k \\ &\cdots \end{aligned} \tag{26}$$

a result which comes from (7).

The information (for some N) represented by C^{k-N} and $D^{k-N}, \ldots, D^k$ is stored instead of C^{k+1}, and compression is achieved by discarding bits from the sequence $\left\{D^{k-\lambda}\right\}$.

Suppose $D^{k-\lambda}$ is replaced by $D^{k-\lambda} + \Delta^{k-\lambda}$, where Δ represents the change made by suppressing "less important" components in the "detail" (error) information (e.g. setting to zero any elements less than a threshold). It is easily seen that C^{k+1} is correspondingly changed by

$$P^{k+1} \ldots P^{k-\lambda+2}Q^{k-\lambda+1}\Delta^{k-\lambda} \tag{27}$$

This change is bounded by

$$\| P^{k+1} \| \cdots \| P^{k-\lambda+2} \| \| Q^{k-\lambda+1} \| \| \Delta^{k-\lambda} \| \tag{28}$$

In particular, if the infinity matrix and vector norms are used (maximum absolute row sum of a matrix and maximum absolute element of a vector [13]), and if we arrange to scale the columns of the Q matrices so that $\| Q^{k-i} \| = 1$, then magnitude of the change to any component of C^{k+1} will be no larger than the maximum magnitude of change made to any component of the D vectors, in view of the fact that the infinity norm of each P matrix is 1 (the rows represent affine combinations).

9. Curve Subdivision Rules

We have experimented with four common curve subdivision rules: the cubic B-spline rule that we have used as an example above, *Faber subdivision* [10], *Chaikin subdivision* [3], and *Dyn-Levin-Gregory subdivision* [9] (considered in Section 10).

The subdivision matrix, P^{k+1}, of the Chaikin rule is:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{3}{4} & \frac{1}{4} & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{3}{4} & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \frac{3}{4} & \frac{1}{4} & \cdots & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{3}{4} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{3}{4} & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 & \cdots & \frac{1}{4} & \frac{3}{4} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \tag{29}$$

the Chaikin normal-equation matrix is:

$$\begin{bmatrix} \frac{5}{4} & \frac{1}{4} & 0 & 0 & \cdots & 0 & 0 & 0 \\ \frac{1}{4} & \frac{7}{8} & \frac{3}{8} & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{3}{8} & \frac{5}{4} & \frac{3}{8} & \cdots & 0 & 0 & 0 \\ 0 & 0 & \frac{3}{8} & \frac{5}{4} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{3}{8} & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & \frac{5}{4} & \frac{3}{8} & 0 \\ 0 & 0 & 0 & 0 & \cdots & \frac{3}{8} & \frac{7}{8} & \frac{1}{4} \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{4} & \frac{5}{4} \end{bmatrix} \tag{30}$$

a suitable extension, Q^{k+1}, is:

$$\begin{bmatrix} \frac{1}{2} & 0 & 0 & \cdots & 0 & 0 \\ -1 & 0 & 0 & \cdots & 0 & 0 \\ \frac{3}{4} & \frac{1}{4} & 0 & \cdots & 0 & 0 \\ -\frac{1}{4} & -\frac{3}{4} & 0 & \cdots & 0 & 0 \\ 0 & \frac{3}{4} & \frac{1}{4} & \cdots & 0 & 0 \\ 0 & -\frac{1}{4} & -\frac{3}{4} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{3}{4} & -\frac{1}{4} \\ 0 & 0 & 0 & \cdots & \frac{1}{4} & \frac{3}{4} \\ 0 & 0 & 0 & \cdots & 0 & -1 \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} \end{bmatrix} \tag{31}$$

and a possible submatrix, $\hat{Q}^{k+1}$, is:

$$\begin{bmatrix} -1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \frac{3}{4} & \frac{1}{4} & \cdots & 0 & 0 \\ 0 & 0 & \frac{3}{4} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & \cdots & -\frac{3}{4} & -\frac{1}{4} \\ 0 & 0 & 0 & \cdots & 0 & -1 \end{bmatrix} \tag{32}$$

The subdivision matrix of the Faber rule is:

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \tag{33}$$

the normal-equation matrix of the Faber rule is:

$$\begin{bmatrix} \frac{5}{4} & \frac{1}{4} & 0 & \cdots & 0 & 0 & 0 \\ \frac{1}{4} & \frac{3}{2} & \frac{1}{4} & \cdots & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{3}{2} & \cdots & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{4} & 0 & 0 \\ 0 & 0 & 0 & \cdots & \frac{3}{2} & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{4} & \frac{3}{2} & \frac{1}{4} \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{4} & \frac{5}{4} \end{bmatrix} \tag{34}$$

a suitable orthogonal extension is:

$$\begin{bmatrix} \frac{1}{2} & 0 & 0 & \cdots & 0 & 0 \\ -1 & 0 & 0 & \cdots & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 \\ 0 & -1 & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & \cdots & 0 & 0 \\ 0 & 0 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & -1 \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} \end{bmatrix} \tag{35}$$

and a possible submatrix, $\hat{Q}^{k+1}$, is simply the negative of the identity, corresponding to selecting the even rows of (35) starting at the second.

10. Periodic Subdivision Rules

The discussion so far has concentrated on *nonperiodic* subdivision. The Dyn-Levin-Gregory rule, for example, was originally given without emphasis on boundary data; i.e., as a *periodic* rule (see [21] for a discussion of the boundary). Periodic rules are provided by P matrices having rows/columns that wrap around. For example, the Dyn-Levin-Gregory matrix (with the authors' parameter w set to $\frac{1}{16}$) is:

$$\begin{bmatrix}
0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
-\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & \cdots & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\
-\frac{1}{16} & 0 & 0 & 0 & 0 & \cdots & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\
\frac{9}{16} & -\frac{1}{16} & 0 & 0 & 0 & \cdots & 0 & -\frac{1}{16} & \frac{9}{16} \\
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
\frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & 0 & 0 & \cdots & 0 & 0 & -\frac{1}{16}
\end{bmatrix} \tag{36}$$

and the closed-curve version of the Chaiken rule is:

$$\begin{bmatrix}
\frac{3}{4} & \frac{1}{4} & 0 & 0 & \cdots & 0 & 0 \\
\frac{1}{4} & \frac{3}{4} & 0 & 0 & \cdots & 0 & 0 \\
0 & \frac{3}{4} & \frac{1}{4} & 0 & \cdots & 0 & 0 \\
0 & \frac{1}{4} & \frac{3}{4} & 0 & \cdots & 0 & 0 \\
0 & 0 & \frac{3}{4} & \frac{1}{4} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{3}{4} & \frac{1}{4} \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{4} & \frac{3}{4} \\
\frac{1}{4} & 0 & 0 & 0 & \cdots & 0 & \frac{3}{4} \\
\frac{3}{4} & 0 & 0 & 0 & \cdots & 0 & \frac{1}{4}
\end{bmatrix} \tag{37}$$

The normal-equation matrix is also cyclic. For example, the Dyn-Levin-Gregory normal-equation matrix is:

$$\begin{bmatrix}
\delta & \gamma & -\beta & \alpha & 0 & 0 & 0 & 0 & \alpha & -\beta & \gamma \\
\gamma & \delta & \gamma & -\beta & \alpha & 0 & 0 & 0 & 0 & \alpha & -\beta \\
-\beta & \gamma & \delta & \gamma & -\beta & \alpha & 0 & 0 & 0 & 0 & \alpha \\
\alpha & -\beta & \gamma & \delta & \gamma & -\beta & \alpha & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \alpha & -\beta & \gamma & \delta & \gamma & -\beta & \alpha \\
\alpha & 0 & 0 & 0 & 0 & \alpha & -\beta & \gamma & \delta & \gamma & -\beta \\
-\beta & \alpha & 0 & 0 & 0 & 0 & \alpha & -\beta & \gamma & \delta & \gamma \\
\gamma & -\beta & \alpha & 0 & 0 & 0 & 0 & \alpha & -\beta & \gamma & \delta
\end{bmatrix} \tag{38}$$

where $\alpha = \frac{1}{256}$, $\beta = \frac{9}{128}$, $\gamma = \frac{63}{256}$, and $\delta = \frac{105}{64}$. Such matrices can be broken into LDL^T factors as in the case of open curves. The factors do not have the banded structure of the open-curve normal-equation matrices, but their structure does contain only a small, fixed number of nonzero entries for each row, in regular positions, which is just as good as a purely banded structure. The profile of the upper triangular factor of (38), for example, is:

$$\begin{bmatrix}
X & X & X & X & 0 & 0 & 0 & \cdots & 0 & X & X & X \\
0 & X & X & X & X & 0 & 0 & \cdots & 0 & X & X & X \\
0 & 0 & X & X & X & X & 0 & \cdots & 0 & X & X & X \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & X & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & X & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & X & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & X & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & X & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & X & X \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & X
\end{bmatrix} \tag{39}$$

The general row in (39) has four entries in a band along the diagonal and three entries in the last three columns. Consequently, the generation of these numbers and the backsolution process costs linear time. This is characteristic of the case with periodic P matrices: their corresponding normal-equation matrices are periodic and essentially banded in the style of (38), while their LDL^T factors are banded with an additional number of final columns that are full.

