The Easy Path Wavelet Transform: A New Adaptive Wavelet Transform for Sparse Representation of Two-dimensional Data

GERLIND PLONKA
Department of Mathematics, University of Duisburg-Essen,
Campus Duisburg, 47048 Duisburg, Germany
gerlind.plonka@uni-due.de

DEDICATED TO MANFRED TASCHE ON THE OCCASION OF HIS 65TH BIRTHDAY

Abstract

We introduce a new locally adaptive wavelet transform, called **Easy Path Wavelet Transform** (EPWT), that works along pathways through the array of function values and exploits the local correlations of the data in a simple appropriate manner. The usual discrete orthogonal and biorthogonal wavelet transform can be formulated in this approach. The EPWT can be incorporated into a multiresolution analysis structure and generates data dependent scaling spaces and wavelet spaces. Numerical results show the enormous efficiency of the EPWT for representation of two-dimensional data.

Key words. wavelet transform along pathways, data compression, adaptive wavelet bases, directed wavelets

AMS Subject classifications. 65T60, 42C40, 68U10, 94A08

1 Introduction

A crucial problem in data analysis is to construct efficient low-level representations, thereby providing a precise characterization of features which compose it, such as edges and texture components. Fortunately, in many relevant applications, the components of the given multidimensional data are not independent, and so the data points can be assumed to lie on or close to a low-dimensional nonlinear manifold embedded in a multidimensional space.

Wavelets are particularly efficient to represent signals. Yet tensor product wavelet bases are suboptimal for representing geometric structures because their support is not adapted to directional geometric properties.

In order to be able to achieve sparser representations of images, correlations along contours and geometry have to be taken into account. This is not straightforward to do with an orthogonal basis, due to the large variety of contour features.

Many nice ideas have been developed to design approximation schemes for a more efficient representation of two-dimensional data.

Curvelets [4, 5] and shearlets [15, 16] are examples of non-adaptive highly redundant function frames with strong anisotropic directional selectivity. For piecewise Hölder continuous functions of order 2 with discontinuities along C^2-curves Candès and Donoho [4] proved that a best approximation f_M of a given function f with M curvelets satisfies

$$\|f - f_M\|^2 \leq C\, M^{-2} \, (\log_2 M)^3,$$

while a (tensor product) wavelet expansion only leads to an approximation error $\mathcal{O}(M^{-1})$ [21]. Up to the $(\log M)^3$ factor, this curvelet approximation result is asymptotically optimal (see [12], Section 7.4). A similar estimation has been achieved by Guo and Labate [15] for shearlet frames.

However these results are not adaptive with respect to the assumed regularity of the image. Thus, these frame approximations lose their near optimal properties when the image is composed of edges which are not exactly piecewise C^2.

Instead of choosing a priori a basis or a frame to approximate f, one can rather adapt the approximation scheme to the image geometry. Within the last years, different approaches have been developed in this direction [1, 2, 3, 6, 7, 9, 11, 13, 14, 17, 18, 19, 20, 22, 23, 25, 26, 27]. For example, one can construct an approximation f_M which is piecewise linear over an optimized triangulation including M triangles and satisfies $\|f - f_M\|^2 \leq C\, M^{-2}$. This requires adapting the triangulation to the edge geometry (see e.g. [11]). Also, different image processing algorithms have been developed to find such approximations by detecting edges and constructing regular approximations between the edges where the image is uniformly regular (see e.g. [26]). Donoho introduced multiscale strategies to approximate the image geometry by wedgelets in order to obtain fast polynomial time algorithms [14]. Wakin et al. proposed a compression scheme that mixes wedgelets and wavelets to obtain better practical approximation results [27]. Recently, Dekel and Leviatan introduced geometric wavelets, based on an adaptive partition of the domain to match the geometry of a given input function [9]. Recently, Jacques and Antoine [19] proposed a wavelet frame construction with adaptive angular selectivity.

In [20], bandelet orthogonal bases and frames are introduced that adapt the geometric regularity of the image. The considered bandelets are anisotropic wavelets that are warped along a geometrical flow and generate bandelet orthonormal bases in different bands. With this construction and an even generalized image model, where the edges may be blurred, LePennec and Mallat [20] succeeded to show that their bandelet dictionary yields asymptotically optimal M-term approximations.

In our opinion, the **key idea** is, not to start with the construction of a basis or frame of scaling functions (and wavelets) but to consider locally adaptive low-pass and high-pass filters that will determine a multiresolution structure.

Nonlinear lifting schemes that have been considered already by Claypoole et al. [6]. Further successful attempts in this direction are the nonlinear edge adapted multiscale decompositions based on ENO schemes in [1, 2, 3, 7, 17, 18, 23]. Based on ideas by Harten [17, 18], one starts with the cell average discretization of an integrable function f at a fixed resolution 2^{-j} and applies a linear decimation operator (usually a simple average filter like the Haar low-pass filter). The choice of the nonlinear prediction operator is now crucial; one faces the problem of accurately reconstructing f from its cell-average data. The constructed nonlinear prediction operators use the Essentially Non-Oscillatory (ENO) techniques, together with subcell resolution (ENO-SR) [1, 2, 23] and its two-dimensional extension (ENO-EA) [3, 7, 23]. Basically, for each subcell I_k one selects among all stencils $\{S_{k-m}, \ldots, S_{k+m}\}$, which contain I_k, the stencil $\tilde{S}_k$ that minimizes a certain measure of oscillation. Hence, the prediction operator essentially depends on the data in the neighborhood of I_k. With this scheme the simple tree-like structure of wavelet decompositions is retained, while at the same time a specific treatment of edges is incorporated within transformation process. We remark that the ENO-EA schemes lead to an optimal N-term approximation, i.e., $\|f - f_M\|^2 \leq C\, M^{-2}$ for piecewise smooth functions with discontinuities along C^2-curves. Moreover, unlike the non-adaptive schemes, the ENO-EA multiresolution techniques provide optimal approximation results also for BV-spaces and L^p spaces, see [3, 24].

Mallat [22] has proposed a different idea, studying harmonic analysis transforms that have similarities with Gestalt groupings. He implements a weighted Haar wavelet transform which is applied successively to points that are grouped by a so-called association field. Again, the low-pass filters are just averaging Haar filters, and the non-linearity lies in the fact, that (depending on the determined subgrids) one can choose the "most suitable" neighbor of each image value for applying the averaging process. This idea leads to a grouplet orthonormal Haar basis that adaptively depends on the underlying signal.

In this paper we want to explore the new idea of nonlinear locally adaptive easy path wavelet transform (EPWT).

As in ENO-EA schemes, our considerations start from the discrete point of view. Given a matrix of data that are obtained e.g. by a cell-average discretization of an integrable function f at a fixed resolution, how one can find a way to process the data thereby exploiting the local correlations of the data efficiently?

We will restrict ourselves to the two-dimensional case but the algorithm presented in the paper can be easily transferred to higher dimensions. The idea is very simple. Starting with some suitable point of a given data set, we look in a first step for a path through all data points, such that there is a strong correlation between neighboring data points in this path. For the *rigorous EPWT*, the values in the path vector can be found consecutively as follows. Having reached one data point, we look at the neighborhood of this point and choose the "best neighbor", which has not been used yet in the path vector. This "best neighbor" will be the next data point in the path. We have to make sure, that all given data points are used in the path and that no data point is taken twice. Then we can apply a suitable one-dimensional discrete wavelet transform to this vector, and the choice of the path will ensure that most wavelet coefficients remain small. The same idea will be repeated to the low-pass part obtained after applying the wavelet transform. The values from the downsampled low-pass vector will be used to determine a low-pass filtered image that has the same size as the original image and consists of pairs of (usually neighboring) data points with equal value. Regarding these pairs of data points as units we now look for a path through these pairs and apply the discrete wavelet transform again to the resulting path vector. This idea will be repeatedly applied. In the jth level we consider sets of 2^j data points in the low-pass filtered image with equal values und apply the discrete wavelet transform along a path through these sets.

In the paper, we shall use the convention that the path vectors $\mathbf{p} = (\mathbf{p}(l))_{l=0}^{N-1}$ are vectors of indices and the wavelet transform will be applied to the corresponding data, i.e., to $(f(\mathbf{p}(l))_{l=0}^{N-1}$.

After a suitable number of such iterations, we can apply a shrinkage procedure to the wavelet coefficients obtained at the different levels in order to find a compressed digital representation of the function.

For reconstruction of the function, one needs the vectors of wavelet coefficients and the low-pass coefficients from the coarsest level as usual. Furthermore, in this approach we also need to have the path vectors of indices along which the wavelet transform has been applied at the different levels.

As we shall see from the numerical experiments, the EPWT can lead to an efficient compression of digital two-dimensional data.

Observe that the EPWT is (usually) not a one-dimensional transform! Only in the trivial case, if the path vectors at the second level and at all further considered levels of EPWT are trivial, i.e., are identities on the (ordered) index sets, a one-dimensional transform arises with rather poor approximation properties. Since we use a new path vector at each level of the transform, the EPWT is highly nonlinear and data dependent.

We will show that the EPWT determines a generalized multiresolution structure and defines scaling and wavelet spaces whose basis functions depend on the considered data. As usual, the hierarchical structure of these spaces is maintained. We shall also focus on the difference between the EPWT and a tensor product wavelet transform. In particular, we show that the usual tensor product wavelet transform can not been seen as a special case of the EPWT with suitably chosen path vectors. The approximation properties of the EPWT scheme will be subject to further research.

Facing the efficient application of the EPWT, we have also to study the cost of adaptivity, i.e., the cost of storing the path vectors. These considerations lead us to a modification of the rigorous EPWT, the so-called *relaxed EPWT*, such that an efficient storing of the path vectors is possible.

The paper is organized as follows. In Section 2 we shall give the definitions and notations that will be used in this paper. Section 3 is devoted to a detailed description of the (rigorous) EPWT. The cost of adaptivity will be investigated in Section 4. In particular, we shall give a procedure for path storing and derive the *relaxed EPWT* in order to obtain path vectors that can be stored in a cheaper way. In Section 5, we study the multiresolution structure of the EPWT for the Haar wavelet transform. In particular, we will emphasize the differences between the EPWT and the tensor product wavelet approach. Finally, in Section 6 we illustrate the efficiency of the new algorithm by numerical examples. We shall apply the EPWT to images and show its ability to sparsely represent directed structures. The EPWT algorithm is particularly efficient for representation of real data sets.

2 Definitions and Notation

In order to explain the new idea of the EPWT, where we want to use the discrete one-dimensional wavelet transform along *path vectors* through the matrix of data, we first need some definitions and notations.

