SIAM J. Numer. Anal.
Vol. 40, No. 4, pp. 1369–1404

ENO-WAVELET TRANSFORMS FOR PIECEWISE SMOOTH FUNCTIONS*

TONY F. CHAN† AND H. M. ZHOU‡

Abstract. We have designed an adaptive essentially nonoscillatory (ENO)-wavelet transform for approximating discontinuous functions without oscillations near the discontinuities. Our approach is to apply the main idea from ENO schemes for numerical shock capturing to standard wavelet transforms. The crucial point is that the wavelet coefficients are computed without differencing function values across jumps. However, we accomplish this in a different way than in the standard ENO schemes. Whereas in the standard ENO schemes the stencils are adaptively chosen, in the ENO-wavelet transforms we adaptively change the function and use the same uniform stencils. The ENO-wavelet transform retains the essential properties and advantages of standard wavelet transforms such as concentrating the energy to the low frequencies, obtaining maximum accuracy, maintained up to the discontinuities, and having a multiresolution framework and fast algorithms, all without any edge artifacts. We have obtained a rigorous approximation error bound which shows that the error in the ENO-wavelet approximation depends only on the size of the derivative of the function away from the discontinuities. We will show some numerical examples to illustrate this error estimate.

Key words. ENO, wavelet, image compression, image denoising, signal processing

AMS subject classifications. 65D15, 65T60, 68P30, 94A08

PII. S0036142900370915

1. Introduction.

In this paper, we develop new wavelet algorithms to approximate piecewise continuous functions, for instance piecewise smooth functions connected by large jumps. It is well known that wavelet linear approximation (i.e., truncating the high frequencies) can approximate smooth functions very efficiently: It can achieve high order accuracy by selecting appropriate wavelet basis; it can concentrate the large wavelet coefficients in the low frequencies; and it has a multiresolution framework and associated fast transform algorithms.

Standard wavelet linear approximation techniques cannot achieve similar results for functions which are not smooth, for example piecewise smooth functions with large jumps in function value or in its derivatives. Several problems arise near jumps, primarily caused by the well-known Gibbs phenomenon. The jumps generate large high frequency wavelet coefficients and thus linear approximation cannot get the same high accuracy near the points of discontinuity as in the smooth region. In fact, the jump points generate oscillations which cannot be removed by mesh refinement.

To overcome these problems within the standard wavelet transform framework, nonlinear data-dependent approximations, which selectively retain certain high frequency coefficients, are often used, e.g., hard and soft thresholding techniques; see [6], [15], [19], [18], [27], and corresponding references therein. The main idea of these thresholding approximations is to truncate both low and high frequency wavelet coefficients by their magnitudes, not frequencies. For instance, hard thresholding sets all coefficients whose magnitudes are less than a given tolerance to zero and retains the

*Received by the editors April 18, 2000; accepted for publication (in revised form) February 26, 2002; published electronically October 23, 2002. This research was supported in part by grants ONR-N00017-96-1-0277 and NSF DMS-96-26755.

http://www.siam.org/journals/sinum/40-4/37091.html

†Department of Mathematics, The University of California, Los Angeles, CA 90095-1555 (chan@math.ucla.edu).

‡Department of Applied and Computational Mathematics, Mail Code 217-50, California Institute of Technology, Pasadena, CA 91125 (hmzhou@acm.caltech.edu).

other coefficients unchanged. It has been verified through many research efforts that such nonlinear processes can effectively reduce Gibbs oscillations, and therefore they have been widely used in many applications such as image compression and denoise, and even computation of partial differential equations. However, these techniques often require more complicated data structure to record the location of the retained wavelet coefficients and still cannot remove the effects of the Gibbs phenomenon completely unless all jump-related coefficients are preserved.

Another fundamental approach is to modify the wavelet transform to not generate large high frequency wavelet coefficients near jumps. A few papers in the literature have discussed this approach. Claypoole et al. [12] proposed an adaptive lifting scheme which lowers the order of approximation near jumps, thus minimizing the Gibbs effect. Consequently, this scheme suffers from reduced approximation accuracy near jumps, and some residual Gibbs phenomenon still exists. Another way due to Donoho is to construct orthonormal basis such as wedgelets [16] and ridgelets [7], [17] to represent the discontinuities.

In this paper, we develop a new wavelet algorithm by borrowing the well-developed essentially nonoscillatory (ENO) technique for shock capturing in computational fluid dynamics (e.g., see [23] and [29]) to modify the standard wavelet transform near discontinuities in order to overcome the above-mentioned difficulties. ENO schemes are systematic ways of adaptively defining piecewise polynomial approximations of the given functions according to their smoothness. There are two crucial points in designing ENO schemes. The first is to use one-sided information near jumps and never differencing across the discontinuities. The second is to adaptively form the divided difference table and select the smoothest *stencil* (the support of the basis) for every grid point. ENO schemes lead to uniform high accuracy approximations for each smooth piece of the function. We will use only the first point in our design of the ENO-wavelet transforms. Preliminary results of this work have been reported in [8].

Combining the ENO idea with the multiresolution data representation is a natural way to avoid oscillations in the approximations. In fact, it has been explored by Harten in his general framework of multiresolution in [20], [21], and [22]. (The lifting scheme of Sweldens [31] uses a similar idea.) Recent studies of his general framework and its application in data compression can be found in [2], [3], [4], and [9]. Harten's approach is to directly blend the two ideas and to fully implement the ENO schemes at every point. This consists of using the adaptive ENO finite difference table to select the stencil and then compute the decomposition as well as the reconstruction process. However, his method cannot be directly applied to the more generally used pyramidal filtering algorithms which the standard wavelet transforms are implemented in because in this context we have to work only with fixed size and fixed value filters, and these rigid filters cannot be directly used to compute the adaptive divided difference tables at each grid point.

Our goal is to design a more direct functional replacement of the standard wavelet transforms such that there are no oscillations at the discontinuities in the approximations. We want to stick with the classical pyramidal filtering framework because they are easy to use and have been successfully applied in many applications. Compared to Harten's multiresolution approach, which is more flexible and easier to adaptively implement than the ENO idea, the standard wavelet transforms are more regular and rigid in algorithmic structure; therefore, directly applying the ENO idea would lead to a more drastic perturbation of the underlying pyramidal filtering algorithms. This is the challenge we face.

Conceptually, the ENO-wavelet transforms that we will introduce in this paper are closely related to the ENO implementation of Harten's multiresolution framework. Both methods share the one-sided information idea, which computes the decomposition and reconstruction from smooth data. However, we achieve this in a different manner. The way we accomplish this is to not change the wavelet transforms or the filter coefficients, which most data-dependent multiresolution algorithms do, but instead locally change the function near the discontinuities in such a way that the standard filters are applied only to smooth data. By recording how the changes are made, the original discontinuous function can be exactly recovered by using the original inverse filters. Indeed, by applying the idea of using one-sided information near the discontinuities, we directly extend the functions from both sides of the discontinuities, thus we can apply the standard wavelet transforms on these extended values such that there are no large coefficients generated in the high frequencies and the low frequency approximations are essentially nonoscillatory, and therefore the Gibbs phenomenon can be completely avoided.

In addition, in this modified wavelet transform, the low frequency part preserves the piecewise smoothness of the original function. In particular, the jumps in the low frequency part is not spread widely as in the standard transform. Therefore, the same ENO idea can be recursively used for the coarser levels of the low pass coefficients. By doing so, the multiresolution framework also can be kept.

We show that the resulting wavelet transform retains all the desirable properties of the standard transform: It can have uniformly maximum accuracy, maintained up to the discontinuities (with a rigorous uniform order of the error bound); it concentrates the large coefficients to the low frequencies; it preserves the multiresolution framework and fast transform algorithms; and it is easy to implement. Furthermore, since we do not fully adopt the ENO schemes, in particular we do not build the divided difference table and compare the smoothness of all possible stencils at every point, the extra cost (in floating point operations) required by the modified ENO-wavelet transforms is insignificant. In fact, it is of the order $O(dl)$, where d is the number of discontinuities and $l+1$ the stencil length. Compared to the cost of the standard wavelet transform, which is of the order $O(nl)$, where n is the size of the data, the ratio of the extra cost over that of the standard transform is of the order $O(\frac{d}{n})$ which is independent of l and negligible when n is large.

Besides, since the designed ENO-wavelet transforms play the same role as the standard wavelet transforms in the applications, in principle, any of the numerous existing algorithms for postprocessing wavelet coefficients can also be used in conjunction with the ENO-wavelet coefficients. For example, ENO-wavelet transforms can be used in conjunction with the standard adaptive nonlinear techniques such as hard and soft thresholding, tree structured (e.g., Shapiro's EZW [28]) coders in image compression, and Coifman and Donoho's translation invariant algorithm [10] in denoising. However, in this paper we focus on the construction of ENO-wavelet transforms, and we will not discuss those applications in detail. Instead, we show a numerical example which illustrates the advantages of using the combination of ENO-wavelet transforms with hard thresholding in section 5.

The arrangement of the paper is as follows. In section 2 , we review the standard continuous and discrete wavelet transforms. In section 3, we give a general algorithm to implement the ENO-wavelet transform discretely. In section 4, we prove an error bound for the ENO-wavelet approximation which shows that the error in the ENO-wavelet approximation depends only on the size of the derivative of the function *away* from the discontinuities. Finally, in section 5, we give some numerical examples to illustrate the main advantage of the ENO-wavelet transforms, including some two-dimensional (2-D) examples.

2. Wavelet transforms.

Before we introduce the adaptive ENO-wavelet transforms, we briefly review the standard wavelet transforms; e.g., see [5], [11], [13], [14], [25], [26], [27], and [30]. We use Daubechies orthonormal wavelets as the framework in all discussion in this paper. We will go over both continuous and discrete wavelet transforms, because we will present our ENO-wavelet transforms in the discrete form and prove the approximation error bound by using the continuous form.

First, we review the standard wavelet transforms. To simplify the notation, we assume zeros have been padded to the data at the boundaries.

The standard wavelet transforms are based on translation and dilation. Suppose $\phi(x)$ and $\psi(x)$ are the scaling function and the corresponding wavelet, respectively, with finite support $[0, l]$, where l is a positive integer. It is well known that $\phi(x)$ satisfies the basic dilation equation

$$\phi(x) = \sqrt{2}\sum_{s=0}^{l} c_s \phi(2x - s) \tag{1}$$

and $\psi(x)$ satisfies the corresponding wavelet equation

$$\psi(x) = \sqrt{2}\sum_{s=0}^{l} h_s \phi(2x - s), \tag{2}$$

where the c_s's and h_s's are constants called low pass and high pass filter coefficients, respectively.

We assume that $\psi(x)$ has p vanishing moments

$$\int \psi(x) x^j dx = 0 \qquad \text{for} \quad j = 0, 1, \dots, p-1. \tag{3}$$

We will use the following standard notations:

$$\phi_{j,i}(x) = 2^{\frac{j}{2}} \phi(2^j x - i) \tag{4}$$

and

$$\psi_{j,i}(x) = 2^{\frac{j}{2}} \phi(2^j x - i). \tag{5}$$

Consider the subspace V_j of L^2 defined by

$$V_j = Span\{\phi_{j,i}(x), i \in Z\}$$

and the subspace W_j of L^2 defined by

$$W_j = Span\{\psi_{j,i}(x), i \in Z\}.$$

The subspaces V_j's, $-\infty < j < \infty$, form a multiresolution of L^2 with the subspace W_j being the difference between V_j and V_{j+1}. In fact, the L^2 space has an orthonormal decomposition as

$$L^2 = V_J \oplus \sum_{j=J}^{\infty} W_j. \tag{6}$$

The projection of a L^2 function $f(x)$ onto the subspace V_j is defined by

$$f_j(x) = \sum_i \alpha_{j,i}\phi_{j,i}(x), \tag{7}$$

where

$$\alpha_{j,i} = \int f(x)\phi_{j,i}(x)dx, \qquad i = \cdots, -1, 0, 1, \dots, \tag{8}$$

which we call low frequency wavelet coefficients. (They are often called scaling coefficients in many literatures.) Similarly, we can project $f(x)$ onto W_j by

$$w_j(x) = \sum_i \beta_{j,i}\psi_{j,i}(x), \tag{9}$$

where

$$\beta_{j,i} = \int f(x)\psi_{j,i}(x)dx, \quad i = \cdots, -1, 0, 1, \dots, \tag{10}$$

which we call high frequency wavelet coefficients (often called wavelet coefficients in many literatures). In this paper, we refer to wavelet coefficients as both low and high frequency coefficients. Therefore, the function $f(x)$ can be decomposed by

$$f(x) = f_j(x) + \sum_{t=j}^{\infty} w_t(x). \tag{11}$$

The projection $f_j(x)$ is called the linear approximation of the function $f(x)$ in the subspace V_j.

From (4) and (5), the projection coefficients $\alpha_{j,i}$ and $\beta_{j,i}$ of $f(x)$ in the subspaces V_j and W_j can be easily computed by the so-called fast wavelet transform

$$\alpha_{j,i} = \sum_{s=0}^{l} c_s \alpha_{j+1,2i+s} \tag{12}$$

and

$$\beta_{j,i} = \sum_{s=0}^{l} h_s \alpha_{j+1,2i+s}. \tag{13}$$

In practice, discrete wavelet transforms are often directly used with a set of discrete numbers which are the low frequency coefficients of the L^2 function $f(x)$ at a fine level subspace V_{j+1}. In many applications, this set of numbers are sample values of the function $f(x)$ on a fine grid (although in [30] this is called a "wavelet crime").

Let us define the following matrices:

$$L = \begin{pmatrix} c_0 & c_1 & \cdots & c_l & & & & & \\ & & c_0 & c_1 & \cdots & c_l & & & \\ & & & & \cdots & \cdots & \cdots & & \\ & & & & & c_0 & c_1 & \cdots & c_l \end{pmatrix}$$

and

$$H = \begin{pmatrix} h_0 & h_1 & \cdots & h_l & & & & & \\ & & h_0 & h_1 & \cdots & h_l & & & \\ & & & & \cdots & \cdots & \cdots & & \\ & & & & & h_0 & h_1 & \cdots & h_l \end{pmatrix}.$$

We also denote $\vec{\alpha}_j = (\ldots, \alpha_{j,i}, \alpha_{j,i+1}, \ldots)^T$ and $\vec{\beta}_j = (\ldots, \beta_{j,i}, \beta_{j,i+1}, \ldots)^T$.

By using matrix and vector forms, the fast wavelet transform equations (12) and (13) can be written as

$$\vec{\alpha}_j = L\vec{\alpha}_{j+1} \tag{14}$$

and

$$\vec{\beta}_j = H\vec{\alpha}_{j+1}. \tag{15}$$

It is well known that the wavelet transform matrices L and H are orthogonal:

$$L^*L + H^*H = I. \tag{16}$$

It follows that the inverse wavelet transform is simply

$$\vec{\alpha}_{j+1} = L^*\vec{\alpha}_j + H^*\vec{\beta}_j. \tag{17}$$

The standard linear wavelet approximation achieves maximum accuracy away from discontinuities, but it oscillates near the jumps. The reason for the oscillations is that some stencils cross jumps and cause the corresponding high frequency coefficients to becoming large and therefore more information is lost when the high frequency coefficients are discarded.

In Figure 1, we display a piecewise continuous function (left) and its DB4 wavelet coefficients (right) with low frequencies at the left end and high frequencies at the right end. From the right picture, we see that most of the high frequency coefficients are zero, except for a few large coefficients which are computed near jumps. Figure 2 displays the linear approximation (solid line) compared to the initial function (dotted line). The right picture is the zoom-in to show the approximation behavior near a jump. In this figure, we clearly see oscillations (people call them the Gibbs phenomenon) near discontinuities.