The Q-extension of a periodic matrix is also easy to find. There are no special columns, so the sign-alternation and shifting for alignment of even numbers of elements, as explained in Section 6, can be carried out easily (and periodically). The orthogonal exten-

sion of (36), for example, is:

$$\begin{bmatrix}
\frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & 0 & 0 & \cdots & 0 & 0 & -\frac{1}{16} \\
0 & -1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
-\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & \cdots & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & -1 & 0 \\
-\frac{1}{16} & 0 & 0 & 0 & 0 & \cdots & -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -1 \\
\frac{9}{16} & -\frac{1}{16} & 0 & 0 & 0 & \cdots & 0 & -\frac{1}{16} & \frac{9}{16} \\
-1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0
\end{bmatrix} \tag{40}$$

The best subselection of rows to take from this matrix, of course, consists of all rows having only the entry -1.

Not all subselections are quite so convenient. The periodic, cubic B-spline P matrix is:

$$\begin{bmatrix}
\frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & \cdots & 0 & 0 & 0 \\
0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\
0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \cdots & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & \cdots & 0 & 0 & 0 \\
0 & 0 & \frac{1}{8} & \frac{3}{4} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\
0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{8} & 0 & 0 & 0 & \cdots & 0 & \frac{1}{8} & \frac{3}{4} \\
\frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 & \frac{1}{2} \\
\frac{3}{4} & \frac{1}{8} & 0 & 0 & \cdots & 0 & 0 & \frac{1}{8} \\
\frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0
\end{bmatrix} \tag{41}$$

The Q-extension is:

$$\begin{bmatrix}
-\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\
\frac{1}{8} & \frac{3}{4} & \frac{1}{8} & 0 & \cdots & 0 & 0 & 0 \\
0 & -\frac{1}{2} & -\frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\
0 & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} & \cdots & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\
0 & 0 & 0 & 0 & \cdots & 0 & -\frac{1}{2} & -\frac{1}{2} \\
\frac{1}{8} & 0 & 0 & 0 & \cdots & 0 & \frac{1}{8} & \frac{3}{4} \\
-\frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 & -\frac{1}{2} \\
\frac{3}{4} & \frac{1}{8} & 0 & 0 & \cdots & 0 & 0 & \frac{1}{8}
\end{bmatrix} \tag{42}$$

No subselected matrix, $\hat{Q}$, is triangular. The following matrix is subselected to be periodic and is made triangular by trivially precomputable eliminations:

$$\begin{bmatrix}
-\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & -\frac{1}{2} & -\frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & -\frac{1}{2} & -\frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & -\frac{1}{2} & -\frac{1}{2} \\
-\frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 & -\frac{1}{2}
\end{bmatrix} \tag{43}$$

A non-periodic $\hat{Q}$ is easier to make triangular. The following subselection of rows requires only one Gaussian elimination:

$$\begin{bmatrix}
-\frac{1}{2} & -\frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & -\frac{1}{2} & -\frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & -\frac{1}{2} & -\frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\
0 & 0 & 0 & 0 & \cdots & 0 & -\frac{1}{2} & -\frac{1}{2}
\end{bmatrix} \tag{44}$$

As a final example, we give our analog of the Daubechies D_4 relationships. This must be regarded as a freak of nature, of course, and it is simply included to show that it can be done. Daubechies' A and B matrices (see Section 13) for D_4 are each sparse with 4 elements per row. Her Q is different from ours, but also contains only 4 elements per column. She needs no LDL^T factorization, whereas we do. In order to achieve this sparsity, on the other hand, her scale functions (Section 13 again) are negative on portions of their support, making them inappropriate for certain aspects of geometric use such as interactive modeling.

The P matrix for closed curves (re-scaled to provide affine combinations) is:

$$\begin{bmatrix} \frac{3-\sqrt{3}}{4} & \frac{1+\sqrt{3}}{4} & 0 & \cdots & 0 \\ \frac{1-\sqrt{3}}{4} & \frac{3+\sqrt{3}}{4} & 0 & \cdots & 0 \\ 0 & \frac{3-\sqrt{3}}{4} & \frac{1+\sqrt{3}}{4} & \cdots & 0 \\ 0 & \frac{1-\sqrt{3}}{4} & \frac{3+\sqrt{3}}{4} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1+\sqrt{3}}{4} \\ 0 & 0 & 0 & \cdots & \frac{3+\sqrt{3}}{4} \\ \frac{1+\sqrt{3}}{4} & 0 & 0 & \cdots & \frac{3-\sqrt{3}}{4} \\ \frac{3+\sqrt{3}}{4} & 0 & 0 & \cdots & \frac{1-\sqrt{3}}{4} \end{bmatrix} \tag{45}$$

The normal-equation matrix is cyclic, with the following column pattern shifted down one row position for each advance of column position:

$$\begin{bmatrix} 0 \\ \vdots \\ 0 \\ \left(\frac{3-\sqrt{3}}{4}\right)\left(\frac{1+\sqrt{3}}{4}\right)+\left(\frac{1-\sqrt{3}}{4}\right)\left(\frac{3+\sqrt{3}}{4}\right) \\ \left(\frac{3-\sqrt{3}}{4}\right)^2+\left(\frac{1-\sqrt{3}}{4}\right)^2+\left(\frac{1+\sqrt{3}}{4}\right)^2+\left(\frac{3+\sqrt{3}}{4}\right)^2 \\ \left(\frac{3-\sqrt{3}}{4}\right)\left(\frac{1+\sqrt{3}}{4}\right)+\left(\frac{1-\sqrt{3}}{4}\right)\left(\frac{3+\sqrt{3}}{4}\right) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{46}$$

Our corresponding Q matrix is:

$$\begin{bmatrix} -\frac{3+\sqrt{3}}{4} & -\frac{1-\sqrt{3}}{4} & 0 & \cdots & 0 \\ \frac{1+\sqrt{3}}{4} & \frac{3-\sqrt{3}}{4} & 0 & \cdots & 0 \\ 0 & -\frac{3+\sqrt{3}}{4} & -\frac{1-\sqrt{3}}{4} & \cdots & 0 \\ 0 & \frac{1+\sqrt{3}}{4} & \frac{3-\sqrt{3}}{4} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1-\sqrt{3}}{4} \\ 0 & 0 & 0 & \cdots & \frac{3-\sqrt{3}}{4} \\ -\frac{1-\sqrt{3}}{4} & 0 & 0 & \cdots & -\frac{3+\sqrt{3}}{4} \\ \frac{3-\sqrt{3}}{4} & 0 & 0 & \cdots & \frac{1+\sqrt{3}}{4} \end{bmatrix} \tag{47}$$

and a suitable subselection, $\hat{Q}$, consists of every other row.