Let N_1, N_2 be two positive integers with $N_1 N_2 = 2^L s$, with $L,\, s \in \mathbb{N}$. For a digital function $f = (f(i,j))_{i=0,j=0}^{N_1-1,N_2-1}$ let $I = \{(i,j) : i = 0,\ldots,N_1-1,\ j = 0,\ldots,N_2-1\}$ be the corresponding *index set*. We say that for a given index (i,j) and a given function f, the function value $f(i,j)$ *corresponds* to the index (i,j). Further, we shall use a one-dimensional representation of the index set I by taking the bijective mapping $J : I \to \{0,\ldots,N_1N_2-1\}$, where

$$J((i,j)) := i + j\, N_1.$$

The inverse mapping is then given by

$$j = \Big\lfloor \frac{J((i,j))}{N_1} \Big\rfloor,\ i = J((i,j)) - jN_1.$$

Let $\mathbf{f}^L \in \mathbb{R}^{N_1N_2}$ be the vector of function values of f corresponding to the index set $J(I)$, obtained by concatenating all columns of the matrix f,

$$\mathbf{f}^L = (f(0,0), f(1,0), \ldots, f(N_1-1,0), f(0,1), f(1,1), \ldots, f(N_1-1, N_2-1))^T.$$

We say that for a given index $l = J((i,j))$ and a given function f, the function value $\mathbf{f}^L(l) = f(i,j)$ corresponds to the index l.

We define a *neighborhood* of an index $(i,j) \in I$ by

$$N(i,j) = \{(i_1,j_1) \in I \setminus \{(i,j)\} : |i-i_1| \le 1, |j-j_1| \le 1\}.$$

Hence, an index that lies not at the "boundary", i.e., $i \notin \{0, N_1-1\}$ and $j \notin \{0, N_2-1\}$, has eight neighbors, an index at the boundary but not at a vertex has five neighbors, and an index at the vertex has only three neighbors, (for instance $(0,0)$ has the neighbors $(0,1)$, $(1,0)$ and $(1,1)$).

Using the one-dimensional index set $J(I) = \{0,1,\ldots,N_1N_2-1\}$ instead of I, a neighborhood of an index $l \in J(I)$ with $l = J((i,j))$ is given by $N(l) = J(N(i,j))$. As before, an index l not lying at the boundary has 8 neighbors,

$$N(l) = \{l+1, l-1, l+N_1, l-N_1, l+N_1+1, l+N_1-1, l-N_1+1, l-N_1-1\}.$$

We shall also consider disjoint *partitions* E of $J(I)$, i.e. $E = \{J_1, J_2, \ldots, J_r\}$ where $J_\mu \cap J_\nu = \emptyset$ for $\mu \neq \nu$ and $\cup_{\nu=1}^{r} J_\nu = J(I)$. We then say that two subsets J_ν and J_μ from E with $\mu \neq \nu$ are *neighbors*, and we denote

$$J_\nu \in N(J_\mu),$$

if there exists an index $l \in J_\nu$ and an index $l_1 \in J_\mu$ such that $l \in N(l_1)$.

We will look for path vectors through index subsets of $J(I)$ and apply a one-dimensional wavelet transform along these path vectors.

We say that a vector of indices $(l_k, \ldots, l_{k+n})$, $0 \le k < k+n \le N_1N_2-1$, is *connected*, if we have $l_{\nu+1} \in N(l_\nu)$ for $\nu = k, \ldots, k+n-1$. Such a connected index vector is called *pathway.* We are interested in a *complete path* through the index set $J(I)$. A complete path through $J(I)$ is an integer vector of length N_1N_2 containing all indices of $J(I)$ in a certain order, i.e., it is a permutation of $(0, 1, 2, \ldots, N_1N_2 - 1)^T$. This complete path should be composed of a number of pathways, i.e. $(\mathbf{p}_1^T \mid \mathbf{p}_2^T \mid \ldots \mid \mathbf{p}_r^T)^T$, where $\mathbf{p}_\nu$, $\nu = 1, \ldots, r$, are connected (column) vectors of indices, and it is regarded as a column vector of length N_1N_2 by concatenating $\mathbf{p}_1, \ldots, \mathbf{p}_r$.

One simple example of such a complete path is to take just $(0, 1, 2, \ldots, N_1N_2-1)^T$, and the pathways used here are $\mathbf{p}_1 = (0, 1, \ldots, N_1 - 1)^T$, $\mathbf{p}_2 = (N_1, N_1 + 1, \ldots, 2N_1 - 1)^T$,. . ., $\mathbf{p}_{N_2} = ((N_2 - 1)N_1, (N_2 - 1)N_1 + 1, \ldots, N_2N_1 - 1)^T$, such that the complete path vector is composed by N_2 pathways. The transition from one pathway to the next in the path vector is called *interruption.*

3 The rigorous EPWT

Let us now describe the (rigorous) easy path wavelet transform in full detail. We start with the decomposition of the data $f \in \mathbb{R}^{N_1 \times N_2}$ resp. $\mathbf{f}^L \in \mathbb{R}^{N_1N_2}$, where we assume that $N_1N_2 = 2^L s$ with $L, s \in \mathbb{N}$. Then we will be able to apply L levels of the EPWT.

Decomposition

First level

At the first level, we determine a complete path vector $\mathbf{p}^L$ through $J(I)$ and then apply the discrete one-dimensional (periodic) wavelet transform to the function values along this path $\mathbf{p}^L$.

We start with $\mathbf{p}^L(0) := 0$. In order to determine the second index $\mathbf{p}^L(1)$, we seek the minimum of absolute differences of the function values corresponding to the neighborhood of the index 0, and put

$$\mathbf{p}^L(1) := \underset{k}{\operatorname{argmin}} \, \{|\mathbf{f}^L(0) - \mathbf{f}^L(k)|, k \in \{1, N_1, N_1 + 1\}\}.$$

For example, the second value $\mathbf{p}^L(1)$ is equal to $N_1 + 1$ if $|\mathbf{f}^L(0) - \mathbf{f}^L(N_1 + 1)| = |f(0, 0) - f(1, 1)|$ attains the minimum above. We proceed in the same way in order to determine the index $\mathbf{p}^L(2)$. Let for example the first two values of $\mathbf{p}^L$ be given by $\mathbf{p}^L(0) = 0$ and $\mathbf{p}^L(1) = N_1 + 1$. We now look for the minimum of absolute differences of the function values corresponding to the neighborhood of the index $N_1 + 1$, but we do not consider the index 0 that has been used already in the path vector $\mathbf{p}^L$. Here, we determine

$$\mathbf{p}^L(2) := \underset{k}{\operatorname{argmin}} \, \{|\mathbf{f}^L(N_1 + 1) - \mathbf{f}^L(k)|, k \in \{1, 2, N_1, N_1 + 2, 2N_1, 2N_1 + 1, 2N_1 + 2\}\}.$$

We proceed in this manner, thereby determining a path vector through the index set $J(I)$ that is locally adapted to the function f (easy path). With the above procedure we obtain a pathway such that the absolute differences between neighboring function values $\mathbf{f}^L(l)$ along the path remain as small as possible. Generally, having given the index $\mathbf{p}^L(l)$, $0 \le l \le N_1N_2 - 2$, we usually determine the next value $\mathbf{p}^L(l + 1)$ by

$$\mathbf{p}^L(l + 1) := \underset{k}{\operatorname{argmin}} \, \{|\mathbf{f}^L(\mathbf{p}^L(l)) - \mathbf{f}^L(k)|, k \in N(\mathbf{p}^L(l)), k \neq \mathbf{p}^L(\nu), \ \nu = 0, \ldots, l\}.$$

It can happen that the choice of the next index value $\mathbf{p}^L(l+1)$ is not unique, since the above minimum is attained by more than one index. In this case, one may just fix a favorite direction in order to determine a unique pathway.

Another situation can occur during the procedure, namely that all indices in the neighborhood of a considered index $\mathbf{p}^L(l)$ have already been used in the path $\mathbf{p}^L$. In this case we need to start with a new pathway, i.e., we have an interruption in the path vector. We need to choose one index $\mathbf{p}^L(l+1)$ from the remaining "free" indices in $J(I)$ that have not been taken in $\mathbf{p}^L$ yet.

There are different possibilities how to start the next pathway. One simple choice is to take just the smallest index from $J(I)$ that has not been used so far. Another choice is to look for a next index, such that again the absolute difference $|\mathbf{f}^L(\mathbf{p}^L(l)) - \mathbf{f}^L(\mathbf{p}^L(l+1))|$ is minimal, i.e., we take in this case

$$\mathbf{p}^L(l+1) = \operatorname*{argmin}_k \{|\mathbf{f}^L(\mathbf{p}^L(l)) - \mathbf{f}^L(k)|, k \in J(I), k \neq \mathbf{p}^L(\nu),\ \nu = 0, \ldots, l\}. \qquad (3.1)$$

We proceed in this manner and obtain finally a complete path vector $\mathbf{p}^L \in \mathbb{Z}^{N_1 N_2}$ that is a permutation of $(0, 1, \ldots, N_1 N_2 - 1)^T$.

After having constructed the path $\mathbf{p}^L$, we apply one level of the discrete one-dimensional Haar wavelet transform or any other discrete orthogonal or biorthogonal periodic wavelet transform to the vector of function values $(\mathbf{f}^L(\mathbf{p}^L(l)))_{l=0}^{N_1 N_2 - 1}$ along the path $\mathbf{p}^L$. We obtain the vector $\mathbf{f}^{L-1} \in \mathbb{R}^{N_1 N_2/2}$ containing the low-pass part and the vector of wavelet coefficients $\mathbf{g}^{L-1} \in \mathbb{R}^{N_1 N_2/2}$. While the wavelet coefficients will be stored in $\mathbf{g}^{L-1}$, we proceed now further with the low-pass vector $\mathbf{f}^{L-1}$ at the second level.

Second level

By slightly blowing up the low-pass vector $\mathbf{f}^{L-1}$, we first construct a *smoothed function vector* $\tilde{\mathbf{f}}^{L-1} \in \mathbb{R}^{N_1 N_2}$ as

$$\tilde{\mathbf{f}}^{L-1}(\mathbf{p}^L(2l)) := \mathbf{f}^{L-1}(l), \qquad \tilde{\mathbf{f}}^{L-1}(\mathbf{p}^L(2l+1)) := \mathbf{f}^{L-1}(l), \quad l = 0, \ldots N_1 N_2/2 - 1,$$

i.e., the obtained smoothed values in $\mathbf{f}^{L-1}$ now determine two (usually neighboring) function values in $\tilde{\mathbf{f}}^{L-1}$. See Figure 1 (middle) for the Haar wavelet transform. The corresponding index sets

$$J_l^{L-1} := \{\mathbf{p}^L(2l), \mathbf{p}^L(2l+1)\}, \qquad l = 0, \ldots N_1 N_2/2 - 1,$$

determine a partition of $J(I)$. Now, each such index set J_l^{L-1} is considered as *one object* and corresponds to the function value $\mathbf{f}^{L-1}(l)$.

Again we want to exploit the local correlations in the smoothed array of function values determined by $\mathbf{f}^{L-1}$ (resp. $\tilde{\mathbf{f}}^{L-1}$). We repeat the same procedure as in the first step, but replacing the single indices with corresponding function values by the new index sets J_l^{L-1} and the corresponding smoothed function values $\mathbf{f}^{L-1}(l)$. The new path vector $\mathbf{p}^{L-1} \in \mathbb{Z}^{N_1 N_2/2}$ is now a permutation of $(0, 1, \ldots,\ N_1 N_2/2 - 1)^T$.