Since the oscillations are generated by discarding large high frequency coefficients which are computed on the stencils crossing discontinuities, to get rid of the oscillations, we want to avoid stencils crossing discontinuities. This motivates us to apply the ENO idea to avoid stencils crossing jumps.

Before we introduce the ENO-wavelet transforms, we give the following definition which we will use in the later sections. Given a function $f(x)$ which has discontinuous set D, then

$$D = \{x_i : f(x) \quad \text{is discontinuous at} \quad x_i\}.$$

Denote t as the closest distance between any two discontinuous points, i.e.,

$$t = \inf\{|x_i - x_j| : x_i, x_j \in D\}.$$

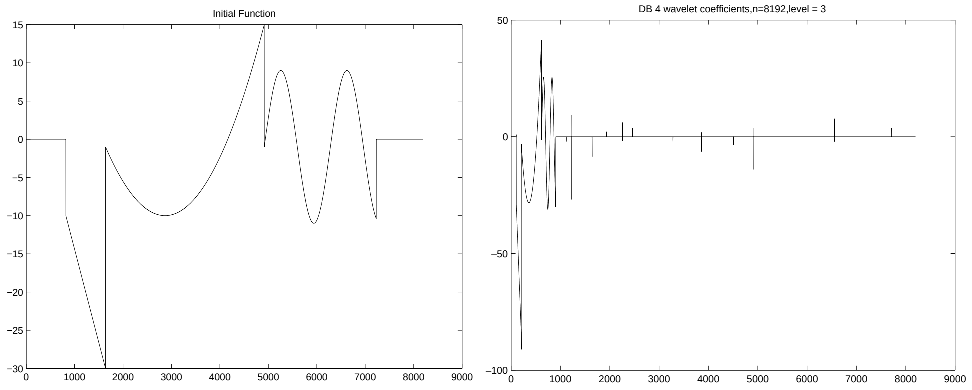

FIG. 1. *The initial function (left) and its DB4 coefficients (right). Most of the high frequency coefficients (right part) are zero except for a few large coefficients computed near the jumps.*

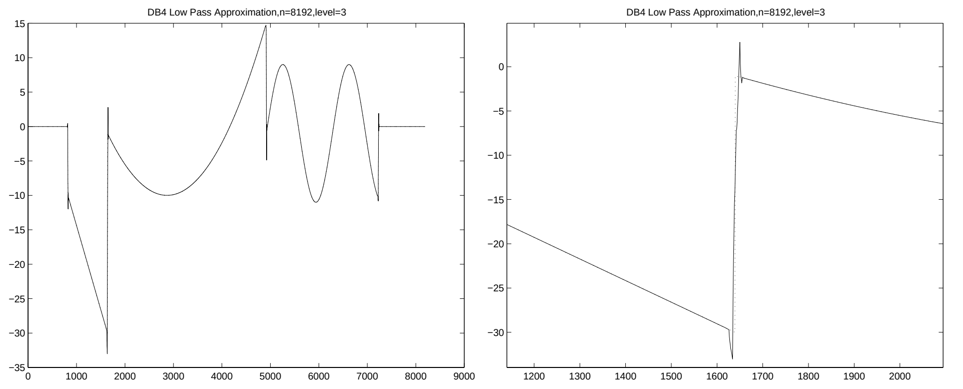

FIG. 2. *The approximation function (left) and its zoom-in (right). Oscillations are generated near the discontinuities in the linear approximation.*

DEFINITION 1. *For a given wavelet filter with stencil length $l + 1$, we say a projection of $f(x)$ in space V_j with spatial step $\Delta x = 2^{-j}$ satisfies the* discontinuity separation property *(DSP) if* $(l + 2)\Delta x < t$.

A projection satisfying the DSP implies that any one discontinuity is located at least one stencil and two data points away from other discontinuities. In other words, there are no two consecutive stencils containing two discontinuities. We assume that all projections we consider in this paper satisfy the DSP. Since our ENO-wavelet transform is essentially using ENO techniques to modify the standard wavelet transform near discontinuities, this property will avoid the modifications near one discontinuity interacting with the modifications near other discontinuities.

Remark. For any piecewise discontinuous function, a projection will satisfy this DSP if j is sufficiently large, i.e., if the discretization is fine enough. On the other hand, at the place where the DSP is invalid, the approximations produced by the ENO-wavelet transforms are comparable to that by the standard wavelet transforms. We will show numerical examples in section 5 illustrating this point.

3. ENO-wavelet transforms.

In this section, we design the ENO-wavelet transforms. In addition to the standard wavelet transforms, our ENO-wavelet transforms are composed of two phases: locating the jumps and forming the approximations at the discontinuities. First, assuming that the location of the jumps are known, we give the ENO-wavelet approximations at the discontinuities by using one-sided information to avoid oscillations. Then we give the methods to detect the exact subinterval on the next finer grid at which the discontinuity is located.

3.1. ENO-wavelet approximation at discontinuities. In this subsection, we assume that the exact subintervals on the next finer grid at which the discontinuities is located are known. We want to modify the standard wavelet transforms near the jumps such that oscillations can be avoided in the approximation. From ENO schemes, we borrow the idea of using one-sided information to form the approximation and avoid applying the wavelet filters crossing the discontinuities. Since we assume the DSP is satisfied by the given projection of the function $f(x)$, we can just consider the local modification near one jump. The main tool which we use to modify the standard wavelet transforms at the discontinuities is function extrapolation in the function spaces or in the wavelet spaces.

Direct function extrapolation. The first way is to extend the function directly at the discontinuity by extrapolation from both sides. Then we can apply the standard wavelet transforms on the extended functions and avoid computing wavelet coefficients using information from both sides.

To maintain the same approximation accuracy near the discontinuity as that for away from the discontinuity, the extrapolation has to be pth order accurate if the wavelet functions have p vanishing moments. For instance, we use constant extrapolation for Haar wavelets and $(p-1)$th order extrapolation for Daubechies-$2p$ orthogonal wavelets which have p vanishing moments.

We use the diagram in Figure 3 to show how to extend the function and compute the ENO-wavelet coefficients.

As shown in Figure 3, the discontinuity is located between $\{x(2i+l-2), x(2i+l-1)\}$. We extend the function from both sides of the discontinuity using $(p-1)$th order extrapolation; i.e., we use the information from the left side of the jump to extrapolate the function over $\hat{x}(2i+l-1), \ldots, \hat{x}(2i+2l-2)$ and use the information from the right side to extrapolate the function over $\bar{x}(2i), \ldots, \bar{x}(2i+l-2)$. And then for $i \leq m \leq i+k-2$, where $l = 2k-1$, we can compute the wavelet coefficients $\hat{\alpha}_{j,m}$ and $\hat{\beta}_{j,m}$ from the left side, and compute $\bar{\alpha}_{j,m}$ and $\bar{\beta}_{j,m}$ from the right side by using the standard wavelet transforms, respectively.

In general, we have the low frequency wavelet coefficients on the finer levels instead of knowing the function values themselves near the discontinuities. We extrapolate these finer level coefficients from both sides of the discontinuities to obtain the values of $\hat{\alpha}_{j+1,m}$ and $\bar{\alpha}_{j+1,m}$, and use the fast wavelet transforms (12) and (13) to compute the coarser level coefficients. For instance, we can compute $\hat{\alpha}_{j,i}$ and $\hat{\beta}_{j,i}$ by

$$\begin{aligned} \begin{pmatrix} \hat{\alpha}_{j,i} \\ \hat{\beta}_{j,i} \end{pmatrix} &= \begin{pmatrix} \sum_{s=0}^{l-2} c_s \alpha_{j+1,2i+s} + c_{l-1}\hat{\alpha}_{j+1,2i+l-1} + c_l \hat{\alpha}_{j+1,2i+l} \\ \sum_{s=0}^{l-2} h_s \alpha_{j+1,2i+s} + h_{l-1}\hat{\alpha}_{j+1,2i+l-1} + h_l \hat{\alpha}_{j+1,2i+l} \end{pmatrix} \\ &\equiv \begin{pmatrix} \delta_{j,i} \\ \gamma_{j,i} \end{pmatrix} + A \begin{pmatrix} \hat{\alpha}_{j+1,2i+l-1} \\ \hat{\alpha}_{j+1,2i+l} \end{pmatrix}, \end{aligned} \tag{18}$$

where $\delta_{j,i}$ and $\gamma_{j,i}$ are $\sum_{s=0}^{l-2} c_s \alpha_{j+1,2i+s}$ and $\sum_{s=0}^{l-2} h_s \alpha_{j+1,2i+s}$, respectively, and depend only on the unextrapolated values of $\alpha_{j+1,m}$, and A a 2×2 matrix defined by

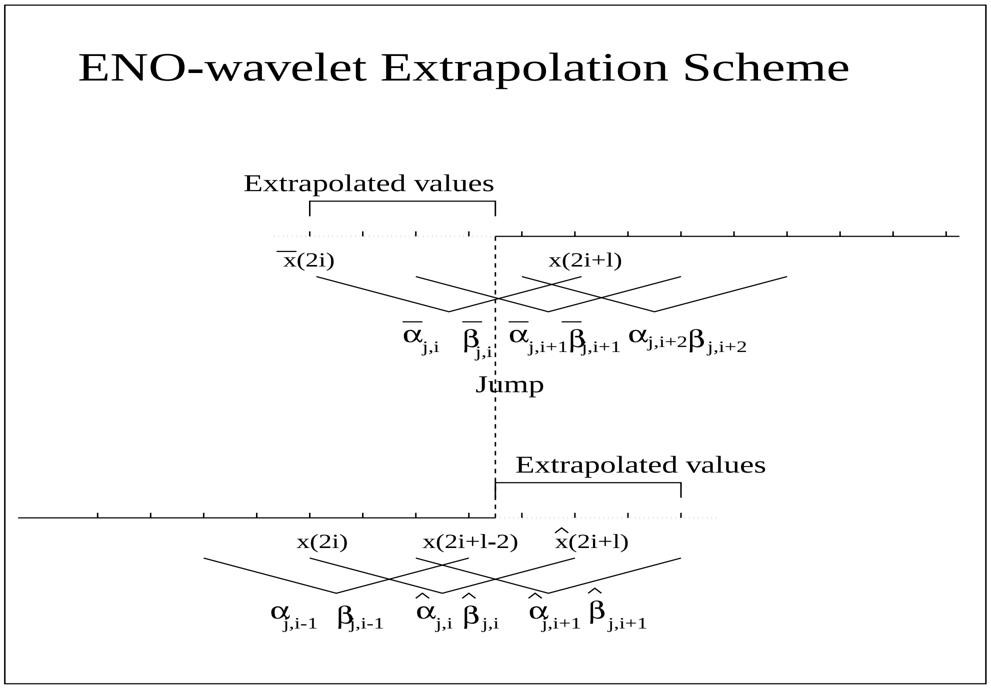
ENO-wavelet Extrapolation Scheme
Extrapolated values
x̄(2i)
x(2i+l)
ᾱj,i β̄j,i ᾱj,i+1 β̄j,i+1 αj,i+2 βj,i+2
Jump
Extrapolated values
x(2i)
x(2i+l-2)
x̂(2i+l)
αj,i-1 βj,i-1 α̂j,i β̂j,i α̂j,i+1 β̂j,i+1

FIG. 3. *Coarse level extrapolation illustration. From the left side of the discontinuity, we extrapolate the low frequency coefficients $\hat{\alpha}_{j,m}$ to determine corresponding high frequency coefficients $\hat{\beta}_{j,m}$ and store them. From the right side of the discontinuity, we extend the high frequency coefficients $\bar{\beta}_{j,m}$ to determine and store the low frequency coefficients $\bar{\alpha}_{j,m}$.*

the filter coefficients as

$$A = \begin{pmatrix} c_{l-1} & c_l \\ h_{l-1} & h_l \end{pmatrix}.$$

In computing $\hat{\alpha}_{j,m}$ and $\hat{\beta}_{j,m}$ by the fast wavelet transforms, the number of extrapolated values we must use is 2 for $m = i$, 4 for $m = i + 1$, and so on. Those extrapolated values are determined from the smooth side of the discontinuity; then the high frequency coefficients $\hat{\beta}_{j,m}$ remain as small values as those of the smooth stencils.

By symmetry, we can compute $\bar{\alpha}_{j,m}$'s and $\bar{\beta}_{j,m}$'s from the right side in a similar way.

There are many methods to extrapolate the extended values. For example, a straightforward way is to use p-point polynomial extrapolation such as Lagrange polynomials or Taylor expansion polynomials. In our numerical experiments in this paper, we use Lagrange polynomial extrapolation. Least square extrapolation can be used too [33], especially for noisy data.

There is a storage problem for this direct function extrapolation. Indeed, it doubles the number of the wavelet coefficients near every discontinuity. To retain the perfect invertible property, we need to store the ENO-wavelet coefficients $\hat{\alpha}_{j,m}$ and $\hat{\beta}_{j,m}$ from the left side, also $\bar{\alpha}_{j,m}$ and $\bar{\beta}_{j,m}$ from the right side. Thus, the output sequences are no longer the same size as the input sequences. In many applications, such as image compression, this extra storage requirement definitely needs to be avoided.

Remark. In the least square extrapolation case, it is possible to reduce the demands of the extra storage because not all the wavelet coefficients $\hat{\alpha}_{j,m}$, $\hat{\beta}_{j,m}$, $\bar{\alpha}_{j,m}$, and $\bar{\beta}_{j,m}$ are linearly independent [33]. However, this requires complicated extra computation.

Our approach: Coarse level extrapolation. To avoid computing the wavelet coefficients using the information from both sides of the discontinuities, to maintain the same high order accuracy near the discontinuities as away from the discontinuities, and also to keep the size of the output sequences the same as that of the input sequences without significant extra computation, we introduce the coarse level extrapolation schemes. The idea is to extrapolate the coarser level wavelet coefficients near the discontinuities instead of the function values or the finer level wavelet coefficients.

We still use Figure 3 to illustrate these schemes. We consider the left side of the jump first.