11. Tensor-Product Surfaces

For a tensor-product surface, the subdivision process given by equation (4) becomes

$$\left[P_L^{k+1}\right]\left[C^k\right]\left[P_R^{k+1\,T}\right] = \left[C^{k+1}\right] \tag{48}$$

where P_L^{k+1} is $m_L \times n_L$, P_R^{k+1} is $m_R \times n_R$, C^k is $n_L \times n_R$, and C^{k+1} is $m_L \times m_R$. Completing each P matrix by a corresponding Q matrix yields

$$\left[P_L^{k+1} Q_L^{k+1}\right] \begin{bmatrix} C^k & F^k \\ G^k & H^k \end{bmatrix} \begin{bmatrix} P_R^{k+1\,T} \\ Q_R^{k+1\,T} \end{bmatrix} \tag{49}$$

where Q_L^{k+1} is $m_L \times m_L - n_L$ and Q_R^{k+1} is $m_R \times m_R - n_R$. Accordingly, the relationship between coarse and fine levels is composed of four terms:

$$\begin{aligned} \left[C^{k+1}\right] = & \left[P_L^{k+1}\right]\left[C^k\right]\left[P_R^{k+1\,T}\right] \\ & + \left[Q_L^{k+1}\right]\left[G^k\right]\left[P_R^{k+1\,T}\right] \\ & + \left[P_L^{k+1}\right]\left[F^k\right]\left[Q_R^{k+1\,T}\right] \\ & + \left[Q_L^{k+1}\right]\left[H^k\right]\left[Q_R^{k+1\,T}\right] \end{aligned} \tag{50}$$

Finding coarse versions of C^{k+1} may proceed with some flexibility. A *left coarse version* is produced by solving

$$\left(\left[P_L^{k+1}\right]^T\left[P_L^{k+1}\right]\right)\left[C_L^k\right] = \left[P_L^{k+1}\right]^T\left[C^{k+1}\right] \tag{51}$$

column by column for C_L^k. The corresponding error is given by:

$$\begin{aligned} & \left(\left[Q_L^{k+1}\right]^T\left[Q_L^{k+1}\right]\right)\left[D_L^k\right] \\ & \quad = \left[Q_L^{k+1}\right]^T\left(\left[C^{k+1}\right] - \left[P_L^{k+1}\right]\left[C_L^k\right]\right) \end{aligned} \tag{52}$$

which may be found by solving

$$\left[\hat{Q}_L^{k+1}\right]\left[D_L^k\right] = \left[C^{k+1}\right] - \left[P_L^{k+1}\right]\left[C_L^k\right] \tag{53}$$

where $\hat{Q}_L^{k+1}$ is the submatrix selected from Q_L^{k+1} as described in Section 7. In terms of the four tensor-product terms

$$\begin{aligned} C_L^k &= \left[C^k\right]\left[P_R^{k+1\,T}\right] + \left[F^k\right]\left[Q_R^{k+1\,T}\right] \\ D_L^k &= \left[G^k\right]\left[P_R^{k+1\,T}\right] + \left[H^k\right]\left[Q_R^{k+1\,T}\right] \end{aligned} \tag{54}$$

Similarly a *right coarse version* of C^{k+1} may be found by solving

$$\left[C_R^k\right]\left(\left[P_R^{k+1}\right]^T\left[P_R^{k+1}\right]\right) = \left[C^{k+1}\right]\left[P_R^{k+1}\right] \tag{55}$$

row by row. The corresponding error, D_R^k, may be found in a way corresponding to D_L^k. The *fully coarse version* of C^{k+1}, C^k, is obtained by finding the left coarse version of C_R^k or the right coarse version of C_L^k.

12. Examples

Several examples are offered here. Figure 1 shows the shoreline of an island off the coast of Norway, given by 512 points around its perimeter. Morten Dæhlen kindly provided the data. Figures 2 through 5 show coarser versions of the shoreline produced by reversing Chaikin subdivision. Each successive coarse version has half the number of points. Figure 6 overlays the 512-point and 32-point version. Figure 7 shows a 512-point version of the coastline reproduced from the 32-point version purely by subdivision; that is, without the error terms. This subdivision curve is overlain on the original data. Figure 8 shows a 512-point version of the coastline reproduced from the 32-point version and all of the intervening points D via the multiresolution reconstruction process given in (7).

The errors, $Q^{k+1}D^k$, produced through this process had the following norms:

$$\begin{array}{ll} \| Q^{512}D^{256} \|: & 0.0063 \\ \| Q^{256}D^{128} \|: & 0.0110 \\ \| Q^{128}D^{64} \|: & 0.0231 \\ \| Q^{64}D^{32} \|: & 0.0341 \end{array}$$

Figure 9 shows the 512-point shoreline and the 32-point version using the reversal process, using cubic B-splines, introduced by Finkelstein and Salesin in [11]. Figure 10 shows the 512-point shoreline and the 32-point version using our reversal process on cubic B-spline subdivision. The corresponding errors were:

$$\begin{array}{ll} \| Q^{512}D^{256} \|: & 0.0063 \\ \| Q^{256}D^{128} \|: & 0.0132 \\ \| Q^{128}D^{64} \|: & 0.0287 \\ \| Q^{64}D^{32} \|: & 0.0448 \end{array}$$

for Finkelstein and Salesin and:

$$\begin{array}{ll} \| Q^{512}D^{256} \|: & 0.0058 \\ \| Q^{256}D^{128} \|: & 0.0139 \\ \| Q^{128}D^{64} \|: & 0.0304 \\ \| Q^{64}D^{32} \|: & 0.0442 \end{array}$$

for our form of cubic B-spline subdivision reversal. Note that all error magnitudes are comparable among the three multiresolution processes exemplified here. A decision to use one or the other would be made on other grounds than on the magnitudes of the error information.

Figure 11 shows a grey-scale image of a fox. Figure 12 shows the same image after two levels of being made coarse through the reverse of tensor-product, cubic B-spline subdivision. Figure 13 is the full reconstruction of the image, and Figure 14 is the reconstruction without error terms. In contrast to the shorline example, which used periodic versions of subdivision and reconstruction, this uses example uses the version defined by the matrices P and Q given in equations (13) and (24).

Figure 15 shows range data of a bust of Victor Hugo. Since we have been taking subdivision rules in as simple a form as possible, we have had to reduce this data somewhat. A typical subdivision rule will transform m_k points into m_{k+1} points. If we are given data that is not exactly m_{k+1} points, we must adjust the number somehow. This is a typical problem familiar from wavelet and FFT decompositions. We have simply reduced the Victor Hugo data by removing all rows from the top of the head downward as needed, and by deleting every other column from the back of the head (where detail is slight and so much data unnecessary.) Thus the data covers the front of the bust with twice the density of that in the back. Figure 16 shows the result of coarse approximation in one direction only via tensor-product, cubic B-spline subdivision. Figure 17 shows coarse approximation for two levels in two directions. Figure 18 is the reconstructed original data fully reconstructed from the surface of Figure 17, and Figure 19 shows a reconstruction that did not include any of the error terms D.

13. Wavelet Connections

The paper by Schröder and Sweldens [17] provides a very compact introduction to wavelets and multiresolution, together with some applications in graphics. Our notation in this paper was, however, adapted from that used by Finkelstein and Salesin [11] for their work on multiresolution cubic B-spline curves. For a more thorough introduction to B-spline wavelets, refer to the book by Chui [4]. For an introduction to wavelets with emphasis on their use in computer graphics see the book by Stollnitz, et. al. [18]. For an extensive general coverage of wavelets, featuring signal processing, see the book by Strang and Nguyen [19].

In the wavelet interpretation, points c_j^κ are interpreted as coefficients of a *basis* of functions ϕ_j^κ for a space V^κ ($\kappa = k, k+1$). Furthermore, $V^k \subset V^{k+1}$. The basis ϕ_j^k for V^k can be completed to a basis for V^{k+1} by adding independent functions ψ_ℓ^k. The functions ϕ are known as *scale functions*, and the functions ψ are called *wavelets* or *wavelet functions*. Taken all together, $V^\kappa, \phi^\kappa, \psi^k$ for $\kappa = k, k+1$, if repeated for a succession $k = 1, 2, 3, \ldots$, form a *multiresolution system*. With respect to an inner product, if the ϕ and the ψ are mutually orthogonal ($<\phi_i^\kappa, \phi_j^\kappa> = \delta_{i,j}$ ($\kappa = k, k+1$), $<\phi_i^k, \psi_\ell^k> = 0$, $<\psi_\ell^k, \psi_r^k> = \delta_{\ell,r}$) the multiresolution system is said to be *orthogonal*. If only

$<\phi_i^k, \psi_\ell^k> = 0$ holds, the system is *semiorthogonal*. A third category of system is *biorthogonal* (refer to [17] for a definition). Biorthogonal systems will not concern us here.