We start again with the first index set J_0^{L-1}, i.e., $\mathbf{p}^{L-1}(0) := 0$. We consider the minimum of the absolute differences of function values of $\mathbf{f}^{L-1}$ for all neighboring index sets, and choose the set for which the minimum is obtained. More precisely, we take

$$\mathbf{p}^{L-1}(1) := \operatorname*{argmin}_k \{|\mathbf{f}^{L-1}(0) - \mathbf{f}^{L-1}(k)| : J_k^{L-1} \in N(J_0^{L-1})\}.$$

We proceed in the same way. Having already found $\mathbf{p}^{L-1}(l)$, $0 \le l \le N_1N_2/2-2$, we determine the next value $\mathbf{p}^{L-1}(l+1)$ by

$$\begin{aligned}\mathbf{p}^{L-1}(l+1) &:= \operatorname*{argmin}_k \{|\mathbf{f}^{L-1}(\mathbf{p}^{L-1}(l)) - \mathbf{f}^{L-1}(k)| : J_k^{L-1} \in N(J^{L-1}_{\mathbf{p}^{L-1}(l)}), \\ &\qquad k \ne \mathbf{p}^{L-1}(\nu),\, \nu = 0,\ldots,l\}.\end{aligned}$$

Recall that the neighborhood of the index set J_l^{L-1} is here

$$N(J_l^{L-1}) = \{J_k^{L-1} : J_k^{L-1} \cap \big(N(\mathbf{p}^L(2l)) \cup N(\mathbf{p}^L(2l+1))\big) \ne \emptyset,\ k \ne l\}.$$

If the new value $\mathbf{p}^{L-1}(l+1)$ is not uniquely determined by the minimizing procedure, we can just fix favorite directions in order to obtain a unique path. If for the set $J_{\mathbf{p}^{L-1}(l)}$ there is no neighboring index set that has not been used yet in the path vector $\mathbf{p}^{L-1}$, then we have to interrupt the path and to find a good index set (that has been not used so far) to start a new pathway. As at the first level, we try to keep the differences of function values along the path small and choose in this case e.g.

$$\begin{aligned}\mathbf{p}^{L-1}(l+1) &:= \operatorname*{argmin}_k \{|\mathbf{f}^{L-1}(\mathbf{p}^{L-1}(l)) - \mathbf{f}^{L-1}(k)|, 0 \le k \le N_1N_2/2-1, \\ &\qquad k \ne \mathbf{p}^{L-1}(\nu),\ \nu = 0,\ldots,l\}.\end{aligned}$$

After having completed the path vector $\mathbf{p}^{L-1}$, we apply again the chosen discrete (periodic) wavelet transform to the vector $(\mathbf{f}^{L-1}(\mathbf{p}^{L-1}(l)))_{l=0}^{N_1N_2/2-1}$ along the path $\mathbf{p}^{L-1}$. Assuming that $N_1N_2/4 \in \mathbb{N}$, i.e. $L \ge 2$, we obtain the vector $\mathbf{f}^{L-2} \in \mathbb{R}^{N_1N_2/4}$ containing the low-pass part and the vector of wavelet coefficients $\mathbf{g}^{L-2} \in \mathbb{R}^{N_1N_2/4}$. While the wavelet coefficients in $\mathbf{g}^{L-2}$ will be stored, we proceed now again with the low-pass vector $\mathbf{f}^{L-2}$ in the next step.

Further levels

If N_1N_2 is of the form $2^L s$ with $s \in \mathbb{N}$ being greater than or equal to the lengths of low-pass and high-pass filters in the used discrete wavelet transform, then we may apply the procedure L times. For a given vector $\mathbf{f}^{L-j}$, $0 < j < L$, we consider in the $(j+1)$-th step the index sets

$$J_l^{L-j} = J^{L-j+1}_{\mathbf{p}^{L-j+1}(2l)} \cup J^{L-j+1}_{\mathbf{p}^{L-j+1}(2l+1)}, \qquad l = 0,\ldots,N_1N_2/2^j - 1$$

with the corresponding function values $\mathbf{f}^{L-j}(l)$. Then we determine a path vector $\mathbf{p}^{L-j}$ of length $N_1N_2/2^j$ as a permutation of $(0,1,\ldots,N_1N_2/2^j-1)^T$. We start with $\mathbf{p}^{L-j}(0) := 0$ as before, and having found $\mathbf{p}^{L-j}(l)$ we determine

$$\begin{aligned}\mathbf{p}^{L-j}(l+1) &:= \operatorname*{argmin}_k \{|\mathbf{f}^{L-j}(\mathbf{p}^{L-j}(l)) - \mathbf{f}^{L-j}(k)| : J_k^{L-j} \in N(J^{L-j}_{\mathbf{p}^{L-j}(l)}), \\ &\qquad k \ne \mathbf{p}^{L-j}(\nu),\, \nu = 0,\ldots,l\},\end{aligned}$$

where

$$N(J_l^{L-j}) = \{J_k^{L-j} :\, J_k^{L-1} \cap \Big(N(J^{L-j+1}_{\mathbf{p}^{L-j+1}(2l)}) \cup N(J^{L-j+1}_{\mathbf{p}^{L-j+1}(2l+1)})\Big) \ne \emptyset,\ k \ne l\}.$$

If $\{|\mathbf{f}^{L-j}(\mathbf{p}^{L-j}(l)) - \mathbf{f}^{L-j}(k)| : J_k^{L-j} \in N(J^{L-j}_{\mathbf{p}^{L-j}(l)}), k \neq \mathbf{p}^{L-j}(\nu),\ \nu = 0, \ldots, l\}$ is an empty set, then an interruption occurs and we put

$$\begin{aligned}\mathbf{p}^{L-j}(l+1) \quad := \quad & \underset{k}{\operatorname{argmin}} \{|\mathbf{f}^{L-j}(\mathbf{p}^{L-j}(l)) - \mathbf{f}^{L-j}(k)|, 0 \leq k \leq N_1 N_2/2^j - 1, \\ & \qquad\qquad k \neq \mathbf{p}^{L-j}(\nu),\ \nu = 0, \ldots, l\}.\end{aligned}$$

We apply the (periodic) wavelet transform to the vector $(\mathbf{f}^{L-j}(\mathbf{p}^{L-j}(l)))_{l=0}^{N_1 N_1/2^j - 1}$ along the path $\mathbf{p}^{L-j}$ thereby obtaining the low-pass vector $\mathbf{f}^{L-j-1} \in \mathbb{R}^{N_1 N_2/2^{j+1}}$ and the vector of wavelet coefficients $\mathbf{g}^{L-j-1} \in \mathbb{R}^{N_1 N_2/2^{j+1}}$.

Output

As output of the complete procedure after L iterations we obtain the coefficient vector

$$\mathbf{g} = ((\mathbf{f}^0)^T, (\mathbf{g}^0)^T, (\mathbf{g}^1)^T, \ldots, (\mathbf{g}^{L-1})^T)^T \in \mathbb{R}^{N_1 N_2}$$

and the vector determining the paths in each iteration step

$$\mathbf{p} = ((\mathbf{p}^1)^T, (\mathbf{p}^2)^T, \ldots, (\mathbf{p}^L)^T)^T \in \mathbb{R}^{2N_1 N_2(1-1/2^L)}.$$

These two vectors now contain the entire information about the originally given function $f \in \mathbb{R}^{N_1 \times N_2}$ (resp. $\mathbf{f}^L \in \mathbb{R}^{N_1 N_2}$).

In order to find a sparse representation of the original function f, we apply a *shrinkage procedure* to the wavelet coefficients in the vectors $\mathbf{g}^j$, $j = 0, \ldots, L-1$. In our experiments in Section 4, we shall use the hard threshold function

$$S_\sigma(x) = \begin{cases} x & |x| \geq \sigma, \\ 0 & |x| < \sigma. \end{cases}$$

Reconstruction

The reconstruction of f resp. $\mathbf{f}^L$ from $\mathbf{g}$ and $\mathbf{p}$ is given as follows.

```
For $j = 0$ to $L-1$ do
    Apply the inverse discrete wavelet transform to the vector $\begin{pmatrix} \mathbf{f}^j \\ \mathbf{g}^j \end{pmatrix} \in \mathbb{R}^{s2^j}$ in order to obtain $\mathbf{f}_p^{j+1} \in \mathbb{R}^{s2^{j+1}}$.
    Apply the permutation
        $\mathbf{f}^{j+1}(\mathbf{p}^{j+1}(k)) := \mathbf{f}_p^{j+1}(k), \qquad k = 0, \ldots, s2^{j+1} - 1.$
end.
```

Example 3.1 *We illustrate the simple idea of the EPWT in a small example. Let the following matrix of function values be given*

$$f = \begin{pmatrix} 0.4492 & 0.4219 & 0.4258 & 0.4375 \\ 0.4141 & 0.4531 & 0.4180 & 0.4258 \\ 0.4375 & 0.4297 & 0.4219 & 0.4219 \\ 0.4219 & 0.4258 & 0.4023 & 0.4141 \end{pmatrix}.$$

We use the one-dimensional numbering of indices given by $J(I)$ *with* $N_1 = N_2 = 4$, *i.e.,* $N_1 N_2 = 2^4$. *Applying the procedure described above and using the condition (3.1), the first path vector reads*

$$\mathbf{p}^4 = (0, 5, 2, 6, 7, 3 \mid 4, 9, 14, 10, 13, 8, 12 \mid 1 \mid 15, 14)^T,$$

where $\mid$ *indicates the interruptions. This path has three interruptions and is illustrated by arrows in Figure 1 (left).*

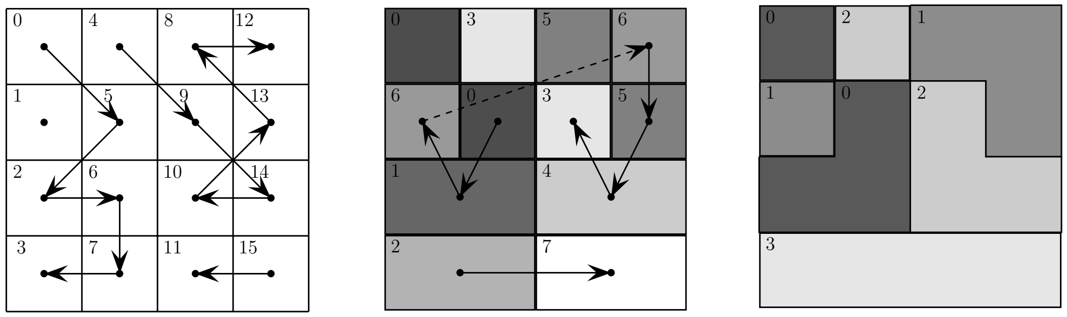

Figure 1. Illustration of the path vectors and the low-pass parts for the first two levels of the EPWT with Haar wavelet transform. Index sets at the second and third level are illustrated by different gray values.