In the direct function extrapolation case, the computation process is to directly extrapolate the finer level wavelet coefficients $\hat{\alpha}_{j+1,m}$, $(2i+l-1) \leq m \leq (2i+2l-2)$, and then compute the extended coarser level wavelet coefficients $\hat{\alpha}_{j,m}$ and $\hat{\beta}_{j,m}$, $i \leq m \leq (i+k-2)$ using the standard filters. We reverse the order of this process in our coarse level extrapolation. More precisely, we extrapolate the coarser level low frequency coefficients $\hat{\alpha}_{j,m}$ using the known low frequency coefficients from the left, and extend the coarser level high frequency coefficients $\hat{\beta}_{j,m}$ to zero (or some predefined values), and then determine the extended finer level wavelet coefficients. For example, in the direct function extrapolation, we extrapolate finer level values $\hat{\alpha}_{j+1,m}$ and then compute the coarser level coefficients $\hat{\alpha}_{j,i}$ and $\hat{\beta}_{j,i}$ by (18). On the contrary, we can first extend the coarser level coefficients $\hat{\alpha}_{j,i}$ and $\hat{\beta}_{j,i}$ and then determine the finer level values. Indeed, if the matrix A is nonsingular, we can uniquely determine the finer level values by solving (18). In this case, we can prescribe both the coarser level coefficients simultaneously. However, in Daubechies orthogonal wavelet transforms, the matrix A is singular, because

$$\frac{h_{l-1}}{c_{l-1}} = \frac{h_l}{c_l}. \tag{19}$$

Thus, in order to have a solution of (18), we must extend the coarser level coefficients $\hat{\alpha}_{j,i}$ and $\hat{\beta}_{j,i}$ in such a way that they satisfy

$$\begin{pmatrix} \hat{\alpha}_{j,i} \\ \hat{\beta}_{j,i} \end{pmatrix} - \begin{pmatrix} \delta_{j,i} \\ \gamma_{j,i} \end{pmatrix} \in R(A),$$

where $R(A)$ is the range space of A. This requirement implies that

$$\begin{pmatrix} -1 & \frac{c_l}{h_l} \end{pmatrix} \left[\begin{pmatrix} \hat{\alpha}_{j,i} \\ \hat{\beta}_{j,i} \end{pmatrix} - \begin{pmatrix} \delta_{j,i} \\ \gamma_{j,i} \end{pmatrix} \right] = 0,$$

which we can also rewrite as

$$\hat{\beta}_{j,i} = \gamma_{j,i} + \frac{h_l}{c_l}(\hat{\alpha}_{j,i} - \delta_{j,i}) \tag{20}$$

or

$$\hat{\alpha}_{j,i} = \delta_{j,i} + \frac{c_l}{h_l}(\hat{\beta}_{j,i} - \gamma_{j,i}). \tag{21}$$

Therefore, we cannot prescribe both $\hat{\alpha}_{j,i}$ and $\hat{\beta}_{j,i}$ simultaneously. Thus we have two choices:

(1) We can extrapolate the low frequency coefficients $\hat{\alpha}_{j,i}$ first and then determine the corresponding high frequency coefficients $\hat{\beta}_{j,i}$ by (20).

(2) Or we can extend $\hat{\beta}_{j,i}$ to zero first and then determine the corresponding $\hat{\alpha}_{j,i}$ by (21).

Once coefficients $\hat{\beta}_{j,i}$ and $\hat{\alpha}_{j,i}$ are obtained, we can determine the finer level coefficients $\hat{\alpha}_{j+1,2i+l-1}$ and $\hat{\alpha}_{j+1,2i+l}$. Since A is not invertible for Daubechies wavelets, $\hat{\alpha}_{j+1,2i+l-1}$ and $\hat{\alpha}_{j+1,2i+l}$ cannot be uniquely determined by $\hat{\beta}_{j,i}$ and $\hat{\alpha}_{j,i}$. There is one more freedom left to use. (In the case of the discontinuity being located between $\alpha_{j+1,2i+l-1}$ and $\alpha_{j+1,2i+l}$, $\hat{\alpha}_{j+1,2i+l}$ can be uniquely determined.) Indeed, there are many ways to completely determine the values of $\hat{\alpha}_{j+1,2i+l-1}$ and $\hat{\alpha}_{j+1,2i+l}$. For instance, one can simply extend $\hat{\alpha}_{j+1,2i+l-1}$ by any extrapolation technique, such as $(p-1)$th order polynomial extrapolation for smooth data or averaging extrapolation techniques for noisy data (we use them in our numerical experiments in section 5), and then determine $\hat{\alpha}_{j+1,2i+l}$ by $\hat{\beta}_{j,i}$ or $\hat{\alpha}_{j,i}$. Another possible way to uniquely extend the coefficients $\hat{\alpha}_{j+1,2i+l-1}$ and $\hat{\alpha}_{j+1,2i+l}$ on the finer level is to leave this extra freedom to be used in the next stencil by requiring some special desire properties in the next extended coarser level coefficients. This involves slightly more complicated formulation which we will not exploit further in this paper. Thereafter, the above procedure can be repeatedly used to the next stencil to compute $\hat{\beta}_{j,i+1}$ and $\hat{\alpha}_{j,i+1}$ by treating $\hat{\alpha}_{j+1,2i+l-1}$ and $\hat{\alpha}_{j+1,2i+l}$ as known values. By the same principle, all extended coefficients $\hat{\beta}_{j,m}$ and $\hat{\alpha}_{j,m}$ can be calculated.

Remark. We notice that in both cases (20) and (21) the coefficients are computed by applying the standard filters to the extended data which is smooth. This implies that there are no large coefficients generated by them.

Again by symmetry, we have two analogous choices for the right side of the jump.

Using this coarse level extrapolation technique, we can easily solve the storage problem which we have in the direct function extrapolation. In fact, we just need to store the high frequency coefficients $\hat{\beta}_{j,m}$ for choice (1) and the low frequency coefficients $\hat{\alpha}_{j,m}$ for choice (2). In our implementation, we use choice (1) for the left side of the jumps and choice (2) for the right side of the jumps; therefore we store $\hat{\beta}_{j,m}$ and $\bar{\alpha}_{j,m}$ for every m. This satisfies the standard wavelet storage scheme, i.e., storing one low frequency and one high frequency coefficient for every stencil.

Remark. We select choice (1) from the left side of the jumps and choice (2) from the right side because we want to keep half of the output sequence to be α's and half to be β's. It is possible to select choice (1) or choice (2) for both sides of the jumps, but that will not give equal number of α's and β's in the output sequence; also, it may destroy the data structure for the next level decomposition.

Since we know the way we extend the data at the discontinuities, we can easily extrapolate the low frequency coefficients $\hat{\alpha}_{j,m}$ from the left sides of the discontinuities. Using them together with the stored high frequency coefficients $\hat{\beta}_{j,m}$, we can exactly recover data at the left sides by applying the standard inverse filters. Similarly, the data at the right sides of the discontinuities can also be exactly restored.

For each stencil crossing a jump, an extra cost (in floating point operation) is required in the extrapolation of low frequency coefficients, which is of the order $O(1)$ per stencil, and in the computation of the corresponding high and low frequency coefficients by (20) and (21), which is of the order $O(l)$ per stencil. Overall, the extra cost over the standard wavelet transform is of the order $O(dl)$, where d is the number of discontinuities. Compared to the cost of the standard wavelet transform, which is of the order $O(nl)$ where n is the size of data, the ratio of the extra cost over that of the standard transform is $O(\frac{d}{n})$, which is independent of l and negligible when n is large.

3.2. Locating the discontinuities. In the previous subsection, we showed how to modify the standard wavelet transforms at the discontinuities to avoid oscillations if we know the exact subinterval on the next finer grid at which the jumps are located. In this subsection, we introduce the methods to detect those exact subintervals for discontinuities for piecewise smooth functions with and without noise. First we give a method for smooth data.

Piecewise smooth functions. Our purpose is to avoid wavelet stencils crossing discontinuities. Theoretically, a discontinuity can be characterized by comparing the left and right limit of the derivatives $f^{(m)}(x)$ at the given point x; i.e., we call a point x a discontinuity if for some $m < p$ we have

$$f^{(m)}(x-) \neq f^{(m)}(x+).$$

We define the intensity of a jump in the mth derivative at x as

$$[f^{(m)}(x)] = |f^{(m)}(x+) - f^{(m)}(x-)|.$$

It is well known that the high pass filters in wavelet transforms measure the smoothness of functions: they produce smaller values at smoother regions and larger values at rougher regions. In fact, it has been shown in [1], [24], and [32] that if a function $f(x)$ is Lipschitz $\gamma \le p$ at x, i.e., $|f(x+\delta) - f(x)| \le \delta^\gamma$ for any small δ, the corresponding high frequency wavelet coefficients are of the order of $O(\Delta x^\gamma)$. From this, it is easy to obtain that at smooth regions the magnitudes of high frequency coefficients $|\beta_{j,i}|$ have the order of $|f^{(p)}(x)|O(\Delta x^p)$. On the other hand, if a stencil contains a discontinuity, no matter if it is a discontinuity in function value ($m = 0$) or in its mth derivative, the magnitude of the corresponding high frequency coefficient $|\beta_{j,i}|$ is of the order of $O(\Delta x^{(m)})$, which is at least one order lower than that at the smooth regions. Therefore, instead of fully adopting the ENO comparison idea which compares the magnitudes of divided differences on all possible stencils, we can use the magnitudes of the high frequency coefficients as our criterion to identify the discontinuities.

The obvious way, also the cheapest way, to identify the discontinuities is to compare the magnitudes of the high frequency coefficients on the current standard stencils $|\beta_{j,i}|$ with that on the previous standard stencils $|\beta_{j,i-1}|$. Since for smooth functions we have $|\beta_{j,i}| = |f^{(p)}(x)|O(\Delta x^p)$, this implies that at smooth regions, by Taylor expansion, we have

$$|\beta_{j,i}| = (1 + O(\Delta x))|\beta_{j,i-1}|, \tag{22}$$

where the constant in the term $O(\Delta x)$ depends on the size of higher order derivatives of $f(x)$ such as $\max_x |f^{(p+1)}(x)|$. In contrast, the magnitudes of high frequency coefficients $|\beta_{j,i}|$ based on the stencils containing the discontinuities are at least one order lower than that at the smooth regions. More precisely, if we assume function $f(x)$ has a jump in its mth derivatives at point $x_0 \in (i\Delta x, (i+l)\Delta x)$ for some integer i, using Taylor expansion, in a small neighborhood of x_0, we can write this function as

$$f(x) = g(x) + \begin{cases} f^{(m)}(x_0-)(x-x_0)^m + O(x-x_0)^{(m+1)}, & x \le x_0, \\ f^{(m)}(x_0+)(x-x_0)^m + O(x-x_0)^{(m+1)}, & x_0 < x, \end{cases}$$

where $g(x)$ is its Taylor polynomial of order $m-1$ near x_0. Then the wavelet coefficients $\beta_{j,i}$ is estimated by using the vanishing moments property as

$$\begin{aligned}|\beta_{j,i}| &= \left|\int f(x)\psi_{j,i}(x)dx\right| \\ &= \left|\int_{i\Delta x}^{x_0}(f^{(m)}(x_0-)(x-x_0)^m + O(x-x_0)^{m+1})\psi_{j,i}(x)dx\right. \\ &\quad \left. + \int_{x_0}^{(i+l)\Delta x}(f^{(m)}(x_0+)(x-x_0)^m + O(x-x_0)^{m+1})\psi_{j,i}(x)dx\right| \\ &= |[f^{(m)}(x_0)]|O(\Delta x^m).\end{aligned}$$

It depends on the Δx^m and also on the intensity of the jump.

Thus, we can design a method to detect the discontinuities as follows: For each standard stencil, suppose we know that the previous standard stencil does not contain any discontinuities, if we have $|\beta_{j,i}| \le a|\beta_{j,i-1}|$, where $a > 1$ is a given constant, and then we treat the current stencil as a smooth stencil. Otherwise, we conclude that there are discontinuities contained in it.

The choice of constant a depends on the grid size Δx and also on the intensity of the jumps. In fact, the ratio between a high frequency coefficient at the rough regions and that at the smooth regions is of the order of $|[f^{(m)}(x)]|O(\Delta x^{(m-p)})$. When Δx becomes small, this ratio is large. We can choose a as any number such that

$$(1+O(\Delta x)) \le a \le \min_x\{|[f^{(m)}(x)]|O(\Delta x^{(m-p)})\}, \tag{23}$$

provided the above minimal number is larger than $1+O(\Delta x)$. This is always true for piecewise smooth functions with small enough grid size Δx.

Remark. When a jump in the mth derivative has very small intensity less than $O(\Delta x^{(p-m)})$, this jump cannot be detected by the above-described method. However, the error caused by missing this jump is also very small, which is at the same order of the error bound we will give in section 4. In practice, especially when we care only about the jumps in function values, we have a large range to select a.

To completely avoid oscillations, we also need to know the exact locations of the discontinuities so that we can avoid computing the wavelet coefficients crossing them. In fact, the above comparison method based on the magnitudes of high frequency coefficients can also help us to locate the exact positions of the discontinuities. We will use the diagram in Figure 4 to explain how to find the exact jump positions.

Assume we consider the wavelet filters with length $(l+1)$. We compare the magnitude of the high frequency coefficient $|\beta_{j,i}|$ on the current stencil, which starts at $x(2i)$ with $|\beta_{j,i-1}|$ on the previous stencil. If we have $|\beta_{j,i}| > a|\beta_{j,i-1}|$, we identify the discontinuity lying in the current stencil. Since there are no discontinuities in the previous stencils, we know that this discontinuity must be located between $\{x(2i+l-2), x(2i+l)\}$, where it has only two possible positions: between $\{x(2i+l-2), x(2i+l-1)\}$ or between $\{x(2i+l-1), x(2i+l)\}$. In fact, we can determine the exact position of the jump by continuing to compare the subsequent values of $\beta_{j,m}$. As shown in Figure 4, we must have at least $(k-1)$ consecutive "large" $\beta_{j,m}$, $i \le m \le (i+k-2)$, because the subsequent $(k-1)$ stencils also include the discontinuity. We compute $\beta_{j,i+k-1}$ and $\beta_{j,i+k}$ on the corresponding standard stencils, if we have $|\beta_{j,i+k-1}| > a|\beta_{j,i+k}|$, and then we have k consecutive stencils containing the discontinuity, which implies that the discontinuity is located between $\{x(2i+l-1), x(2i+2l-1)\}$ (see Figure 4(a)).

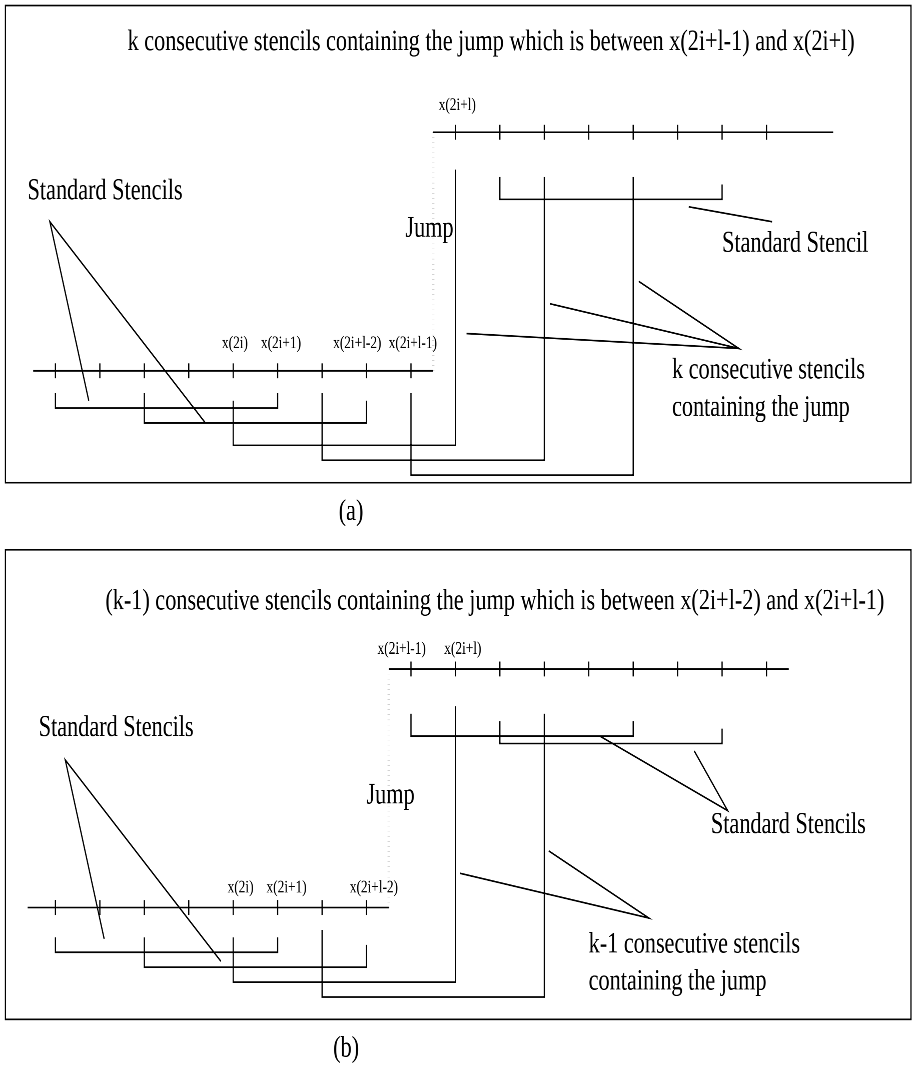
k consecutive stencils containing the jump which is between x(2i+l-1) and x(2i+l)
x(2i+l)
Standard Stencils
Jump
Standard Stencil
x(2i)
x(2i+1)
x(2i+l-2)
x(2i+l-1)
k consecutive stencils containing the jump
(a)
(k-1) consecutive stencils containing the jump which is between x(2i+l-2) and x(2i+l-1)
x(2i+l-1)
x(2i+l)
Standard Stencils
Jump
Standard Stencils
x(2i)
x(2i+1)
x(2i+l-2)
k-1 consecutive stencils containing the jump
(b)

FIG. 4. *Locating the exact position of the jump by counting the number of consecutive stencils containing the jump.* (a) *If k stencils contain the jump, then the jump position is between $x(2i+l-1)$ and $x(2i+l)$.* (b) *If $(k-1)$ consecutive stencils contain the jump, then the jump is located between $x(2i+l-2)$ and $x(2i+l-1)$.*

If we have exactly $(k-1)$ consecutive standard stencils containing the discontinuity, then this implies that the jump must be located between $\{x(2i+l-2), x(2i+l-1)\}$ (see Figure 4(b)). We summarize the above arguments in the following proposition.