Since the spaces are nested, we have

$$\phi_j^k = \sum_i p_{i,j}^{k+1} \phi_i^{k+1} \tag{56}$$

$$\psi_\ell^k = \sum_i q_{i,\ell}^{k+1} \phi_i^{k+1} \tag{57}$$

and conversely

$$\phi_i^{k+1} = \sum_j a_{j,i}^{k+1} \phi_j^k + \sum_\ell b_{\ell,i}^{k+1} \psi_\ell^k \tag{58}$$

for some coefficients $a_{j,i}^{k+1}, b_{\ell,i}^{k+1}, p_{i,j}^{k+1}, q_{i,\ell}^{k+1}$. Thus, for any element of V^{k+1} we can write

$$\begin{aligned} \textstyle\sum_i c_i^{k+1}\phi_i^{k+1} &= \textstyle\sum_i c_i^{k+1}\left(\sum_j a_{j,i}^{k+1}\phi_j^k + \sum_\ell b_{\ell,i}^{k+1}\psi_\ell^k\right) \\ &= \textstyle\sum_j \left(\sum_i c_i^{k+1} a_{j,i}^{k+1}\right)\phi_j^k \\ &\quad + \textstyle\sum_\ell \left(\sum_i c_i^{k+1} b_{\ell,i}^{k+1}\right)\psi_\ell^k \\ &= \textstyle\sum_j c_j^k \phi_j^k + \sum_\ell d_\ell^k \psi_\ell^k \end{aligned} \tag{59}$$

This provides the *decomposition* process

$$c_j^k = \sum_i c_i^{k+1} a_{j,i}^{k+1} \tag{60}$$

$$d_\ell^k = \sum_i c_i^{k+1} b_{\ell,i}^{k+1} \tag{61}$$

whereby $\sum_j c_j^k \phi_j^k$ is the approximation of $\sum_i c_i^{k+1}\phi_i^{k+1}$ in V^k and $\sum_i c_i^{k+1} b_{\ell,i}^{k+1}$ is the corresponding error in $W^k = V^{k+1} \setminus V^k$. The c are the *scale coefficients*, and the d are the *error* (or *detail*) *coefficients*. In matrix terms,

$$\begin{bmatrix} A^{k+1} \\ B^{k+1} \end{bmatrix} \left[C^{k+1}\right] = \begin{bmatrix} C^k \\ D^k \end{bmatrix} \tag{62}$$

Conversely,

$$\begin{aligned} \textstyle\sum_j c_j^k\phi_j^k &+ \textstyle\sum_\ell d_\ell^k\psi_\ell^k \\ &= \textstyle\sum_j c_j^k \left(\sum_i p_{i,j}^{k+1}\phi_i^{k+1}\right) \\ &+ \textstyle\sum_\ell d_\ell^k \left(\sum_i q_{i,\ell}^{k+1}\phi_i^{k+1}\right) \\ &= \textstyle\sum_i \left(\sum_j c_j^k p_{i,j}^{k+1} + \sum_\ell d_\ell^k q_{i,\ell}^{k+1}\right)\phi_i^{k+1} \\ &= \textstyle\sum_i c_i^{k+1}\phi_i^{k+1} \end{aligned} \tag{63}$$

This provides the *reconstruction* process

$$c_i^{k+1} = \sum_j c_j^k p_{i,j}^{k+1} + \sum_\ell d_\ell^k q_{i,\ell}^{k+1} \tag{64}$$

which in matrix format is equation (5).

The ideal situation in wavelets is to have the matrices A, B, P, and Q be *sparse*. More specifically, they should each possess only a small number of nonzeros clustered together in each column. In this case the decomposition and reconstruction processes will be extremely fast and efficient. We cannot achieve sparsity. The A matrix is easily seen to be $\left(P^T P\right)^{-1} P^T$, which will invariably be a full matrix. Equation (12) will show that a like comment holds for B and $\left(Q^T Q\right)^{-1} Q^T$. However, we can still achieve speed and efficiency by avoiding the computation of the inverse, as has been discussed in Section 5.

The equations in this section also lead to a matrix view of what constitutes a semiorthogonal system:

$$\begin{aligned} 0 &= \left\langle \phi_i^k, \psi_j^k \right\rangle = \left\langle \textstyle\sum_r p_{r,i}^{k+1}\phi_r^{k+1}, \sum_s q_{s,j}^{k+1}\phi_s^{k+1} \right\rangle \\ &= \textstyle\sum_r \sum_s p_{r,i}^{k+1} q_{s,j}^{k+1} \left\langle \phi_r^{k+1}, \phi_s^{k+1} \right\rangle \\ &= \left[P^{k+1\,T}\right]\left[\Phi^{k+1}\right]\left[Q^{k+1}\right] \end{aligned} \tag{65}$$

where Φ^{k+1} is the *Gram matrix* formed by the scale functions for V^{k+1} and the inner product $<\cdot,\cdot>$. Equations (71) and (70) will indicate how a change in the Gram matrix can change the support of the wavelets represented by Q^{k+1}.

The matrix observations we made in equation (23) are related to the observations in the literature on 2-scale relationships that associate $\phi(x) = \sum_n p_n \phi(2x - n)$ with $\psi(x) = \sum_n (-1)^n p_{-n} \phi(2x - n)$.

Note that the subdivision matrix of equation (13) is the same one used by Finkelstein and Salesin in [11]. However, our matrix Q^{k+1} is significantly simpler than the one presented in that paper. More surprisingly, since the columns of Q^{k+1} provide the representation of the level-k wavelets, ψ_ℓ^k, in terms of the scale functions at level $k+1$, ϕ_i^{k+1}, and since the scale functions in question for this example are cubic B-splines, we see that (24) defines wavelets that (except for the special ones at the extremes of the domain) have the same support as the level-k scale functions. Since Chui [4] proves that the minimal-support B-spline wavelet must have support that is essentially *twice* that of the B-splines at the corresponding level, we have some explaining to do.

The reason for the unusually compact wavelets we are defining lies in the form of semiorthogonality we are using. Since we do not assume that we know the underlying scale functions, we have constructed Q^{k+1}

to satisfy

$$\left[P^{k+1\,T}\right]\left[Q^{k+1}\right] = 0 \tag{66}$$

which, while it is appropriate for least-squares data fitting, agrees with equation (65) only if the Gram matrix Φ^{k+1} is the identity. This, in turn, derives from the fact that we are using a different inner product than the usual one. In [4], as in most literature on wavelets, development proceeds in the space L_2; that is, the inner product used is

$$\langle f, g\rangle = \int_{-\infty}^{\infty} f(z)\bar{g}(z)dz \tag{67}$$

In our case, the inner product is defined implicitly to yield the following relationships:

$$\left\langle \phi_i^{k+1}, \phi_j^{k+1}\right\rangle_{k+1} = \delta_{i,j} \tag{68}$$

which defines the inner product on V^{k+1} by

$$\begin{aligned} \langle f, g\rangle_{k+1} &= \textstyle\sum_\gamma f_\gamma g_\gamma \\ &\text{where} \\ f &= \textstyle\sum_\gamma f_\gamma \phi_\gamma^{k+1} \\ g &= \textstyle\sum_\gamma g_\gamma \phi_\gamma^{k+1} \end{aligned} \tag{69}$$

The inner product is flagged with the subscript "$k+1$" because it is a different inner product for each space V^{k+1} in the nested spaces of the multiresolution analysis and corresponds to the Euclidian inner product on each such space; that is, the inner product appropriate for least-squares data fitting.