Application of the Haar wavelet transform with (not normalized) filters $h(0) = h(1) = 1/2$ *and* $g(0) = -1/2$, $g(1) = 1/2$ *gives (with truncation after four digits)*

$$\begin{aligned}\mathbf{f}^3 &= (0.4512, 0.4336, 0.4238, 0.4199, 0.4219, 0.4258, 0.4258, 0.4082),\\ \mathbf{g}^3 &= (0.0020, -0.0039, -0.0020, -0.0020, 0, 0, -0.0117, -0.0059).\end{aligned}$$

We now proceed to the second level. For the 'smoothed array' of function values $\tilde{f}^3$,

$$\tilde{f}^3 = \begin{pmatrix} 0.4512 & 0.4199 & 0.4258 & 0.4258 \\ 0.4258 & 0.4512 & 0.4199 & 0.4258 \\ 0.4336 & 0.4336 & 0.4219 & 0.4219 \\ 0.4238 & 0.4238 & 0.4082 & 0.4082 \end{pmatrix},$$

we obtain the path $\mathbf{p}^3 = (0, 1, 6, 5, 4, 3 \mid 2, 7)$ *with one interruption illustrated by arrows in Figure 1 (middle). Application of Haar wavelet transform gives*

$$\begin{aligned}\mathbf{f}^2 &= (0.4424, 0.4258, 0.4209, 0.4160),\\ \mathbf{g}^2 &= (-0.0088, 0, -0.0010, -0.0078).\end{aligned}$$

At the third level we start with the smoothed matrix $\tilde{f}^2$,

$$\tilde{f}^2 = \begin{pmatrix} 0.4424 & 0.4209 & 0.4258 & 0.4258 \\ 0.4258 & 0.4424 & 0.4209 & 0.4258 \\ 0.4424 & 0.4424 & 0.4209 & 0.4209 \\ 0.4160 & 0.4160 & 0.4160 & 0.4160 \end{pmatrix}$$

illustrated in Figure 1 (right), and obtain the path $\mathbf{p}^2 = (0, 1, 2, 3)$*. This leads to*

$$\begin{aligned}\mathbf{f}^1 &= (0.4341, 0.4185),\\ \mathbf{g}^1 &= (-0.0083, -0.0024).\end{aligned}$$

Finally, for $\mathbf{p}^1 = (1, 2)$ *the last transform yields* $\mathbf{f}^0 = (0.4263)$ *and* $\mathbf{g}^0 = (-0.0078)$*.*

4 Cost of adaptivity: The relaxed EPWT

Compared to the usual tensor product wavelet transform, the rigorous EPWT introduced in Section 3 strongly reduces the number of wavelet coefficients needed for lossy compression of two-dimensional data with same quality, see Section 6.

However, for reconstruction using the EPWT, we have to store the path vector $\mathbf{p} = ((\mathbf{p}^1)^T, \dots, (\mathbf{p}^L)^T)^T \in \mathbb{R}^{2N_1N_2(1-1/2^L)}$, and the cost of storing this vector also needs to be taken into account. In this section we want to study how this can be done efficiently.

At the first level of the EPWT, in $\mathbf{p}^L$, the path usually connects neighboring indices, and we only need to store the next direction of the path instead of the whole next index. Obviously, having eight directions (i.e., at most eight neighbors of one index), we can store them with at most three bits. In the case of an interruption of the path, we may have to store the complete new index in the worst case (depending on the choice of the new starting index for the next pathway).

Compared with the costs for storing the wavelet coefficients, these costs of adaptivity are not negligible.

Therefore we want to propose a procedure for a cheaper storing of the information in the path vector $\mathbf{p}$. This considerations will also lead us to a modification of the rigorous EPWT algorithm presented in Section 3, the so-called *relaxed EPWT*, where a balance between costs of storage of wavelet coefficients and costs of adaptivity (storage of $\mathbf{p}$) can be found.

4.1 First level: How to store the information of $\mathbf{p}^L$?

First, we want to present a procedure for cheaper storing the path $\mathbf{p}^L$ as found with the rigorous EPWT algorithm in Section 3. In order to decrease these storing costs further we will later modify the rigorous EPWT algorithm suitably.

At the first level, the complete information of $\mathbf{p}^L$ of length N_1N_2 can now be stored in $\tilde{\mathbf{p}}^L$ as follows.

(i) Put $\tilde{\mathbf{p}}^L(0) := 0$ since $\mathbf{p}^L(0) = 0$ is the fixed starting point.

(ii) For determining the second value in $\tilde{\mathbf{p}}^L$ look clockwise through the neighbors of $\mathbf{p}^L(0)$ and put

$$\tilde{\mathbf{p}}^L(1) := \begin{cases} 0 & \text{if } \mathbf{p}^L(1) = N_1, \\ 1 & \text{if } \mathbf{p}^L(1) = N_1 + 1, \\ 2 & \text{if } \mathbf{p}^L(1) = 1. \end{cases}$$

i.e., 'right' is the fixed *favorite direction*, where we only need to store a zero in $\tilde{\mathbf{p}}^L(1)$.

(iii) For all further path values use the following convention.

The *favorite direction* (that will be mapped to 0 in $\tilde{\mathbf{p}}^L$) is the direction which is kept from the previous step, i.e., if

$$\mathbf{p}^L(l+1) = \mathbf{p}^L(l) + (\mathbf{p}^L(l) - \mathbf{p}^L(l-1)) = 2\mathbf{p}^L(l) - \mathbf{p}^L(l-1),$$

then $\tilde{\mathbf{p}}^L(l+1) := 0$. If the direction of the path is changed after the previous step then look clockwise through the neighbors (starting with the favorite direction) and determine $\tilde{\mathbf{p}}^L$ accordingly. More precisely, consider the vector $\mathbf{q} = (\mathbf{q}(\mu))_{\mu=0}^7 \in \mathbb{Z}^8$,

$$\mathbf{q} =: (N_1, N_1+1, 1, -N_1+1, -N_1, -N_1-1, -1, N_1-1)^T.$$

Find the index $\tilde{\mu} \in \{0, \ldots, 7\}$ with

$$\mathbf{q}(\tilde{\mu}) = \mathbf{p}^L(l) - \mathbf{p}^L(l-1),$$

such that $\mathbf{q}(\tilde{\mu})$ determines the *favorite direction*. Consider a cyclic shift $\tilde{\mathbf{q}}$ of $\mathbf{q}$ of the form

$$\tilde{\mathbf{q}} := (\mathbf{q}(\tilde{\mu}), \ldots, \mathbf{q}(7), \mathbf{q}(0), \ldots, \mathbf{q}(\tilde{\mu}-1))^T = (\tilde{\mathbf{q}}(\mu))_{\mu=0}^7,$$

such that $\tilde{\mathbf{q}}(0) = \mathbf{q}(\tilde{\mu})$. Observe that $\mathbf{p}^L(l-1)$ is a neighbor of $\mathbf{p}^L(l)$ that has been already taken in the path. In case that all seven remaining neighbors of $\mathbf{p}^L(l)$ have not been used in the path yet, put

$$\tilde{\mathbf{p}}^L(l+1) := \begin{cases} 0 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(0), \\ 1 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(1), \\ 2 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(2), \\ 3 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(3), \\ 4 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(5), \\ 5 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(6), \\ 6 & \mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(7). \end{cases}$$

The direction $\mathbf{p}^L(l+1) = \mathbf{p}(l) + \tilde{\mathbf{q}}(4)$ cannot occur in the path since $\tilde{\mathbf{q}}(4) = \mathbf{p}^L(l-1) - \mathbf{p}^L(l)$.

Let us call a neighbor index of $\mathbf{p}^L(l)$ *admissible*, if it is in $J(I)$, and if this index did not occur in the path already, i.e., it is different from $\mathbf{p}^L(\nu)$ for $\nu = 0, \ldots, l$.

If not all seven remaining neighbor indices of $\mathbf{p}^L(l)$ are admissible, we slightly change the above scheme by deleting first all components $\tilde{\mathbf{q}}(\mu)$ in $\tilde{\mathbf{q}}$ that do not lead to an admissible index $\mathbf{p}^L(l+1) = \mathbf{p}^L(l) + \tilde{\mathbf{q}}(\mu)$. Let $\tilde{\mathbf{q}}^{sh} = (\tilde{\mathbf{q}}^{sh}(\mu))_{\mu=0}^{m-1} \in \mathbb{Z}^m$, $m \leq 7$, contain only the admissible components of $\tilde{\mathbf{q}}$ (without changing the order!). Now put

$$\tilde{\mathbf{p}}^L(l+1) := \mu \quad \text{if} \quad \mathbf{p}^L(l+1) = \mathbf{p}^L(l) + \tilde{\mathbf{q}}^{sh}(\mu).$$

(iv) If the path $\mathbf{p}^L$ contains an interruption, i.e., one pathway ends at $\mathbf{p}^L(l)$ and $\mathbf{p}^L(l+1)$ is not a neighbor of $\mathbf{p}^L(l)$, then the determination of $\tilde{\mathbf{p}}(l+1)$ depends on the procedure used for finding the new starting index $\mathbf{p}^L(l+1)$. If $\mathbf{p}^L(l+1)$ is just the smallest index of all free indices left, then we simply take $\tilde{\mathbf{p}}^L(l+1) := 0$. (In fact we need not to store $\tilde{\mathbf{p}}^L(l+1)$ in this case.) Now we proceed further similarly as before with $\mathbf{p}^L(l+1) = \mathbf{p}^L(l) + N_1$ as the favorite direction. In our numerical examples (see Section 6), we have taken a special procedure for finding a suitable new starting point without high cost.

It can be simply observed that the above procedure, that maps $\mathbf{p}^L$ into $\tilde{\mathbf{p}}^L$, is reversible, i.e., $\mathbf{p}^L$ can be uniquely reconstructed from $\tilde{\mathbf{p}}^L$.

Example 4.1 *Consider again the 4×4 matrix of function values in Example 3.1. With the above procedure, we translate the path $\mathbf{p}^4 = (0,5,2,6,7,3\,|\,4,9,14,10,13,8,12 \mid 1 \mid 15,14)^T$ into the vector*

$$\tilde{\mathbf{p}}^4 = (0,1,2,1,2,0 \mid 1,1,0,3,0,0,0 \mid 0 \mid 1,0)^T,$$

with the convention that for a starting point of a new pathway, we store in $\tilde{\mathbf{p}}^4$ the index of this starting point in a vector of ordered remaining admissible indices.

Using the above procedure for storing the path $\tilde{\mathbf{p}}^L$, it is obvious that the cost of adaptivity decreases if the path changes directions less often. This observation leads to the concept of **relaxed EPWT algorithm**.