PROPOSITION 1. *Consider the wavelet filters with length $l+1$, where $l = 2k-1$. For a given index i, assume we have $|\beta_{j,i-1}| \leq a|\beta_{j,i-2}|$ but $|\beta_{j,i}| > a|\beta_{j,i-1}|$. Then*

(1) *if $|\beta_{j,i+k-1}| > a|\beta_{j,i+k}|$, which means there are k consecutive standard stencils containing the jump, then the discontinuity is located between $\{x(2i+l-1), x(2i+l)\}$;*
(2) *or else we have $|\beta_{j,i+k-1}| \leq a|\beta_{j,i+k}|$, which implies that there are $(k-1)$ consecutive standard stencils containing the jump, and then the discontinuity*

is located between $\{x(2i+l-2),\, x(2i+l-1)\}$.

The extra cost introduced by this comparison jump identification method over the standard wavelet transforms is just the comparison $|\beta_{j,i}| > a|\beta_{j,i-1}|$ for each stencil. In section 5, we use this detection method for all noise-free numerical examples.

Noisy data. The above-described detection method may not be reliable if the function is polluted by noise, especially when the noise is "large." This is because the high frequency coefficients β's may not be able to measure the correct order of smoothness of the functions. Indeed, the high frequency coefficients have the order $\|f^{(p)}(x)+\sigma n^{(p)}(x)\|O(\Delta x^p)$, where $n(x)$ is the random noise and σ a positive number indicating the noise level. In general, the derivatives of the noise $n^{(p)}(x)$ have large values. The noise term $\sigma n^{(p)}(x)$ can dominate the function term $f^{(p)}(x)$ if the noise level σ is large and thus the high frequency coefficients β's may not be able to detect certain discontinuities, e.g., if the jump is small or the discontinuity is in the higher derivatives. In this situation, we need to use heuristics to locate the exact position of the essential discontinuities. Here, we give a simple method to detect the significant large jumps in function values in noisy data.

In many applications, such as in image processing, large discontinuities in function value are the most significant features. Using the standard wavelet transforms, these large discontinuities will generate high frequency coefficients which can be much larger than those generated by the noise. (This is also the fundamental principle in the design of wavelet thresholding.) A simple way to detect these kinds of discontinuities is to look for these large magnitude high frequency coefficients and then compare the data values in the corresponding stencils to locate the exact jump positions. For example, we can look for the places which have the largest difference between two adjacent data values within the stencils. In our numerical experiments, we found that this simple way works very well in practice. In section 5, we will show an example using this method.

Remark. Other jump detection methods can be used for noisy data. As long as the exact subintervals of the discontinuities on the next finer grid are correctly determined, the coarse level ENO-wavelet approximations can be formed at the discontinuities, and our experience shows that it is not sensitive to the presence of noise.

In the ENO-wavelet transforms, to retain the perfect invertibility property, we need to store the adaptive information near every discontinuity, i.e., the exact location of the jump. The reason can also be illustrated by using Figure 4. If there is a jump in the low frequency coefficients (after down sampling) on the coarser level, one can predict a jump in the finer level coefficients. One can further identify the jump existing, for example, between $\{x(2i+l-2),\, x(2i+l)\}$ due to the down sampling. However, as shown in the diagram in Figure 4, for each identified jump, there are two possible locations, i.e., between $\{x(2i+l-2),\, x(2i+l-1)\}$ or between $\{x(2i+l-1),\, x(2i+l)\}$, in the finer level coefficients. Therefore, in order to achieve the perfect reconstruction, the exact locations of discontinuities have to be recorded. In our implementation, we just use one extra bit for each stencil near the discontinuities to indicate it contains a discontinuity. In the application of compression, which aims to reduce the total storage of representing an image, these extra bits need to be taken into account carefully. However, this is beyond the scope of this paper, and we will not discuss it here.

3.3. Forward and inverse transform algorithms. In this subsection, we explicitly present the complete one level forward and inverse ENO-wavelet transform algorithms for the noise-free piecewise smooth data.

We consider the forward transform algorithms first. We denote by $\{c_0, \ldots, c_l\}$ and $\{h_0, \ldots, h_l\}$ the standard wavelet filter coefficients, and by $\{r_0, \ldots, r_l\}$ and $\{d_0, \ldots, d_l\}$ the corresponding inverse filter coefficients. In this paper, since we consider Daubechies orthonormal wavelets, these inverse filter coefficients are defined as $r_s = (-1)^{s+1}h_s$, and $d_s = (-1)^s c_s$, for $s = 0, 1, \ldots, l$. We use a one-bit variable s_i to indicate whether a stencil contains a jump in our algorithms.

Forward Transform Algorithm.

For each i,

(i) compute $\beta_{j,i}$ by (13).

(ii) If $|\beta_{j,i}| \geq a|\beta_{j,i-1}|$ and $|\beta_{j,i}| \geq \epsilon$, then

- compute $\beta_{j,i+k-1}$ and $\beta_{j,i+k}$ by (13).
- Find the exact subinterval of the jump by Proposition 1. For $i \leq m \leq i+k$ or $i \leq m \leq i+k-1$,
  - for the left side of the jump, compute $\hat{\alpha}_{j,m}$ by extrapolation, compute $\hat{\beta}_{j,m}$ by (20), and then set

$$\beta_{j,m} = \hat{\beta}_{j,m}, s_i = 1;$$

  - for the right side of the jump, set $\bar{\beta}_{j,m} = 0$ and compute $\bar{\alpha}_{j,m}$ by (21), and set

$$\alpha_{j,m} = \bar{\alpha}_{j,m}.$$

(iii) Otherwise, compute $\alpha_{j,i}$ by (12). Set $s_i = 0$.

In the algorithm, ϵ is a predefined small positive number which is used to prevent the numerical instability caused by small $\beta_{j,i}$. More precisely, if both $\beta_{j,i}$ and $\beta_{j,i-1}$ are less than the given tolerance ϵ, we treat the current standard stencil as a smooth stencil.

In step (ii), it is possible to use any extrapolation techniques to handle the discontinuities.

Here, we just described the algorithm for one level ENO-wavelet transform with input data sequence $\alpha_{j+1,i}$, and output data $\alpha_{j,i}$ and $\beta_{j,i}$. The coefficient sequences $\alpha_{j,i}$ and $\beta_{j,i}$ have the same size, and their combined size is the same as the input data size at level $j+1$. The multiresolution transform algorithms can be constructed straightforwardly by recursively applying the one level transform to the low frequency coefficients $\alpha_{j,i}$. This is accomplished in the same way as that of the standard multiresolution algorithms. We do not explicitly include them in this paper. Similarly, we present the one level inverse transforms next.

Inverse Transform Algorithm.

For each i,

(i) if $s_i = 0$ and $s_j = 0$, $j = i-k, \ldots, i-1$, then the standard inverse wavelet transforms are applied:

$$\alpha_{j+1,2i} = \sum_{s=0}^{l}(r_{2s+1}\alpha_{j,i-s} + d_{2s+1}\beta_{j,i-s}) \tag{24}$$

and

$$\alpha_{j+1,2i+1} = \sum_{s=0}^{l}(r_{2s}\alpha_{j,i-s} + d_{2s}\beta_{j,i-s}). \tag{25}$$

(ii) If $s_j = 1$, $i-k \leq j \leq i$ or $i-k+1 \leq j \leq i$,

- use Proposition 1 to locate the position of the jump by counting the number of consecutive $s_i = 1$;
- extrapolate $\hat{\alpha}_{j,i}$ from the left side of the jump;
- set $\bar{\beta}_{j,i}$ as zero for the right side of the jump;
- use $\hat{\alpha}_{j,k}$ and $\beta_{j,k}$ to restore the left side by (24) and (25);
- use $\alpha_{j,k}$ and $\bar{\beta}_{j,k}$ to restore the right side of the jump by (24) and (25).

Two simple examples. We give two simple examples in the ENO-Haar and ENO-DB4 cases to illustrate the algorithms. First, we consider computing the transform coefficients of the following initial data:

$$\begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 2 \end{pmatrix}.$$

The standard Haar produces the low and high frequency coefficients

$$\alpha = \begin{pmatrix} \frac{2}{\sqrt{2}} & \frac{3}{\sqrt{2}} & \frac{4}{\sqrt{2}} \end{pmatrix}, \qquad \beta = \begin{pmatrix} 0 & -\frac{1}{\sqrt{2}} & 0 \end{pmatrix}.$$

The corresponding linear approximation is

$$\begin{pmatrix} 1 & 1 & \frac{3}{2} & \frac{3}{2} & 2 & 2 \end{pmatrix},$$

which cannot recover the discontinuity correctly.

Using the ENO-Haar wavelet, we break the initial data sequence into two smooth pieces as shown in the following two rows:

$$\begin{pmatrix} & & y & 2 & 2 & 2 \\ 1 & 1 & 1 & x & & \end{pmatrix},$$

where x and y are some smooth extensions of the corresponding pieces. In fact, we extend x in a way such that the low frequency coefficient $\hat{\alpha}_2$ (boxed in (26)) based on the stencil $(1, x)$ is the same as the previous α_1, which is based on the stencil $(1, 1)$ giving $x = 1$. Similarly, we extend y in a way such that the high frequency coefficient $\bar{\beta}_2$ (boxed in (26)) is zero giving $y = 2$. Therefore we compute the high frequency coefficients $\hat{\beta}_2$ based on stencil $(1, x)$ and the low frequency coefficients $\bar{\alpha}_2$ based on stencil $(y, 2)$ by using the corresponding standard filters giving $\hat{\beta}_2 = 0$ and $\bar{\alpha}_2 = \frac{4}{\sqrt{2}}$. Thus we have the coefficients

$$\alpha = \begin{pmatrix} & \frac{4}{\sqrt{2}} & \frac{4}{\sqrt{2}} \\ \frac{2}{\sqrt{2}} & \boxed{\frac{2}{\sqrt{2}}} & \end{pmatrix}, \qquad \beta = \begin{pmatrix} & \boxed{0} & 0 \\ 0 & 0 & \end{pmatrix}. \tag{26}$$

Since we know how we extended $\hat{\alpha}_2$ and $\bar{\beta}_2$, we do not need to store them. In fact, we just need to store the low and high frequency coefficients as

$$\alpha = \begin{pmatrix} \frac{2}{\sqrt{2}} & \frac{4}{\sqrt{2}} & \frac{4}{\sqrt{2}} \end{pmatrix}, \qquad \beta = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix},$$

which have the same storage schemes as the standard Haar wavelet transform.

When we reconstruct the linear approximation, we can first recover $\hat{\alpha}_2$ and $\bar{\beta}_2$ the same way as in the forward transform and then apply the standard inverse filters to the smooth data to build the approximation. In fact, in this case the linear approximation is exactly the initial data.

In the next example, we show a similar example in which the ENO-DB4 linear approximation is not exactly the same as the initial data, but it still preserves the jump well. The initial data is given as

$$a = \begin{pmatrix} 0 & 1 & 2 & 3 & 4.1 & 5 & 20 & 21 & 22 & 23 \end{pmatrix}.$$

To better demonstrate the coarse level extrapolation idea, we ignore the boundary extension at two ends of the array. We leave out the coefficients based on the boundary extension at the two ends and display only the coefficients corresponding to the middle part of the array. The DB4 filters are given by the low pass filter (0.4830 0.8365 0.2241 −0.1294) and the high pass filter (−0.1294 −0.2241 0.8365 −0.4830). The standard DB4 low and high frequency coefficients (α_2 to α_5 and β_2 to β_5) are

$$\alpha = \begin{pmatrix} 0.8966 & 3.7474 & 7.9280 & 29.1808 \end{pmatrix}$$

and

$$\beta = \begin{pmatrix} -0.0000 & 0.0837 & 4.9368 & -0.0000 \end{pmatrix}.$$

Notice that in this case we have a large high frequency coefficient β_3 which corresponds to the discontinuity between $a_6 = 5$ and $a_7 = 20$ in the array. If we discard the high frequency part, the corresponding linear approximation for the central part of the array around the jump (from $a_3 = 2$ to $a_8 = 21$) is

$$\begin{pmatrix} 2.0108 & 3.0187 & 4.6689 & 6.1470 & 15.8703 & 23.3843 \end{pmatrix},$$

and the discontinuity cannot be preserved.

Using the ENO-DB4 wavelet, we break the initial data sequence into two smooth pieces as shown in the following two rows:

$$\begin{pmatrix} & & & & u & v & 20 & 21 & 22 & 23 \\ 0 & 1 & 2 & 3 & 4.1 & 5 & x & y & & \end{pmatrix},$$

where (x, y) and (u, v) are some smooth extensions of the corresponding pieces. In fact, we extend (x, y) in such a way that the low frequency coefficient $\hat{\alpha}_3 = 6.5983$ (boxed in (27)) based on the stencil $(4.1, 5, x, y)$ is the linear extension of the previous α_1 and α_2. Similarly, we extend (u, v) in such a way that the high frequency coefficient $\bar{\beta}_3$ (boxed in (28)) is zero. Therefore we compute the high frequency coefficients $\hat{\beta}_3$ based on stencil $(4.1, 5, x, y)$ by (20) giving $\hat{\beta}_3 = -0.0259$ and the low frequency coefficients $\bar{\alpha}_3$ based on stencil $(u, v, 20, 21)$ by the analogy of (20) at the right side of a jump giving $\bar{\alpha}_3 = 26.3524$. Thus we have the coefficients

$$\alpha = \begin{pmatrix} & & 26.3524 & 29.1808 \\ 0.8966 & 3.7474 & \boxed{6.5983} & \end{pmatrix} \tag{27}$$

and

$$\beta = \begin{pmatrix} & & \boxed{0} & 0 \\ 0 & 0.0837 & -0.0259 & \end{pmatrix}. \tag{28}$$

Since we know how we extended $\hat{\alpha}_3$ and $\bar{\beta}_3$, we do not need to store them. In fact, we just need to store the low and high frequency coefficients as

$$\alpha = \begin{pmatrix} 0.8966 & 3.7474 & 26.3524 & 29.1808 \end{pmatrix}, \qquad \beta = \begin{pmatrix} 0 & 0.0837 & -0.0259 & 0 \end{pmatrix},$$

which have the same storage schemes as the standard DB4 wavelet transform.