The claim that the difference between our wavelets and the conventional ones is due to the inner product being used can be strengthened. Referring to equation (65), the conventional matrix Φ^{k+1} for the cubic B-spline case (whose elements are the integrals of equation (69) with f and g replaced by pairs of basis splines on the closed, bounded interval), has (aside from some special columns on the left and right) the following general columns:

$$\begin{bmatrix} \vdots & \vdots & \vdots \\ 0 & 0 & 0 \\ \frac{1}{5040} & 0 & 0 \\ \frac{1}{42} & \frac{1}{5040} & 0 \\ \frac{397}{1680} & \frac{1}{42} & \frac{1}{5040} \\ \frac{151}{315} & \frac{397}{1680} & \frac{1}{42} \\ \frac{397}{1680} & \frac{151}{315} & \frac{397}{1680} \\ \frac{1}{42} & \frac{397}{1680} & \frac{151}{315} \\ \frac{1}{5040} & \frac{1}{42} & \frac{397}{1680} \\ 0 & \frac{1}{5040} & \frac{1}{42} \\ 0 & 0 & \frac{1}{5040} \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{70}$$

The product $\left[P^{k+1\,T}\right]\left[\Phi^{k+1}\right]$ yields a "smeared out" version of P^{k+1} with (aside from some special columns on the left and right) a succession columns as follows:

$$\begin{bmatrix} \vdots & \vdots & \vdots \\ 0 & 0 & 0 \\ \frac{1}{40320} & 0 & 0 \\ \frac{31}{10080} & 0 & 0 \\ \frac{559}{13440} & \frac{1}{40320} & 0 \\ \frac{247}{1260} & \frac{31}{10080} & 0 \\ \frac{9241}{20160} & \frac{559}{13440} & \frac{1}{40320} \\ \frac{337}{560} & \frac{247}{1260} & \frac{31}{10080} \\ \frac{9241}{20160} & \frac{9241}{20160} & \frac{559}{13440} \\ \frac{247}{1260} & \frac{337}{560} & \frac{247}{1260} \\ \frac{559}{13440} & \frac{9241}{20160} & \frac{9241}{20160} \\ \frac{31}{10080} & \frac{247}{1260} & \frac{337}{560} \\ \frac{1}{40320} & \frac{559}{13440} & \frac{9241}{20160} \\ 0 & \frac{31}{10080} & \frac{247}{1260} \\ 0 & \frac{1}{40320} & \frac{559}{13440} \\ 0 & 0 & \frac{31}{10080} \\ 0 & 0 & \frac{1}{40320} \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{71}$$

Using the scheme of producing columns of Q^{k+1} by reversing the order of elements in the columns of (71), alternating their signs, and shifting the result by one row position, we produce precisely the cubic B-spline wavelets given in Chui [4] (up to a scale factor). A similar outcome with respect to the wavelets of Chui, Quack and Weyrich [5 16] is found for the special columns on the left and right. Figure 20 shows a graphic comparison between a conventional wavelet, from a general column of matrix (71), shown as a dashed line, and the least-squares inner-product wavelet, from a general columns of matrix (24), shown as a solid line. (The conventional wavelet has support on the interval $[0, 14]$, but its deviation from zero on the intervals $[0, 2]$ and $[12, 14]$ is so slight that it is covered by the line representing the x-axis.)

14. Summary and Future Work

We have presented an approach to the least squares data fitting to given data of curves, tensor-product surfaces, and images. The fit is specified according to a given subdivision rule. The approach and the computations have been related to the formalisms of wavelets.

We have seen that the least squares data fitting problem, which conventionally takes place in a finite-dimensional ℓ_2 space with a Euclidian inner product (that is, with a Gram matrix $\Phi = I$) can result in wavelets that have smaller support than those found in the conventional L_2-space, semiorthogonal setting (where the Gram matrix will be banded or full).

We have provided a simple, straightforward, matrix-oriented method for constructing wavelets. The method can be used whatever the Gram matrix, and have given an example where it produces wavelets for B-splines consistent with the conventional ones, for the L_2 setting, as well as wavelets of half that support, for the ℓ_2 (data-fitting) setting.

We have shown how matrix factorization and the selection of submatrices assists in providing efficient decomposition and reconstruction algorithms, even though we are working in a semiorthogonal setting for which A and B are full matrices.

We have presented illustrations of our approach for several subdivision rules, looking at both the periodic and compact-domain setting. Finally, we have shown examples of the subdivision-guided least-squares decomposition and reconstruction applied to a parametric curve, a tensor-product surface, and a grey-scale image.

The following questions might be asked:

1. If the inner product used has such an effect on the sparsity of the wavelet matrix, Q, could it also influence the sparsity of A and B?
2. How can the methods of this paper be extended to non-tensor-product surfaces?
3. The considerations of this paper are addressed entirely to the *static* case of generating multiresolution representations of given data that are related to a specified subdivision rule. How do these representations perform dynamically; specifically, under hierarchical editing?
4. To what extent can these techniques be used for data compression?

We have been addressing the first question in two ways. By leaving the inner product initially unspecified and considering only local conditions like equation (17), we have found a construction process for a sparse A, B, and Q (for given sparse P) that yields an inner product implicitly [2]. We have also found a process that, given P and Φ, will modify Φ (that is, revise the inner product) so as to yield a sparse A, B, and Q. These results will be submitted in forthcoming papers.

The construction from local conditions that we have developed in view of the first question is applicable to non-tensor-product surfaces, which thereby provides an answer to the second question. This, too, will be submitted in a forthcoming paper.

We have produced an experimental curve editor in *Java* that accepts a subdivision rule as a module, creates sparse A, B, and Q matrices on the fly and then allows the design and editing of curves at multiple levels of detail. This work is currently being written up as a University of Waterloo Master's Thesis.

We have not currently addressed the forth question.

Acknowledgments

The authors are indebted to Guenther Greiner of the University of Erlangen in Germany for very enlightening discussions at the inception of this work. The second author thanks him and Hans-Peter Seidel for their kind hospitality in Erlangen during these discussions. We also wish to thank Alexander Nicolaou for insightful comments and questions during the later stages of development. The collaboration on this work took place while the first author was visiting the University of Waterloo. His visit was enabled through arrangements made by Prof. Nezam Mahdavi-Amiri of Sharif University of Technology. We are especially grateful to Prof. Mahdavi-Amiri for this. This work has been supported by NATO, by the National Science and Engineering Research Council of Canada, and by the Sharif University of Technology.

References

1. A. Aldroubi, M. Eden, and M. Unser. Discrete spline filters for multiresolutions and wavelets of ℓ_2. *SIAM Journal on Mathematical Analysis*, 25(5):1412–1432, 1994.
2. R. Bartels and F. Samavati. Reversing subdivision rules: Local linear conditions and observations on inner products. Available as `http://www.cgl.uwaterloo.ca/~rhbartel/Papers/LocalLS.ps.gz` and submitted to the Journal of Computational and Applied Mathematics, 1999.
3. G. Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3:346–349, 1974.
4. C. K. Chui. *An Introduction to Wavelets*, volume 1 of *Wavelet Analysis and its Applications*. Academic Press, Inc., 1992.
5. C. K. Chui and E. Quack. Wavelets on a bounded interval. In D. Braess and L. L. Schumaker, editors, *Numerical methods in approximation theory*, volume 9 of *International Series of Numerical Mathematics, 105*, pages 53–75. Birkhäuser, Basel, 1992.
6. W. Dahmen and C. Micchelli. Banded matrices with banded inverses ii: Locally finite decompositions of spline spaces. *Construction approximation*, 9:263–281, 1993.
7. I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications of Pure and Applied Mathematics*, 41:909–996, 1988.
8. I. Daubechies. *Ten Lectures on Wavelets*. CBMS-NSF Regional Conference Series in Applied Mathematics: 61. SIAM, 1992.
9. N. Dyn, D. Levin, and J. Gregory. A 4-point interpolatory subdivision scheme for curve design. *Computer Aided Geometric Design*, 4(4):257–268, 1987.
10. G. Faber. Über stetige functionen. *Mathematische Annalen*, 66:81–94, 1909.
11. Adam Finkelstein and David H. Salesin. Multiresolution curves. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 261–268. ACM SIGGRAPH, ACM Press, July 1994.
12. R. Goldman. Illicit expressions in vector algebra. *ACM Transactions on Graphics*, 4(3):223–243, 1985.
13. G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, second edition, 1989.
14. T. Lyche and K. Mørken. Spline-wavelets of minimal support. In D. Braess and L. L. Schumaker, editors, *Numerical methods in approximation theory*, volume 9 of *International Series of Numerical Mathematics, 105*, pages 177–194. Birkhäuser, Basel, 1992.
15. Tom Lyche and Knut Mørken. Knot removal for parametric B-spline curves and surfaces. *Computer Aided Geometric Design*, 4(3):217–230, November 1987.
16. E. Quack and N. Weyrich. Decomposition and reconstruction algorithms for spline wavelets on a bounded interval. *Applied and Computational Harmonic Analysis*, 1(3):217–231, 1994.
17. Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 161–172. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.
18. E. J. Stollnitz, T. D. DeRose, and D. H. Salesin. *Wavelets for Computer Graphics*. Morgan Kaufmann Publishers, 1996.
19. G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press, 1996.
20. J. Warren. Sparse filter banks for binary subdivision schemes. In T. Goodman and R. Martin, editors, *The Mathematics of Surfaces VII*, pages 427–438. IMA Press, 1996.
21. Denis Zorin, Peter Schroeder, and Wim Sweldens. Interpolating subdivision for meshes with arbitrary topology. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 189–192. ACM SIGGRAPH, Addison Wesley, August 1996. held in New Orleans, Louisiana, 04-09 August 1996.