4.2 The relaxed EPWT (first level)

We completely adapt the basic idea of the rigorous EPWT but use a certain predetermined bound θ_1 as follows. Forcing to continue the pathway in the same direction, we allow to take the direction that leads to the entry zero in $\tilde{\mathbf{p}}^L$ if the difference of function values satisfies

$$|f(\mathbf{p}^L(l)) - f(2\mathbf{p}^L(l) - \mathbf{p}^L(l-1))| \le \theta_1,$$

independently from the other absolute differences $|f(\mathbf{p}^L(l)) - f(\mathbf{p}^L(k))|$ with $k \in N(\mathbf{p}^L(l))$. If the index $2\mathbf{p}^L(l) + \mathbf{p}^L(l-1)$ is not an admissible neighbor index of $\mathbf{p}^L(l)$ then take the smallest index of components in $\tilde{\mathbf{q}}^{sh}$ for determining $\mathbf{p}^L(l+1)$ such that

$$|f(\mathbf{p}^L(l)) - f(\mathbf{p}^L(l) + \tilde{\mathbf{q}}^{sh}(\mu))| \le \theta_1 \tag{4.1}$$

is satisfied. If the above inequality (4.1) is not satisfied for all components of $\tilde{\mathbf{q}}^{sh}$, then we take as in the rigorous EPWT

$$\mathbf{p}^L(l+1) := \underset{k}{\operatorname{argmin}}\{|f(\mathbf{p}^L(l) - f(\mathbf{p}^L(k))|,\ k \in N(\mathbf{p}^L(l)),\ k \neq \mathbf{p}^L(\nu),\ \nu = 0,\dots,l\},$$

or equivalently, with the above notation, $\mathbf{p}^L(l+1) := \mathbf{p}^L(l) + \tilde{\mathbf{q}}^{sh}(\mu^*)$, where

$$\mu^* = \underset{\mu}{\operatorname{argmin}}\{|f(\mathbf{p}^L(l)) - f(\mathbf{p}^L(l) + \tilde{\mathbf{q}}^{sh}(\mu))|\}.$$

If the path vector $\mathbf{p}^L$ contains an interruption at $\mathbf{p}^L(l)$, then we may e.g. take as $\mathbf{p}^L(l+1)$ the smallest index of all free indices left. The corresponding path vector $\tilde{\mathbf{p}}^L$ can now be obtained using the procedure in Section 4.1.

Obviously, for $\theta_1 = 0$ the relaxed EPWT coincides with the rigorous EPWT. As we will see, the relaxed EPWT may essentially reduce the costs for path storing.

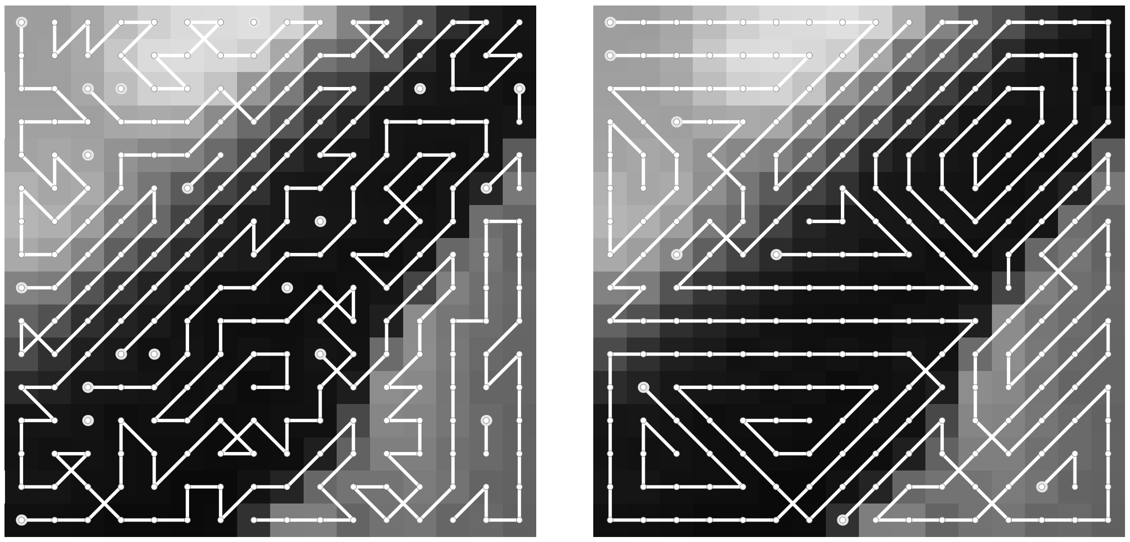

Figure 2. Representation of the first path with $\theta_1 = 0$ (left) and with $\theta_1 = 0.14$ (right)

Considering again the 4×4 matrix of function values in Example 3.1, we now get with a parameter $\theta_1 = 0.1$ the path $\mathbf{p}^4 = (0, 4, 8, 12, 13, 14, 15, 11, 7, 3, 2, 1, 5, 9, 10, 6)$ and $\tilde{\mathbf{p}}^4$ is the zero-vector of length 16.

Example 4.2 *We consider a* 16×16 *image (a part of the pepper image) and are interested in the path* $\mathbf{p}^L$ *(and* $\tilde{\mathbf{p}}^L$*, respectively) for the first level of EPWT. The image values are normalized to the range* $[0, 1)$*. In Figure 2 we compare the path vectors for* $\theta_1 = 0$ *(rigorous EPWT) and for* $\theta_1 = 0.14$ *(relaxed EPWT). Starting points of a new pathway are indicated by circles. They have been obtained here by taking the smallest remaining admissible index.*

Figure 2 shows that great differences between gray levels are avoided in the two paths. Interestingly, the path for $\theta_1 = 0$ *is not only more expensive but also contains more interruptions than the other path. The path* $\tilde{\mathbf{p}}^L$ *for* $\theta_1 = 0$ *consists of* 120 *zeros,* 57 *ones,* 42 *twos,* 18 *threes* 10 *fours,* 4 *fives and* 5 *times a six. The entropy of this path is* 2.08 *bit per pixel. For* $\theta_1 = 0.14$ *the obtained path* $\tilde{\mathbf{p}}^L$ *consists of* 241 *zeros,* 10 *ones,* 2 *twos, and* 3 *threes. This time the entropy of the path is* 0.39*. Observe that for larger images the entropy of the path can be even smaller, see Section 6.*

4.3 Further levels

In this subsection we describe the relaxed EPWT for the further levels and (at the same time) the cheap storage of corresponding path vectors. At the jth level the procedure is as follows.

(i) Put $\mathbf{p}^{L-j}(0) := 0$ and $\tilde{\mathbf{p}}^{L-j}(0) := 0$.

(ii) For $l \geq 0$ do the following.

If $\mathbf{p}^{L-j}(l) + 1$ is an admissible neighbor of $\mathbf{p}^{L-j}(l)$ and

$$|f^{L-j}(\mathbf{p}^{L-j}(l)) - f^{L-j}(\mathbf{p}^{L-j}(l) + 1)| \leq \theta_1, \tag{4.2}$$

then take $\mathbf{p}^{L-j}(l+1) := \mathbf{p}^{L-j}(l)+1$ and $\tilde{\mathbf{p}}^{L-j}(l+1) := 0$.

(iii) If $\mathbf{p}^{L-j}(l)+1$ is not an admissible neighbor of $\mathbf{p}^{L-j}(l)$ or does not satisfy (4.2), then we consider $\mathbf{p}^{L-j}(l)-1$ as the next candidate: If $\mathbf{p}^{L-j}(l)-1$ is an admissible neighbor of $\mathbf{p}^{L-j}(l)$ and

$$|f^{L-j}(\mathbf{p}^{L-j}(l)) - f^{L-j}(\mathbf{p}^{L-j}(l)-1)| \le \theta_1,$$

then take $\mathbf{p}^{L-j}(l+1) := \mathbf{p}^{L-j}(l)-1$ and

$$\tilde{\mathbf{p}}^{L-j}(l+1) := \begin{cases} 0 & \text{if } \mathbf{p}^{L-j}(l)+1 \quad \text{is not admissible,} \\ 1 & \text{if } \mathbf{p}^{L-j}(l)+1 \quad \text{is admissible.} \end{cases}$$

(iv) If (ii) and (iii) do not apply then do the following.

• Determine all indices in $\{0,\ldots,N_1N_2/2^j-1\}$ that are admissible neighbors of $\mathbf{p}^{L-j}(l)$, i.e., have not been already used in $\mathbf{p}^{L-j}$.

• Put these neighbor indices ordered by size into a vector $\mathbf{n}$, taking the smallest number first. If $\mathbf{p}^{L-j}(l)+1$ and/or $\mathbf{p}^{L-j}(l)-1$ are in $\mathbf{n}$ then remove them and let $i_0 \in \{0,1,2\}$ be the number of removed indices.

• Take the component $\mathbf{n}(\mu^*)$ of $\mathbf{n} = (\mathbf{n}(0), \mathbf{n}(1), \ldots)^T$ with smallest index μ^* such that

$$|f^{L-j}(\mathbf{p}^{L-j}(l)) - f^{L-j}(\mathbf{n}(\mu^*))| \le \theta_1 \tag{4.3}$$

is satisfied, and put $\mathbf{p}^{L-j}(l+1) := \mathbf{n}(\mu^*)$, $\tilde{\mathbf{p}}^{L-j}(l+1) := \mu^* + i_0$. If no component $\mathbf{n}(\mu)$ satisfies the inequality (4.3), then take

$$\mu^* := \underset{\mu}{\operatorname{argmin}}\{|f^{L-j}(\mathbf{p}^{L-j}(l)) - f^{L-j}(\mathbf{n}(\mu))|\}$$

and put $\mathbf{p}^{L-j}(l+1) := \mathbf{n}(\mu^*)$, $\tilde{\mathbf{p}}^{L-j}(l+1) := \mu^* + i_0$.

(v) In case of path interruption use a predetermined convention to find the starting index for the new pathway.

Obviously, for $\theta_1 = 0$, again the relaxed EPWT coincides with the rigorous EPWT, and the above procedure for determining $\tilde{\mathbf{p}}^{L-j}$ also applies for this case.

5 Adaptive Haar wavelet bases

How can the EPWT introduced in Section 2 be understood in terms of a multiresolution analysis? What are the main differences between the tensor product wavelet transform and the EPWT? In this section, we want to study these questions for the Haar wavelet transform.

Let $k \in \mathbb{N}$ be a given integer such that 2^{-k} is the finest resolution of a given function f on $Q := [0,1)^2$,

$$f = \sum_{\mu \in I_k} a_\mu \, \chi_{[0,1)^2}(2^k \cdot - \mu)$$

with $I_k = \{0, \ldots, 2^k - 1\} \times \{0, \ldots, 2^k - 1\}$. Thus, f is a piecewise constant function on the partition $\mathcal{D}_k(Q)$, where $\mathcal{D}_k(Q)$ denotes the set of dyadic squares with edge length 2^{-k} and measure 2^{-2k}.

We consider the space

$$V_{2k} = \mathrm{span}\,\{\phi_\nu^{2k},\, \nu \in J(I_k)\},$$

where each ϕ_ν^{2k} is the L^2-normalized characteristic function on one square in $\mathcal{D}_k(Q)$, i.e., $\phi_\nu^{2k} := 2^k \chi_{[0,1)^2}(2^k \cdot - \mu)$ with $\nu = J(\mu)$. Obviously, $\{\phi_\nu^{2k},\, \nu \in J(I_k)\}$ forms an orthonormal basis of V_{2k} and we have $f \in V_{2k}$, i.e., $f = \sum_{\nu \in J(I_k)} \tilde{a}_\nu\, \phi_\nu^{2k}$.