The recovered linear approximation for a_3 to a_8 is

$$\begin{pmatrix} 2.0108 & 3.0187 & 4.0267 & 5.0346 & 20.0000 & 21.0000 \end{pmatrix}.$$

In this case, although the linear approximation is not the same as the initial data, it forms a much more accurate approximation than that of the standard DB4 transform. More importantly, this approximation preserves the discontinuity sharply in contrast to the standard DB4 wavelet which smears the discontinuity.

Remarks.

(i) The ENO-wavelet transforms are just simple modifications of the standard wavelet transforms near discontinuities. The computational complexity of the algorithms remains $O(n)$, and they are relatively easy to implement.

(ii) In the transform algorithms and the corresponding inverse algorithms, the extended low frequency coefficients $\hat{\alpha}_{j,m}$ and the high frequency coefficients $\bar{\beta}_{j,m}$ can be computed by other extrapolation schemes such as least square extrapolation. This may be more robust, especially for noisy data.

(iii) The adaptive ENO-wavelet idea can also be used for other kind of wavelets. They do not necessarily have to be orthogonal wavelets. For instance, one can apply it to the biorthonormal wavelets.

(iv) Like other wavelet transforms, 2-D or even higher-dimensional transforms can be formed by tensor products. In the numerical example section, we will give a 2-D example.

(v) The adaptive ENO-wavelet idea can be recursively used even if the projections do not satisfy the DSP. In such a case, of course we will not get the nice error bound (see section 4), but the approximation errors are comparable to that of the standard wavelet transforms. Also, it is easy to modify the algorithms such that the standard wavelet transforms are applied at the place where the DSP is invalid.

4. Approximation error.

4. Approximation error. In this section, we consider the ENO-wavelet approximation error for piecewise continuous functions.

Given a function $f(x)$ in L^2, in standard wavelet theory [27], [14], [30], it can be linearly approximated by its projection $f_j(x)$ in V_j as in (7) and (8). This linear approximation has a standard error estimate which we state in the following theorem; see also [30].

THEOREM 1. *Suppose the wavelet $\psi(x)$ generated by scaling function $\phi(x)$ has p vanishing moments and $f_j(x)$ is the approximation of $f(x)$, which has bounded pth order derivative, in V_j with basis $\phi_{j,k}(x)$; then,*

$$\|f(x) - f_j(x)\| \leq C(\Delta x)^p \|f^{(p)}(x)\|, \tag{29}$$

where $\Delta x = 2^{-j}$ and C is a constant which is independent of j.

This theorem holds for the L^2 norm in general. Moreover, if the scaling function and its wavelet have finite support, then it also holds for the L^∞ norm.

In this theorem, we can see that the approximation error is controlled by two factors. One is the pth power of the spatial step Δx; the other is the norm of the pth derivative of the function. This error bound does not hold if the function does not have the finite pth derivative. This implies that the approximation could be poor for irregular functions even if the spatial step Δx is small. For piecewise continuous functions, especially functions with large jumps, the approximation error cannot be controlled as smooth functions. In fact, in the standard approximation function $f_j(x)$, oscillations are generated near the discontinuous points, and they will not disappear even if the spatial step size is reduced (the Gibbs phenomenon).

In contrast, in our ENO-wavelet transforms, since no approximation coefficients are computed using information from both sides of the discontinuities, we can obtain a similar error estimate without taking derivatives across the jumps. In the next theorem, we state the estimation and prove it in the rest of this section.

THEOREM 2. *Suppose the scaling function $\phi(x)$ and its $\psi(x)$ have finite support in $[0, l]$, $\psi(x)$ has p vanishing moments, $f(x)$ is a piecewise continuous function in $[a, b]$ with bounded pth derivatives in each piece of smooth regions, and $f_j(x)$ is its jth level ENO-wavelet projection obtained by using any one of the three extrapolation methods given in section* 2.4 *with the choice of a satisfying* (23)*. If the projection $f_{j+1}(x)$ satisfies the DSP, then*

$$\|f(x) - f_j(x)\| \leq C(\Delta x)^p \|f^{(p)}(x)\|_{(a,b)\setminus D}, \tag{30}$$

where $\Delta x = 2^{-j}$ and D is the set where $f(x)$ has jumps in the function value or up to the pth derivatives. The norm $\|\cdot\|$ can be either the L^2 or the L^∞ norm.

Proof. We prove the inequality (30) under the L^∞ norm, and the L^2 result can be obtained in a similar way.

According to section 3.2, with the choice of a, all jumps in set D will be detected by the algorithms described for the piecewise smooth data unless the intensity of the jump is less than $O(\Delta x^{(p-m)})$, where the jump is in the mth derivative. In the latter case, the error caused by missing the jump is of the order of $O(\Delta x^p)$, which can be absorbed by the right-hand side of (30).

The DSP allows us to separate the discontinuities and individually consider a small neighborhood around each jump. Therefore, to simplify the discussion without loss of generality, we consider a piecewise function $f(x)$ with one jump at the origin. In other words,

$$f(x) = \begin{cases} f_1(x), & a \leq x < 0, \\ f_2(x), & 0 \leq x \leq b, \end{cases}$$

where $f_1(x) \in C^p[a, 0]$ and $f_2(x) \in C^p[0, b]$. Because both $\phi_{j,i}(x)$ and $\psi_{j,i}(x)$ have support $[i, (l+i)\Delta x]$, the small neighborhood affected by the ENO decision is $[-l\Delta x, l\Delta x]$. In fact, the ENO-wavelet coefficients depend only on one-sided information and therefore, by symmetry, we just need to prove (30) in $[-l\Delta x, 0]$.

Before we prove that (30) holds for the three types of extrapolation methods, namely direct function extrapolation and the two choices of coarse level extension ((1) and (2) in section 3.1), we give some notations which we will frequently use in the proof.

Denote by $g_1(x)$ the $(p-1)$th order polynomial which is the first p term of the Taylor expansion of $f_1(x)$ at the origin, i.e.,

$$f_1(x) = g_1(x) + \frac{f_1^{(p)}(\xi)}{p!} x^p, \tag{31}$$

where ξ is in interval $[-l\Delta x, 0]$. Also denote by $\alpha_{j,m}$ and $\beta_{j,m}$ the ENO-wavelet low and high frequency coefficients, respectively, and $\bar{\alpha}_{j,m}$ the low frequency coefficients of the polynomial $g_1(x)$, i.e.,

$$\bar{\alpha}_{j,m} = \int g_1(x)\phi_{j,m}(x)dx$$

and

$$g_{1,j}(x) = \sum_m \bar{\alpha}_{j,m}\phi_{j,m}(x).$$

As we mentioned in section 3.1, different techniques can be used for extrapolation. Here we select the extrapolation by Taylor expansion as our starting point throughout the proof because of its simplicity. For other types of extrapolation techniques, the proof can be directly generalized by taking into account the difference between that type of extrapolation and the Taylor expansion extrapolation. For instance, the classical approximation result shows us that the difference between the Lagrange extrapolation that we use in the numerical experiments in this paper and Taylor expansion extrapolation is of the order of $O(\Delta x^p)$, which will be absorbed into the right-hand side of the estimate (30).

Now we are ready to prove that (30) holds for the three types of extrapolation methods. We first prove (30) for direct function extrapolation.

Direct function extrapolation. The direct function extrapolation extends $f_1(x)$ to interval $[0, l\Delta x]$ by defining

$$f_d(x) = \begin{cases} f_1(x), & -l\Delta x \le x < 0, \\ g_1(x), & 0 \le x \le l\Delta x. \end{cases}$$

The corresponding ENO-wavelet low frequency coefficients $\alpha_{j,m}$ are computed by

$$\alpha_{j,m} = \int f_d(x)\phi_{j,m}(x)dx, \tag{32}$$

and the approximation function is defined as

$$f_{d,j}(x) = \sum_m \alpha_{j,m}\phi_{j,m}(x), \qquad x \in [-l\Delta x, 0]. \tag{33}$$

For any point $x_0 \in [-l\Delta x, 0]$, by using (31) and the fact that since $g_1(x)$ is a $(p-1)$th order polynomial, $g_{1,j}(x) = g_1(x)$, we have

$$\begin{aligned} |f_1(x_0) - f_{d,j}(x_0)| &\le |f_1(x_0) - g_1(x_0)| + |g_{1,j}(x_0) - f_{d,j}(x_0)| \\ &\le C(\Delta x)^p \|f_1^{(p)}\| + |g_{1,j}(x_0) - f_{d,j}(x_0)|. \end{aligned} \tag{34}$$

Let q be an integer in $[-l, 0]$ such that $x_0 \in [q\Delta x, (q+1)\Delta x)$; then the last term of (34) can be bounded by

$$\begin{aligned} |g_{1,j}(x_0) - f_{d,j}(x_0)| &= \left| \sum_m (\bar{\alpha}_{j,m} - \alpha_{j,m})\phi_{j,m}(x_0) \right| \\ &\le \sum_{q-l \le m \le q} |\bar{\alpha}_{j,m} - \alpha_{j,m}||\phi_{j,m}(x_0)| \\ &= \sum_{q-l \le m \le q} |\bar{\alpha}_{j,m} - \alpha_{j,m}||(\Delta x)^{-\frac{1}{2}}\phi(2^j x_0 - m)|. \end{aligned} \tag{35}$$

To prove (30), we now need to estimate $|\bar{\alpha}_{j,m} - \alpha_{j,m}|$. Since when $m \le -l$ the coefficients are computed in the standard manner, i.e., $\bar{\alpha}_{j,m} = \alpha_{j,m}$, we just need to consider all m with $-l+1 \leq m \leq 0$. In fact, we have

$$\begin{aligned}|\bar{\alpha}_{j,m} - \alpha_{j,m}| &= \left|\int (f_d(x) - g_1(x))\phi_{j,m}(x)dx\right| \\ &\leq \left|\int_{m\Delta x}^{0} (f_d(x) - g_1(x))\phi_{j,m}(x)dx\right| \\ &\quad + \left|\int_{0}^{(m+l)\Delta x} (f_d(x) - g_1(x))\phi_{j,m}(x)dx\right|.\end{aligned}$$

Because $f_d(x)$ is the same as $g_1(x)$ in $[0,(m+l)\Delta x]$, using (31), we have

$$\begin{aligned}|\bar{\alpha}_{j,m} - \alpha_{j,m}| &= \left|\int_{m\Delta x}^{0} (f_1(x) - g_1(x))\phi_{j,m}(x)dx\right| \\ &\leq \left(\int_{m\Delta x}^{0} |f_1(x) - g_1(x)|^2 dx\right)^{\frac{1}{2}} \left(\int_{m\Delta x}^{0} |\phi_{j,m}(x)|^2 dx\right)^{\frac{1}{2}} \\ &\leq C(\Delta x)^p \|f^{(p)}\| (\Delta x)^{\frac{1}{2}} \\ &\leq C(\Delta x)^{p+\frac{1}{2}} \|f^{(p)}\|.\end{aligned}$$

Therefore, combining this with (35), we have

$$|g_{1,j}(x_0) - f_{d,j}(x_0)| \leq C(\Delta x)^p \|f^{(p)}\|.$$

This and (34) complete the proof of (30) for the case of direct function extrapolation.

Coarse level extrapolation. As described in section 3.1, there are two ways of extrapolating coefficients on the coarse level. One way is to set the extended high frequencies to zero. The other way is to extrapolate the low frequency coefficients by a $(p-1)$th order polynomial in wavelet space. In the following part of the proof, we consider them separately.

We consider the high frequency zero extension first.

Similar to the direct function extrapolation, we also extend $f_1(x)$ to the interval $[0, l\Delta x]$ and denote it by

$$f_h(x) = \begin{cases} f_1(x), & x \in [-l\Delta x, 0], \\ g_h(x), & x \in (0, l\Delta x], \end{cases}$$

where $g_h(x)$ is implicitly defined such that it makes $f_h(x)$ satisfy

$$\int f_h(x)\psi_{j,m}(x)dx = 0, \qquad -l+1 \leq m \leq 0, \tag{36}$$

and

$$\int f_h(x)\phi_{j,m}(x)dx = \alpha_{j,m}, \qquad -l+1 \leq m \leq 0. \tag{37}$$

The difference between $f_d(x)$ and $f_h(x)$ is that in the direct function extrapolation $f_d(x)$ we know that $g_1(x)$ is the $(p-1)$th order polynomial, but in this case $g_h(x)$ is unknown.

Formally following the proof of (30) for the direct function extrapolation, (34) and (35) also hold for this case. Therefore, we need only to estimate $|\bar{\alpha}_{j,m} - \alpha_{j,m}|, -l+1 \leq$

$m \leq 0$. We consider $m = -l+1$ first. Unlike in the direct function extrapolation, where $|\bar{\alpha}_{j,-l+1} - \alpha_{j,-l+1}|$ can be computed directly by the Taylor expansion, here we cannot bound $|\bar{\alpha}_{j,-l+1} - \alpha_{j,-l+1}|$ in the same way. Instead, we use the following trick to obtain the estimate we need.

From the dilation equation (1) and the wavelet equation (2), we have the following relationships:

$$\phi_{j,m}(x) = \sum_{s=0}^{l} c_s \phi_{j+1,s+2m}(x) \tag{38}$$

and

$$\psi_{j,m}(x) = \sum_{s=0}^{l} h_s \phi_{j+1,s+2m}(x). \tag{39}$$

Using (38), $\bar{\alpha}_{j,-l+1}$ and $\alpha_{j,-l+1}$ can be computed by

$$\alpha_{j,-l+1} = \int f_h(x)\phi_{j,-l+1}(x)dx = \sum_{s=0}^{l} c_s \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} f_h(x)\phi_{j+1,s-2l+2}(x)dx$$

and

$$\bar{\alpha}_{j,-l+1} = \int g_1(x)\phi_{j,-l+1}(x)dx = \sum_{s=0}^{l} c_s \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} g_1(x)\phi_{j+1,s-2l+2}(x)dx.$$

Therefore, we have

$$\begin{aligned} |\bar{\alpha}_{j,-l+1} - \alpha_{j,-l+1}| \leq & \left| \sum_{s=0}^{l-2} c_s \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} (g_1(x) - f_h(x))\phi_{j+1,s-2l+2}(x)dx \right| \\ & + \left| c_{l-1} \int_{\frac{\Delta x}{2}(-l+1)}^{\frac{\Delta x}{2}} (g_1(x) - f_h(x))\phi_{j+1,1-l}(x)dx \right. \\ & \left. + c_l \int_{\frac{\Delta x}{2}(-l+2)}^{\Delta x} (g_1(x) - f_h(x))\phi_{j+1,2-l}(x)dx \right|. \end{aligned} \tag{40}$$

We know that only the last two terms involve the value of $f_h(x)$ in $[0, \Delta x]$. The other terms use $f_h(x)$ in $[-l\Delta x, 0]$, which is $f_1(x)$. Then, by Taylor expansion and Schwartz inequality,

$$\left| \sum_{s=0}^{l-2} c_s \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} (g_1(x) - f_h(x))\phi_{j+1,s-2l+2}(x)dx \right| \leq C(\Delta x)^p \|f_1^{(p)}\| 2^{\frac{-(j+1)}{2}}. \tag{41}$$

Thus, to bound $|\bar{\alpha}_{j,-l+1} - \alpha_{j,-l+1}|$, the only remaining task is to estimate the last two terms in (40).