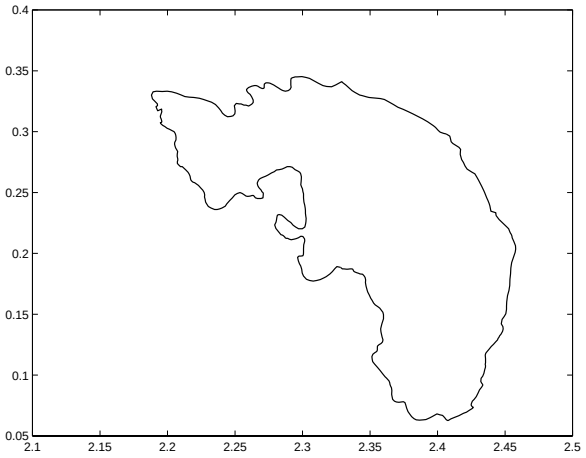

Figure 1: *512-point shoreline*

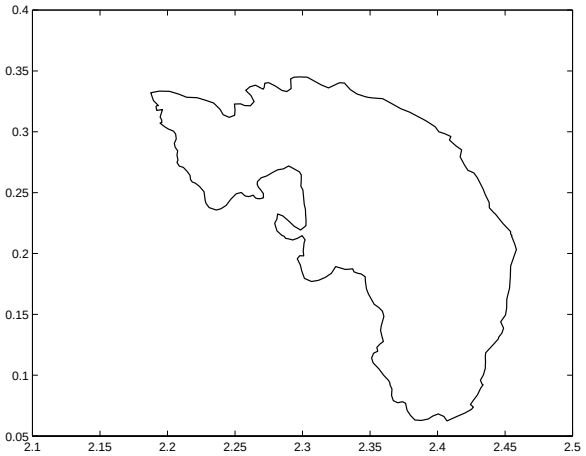

Figure 2: *One Chaikin reversal producing a 256-point shoreline*

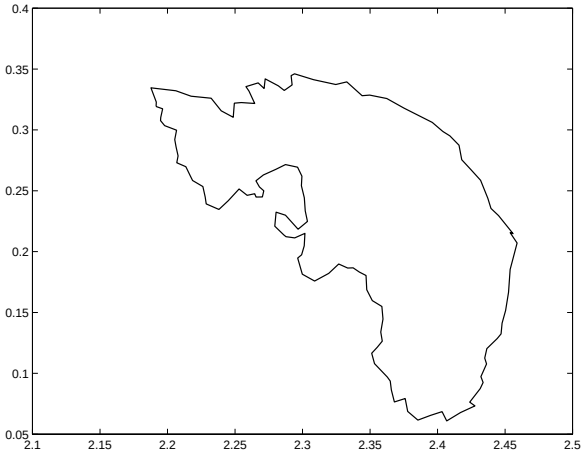

Figure 3: *Two Chaikin reversals producing a 128-point shoreline*

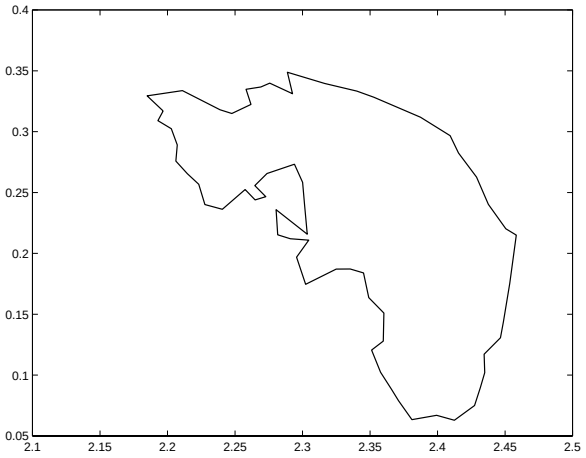

Figure 4: *Three Chaikin reversals producing a 64-point shoreline*

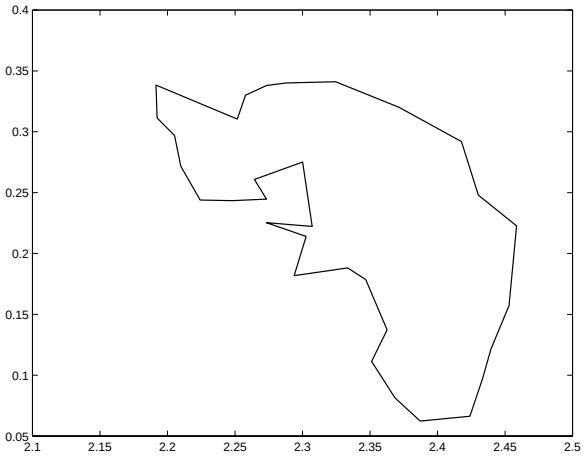

Figure 5: *Four Chaikin reversals producing a 32-point shoreline*

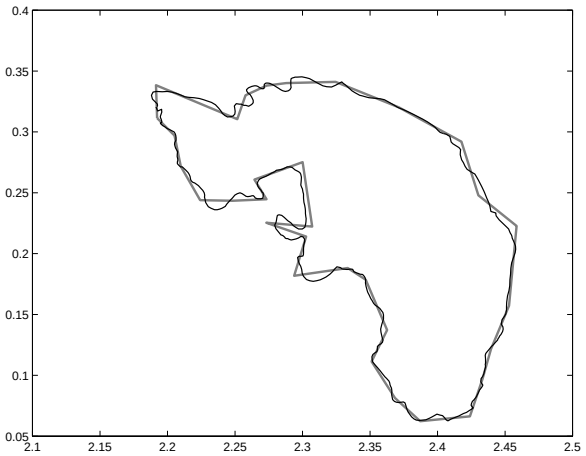

Figure 6: *512-point and 32-point shoreline compared*

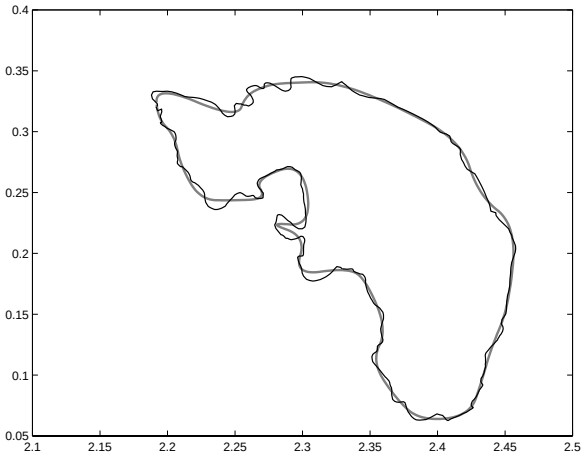

Figure 7: *512-point subdivision curve from 32-point shoreline with original data*

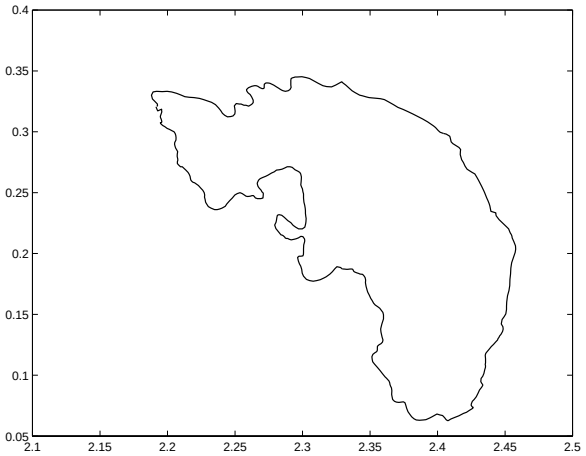

Figure 8: *Original 512-point shoreline reconstructed from 32-point shoreline and error information*