Let us fix the (normalized) Haar wavelet filter bank given by the analysis filters h, g with $h(0) = h(1) = 1/\sqrt{2}$ and $g(0) = -1/\sqrt{2}$, $g(1) = 1/\sqrt{2}$. The synthesis filters $\tilde{h}$ and $\tilde{g}$ coincide with h and g, respectively. As we have seen in Sections 3 and 4, applying the first level of the (rigorous or relaxed) EPWT with Haar filters, we first determine a path $\mathbf{p}^{2k}$ and then apply the one-dimensional Haar wavelet transform to the vector $(f(\mathbf{p}(\mu))_{\mu=0}^{4^k-1}$. This transform determines the new function spaces

$$\begin{aligned} V_{2k-1}(f) &:= \mathrm{span}\,\{\phi_\nu^{2k-1} := \frac{1}{\sqrt{2}}\left(\phi^{2k}_{\mathbf{p}^{2k}(2\nu)} + \phi^{2k}_{\mathbf{p}^{2k}(2\nu+1)}\right) :\, \nu = 0, \ldots, 2^{2k-1} - 1\}, \\ W_{2k-1}(f) &:= \mathrm{span}\,\{\phi_\nu^{2k-1} := \frac{1}{\sqrt{2}}\left(\phi^{2k}_{\mathbf{p}^{2k}(2\nu)} - \phi^{2k}_{\mathbf{p}^{2k}(2\nu+1)}\right) :\, \nu = 0, \ldots, 2^{2k-1} - 1\}. \end{aligned}$$

The space $V_{2k-1}(f)$ is generated by characteristic functions whose support consists of two (usually) neighboring squares of $\mathcal{D}_k(Q)$. We have $V_{2k-1}(f) \subset V_{2k}$ and $W_{2k-1}(f) \subset V_{2k}$, and moreover

$$V_{2k-1}(f) + W_{2k-1}(f) = V_{2k}, \qquad V_{2k-1}(f) \perp W_{2k-1}(f).$$

Clearly, there is a unique orthogonal decomposition of $f = f^{2k}$ into $f^{2k-1} \in V_{2k-1}(f)$ and $g^{2k-1} \in W_{2k-1}(f)$. We proceed in this way and determine the spaces according to the next path vectors $\mathbf{p}^l$, $l = 2k-1, \ldots, 1$ of length 2^l,

$$\begin{aligned} V_l(f) &:= \mathrm{span}\,\{\phi_\nu^{l} := \frac{1}{\sqrt{2}}\left(\phi^{l+1}_{\mathbf{p}^{l+1}(2\nu)} + \phi^{l+1}_{\mathbf{p}^{l+1}(2\nu+1)}\right) :\, \nu = 0, \ldots, 2^{l} - 1\}, \\ W_l(f) &:= \mathrm{span}\,\{\phi_\nu^{l} := \frac{1}{\sqrt{2}}\left(\phi^{l+1}_{\mathbf{p}^{l+1}(2\nu)} - \phi^{l+1}_{\mathbf{p}^{l+1}(2\nu+1)}\right) :\, \nu = 0, \ldots, 2^{l} - 1\} \end{aligned}$$

for $l = 2k-1, 2k-2, \ldots, 0$. Then the support of functions $\phi_\nu^l \in V_l$ consists of (usually) connected areas generated by 2^l squares from $\mathcal{D}_k(Q)$. Again we have $V_l(f) + W_l(f) = V_{l+1}$ and $V_l(f) \perp W_l(f)$ as before.

Example 5.1 *In $V_{2k-1}(f)$ and $V_{2k-2}(f)$, the basis functions can be characteristic functions with supports of the form illustrated in Figure 2 and Figure 3, respectively.*

Observe that the spaces $V_l(f)$ and $W_l(f)$, $l = 1, \ldots, 2k-1$, adaptively depend on the considered original function $f \in V_{2k}$. With the Haar filters given above we obtain the unique orthogonal decomposition of $f = f^{2k} \in V_{2k}$,

$$f^{2k} = f^0 + \sum_{l=0}^{2k-1} g^l,$$

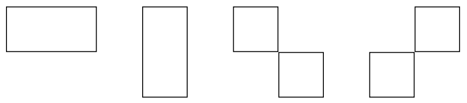

Figure 3: Shapes of supports of possible generating characteristic functions of $V_{2k-1}(f)$.

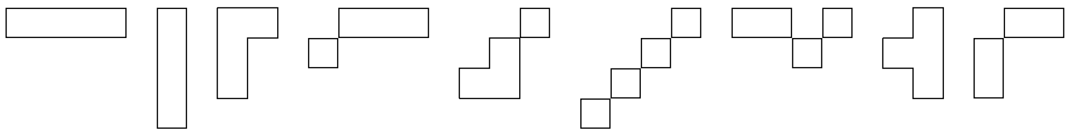

Figure 4: Some shapes of supports of possible generating characteristic functions of $V_{2k-2}(f)$.

where $f^0 \in V_0(f) = V_0$ is the average of f in Q, i.e., $f_0 = c\,\chi_Q$ with $c = \int_Q f(x)dx$, and $g^l \in W_l(f)$ for $l = 0, \ldots, 2k-1$.

In the second part of this section we want to consider the EPWT in matrix form and compare it with the usual tensor product wavelet transform.

Consider again the image $f = (f(i,j))_{i=0,j=0}^{N_1-1,N_2-1}$ as in Section 2, and the corresponding column vector $\mathbf{f}^L \in \mathbb{R}^{N_1 N_1}$,

$$\mathbf{f}^L = (f(0,0), f(1,0), \ldots, f(N_1-1,0), f(0,1), \ldots, f(N_1-1,N_2-1))^T$$

obtained by concatenating all columns of f. Let us introduce the transform matrix of the Haar wavelet transform of length $N \in 2\mathbb{N}$,

$$W_N := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & \ldots & 0 & 0 \\ & & & & & \ldots & & \\ 0 & 0 & 0 & 0 & 0 & \ldots & 1 & 1 \\ -1 & 1 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & \ldots & 0 & 0 \\ & & & & & \ldots & & \\ 0 & 0 & 0 & 0 & 0 & \ldots & -1 & 1 \end{pmatrix} \in \mathbb{R}^{N \times N}.$$

(For other periodic wavelet transforms the transform matrix can be determined similarly.) The tensor product Haar wavelet transform (one level) applied to f now reads

$$\begin{pmatrix} f_{00} & g_{01} \\ g_{10} & g_{11} \end{pmatrix} = W_{N_1}\, f\, W_{N_2}, \tag{5.1}$$

where N_1, N_2 indicate the dimensions of the transform matrices, and f_{00}, g_{01}, g_{10}, g_{11} are matrices in $\mathbb{R}^{N_1/2 \times N_2/2}$; f_{00} represents the low-pass part, g_{01}, g_{10}, g_{11} contain the wavelet coefficients.

In order to compare this transform with the EPWT, we want to rewrite (5.1) with the help of the column vector $\mathbf{f}^L$. The matrix multiplication $W_{N_1}\,\mathbf{f}$ in (5.1) corresponds to the (one-dimensional) Haar wavelet transform of columns of f and can be represented by

$$\mathbf{f}_c^L = (I_{N_2} \otimes W_{N_1})\, \mathbf{f}^L.$$

Here we have used the notation of a Kronecker product of matrices, i.e., $I_{N_2} \otimes W_{N_1}$ is a blockdiagonal matrix of dimension N_1N_2 with blocks W_{N_1}, and I_{N_2} denotes the identity matrix of size N_2. For the wavelet transform of the rows, performed by the multiplication with W_{N_2} in (5.1), we first need to permute $\mathbf{f}_c^L$ suitably. Let the permutation matrix $P_{N_1N_2}$ be given by

$$\begin{aligned} P_{N_1N_2}\, \mathbf{f}_c^L \;=\;& (f_c^L(0), f_c^L(N_1), \ldots, f_c^L((N_2-1)N_1), \\ & f_c^L(1), f_c^L(N_1+1), \ldots, f_c^L((N_2-1)N_1+1), f_c^L(2), \ldots, f_c^L(N_1N_2-1))^T. \end{aligned}$$

Then we can perform the multiplication with $I_{N_2} \otimes W_{N_2}$. One level of the tensor product Haar wavelet transform now reads

$$\mathbf{f}_{cr}^{L-1} = P_{N_1N_2}^T\, (I_{N_1} \otimes W_{N_2})\, P_{N_1N_2}\, (I_{N_2} \otimes W_{N_1})\, \mathbf{f}^L.$$

A reshaping of the column vector $\mathbf{f}_{cr}^{L-1}$ to a matrix gives the same result as in (5.1).

Let us compare this formula with the EPWT. Observe that one level of the tensor product wavelet transform is comparable to two levels of the EPWT.

At the first level we determine the path $\mathbf{p}^L = \mathbf{p}^L(f)$ of length N_1N_2 that yields a permutation matrix $P_{\mathbf{p}^L}$ of size N_1N_2 with entries

$$P_{\mathbf{p}^L}(i,j) = \begin{cases} 1 & j = \mathbf{p}^L(i), \\ 0 & \text{elsewhere.} \end{cases}$$

Then the one-dimensional Haar wavelet transform is applied to the permuted function vector,

$$\begin{pmatrix} \mathbf{f}^{L-1} \\ \mathbf{g}^{L-1} \end{pmatrix} = W_{N_1N_2}\, P_{\mathbf{p}^L}\, \mathbf{f}^L.$$

At the second level we determine the path $\mathbf{p}^{L-1}$ of length $N_1N_2/2$ yielding the permutation matrix $P_{\mathbf{p}^{L-1}}$. The transform is now only applied to the low-pass part $\mathbf{f}^{L-1}$,

$$\begin{pmatrix} \mathbf{f}^{L-2} \\ \mathbf{g}^{L-2} \end{pmatrix} = W_{N_1N_2/2}\, P_{\mathbf{p}^{L-1}}\, \mathbf{f}^{L-1}.$$

Hence, two levels of the EPWT read

$$\begin{pmatrix} \mathbf{f}^{L-2} \\ \mathbf{g}^{L-2} \\ \mathbf{g}^{L-1} \end{pmatrix} = \begin{pmatrix} W_{N_1N_2/2}\, P_{\mathbf{p}^{L-1}} & \\ & I_{N_1N_2/2} \end{pmatrix} W_{N_1N_2}\, P_{\mathbf{p}^L}\, \mathbf{f}^L. \tag{5.2}$$

Formula (5.2) nicely points out that the EPWT is a nonlinear transform, since the permutation matrices $P_{\mathbf{p}^L}$ and $P_{\mathbf{p}^{L-1}}$ strongly depend on the data in $\mathbf{f}^L$ and $\mathbf{f}^{L-1}$.