Considering that $g_1(x)$ is a $(p-1)$th order polynomial, we obtain

$$\int f_h(x)\psi_{j,-l+1}(x)dx = 0 = \int g_1(x)\psi_{j,-l+1}(x)dx.$$

Substituting the wavelet equation (39) into the above equation, we have

$$\sum_{s=0}^{l} h_s \int (f_h(x) - g_1(x))\phi_{j+1,s-2l+2}(x)dx = 0.$$

We can rewrite this equation in the following form:

$$\begin{aligned} &h_{l-1} \int_{\frac{\Delta x}{2}(-l+1)}^{\frac{\Delta x}{2}} (f_h(x) - g_1(x))\phi_{j+1,1-l}(x)dx \\ &+ h_l \int_{\frac{\Delta x}{2}(-l+2)}^{\Delta x} (f_h(x) - g_1(x))\phi_{j+1,2-l}(x)dx \\ &\qquad = -\sum_{s=0}^{l-2} h_s \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} (f_h(x) - g_1(x))\phi_{j+1,s-2l+2}(x)dx. \end{aligned}$$

Notice that we have $\frac{h_{l-1}}{c_{l-1}} = \frac{h_l}{c_l}$. We find that the left-hand side contains the term we need to estimate, whereas the right-hand side uses only $f_h(x)$ at the left side of the origin and thus can be controlled again by Taylor expansion. This means that we have

$$\begin{aligned} &\Bigg| c_{l-1} \int_{\frac{\Delta x}{2}(-l+1)}^{\frac{\Delta x}{2}} (f_h(x) - g_1(x))\phi_{j+1,1-l}(x)dx \\ &+ c_l \int_{\frac{\Delta x}{2}(-l+2)}^{\Delta x} (f_h(x) - g_1(x))\phi_{j+1,2-l}(x)dx \Bigg| \\ &\qquad \leq \left|\frac{c_l}{h_l}\right| \sum_{s=0}^{l-2} |h_s| \int_{\frac{\Delta x}{2}(s-2l+2)}^{\frac{\Delta x}{2}(s-l+2)} |f_1(x) - g_1(x)||\phi_{j+1,s-2l+2}(x)|dx \\ &\qquad \leq C(\Delta x)^p \|f_1^{(p)}\| 2^{\frac{-(j+1)}{2}}. \end{aligned} \tag{42}$$

Combining (40), (41), and (42), we have

$$|\bar{\alpha}_{j,-l+1} - \alpha_{j,-l+1}| \leq C(\Delta x)^p \|f_1^{(p)}\| 2^{-\frac{(j+1)}{2}}.$$

Similarly, we can prove that, for all $m, -l+1 < m \leq 0$,

$$|\bar{\alpha}_{j,m} - \alpha_{j,m}| \leq C(\Delta x)^p \|f_1^{(p)}\| 2^{-\frac{(j+1)}{2}}.$$

Substituting them into (35), we prove that (30) holds for the high frequency extension case.

The last case we need to consider is the coarse level extrapolation of low frequency coefficients. To prove (30), we use the result obtained for the direct function extrapolation.

We denote by $\alpha^d_{j,m}$ the low frequency coefficients for $f_d(x)$. The jth level low frequency extrapolation approximation $f_{l,j}(x)$ is defined as

$$f_{l,j}(x) = \sum_m \alpha_{j,m}\phi_{j,m}(x).$$

For any point $x_0 \in [q\Delta x, (q+1)\Delta x) \subset [-l\Delta x, 0]$, we have

$$|f_1(x_0) - f_{l,j}(x_0)| \leq |f_1(x_0) - f_{d,j}(x_0)| + |f_{d,j}(x_0) - f_{l,j}(x_0)|. \tag{43}$$

Using (30) for the direct function extrapolation case, we know that

$$|f_1(x_0) - f_{d,j}(x_0)| \le C(\Delta x)^p \|f_1^{(p)}\|. \tag{44}$$

And the remaining term can be bounded by

$$|f_{d,j}(x_0) - f_{l,j}(x_0)| \le \sum_{q-l\le m\le q} |\alpha^d_{j,m} - \alpha_{j,m}||2^{\frac{j}{2}}\phi(2^j x_0 - m)|. \tag{45}$$

Again, we need to estimate $|\alpha^d_{j,m} - \alpha_{j,m}|$.

Unlike the previous two cases where the low frequency coefficients $\alpha_{j,m}$ are computed by integration (32) or (37), in this case $\alpha_{j,m}$ are determined by the low frequency extrapolation on the coarse level in wavelet space. So, to estimate $|\alpha^d_{j,m} - \alpha_{j,m}|$, we need to consider them in wavelet space. We introduce the following operator notations first.

Define the continuous wavelet transform (WT) of any function $f(x)$ in space V_j by

$$WT(f)(s) = \int f(x)\phi_{j,s}(x)dx = 2^{\frac{j}{2}} \int f(x)\phi(2^j x - s)dx.$$

Also define the following Taylor extrapolation operator (EX) of $f(x)$:

$$EX(f)(x) = \begin{cases} f(x), & x \le 0, \\ g(x), & x > 0, \end{cases}$$

where $g(x)$ is the $(p-1)$th order Taylor polynomial of $f(x)$. Using these notations, we can represent the low frequency wavelet coefficients

$$\alpha_{j,m} = EX_w(WT(f_1))(m), \quad \text{for} \quad -l+1 \le m \le 0,$$

and

$$\alpha^d_{j,m} = WT(EX_f(f_1))(m), \quad \text{for} \quad -l+1 \le m \le 0,$$

where EX_w and EX_f represent the extrapolation operator EX in the wavelet and physical space, respectively.

Instead of estimating $|\alpha^d_{j,m} - \alpha_{j,m}|$ directly, we prove the following more general result.

LEMMA 1. *Given a smooth function $g(x)$, let $g_{we}(s) = WT(EX_f(g))(s)$ and $g_{ew}(s) = EX_w(WT(g))(s)$; then*

$$|g_{we}(s) - g_{ew}(s)| \le C(\Delta x)^p \|g^{(p)}\| 2^{-\frac{j}{2}}.$$

Using this lemma, we obtain the desired bounds for $|\alpha^d_{j,m} - \alpha_{j,m}|$ easily by taking $s = m$. Combining them with (44) and (45), we prove that (30) holds for the low frequency coefficient extrapolation case.

Proof. Denote $\bar{g}(x) = EX_f(g)(x)$, and then

$$\begin{aligned} g_{we}(s) &= 2^{\frac{j}{2}} \int \bar{g}(x)\phi(2^j x - s)dx \\ &= 2^{-\frac{j}{2}} \int \bar{g}(2^{-j}(y - s))\phi(y)dy. \end{aligned}$$

By changing variable $z = 2^{-j}s$, and denoting

$$e_j(z) = \int \bar{g}(2^{-j}y - z)\phi(y)dy,$$

we have

$$g_{we}(s) = 2^{-\frac{j}{2}} e_j(2^{-j}s).$$

We also know that $e_j(z)$ is a smooth function and, by differentiating p times, we have

$$(46)\ \|e_j^{(p)}\| = \left\| \int (-1)^p \bar{g}^{(p)}(2^{-j}y - z)\phi(y)dy \right\| \le C\|g^{(p)}\| \left\| \int \phi(y)dy \right\| \le C\|g^{(p)}\|.$$

Taking the $(p-1)$th order Taylor expansion of $e_j(z)$ at $z = -l\Delta x$, we have

$$e_j(z) = \hat{e}_j(z) + e_j^{(p)}(\xi)\frac{(z + l\Delta x)^p}{p!},$$

where $\hat{e}_j(z)$ is the $(p-1)$th order Taylor polynomial and $\xi \in [2l, 0]$. Since $g_{ew}(s)$ is the same as $g_{we}(s)$ if $s \le -l$, it is defined as the Taylor polynomial for $s > -l$ according to the definition of EX; i.e., we have

$$g_{ew}(s) = \begin{cases} 2^{-\frac{j}{2}} e_j(2^{-j}s), & s \le -l, \\ 2^{-\frac{j}{2}} \hat{e}_j(2^{-j}s), & s > -l. \end{cases}$$

Therefore,

$$\begin{aligned} |g_{we}(s) - g_{ew}(s)| &\le 2^{-\frac{j}{2}} |e_j(2^{-j}s) - \hat{e}_j(2^{-j}s)| \\ &\le C(\Delta x)^p \|g^{(p)}\| 2^{-\frac{j}{2}}. \end{aligned}$$

This completes the proof of Lemma 1 and also completes the proof of Theorem 2.

5. Numerical examples.

In this section, we give some one-dimensional (1-D) and 2-D numerical examples by using the ENO-wavelet transforms. In particular, we show results of the ENO-Haar, ENO-DB4, and ENO-DB6 wavelet transforms.

In all examples, for simplicity, we just consider functions with zero values at the boundary. For nonzero boundary functions, we can easily extend the function by zero and treat the boundaries as discontinuities.

To illustrate the performance of ENO-wavelet transforms, we show picture comparisons of the standard wavelet approximations and corresponding ENO-wavelet approximations. In addition, we compare the L_∞ and L_2 errors of the standard wavelet approximations and the ENO-wavelet approximations at different levels by measuring $E_{\infty,j} = \inf_x \|f(x) - f_j(x)\|$, which is computed by finding the largest difference on the finest grid, and $E_{2,j} = \|f(x) - f_j(x)\|_2$. Using them, we compute the orders of accuracy defined by

$$Order_\infty = \log_2 \frac{E_{\infty,i}}{E_{\infty,i-1}}$$

and

$$Order_2 = \log_2 \frac{E_{2,i}}{E_{2,i-1}},$$

Table 1

Comparison of the maximum error of the standard Haar and the ENO-Haar wavelet approximation for the smooth function $\sin(x)$*. We see that they have the same approximation error for the smooth functions.*

Level	Haar E_∞	ENO-Haar E_∞	$Order_\infty$
4	0.0919	0.0919	
3	0.0430	0.0430	1.070
2	0.0184	0.0184	1.202
1	0.0061	0.0061	1.585

Table 2

*Comparison of the maximum error of the standard DB*4 *and the ENO-DB*4 *approximations for the smooth function* $f(x) = \exp[-(\frac{1}{x} + \frac{1}{1-x})], 0 < x < 1$*. They have the same error and both achieve second order accuracy which agrees with the results in Theorem* 1 *for the smooth functions.*

Level	DB4 E_∞	ENO-DB4 E_∞	$Order_\infty$
4	3.316e-5	3.316e-5	
3	7.650e-6	7.650e-6	2.104
2	1.590e-6	1.590e-6	2.232
1	2.972e-7	2.973e-7	2.406

which indicates the order of accuracy of the approximation in the L_∞ norm and L_2 norm, respectively.

For all noise-free examples, we use the method described in section 3.2 to locate the exact positions of the discontinuities. And we select $a = 2$ and $\epsilon = 0.0001$ (as used in the algorithms in section 3.3) for all 1-D examples.

First, we compare the approximations for smooth functions. Table 1 is the comparison of Haar and ENO-Haar approximations for the smooth function $f(x) = \sin(x), 0 \leq x \leq 2\pi$ at different levels, and Table 2 is the comparison of DB4 and ENO-DB4 approximations for the function $f(x) = \exp[-(\frac{1}{x} + \frac{1}{1-x})], 0 < x < 1$.

We see from these tables that for smooth functions the ENO-wavelet transforms have exactly the same approximation error as the standard wavelet transforms. Both of them maintain the approximation order 1 and 2 for Haar and DB4, respectively, which agree with the results in Theorem 1. In fact, we notice that in this situation no singularity is detected, and the ENO-wavelet algorithms perform the standard transforms for completely smooth functions as we expected.

Next, we consider a piecewise smooth function defined by

$$f(x) = \begin{cases} 0, & 0 \leq x < 0.2, \\ -50x - 5, & 0.2 \leq x < 0.4, \\ 10\sin(4\pi x + 0.8\pi) - 1, & 0.4 \leq x < 1.1, \\ 5e^{2x} - 100, & 1.1 \leq x < 1.6, \\ 0, & 1.6 \leq x \leq 2. \end{cases}$$

We apply Haar and ENO-Haar, DB4 and ENO-DB4, and DB6 and ENO-DB6 transforms to this function and compare the approximation error. Figure 5 shows the comparison of the order of accuracy in the L_∞ and L_2 norm. It is clear that both L_∞ and L_2 order of accuracy for ENO-wavelet transforms are of the order 1, 2, and 3 for ENO-Haar, ENO-DB4, and ENO-DB6, respectively, and they agree with the results in Theorem 2. In contrast, standard wavelet transforms do not retain the corresponding order of accuracy for piecewise smooth functions.

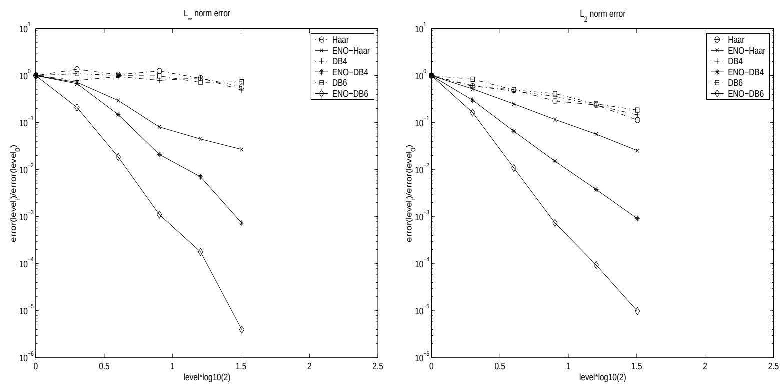

FIG. 5. *The approximation accuracy comparison of ENO-wavelet and wavelet transforms. Both L_∞ (left) and L_2 (right) order of accuracy show that ENO-wavelet transforms maintain the order* 1, 2, *and* 3 *for ENO-Haar, ENO-DB*4, *and ENO-DB*6, *respectively, and they agree with the results of Theorem* 2. *In contrast, standard wavelet transforms do not retain the order of accuracy for piecewise smooth functions.*

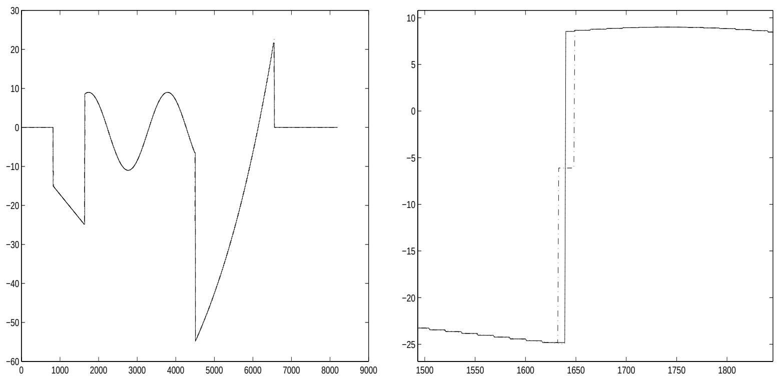

FIG. 6. *The* 4*-level ENO-Haar and Haar approximation. The left picture shows the original function (dotted line), the standard Haar approximation (dash-dotted line) and the ENO-Haar approximation (solid line). The right picture is a zoom-in near a discontinuity. We see the Gibbs phenomenon (staircase) in the standard Haar approximation but not in the ENO-Haar approximation.*

To see the Gibbs oscillations, we display the 4-level ENO-wavelet and standard wavelet approximations in Figures 6, 7, and 8 for ENO-Haar, ENO-DB4, and ENO-DB6 approximations, respectively. In the left column, we show the original function (dotted line), the standard wavelet linear approximations (dash-dotted), and the ENO-wavelet approximations (solid line). The right pictures are zoom-ins of the left pictures near a discontinuity. We clearly see the Gibbs oscillations in the standard approximations; in contrast, the ENO-wavelet approximations preserve the jump accurately.