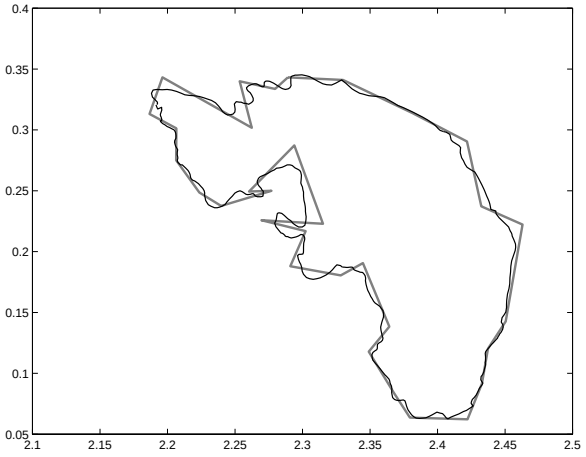

Figure 9: *512-point and 32-point shoreline using Finklestein and Salesin's methods*

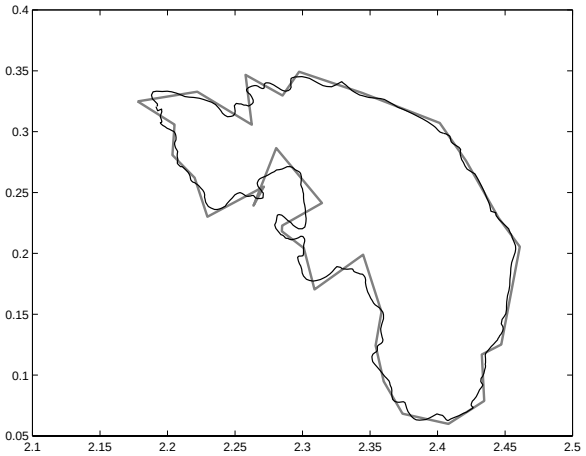

Figure 10: *512-point and 32-point shoreline using our cubic B-spline methods*

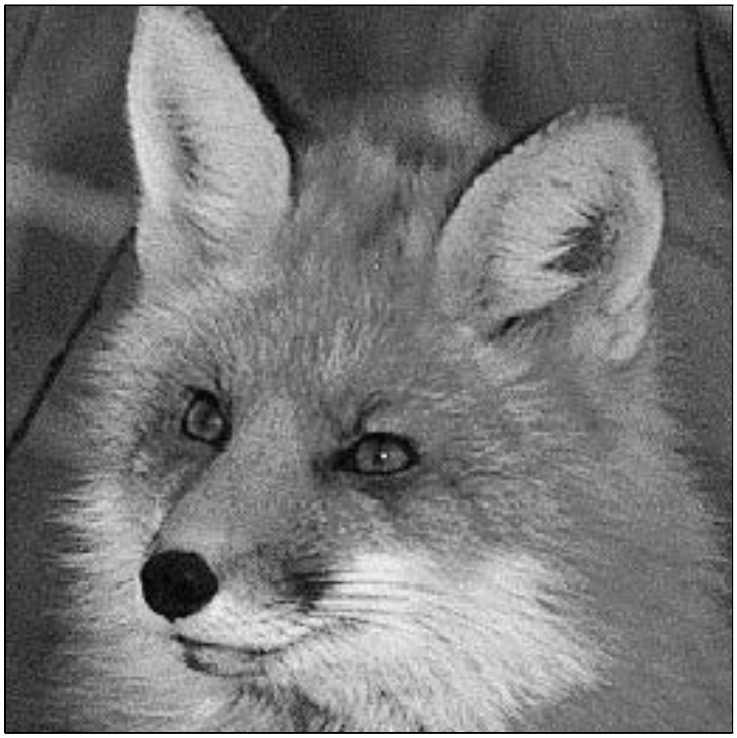

Figure 11: *Original fox image* (259×259)

Figure 12: *Fox image after two levels of approximation* (67×67)

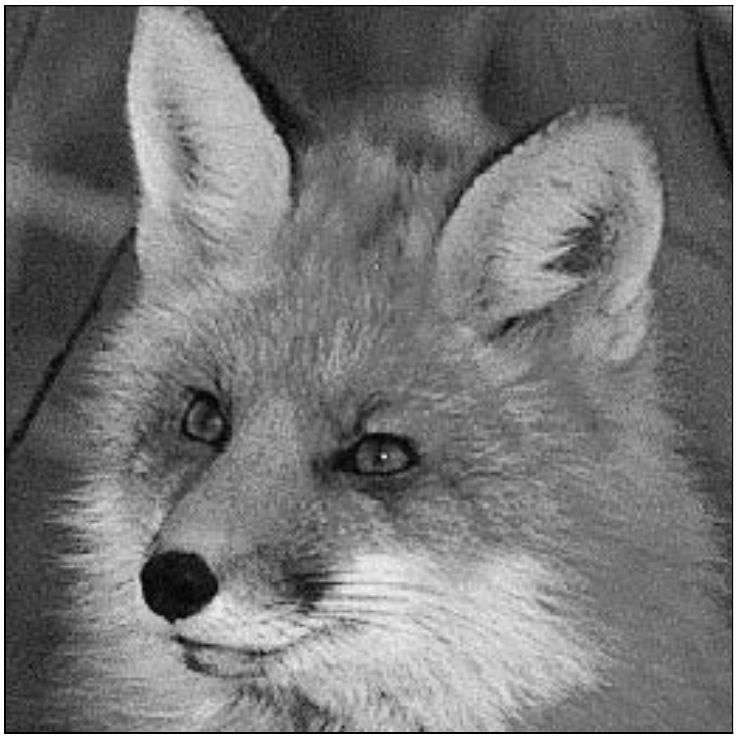

Figure 13: *Fully reconstructed fox image* (259×259)