A comparison of (5.2) and the tensor product transform shows the essential differences between both approaches. While the "first part" of the tensor product wavelet transform, the wavelet transform of columns of f, $(I_{N_2} \otimes W_{N_1})\, \mathbf{f}^L$, can also be realized by the EPWT at the first level by using a suitable path vector $\mathbf{p}^L$, the important difference lies in the "second part". Applying the tensor product wavelet transform, the high-pass coefficients obtained in $\mathbf{f}_c^L$ are used a second time in the wavelet transform of rows. Hence, the wavelet coefficients in g_{00} result from the application of two (one-dimensional) high-pass filters, g_{10} is obtained from a high-pass filtering and a low-pass filtering etc. By contrast, with the EPWT we do never apply a wavelet transform to wavelet coefficients that have been obtained at previous levels. This observation clarifies that the tensor product wavelet transform can not be found as a special case of the EPWT with suitably chosen path vectors.

6 Numerical experiments

In this section we want to give some numerical examples to show the strong efficiency of the new algorithm for sparse representation of data.
As filters we will use

a) the Haar wavelet filters given by

$$(h(k))_{k=0}^{1} = (1/\sqrt{2}, 1/\sqrt{2}),\ (g(k))_{k=0}^{1} = (1/\sqrt{2}, -1/\sqrt{2})$$

and corresponding reconstruction filters $h^* = h$, $g^* = g$;

b) the periodic orthogonal wavelet transform with Daubechies filters

$$\begin{aligned}(h(k))_{k=0}^{3} &= \sqrt{2}\,((1+\sqrt{3})/8, (3+\sqrt{3})/8, (3-\sqrt{3})/8, (1-\sqrt{3})/8),\\ (g(k))_{k=-2}^{1} &= \sqrt{2}\,((1-\sqrt{3})/8, -(3-\sqrt{3})/8, (3+\sqrt{3})/8, -(1+\sqrt{3})/8),\end{aligned}$$

and corresponding reconstruction filters $h^* = h$, $g^* = g$;

c) the periodic biorthogonal wavelet transform based on 7-9 biorthogonal filters as given e.g. in [8] p. 279., Table 8.3.

We shall compare the EPWT for two-dimensional data compression with the usual discrete tensor product periodic wavelet transform and with the discrete curvelet transform. The compression is obtained by hard threshold of the wavelet coefficients. We want to compare the number of coefficients used for the sparse representation and the peak signal-to-noise ratio (PSNR), given by

$$\mathrm{PSNR} = 20 \log_2 \frac{\max |f(i,j)|}{\|\mathbf{f} - \tilde{\mathbf{f}}\|_2},$$

where $\mathbf{f}$ stands for the original data and $\tilde{\mathbf{f}}$ is the reconstructed sparse function.

In the first example we consider the 256×256 pepper image, i.e. 65536 coefficients (normalized to the range $[0, 1)$). We are interested in a lossy compression of this image by 1024 coefficients, i.e. by 1/64 of the original data. Here, we consider the relaxed EPWT with different bounds $\theta_1 \in \{0,\ 0.05,\ 0.1,\ 0.15\}$ and use the three wavelet transforms given in a), b), c). We compare these results to the tensor product wavelet transform and to curvelets. The results are summarized in Table 6.1. In Figure 5, we show our compression results for the (relaxed) EPWT with the 7-9 biorthogonal wavelet transform and compare it with the tensor product wavelet transform and the curvelet transform.

wavelet transform	θ_1	decomp. levels	number of coeff. after threshold	PSNR of reconstr. image	figure
tensor prod. Haar		8	1024	23.90	
tensor prod. Daub.		7	1024	24.81	
tensor prod. 7-9		5	1024	24.39	4(a)
tensor prod. Haar		8	4096	29.88	
tensor prod. Daub		7	4096	31.16	
tensor prod. 7-9		5	4096	30.57	
EPWT Haar	0.00	16	1024	30.44	
EPWT Haar	0.05	16	1024	30.55	
EPWT Haar	0.10	16	1024	29.65	
EPWT Haar	0.15	16	1024	28.59	
EPWT Daubechies	0.00	14	1024	31.20	
EPWT Daubechies	0.05	14	1024	31.43	
EPWT Daubechies	0.10	14	1024	30.71	
EPWT Daubechies	0.15	14	1024	29.73	
EPWT 7-9 filter	0.00	12	1024	30.63	4(c)
EPWT 7-9 filter	0.05	12	1024	31.03	4(d)
EPWT 7-9 filter	0.10	12	1024	30.36	4(e)
EPWT 7-9 filter	0.15	12	1024	29.55	4(f)

Table 6.1. Comparison of tensor product wavelet transform and easy path wavelet transform for the pepper image.

Applying the redundant discrete curvelet transform (see in the Matlab toolbox www.curvelet.org the function fdct_wrapping_demo_recon.m) one obtains 184985 curvelet coefficients for the pepper image. Using 1% threshold, i.e. 1850 coefficients, the reconstructed image achieves an PSNR of 22.48, see Figure 5(b).

Let us now consider the cost of adaptivity for the rigorous and the relaxed EPWT. With the method described in Section 4, we determine the path $\tilde{\mathbf{p}}^{16}$ of length $2^{16} = 65536$ for the pepper image. In particular, if the path is interrupted after $\mathbf{p}^{16}(l)$ (resp. $\tilde{\mathbf{p}}^{16}(l)$) we use the following procedure for finding a starting index $\mathbf{p}^{16}(l+1)$ for a new pathway.

We take the vector $\mathbf{n}$ of all admissible indices, i.e., all indices in $\{0, 1, \ldots, 65535\}$ that have not been used in $(\mathbf{p}^{16}(\nu))_{\nu=0}^{l}$ (ordered by size). Let K be the lenght of this vector and compute $k_0 := \lfloor K/7 \rfloor$. If $k_0 = 0$, i.e., $K < 7$ then let $\mathbf{n}_0 := \mathbf{n}$. If $k_0 > 0$ we consider the vector $\mathbf{n}_0 := (\mathbf{n}(1), \mathbf{n}(1+k_0), \mathbf{n}(1+2k_0), \ldots, \mathbf{n}(1+6k_0))$ of length 7. As a starting index $\mathbf{p}^{16}(l+1)$ for a new pathway we now choose $\mathbf{n}(1+\nu_0 k_0)$, where

$$\nu_0 := \underset{\nu}{\operatorname{argmin}} \, |f^{16}(\mathbf{p}^{16}(l)) - f^{16}(\mathbf{n}(1+\nu k_0))|, \quad \nu = 0, \ldots, 6,$$

and we take $\tilde{\mathbf{p}}^{16}(l+1) := \nu_0$.

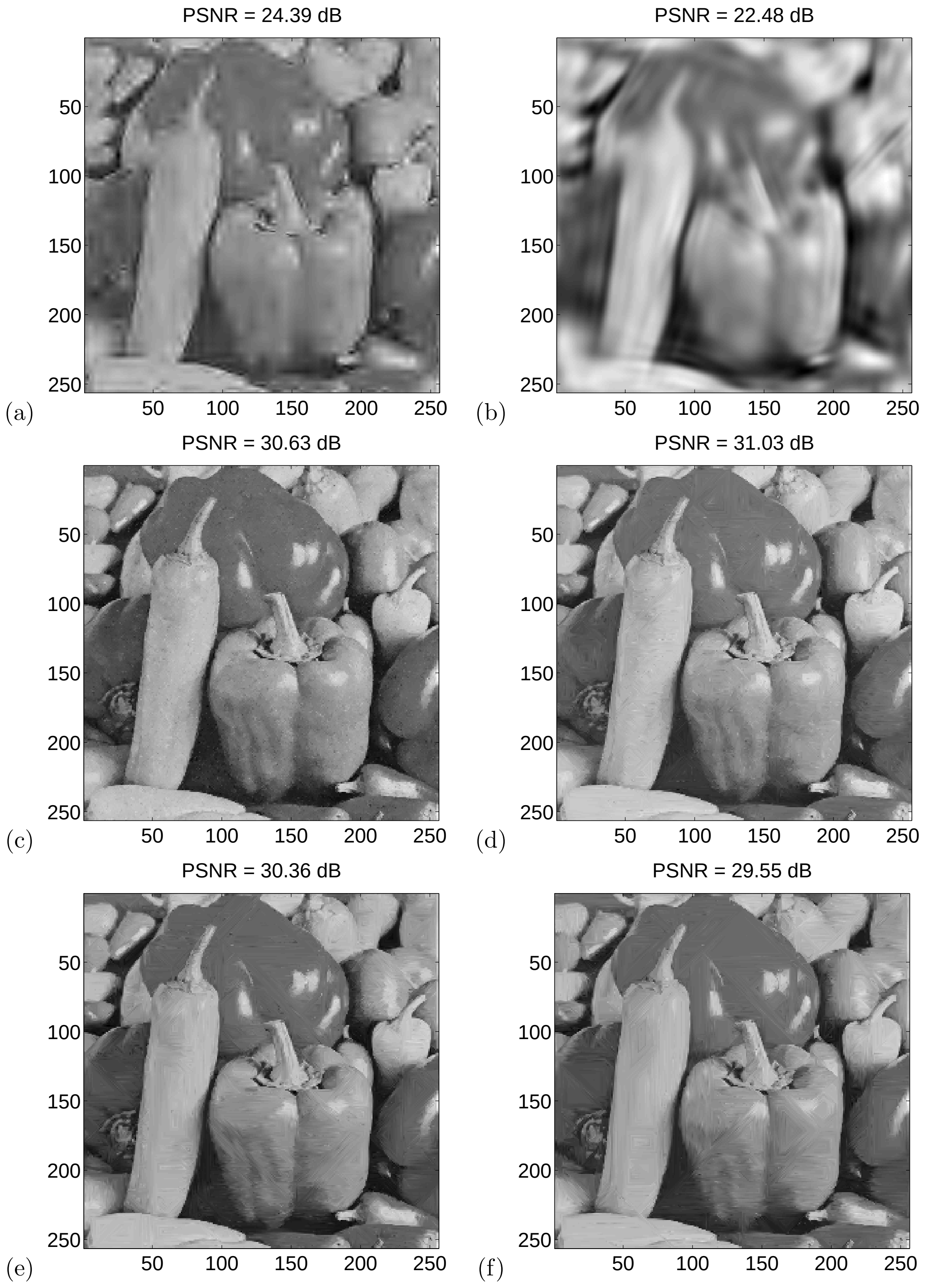

Figure 5. Image compression using (a) the tensor product wavelet transform with 7-9 filter, (b) the discrete curvelet transform, (c) the rigorous EPWT with 7-9 filter, and the relaxed EPWT with 7-9 filter and (d) parameter $\theta_1 = 0.05$, (e) $\theta_1 = 0.1$, (f) $\theta_1 = 0.15$, see Table 6.1.

Observe, that with this procedure one may obtain a starting point of a new pathway whose eight neighbors are all admissible such that also the number 7 can occur in $\tilde{\mathbf{p}}^{16}$. Table 6.2 shows the distribution of the numbers $0, \ldots, 7$ in the path $\tilde{\mathbf{p}}^{16}$ and its entropy for different bounds θ_1. Observe that the path vectors $\tilde{\mathbf{p}}^{16}$ resp. $\mathbf{p}^{16}$ do not depend on the used wavelet transform.