In Figures 9, 10, and 11, we also present the standard Haar, DB4, and DB6 wavelet coefficients (dotted line) and the ENO-Haar, ENO-DB4, and ENO-DB6 wavelet coefficients (solid line), respectively. The left part corresponds to the low frequency coefficients and the right part to the high frequency coefficients. We notice that there are some large standard high frequency coefficients near the discontinuities. On the other hand, no large high frequency coefficients are present in the ENO-wavelet

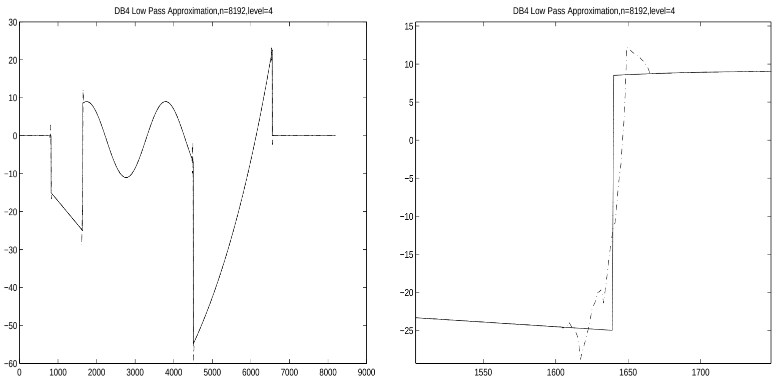

FIG. 7. *The* 4*-level ENO-DB*4 *and the standard DB*4 *approximations. The original discontinuous function (dotted line), the standard DB*4 *approximation (dash-dotted line), and the ENO-DB*4 *approximation (solid line) are displayed. The Gibbs phenomenon is clearly seen for the standard DB*4 *approximation but not for the ENO-DB*4 *approximation.*

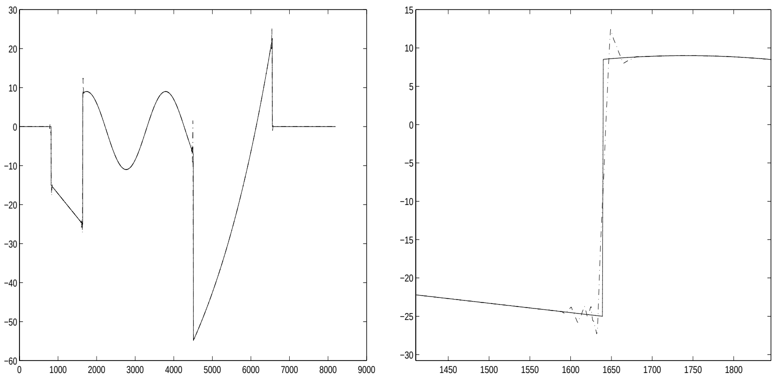

FIG. 8. *The* 4*-level ENO-DB*6 *(solid line) and the standard DB*6 *(dash-dotted line) approximation. The standard DB*6 *generates oscillations near discontinuities, but the ENO-DB*6 *does not.*

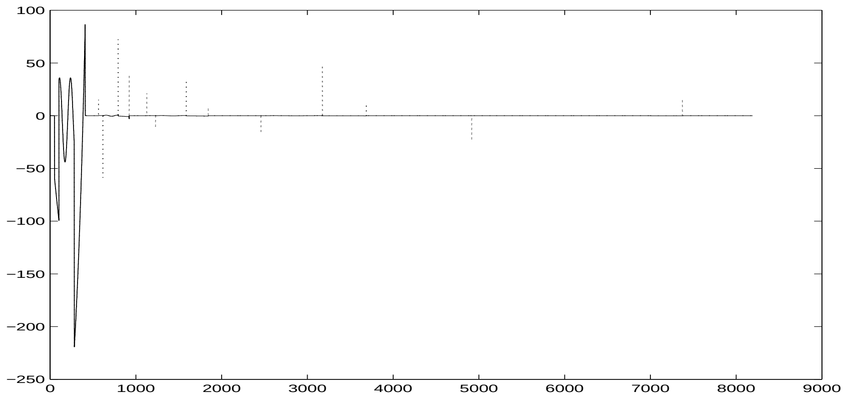

FIG. 9. *The* 4*-level ENO-Haar (solid line) and the standard Haar coefficients (dotted line). The left part corresponds to the low frequencies, the right part to the high frequencies. In the standard Haar coefficients, large high frequency coefficients present near discontinuities, while in the ENO-Haar case there are no large high frequency coefficients.*

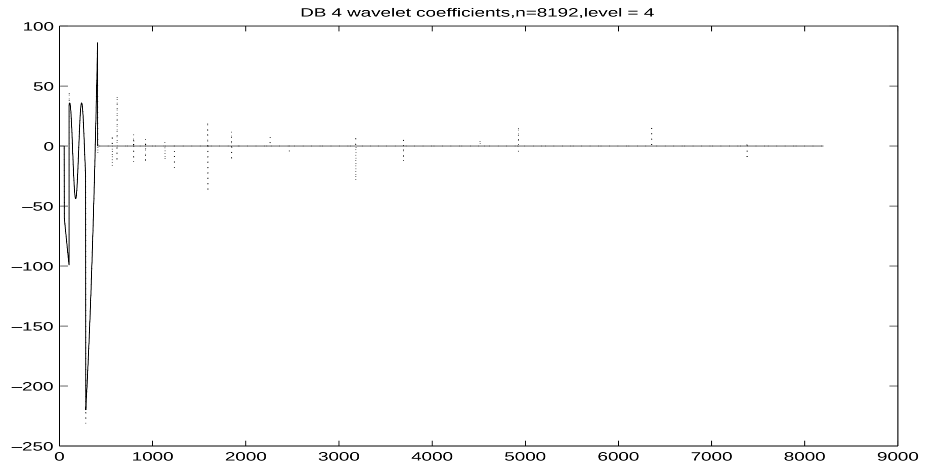

FIG. 10. *The* 4*-level ENO-DB*4 *coefficients (solid line) and the standard DB*4 *coefficients (dotted line). There are large high frequency coefficients (right part) near the discontinuities in the standard DB*4 *transform but not in the ENO-DB*4 *transform.*

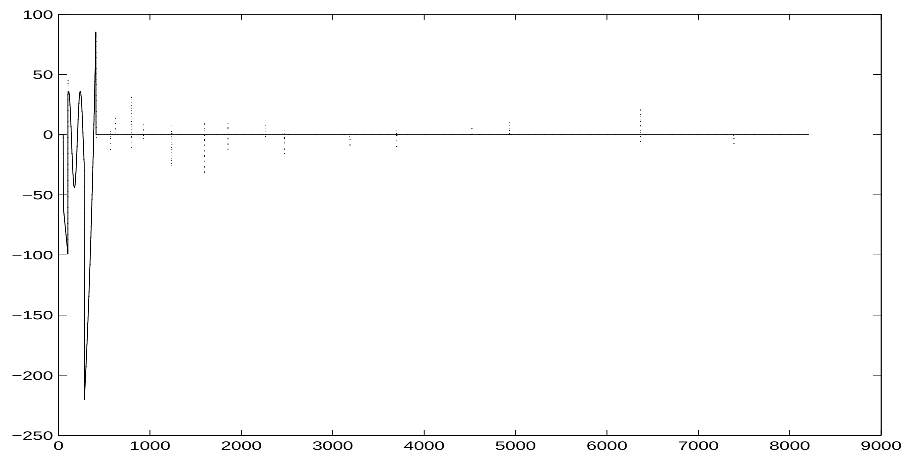

FIG. 11. *The* 4*-level ENO-DB*6 *coefficients (solid line) and the standard DB*6 *coefficients (dotted line). There are large high frequency coefficients near the discontinuities in the standard DB*6 *transform but not in the ENO-DB*6 *transform.*

coefficients. This illustrates that the ENO-wavelet coefficients have better distribution than standard wavelet coefficients; i.e., they have no large coefficients in the high frequencies, and the energy is concentrated in the low frequency end.

The next 1-D example we present here (Figure 12) is a comparison of the standard DB6 and the ENO-DB6 transforms to illustrate the performance at places where the DSP is not valid and also at jumps in the derivative. The original data (circles) has two discontinuities (the middle bump) which violate the DSP assumption, which requires that there are at least eight data points between any pair of discontinuities. Although the ENO-DB6 approximation (solid line) does not preserve this pair of discontinuities exactly, its approximation error is still comparable (actually better in this case) to that of the standard DB6 approximation (dotted line). At the left bump where the DSP holds, the ENO-DB6 does preserve the discontinuities exactly as we

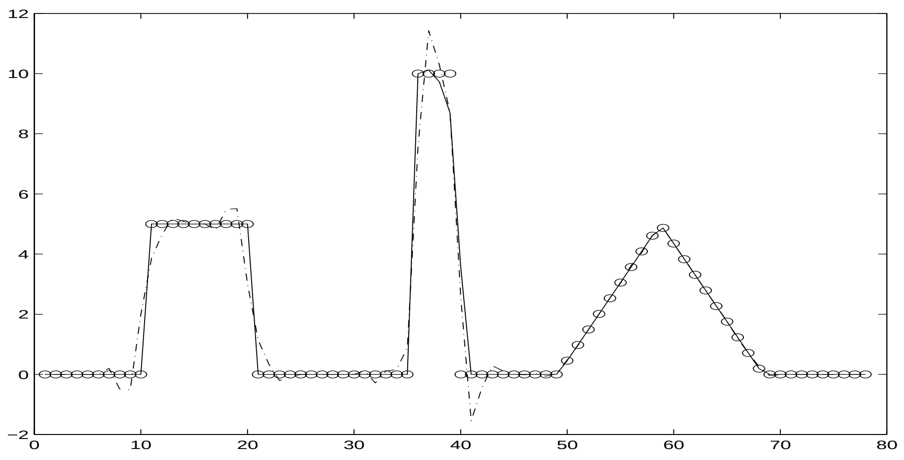

FIG. 12. *The level-*1 *approximation comparison of the ENO-DB*6 *and the standard DB*6 *wavelets at places where the DSP is invalid (the middle bump). The initial data (circles) has two close discontinuities. The ENO-DB*6 *approximation (solid line) error is comparable to that of the standard DB*6 *approximation (dotted line). The left bump satisfies the DSP and therefore the ENO-DB*6 *exactly recovers it. The right kink is a discontinuity in the first derivative, and the standard DB*6 *still generates oscillations although their magnitudes are not significant. The ENO-DB*6 *restores it perfectly, We display a zoom-in picture of this kink in Figure* 13.

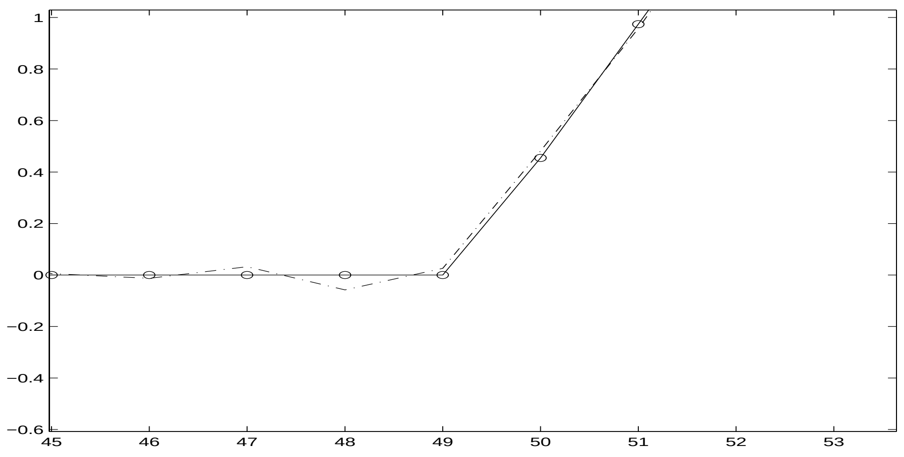

FIG. 13. *The zoom-in of Figure* 12 *at the kink where there is a discontinuity in its derivative. The ENO-DB*6 *(solid line) can recover it perfectly, but the standard DB*6 *(dash-dotted line) generates oscillations.*

expected. In the same example, we also display the comparison of the ENO-DB6 and the standard DB6 approximations at the right kink, which is not a discontinuity in function values but in its first order derivative. The standard DB6 approximation has oscillations, although their magnitudes are small, but the ENO-DB6 restores it exactly (see Figure 13).