Figure 14: *Reconstructed fox image without error terms (*259×259*)*

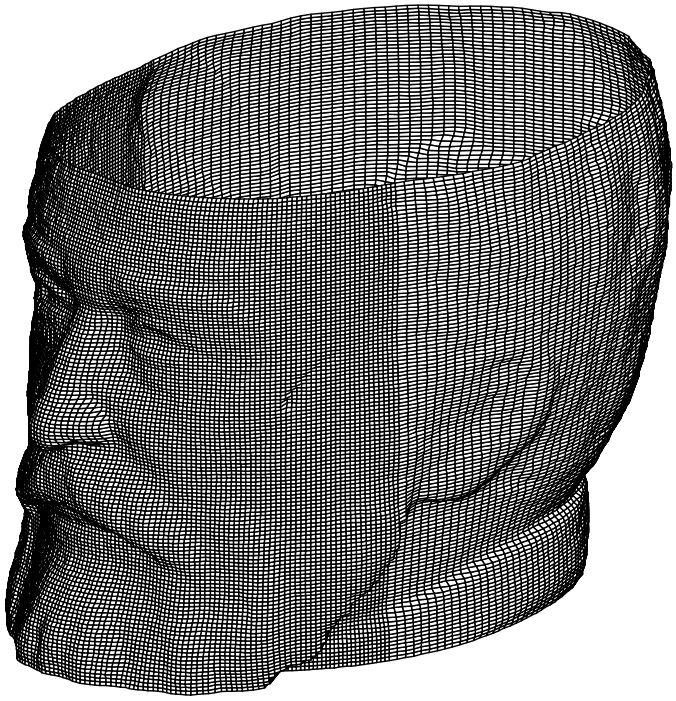

Figure 15: *Original Hugo data (*259×131*)*

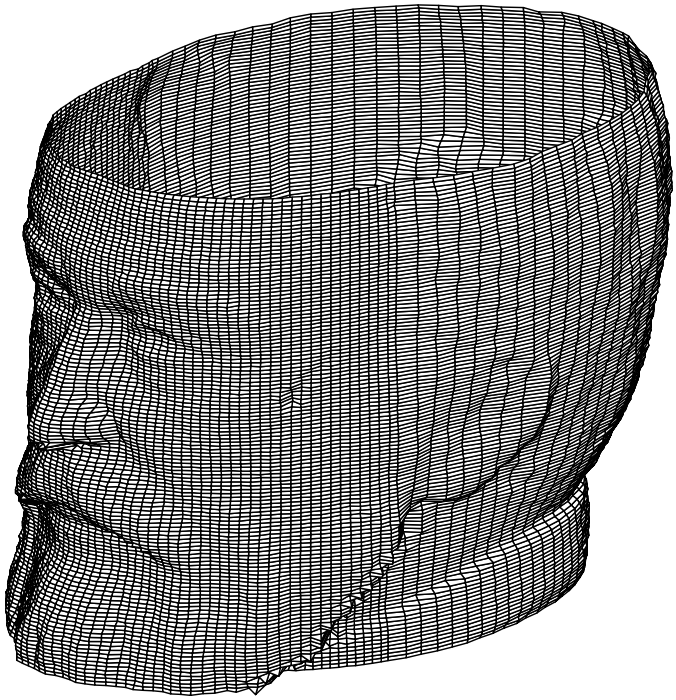

Figure 16: *Hugo data approximated one level in one direction (*131×131*)*

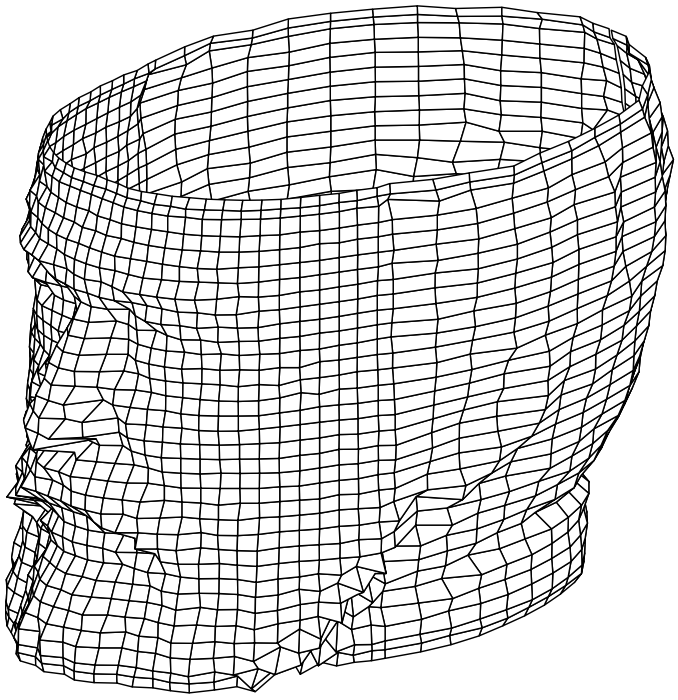

Figure 17: *Hugo data approximated two levels in both directions (*67×35*)*

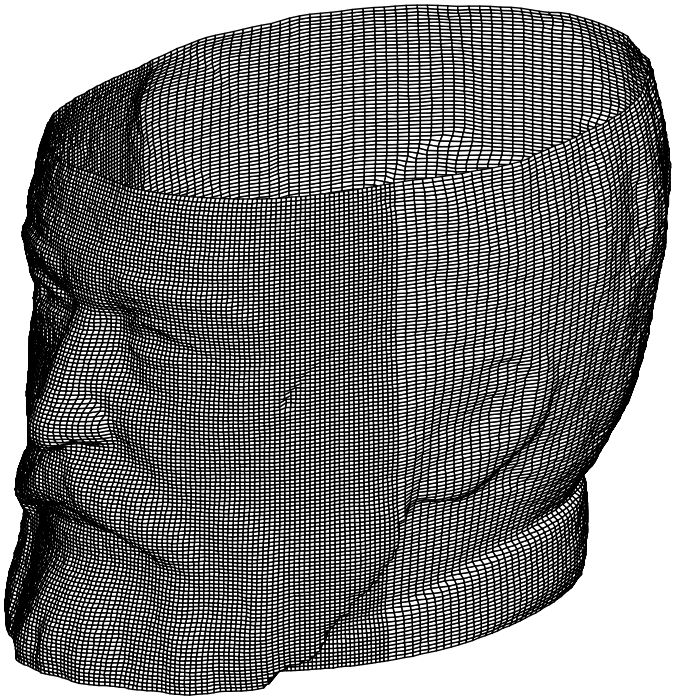

Figure 18: *Fully reconstructed Hugo from two levels (*259×131*)*

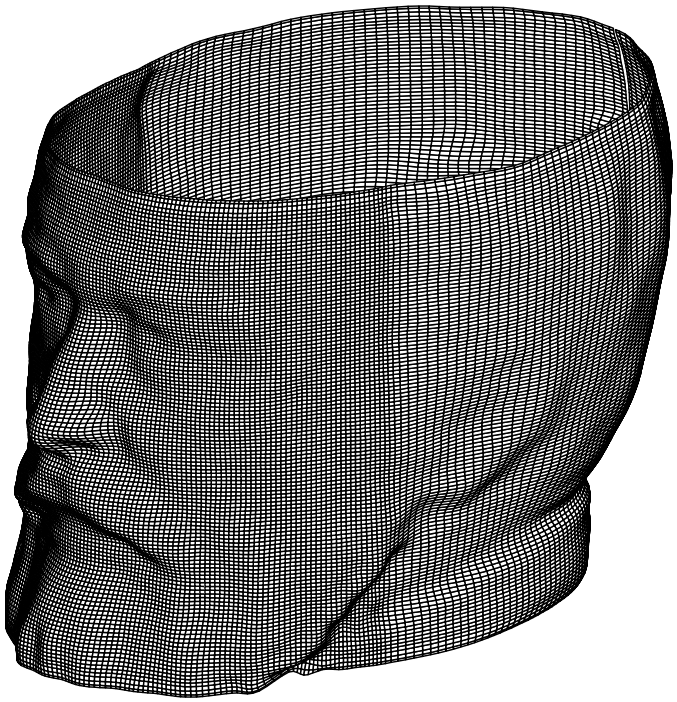

Figure 19: *Reconstructed Hugo from two levels without error terms (*259×131*)*

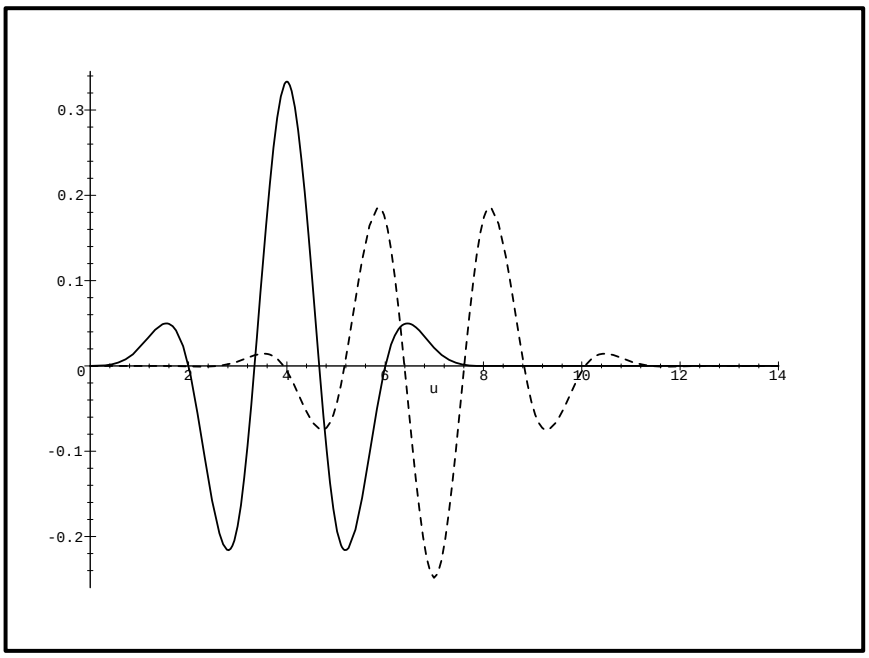

Figure 20: *Comparison of wavelets*