The last 1-D example is applying the ENO-DB6 wavelet transform to a piecewise constant function polluted by Gaussian random noise (see Figure 14). For this example, the jump detection method corresponding to Lemma 1 does not work. Instead, we use the simple method given in section 3.2, which detects jumps by looking for

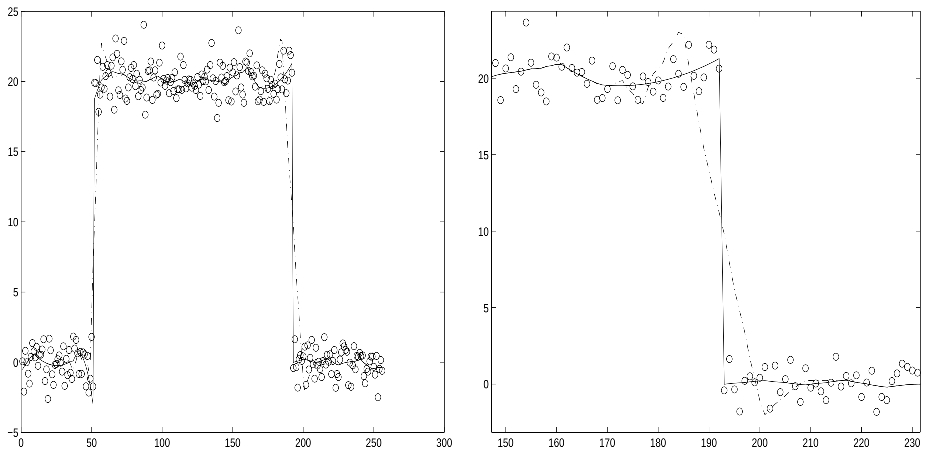

FIG. 14. *Left: The comparison of the* 3*-level ENO-DB*6 *approximation (solid line) with the standard DB*6 *approximation (dash-dotted line) for noisy initial data (circles). The ENO-DB*6 *approximation retains the sharp jumps, but the standard DB*6 *approximation does not (right picture). Right: A zoom-in of the left example at the discontinuities.*

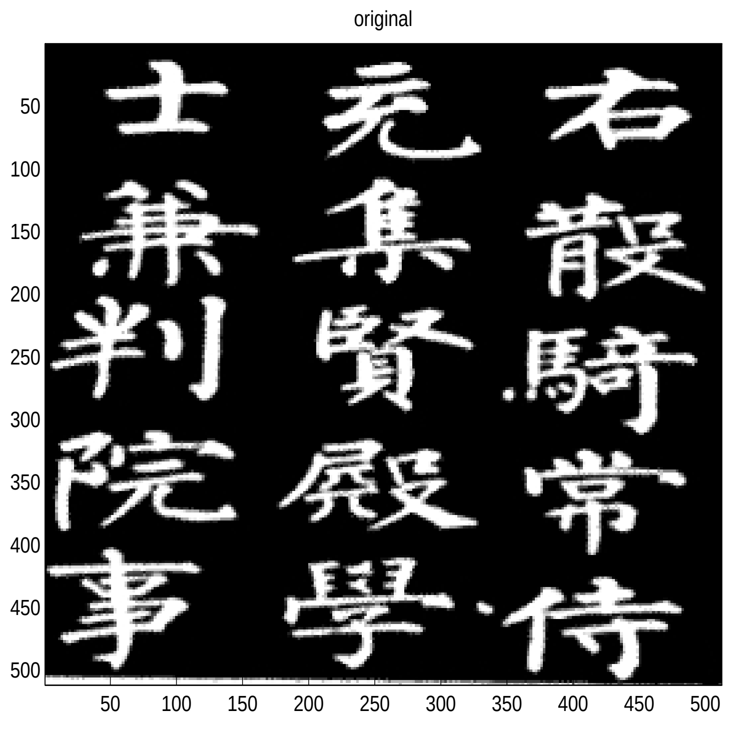

FIG. 15. *Original* 2*-D function.*

stencils with significant larger high frequency coefficients than their neighbors and then locates the exact jump locations by directly comparing the differences between two adjacent function values within the stencil. Despite the presence of noise in the initial data (circles), the level-3 ENO-DB6 approximation (solid line) still retains the sharp edges (see zoom-in in the right picture in Figure 14) compared to the stan-

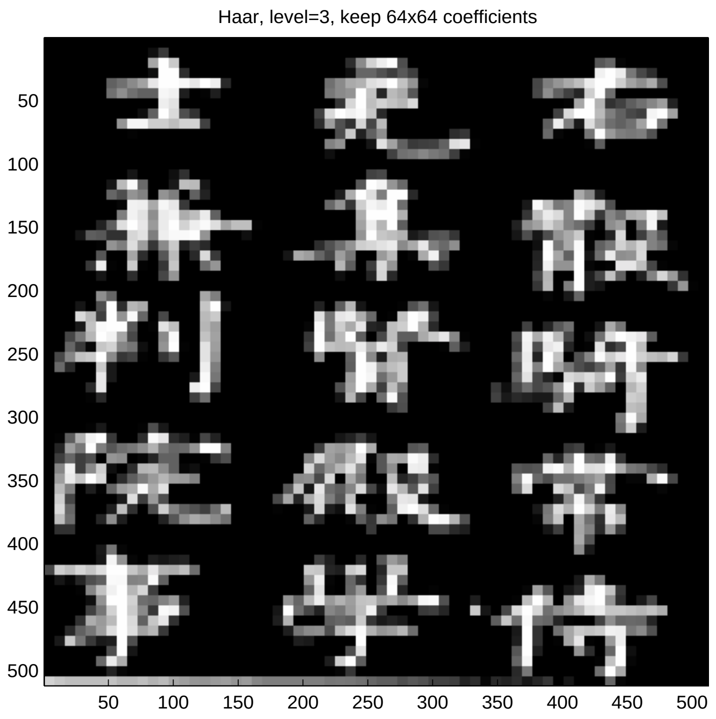

FIG. 16. *The 3-level standard Haar approximation: The reconstructions are obtained from low frequency coefficients α_{J-3} only, where α_J is the original image. The edges are fuzzier than those in the next picture.*

dard DB6 approximation (dash-dotted line) which not only has oscillations at the discontinuities but also smears them.

Finally, we give a 2-D testing image example to compare the standard Haar and the ENO-Haar approximations. Here we use tensor products of 1-D transforms. The original picture is shown in Figure 15. Figure 16 is the 3-level standard Haar approximation and Figure 17 is the 3-level ENO-Haar approximation. Both use low frequency approximations (the reconstructions are obtained from low frequency coefficients α_{J-3} only, where α_J is the original image) and store the same number of coefficients ($\frac{1}{64}$ of the original data). It is clear that in the standard Haar case the function becomes fuzzier than in the ENO-Haar case. This illustrates that the ENO-Haar approximation can reduce the edge oscillations for 2-D functions. In addition, as we mentioned in the introduction, we designed ENO-wavelet transforms not to replace the standard nonlinear adaptive wavelet techniques; rather we think it would be beneficial to use them in conjunction with the standard adaptive nonlinear techniques. For instance, we can combine ENO-wavelets with hard thresholding techniques as one can do it for the standard wavelet transforms. We show the standard hard thresholding approximation image by retaining the largest 64×64 coefficients in Figure 18, and we note that sharper edges are recovered comparing to the linear approximations. Similarly, we can apply the same thresholding techniques to the ENO-wavelet transforms. In Figure 19, we give the approximate image by using the ENO-Haar hard thresholding technique by keeping the largest 3506 ENO-Haar coefficients, which is 70% of number of coefficients retained in the previous image. In this image, edges are almost perfectly recovered.

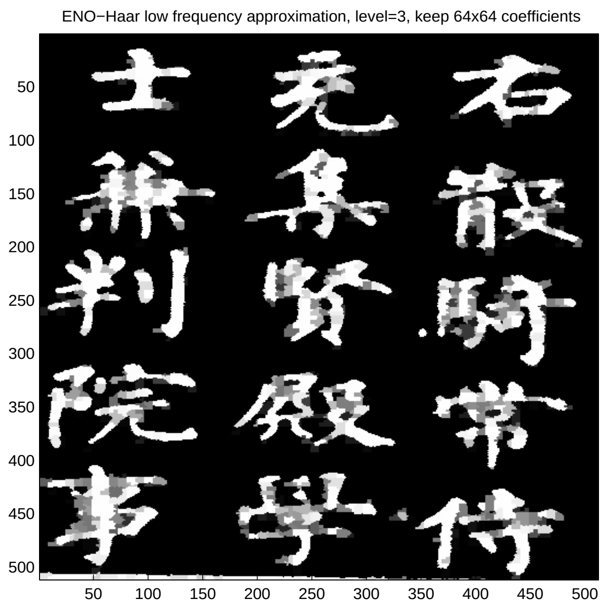
ENO−Haar low frequency approximation, level=3, keep 64x64 coefficients
50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500

FIG. 17. *The* 3*-level ENO-Haar approximation: Similar to Figure* 16*, the reconstruction is obtained from low frequency coefficients* α_{J-3} *only. Both the edges and the interior of the characters are clearer than those in the standard Haar linear approximation.*

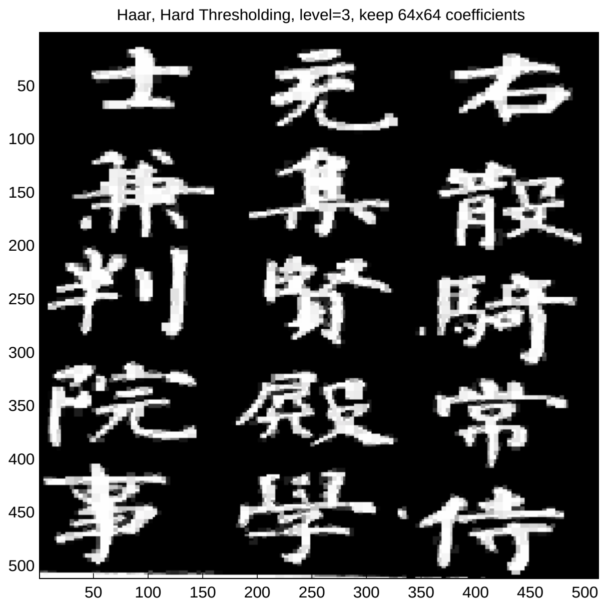

FIG. 18. *The* 3*-level standard Haar hard thresholding approximation: The image is reconstructed from the largest* 64×64 *wavelet coefficients (including* $\alpha_{J-3}, \beta_{J-3}, \beta_{J-2}, \beta_{J-1}$*). The edge artifacts are less severe than the standard linear approximation. On the other hard, the picture is comparable to the ENO-Haar low frequency approximation.*

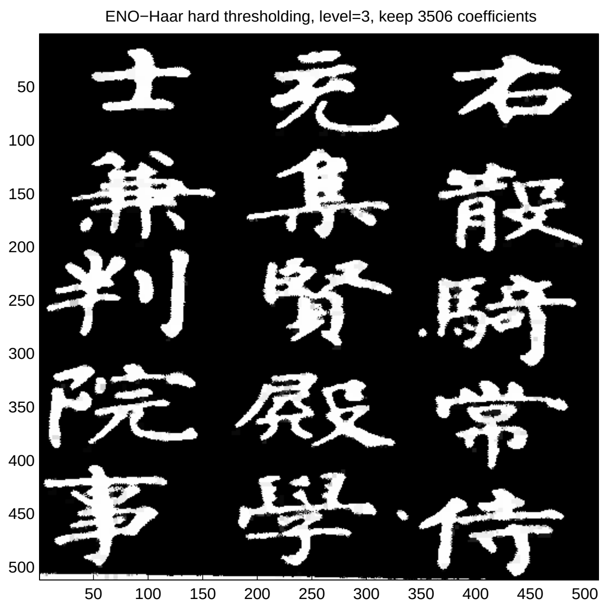

FIG. 19. *The 3-level ENO-Haar hard thresholding approximation: Similar to Figure 18, the image is reconstructed from the largest 64×64 ENO-wavelet coefficients (including $\alpha_{J-3}, \beta_{J-3}, \beta_{J-2}, \beta_{J-1}$). Less severe edge artifacts are generated compared to the previous images.*

REFERENCES

[1] A. ARNEODO, *Wavelet analysis of fractals: From the mathematical concepts to experimental reality*, in Wavelets: Theory and Applications, G. Erlebacher, M. Hussaini, and L. Jameson, eds., Oxford University Press, New York, 1996.

[2] S. AMAT, F. ARANDIGA, A. COHEN, R. DONAT, G. GARCIA, AND M. VON OEHSEN, *Data compression with ENO schemes: A case study*, Appl. Comput. Harmon. Anal., 11 (2001), pp. 273–288.

[3] S. AMAT, F. ARANDIGA, A. COHEN, AND R. DONAT, *Tensor product multiresolution analysis with error control for compact image representations*, Signal Process., to appear.

[4] F. ARANDIGA AND R. DONAT, *Nonlinear multiscale decompositions: The approach of A. Harten*, Numer. Algorithms, 23 (2000), pp. 175–216.

[5] G. BEYLKIN, R. COIFMAN, AND V. ROKHLIN, *Fast wavelet transforms and numerical algorithms*, Comm. Pure Appl. Math., 44 (1991), pp. 141–183.

[6] A. CHAMBOLLE, R. DEVORE, N. LEE, AND B. LUCIER, *Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage*, IEEE Trans. Image Process., 7 (1998), pp. 319–335.

[7] E. CANDES AND D. DONOHO, *Ridgelets: A Key to higher-dimensional intermittency?*, R. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci., 357 (1999), pp. 2495–2509.

[8] T. F. CHAN AND H. M. ZHOU, *Adaptive ENO-wavelet Transforms for Discontinuous Functions*, CAM Report 99-21, UCLA, Los Angeles, CA, 1999.

[9] A. COHEN AND B. MATEI, *Compact representations of images by edge adapted multiscale transforms*, in Proceedings of the IEEE ICIP Conference, Tessaloniki, Greenland, 2001, to appear.

[10] R. COIFMAN AND D. DONOHO, *Translation invariant de-noising*, in Wavelets and Statistics, A. Antoniadis and G. Oppenheim, eds., Springer-Verlag, New York, 1995, pp. 125–150.

[11] C. K. CHUI, *Wavelet: A Mathematical Tool for Signal Analysis*, SIAM Monogr. Math. Model. Comput. 1, SIAM, Philadelphia, 1997.

[12] P. CLAYPOOLE, G. DAVIS, W. SWELDENS, AND R. BARANIUK, *Nonlinear Wavelet Transforms for Image Coding*, preprint, 1999, IEEE Trans. Image Process., submitted.

[13] I. Daubechies, *Orthonormal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 41 (1988), pp. 909–996.

[14] I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 61, SIAM, Philadelphia, 1992.

[15] D. Donoho, *De-noising by soft thresholding*, IEEE Trans. Inform. Theory, 41 (1995), pp. 613–627.

[16] D. Donoho, *Wedgelets: Nearly-Minimax Estimation of Edges*, Technical report, Department of Statistics, Stanford University, Stanford, CA, 1997.

[17] D. Donoho, *Orthonormal Ridgelets and Linear Singularities*, Technical report, Department of Statistics, Stanford University, Stanford, CA, 1998.

[18] D. Donoho, I. Daubechies, R. DeVore, and M. Vetterli, *Data Compression and Harmonic Analysis*, preprint, Stanford University, Stanford, CA, 1998.

[19] D. Donoho and I. Johnstone, *Adapting to unknown smoothness via wavelet shrinkage*, J. Amer. Statist. Assoc., 90 (1995), pp. 1200–1224.

[20] A. Harten, *Discrete multi-resolution analysis and generalized wavelet*, Appl. Numer. Math., 12 (1993), pp. 153–192.

[21] A. Harten, *Multiresolution Representation of Data,* II. *General Framework*, CAM Report 94-10, UCLA, Los Angeles, CA, 1994.

[22] A. Harten, *Multiresolution Representation of Cell-Averaged Data*, CAM Report 94-21, UCLA, Los Angeles, CA, 1994.

[23] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy, *Uniformly high order essentially non-oscillatory schemes,* III, J. Comput. Phys., 71 (1987), pp. 231–303.

[24] S. Jaffard, *Exposants de Hölder en des points donnés et coefficients d'ondelettes*, C. R. Acad. Sci. Paris Sér. I Math., 308 (1989), pp. 79–81.

[25] S. Mallat, *Multiresolution approximation and wavelet orthonormal bases of $L^2(R)$*, Trans. Amer. Math. Soc., 315 (1989), pp. 69–87.

[26] S. Mallat, *A theory of multiresolution signal decomposition: The wavelet representation*, IEEE Trans. PAMI, 11 (1989), pp. 674–693.

[27] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, CA, 1998.

[28] J. Shapiro, *Embedded image coding using zerotrees of wavelet coefficients*, IEEE Trans. Signal Process., 41 (1993), pp. 3445–3462.

[29] C.W. Shu, *High order ENO and WENO schemes for computational fluid dynamics*, in High-Order Methods for Computational Physics, Lect. Notes Comput. Sci. Eng. 9, T. Barth and H. Deconinck, eds., Springer, Berlin, 1999, pp 439–582.

[30] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, MA, 1996.

[31] W. Sweldens, *The lifting scheme: A construction of second generation wavelets*, SIAM J. Math. Anal., 29 (1998), pp. 511–546.

[32] P. Tchamitchian, *Wavelets, Functions, and Operators*, in Wavelets: Theory and Applications, G. Erlebacher, M. Hussaini, and L. Jameson, eds., Oxford University Press, New York, 1996.

[33] J. R. Williams and K. Amaratunga, *A Discrete Wavelet Transform without Edge Effects*, IESL Technical report 95-02, MIT, Cambridge, MA, 1995.