θ_1	0.0	0.05	0.1	0.15
occurrence of 0	27636	58445	62556	63753
occurrence of 1	13494	3117	1207	705
occurrence of 2	9250	1610	694	359
occurrence of 3	6889	1031	438	275
occurrence of 4	4107	590	260	172
occurrence of 5	2456	361	185	117
occurrence of 6	1685	382	195	155
occurrence of 7	19	0	1	0
entropy (bit per pixel)	2.30	0.73	0.37	0.24

Table 6.2. Distribution of numbers $0, \ldots, 7$ in the path $\tilde{\mathbf{p}}^{16}$ and its entropy for different bounds θ_1 for the pepper image.

The path vectors for the further levels $(\mathbf{p}^{15}, \ldots, \mathbf{p}^{1})$ have together the length $2^{16} - 1$ and storing of them usually doubles the cost given in Table 6.2. Using different wavelet transforms, we have found no essential differences in adaptivity costs.

For a rough comparison of the complete storing costs of the compressed pepper image using the tensor product transform and the relaxed EPWT we apply the following simplified scheme. We compute the cost for coding of the position of M non-zero wavelet coefficients by $-\frac{M}{N}\log_2\frac{M}{N} - \frac{(N-M)}{N}\log_2\frac{(N-M)}{N}$, where $N = 256 \times 256$ is the number of all coefficients in the image. Further, let b be the number of bits for encoding of one wavelet coefficient. For the tensor product wavelet transform, the storing costs are composed of the costs for encoding of the position of the non-zero coefficients and the costs for storage of these coefficients. For the EPWT, we have in addition the storage cost of the path. For $b = 8$ and $b = 16$, we have compared these costs for the Haar wavelet transform in Table 6.3, where we have used the above computed entropy of the path $\tilde{\mathbf{p}}^{16}$ as additional path cost. Obviously, the EPWT performs especially well for large b.

wavelet transform	θ_1	number of coeff. after threshold	b	storage cost (in bpp)
tensor prod. Haar	-	4096	8	0.62
EPWT Haar	0.10	1024	8	0.61
EPWT Haar	0.15	1024	8	0.48
tensor prod. Haar	-	4096	16	1.12
EPWT Haar	0.10	1024	16	0.74
EPWT Haar	0.15	1024	16	0.61

Table 6.3. Comparison of complete storage costs for the pepper image.

In the second example we are particularly interested in the ability of the EPWT to compress directional structures of a function. For that purpose we use the 128×128 image of a door lock with $2^{14} = 16384$ coefficients (normalized to the range $[0, 1)$), see Figure 6(a). As before, we compare the reconstructed images after a wavelet shrinkage procedure with 512 remaining coefficients (compression ratio 1/64) for tensor product periodic wavelet transform and the relaxed EPWT with different bounds θ_1. The results for the tensor product wavelet transform and for the EPWT are summarized in Table 6.4.

wavelet transform	θ_1	decomp. levels	number of coeff. after threshold	PSNR of recon. image	entropy of $\tilde{\mathbf{p}}^{14}$	fig.
tensor prod. Haar	-	7	512	22.16	-	
tensor prod Daub.	-	6	512	22.94	-	5(b)
tensor prod 7-9	-	4	512	22.49	-	
EPWT Haar	0.00	14	512	28.04	2.22	
EPWT Haar	0.05	14	512	28.37	1.11	
EPWT Haar	0.10	14	512	27.74	0.55	
EPWT Haar	0.15	14	512	26.85	0.32	
EPWT Daub.	0.00	12	512	28.63	2.22	5(c)
EPWT Daub.	0.05	12	512	29.23	1.11	5(d)
EPWT Daub.	0.10	12	512	28.67	0.55	5(e)
EPWT Daub.	0.15	12	512	27.65	0.32	5(f)
EPWT 7-9	0.00	10	512	28.35	2.22	
EPWT 7-9	0.05	10	512	28.99	1.11	
EPWT 7-9	0.10	10	512	28.38	0.55	
EPWT 7-9	0.15	10	512	27.69	0.32	

Table 6.4. Comparison of tensor product wavelet transform and easy path wavelet transform for the door lock image.

7 Conclusion

As we can see from the examples (see Table 6.1), compared with the traditional tensor product wavelet transform, the new EPWT needs less than *one fourth* of wavelet coefficients in order to achieve a similar PSNR. Here one needs to keep in mind, that with the two transforms not only the real wavelet coefficients themselves but also there positions in the wavelet vectors have to be stored.

For the EPWT we also have to store the path vectors, thus it is highly desirable to make these vectors as cheap as possible. The idea of *relaxed EPWT* is a first step in this direction. In our numerical experiments we have observed that the performance of the relaxed EPWT crucially depends on the choice of path vectors in all levels. Therefore one should think about other versions of relaxed EPWT with high performance and much cheaper path vectors. Instead of the simple method of *favorite directions* one may use more sophisticated extrapolation methods for obtaining the next component of the path.

The EPWT may be also interesting for further theoretical investigations regarding the nonlinear approximation of two-dimensional functions. In the future, we want to

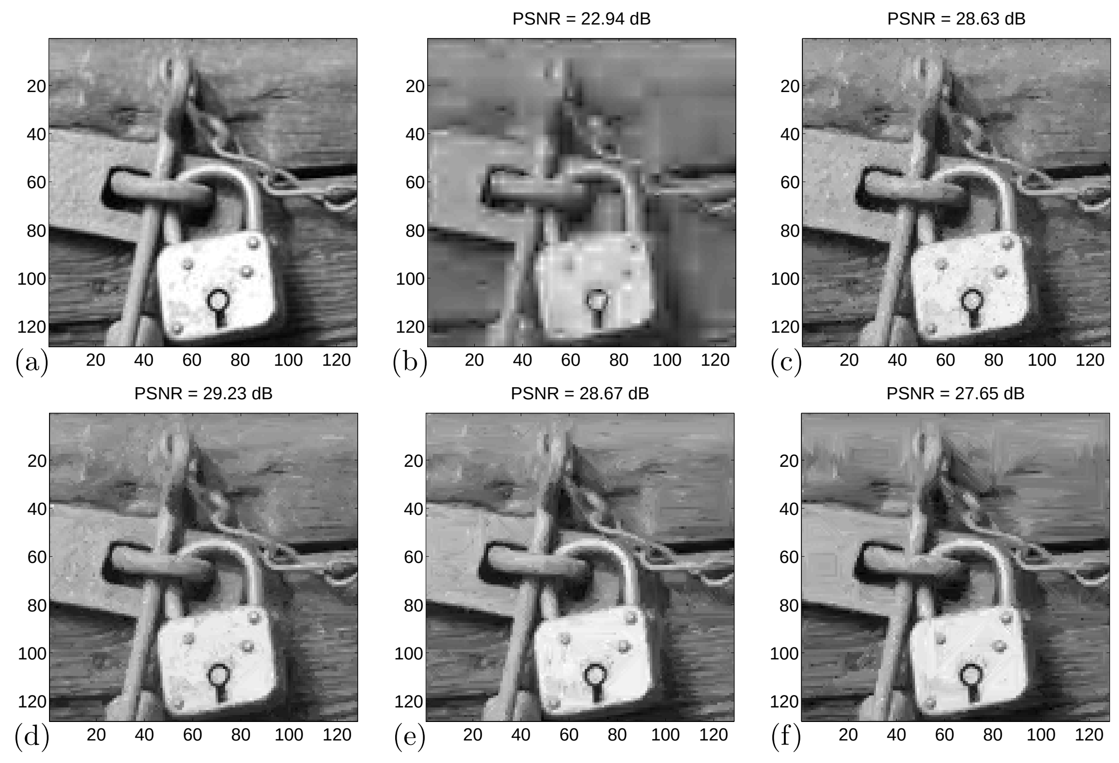

Figure 6. (a) Original image. Image compression using (b) the tensor product wavelet transform with Daubechies filter, (c) the rigorous EPWT with Daubechies filter, and the relaxed EPWT with Daubechies filter and (d) parameter $\theta_1 = 0.05$, (e) $\theta_1 = 0.1$ and (f) $\theta_1 = 0.15$, see Table 6.4.

study whether a characterization of certain function spaces in terms of decreasing order of wavelet coefficients is also possible with the EPWT. Such results are e.g. available for ENO-EA schemes (see [3]). In Section 4, we have used the characteristic function on the square as a "basic function" in order to make the spaces $V_j(f)$ and $W_j(f)$ be subspaces of $L^2([0,1)^2)$. For further analysis one may think also about other "basic functions" for the interpretation of the EPWT in the two-dimensional case (as e.g. interpolating functions with small support) in order to create an MRA with smoother function spaces $V_j(f)$ and $W_j(f)$.

Acknowledgement.

This research in this paper is supported by the project PL 170/13-1 and 436 RUM 113/31/0-1. This is grateful acknowledged. The author thanks J. Krommweh and D. Roşca for valuable comments for improving the presentation of the first version of the paper.
S. Dekel deserves special thanks for his ideas about a cheaper storing of path vectors, that I have been able to realize only partially. Further, I would like to thank the unknown referees for valuable comments that led to an essential revision of the paper, e.g. by stressing the differences between the EPWT and tensor product wavelet transform, by a more detailed study of adaptivity cost of EPWT that resulted in the relaxed EPWT, and by the improvement of the presentation and the English language.

References

[1] 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** (2002), 273–288.

[2] F. ARANDIGA, A. COHEN, R. DONAT, N. DYN, Interpolation and approximation of piecewise smooth functions, *SIAM J. Numer. Anal.* **43** (2005), 41–57.

[3] F. ARANDIGA, A. COHEN, R. DONAT, N. DYN, B. MATEI, Approximation of piecewise smooth functions and images by edge-adapted (ENO-EA) nonlinear multiresolution techniques, *Appl. Comput. Harmon. Anal.*, to appear.

[4] E.J. CANDÈS, D.L. DONOHO, New tight frames of curvelets and optimal representations of objects with piecewise singularities, *Comm. Pure Appl. Math.* **57** (2004), 219–266.

[5] E.J. CANDÈS, L. DEMANET, D.L. DONOHO, L. YING, Fast discrete curvelet transforms, *Multiscale Model. Simul.* **5** (2006), 861–899.

[6] R.L. CLAYPOOLE, G.M. DAVIS, W. SWELDENS, R.G. BARANIUK, Nonlinear wavelet transforms for image coding via lifting, *IEEE Trans. Image Process.* **12** (2003) 1449–1459.

[7] A. COHEN, B. MATEI, Compact representation of images by edge adapted multiscale transforms. in Proc. IEEE Int. Conf. on Image Proc. (ICIP), 2001, Thessaloniki, Greece, pp. 8-11.

[8] I. DAUBECHIES, Ten Lectures on Wavelets, SIAM, Philadelphia, 1992.

[9] S. DEKEL and D. LEVIATAN, Adaptive multivariate approximation using binary space partitions and geometric wavelets, *SIAM J. Numer. Anal.* **43** (2006), 707–732.

[10] L. DEMANET and L. YING, Wave atoms and sparsity of oscillatory patterns, *Appl. Comput. Harmon. Anal.* **23** (2007) 368–387.

[11] L. DEMARET, N. DYN, and A. ISKE, Image compression by linear splines over adaptive triangulations. *Signal Processing* **86** (2006), 1604–1616.

[12] R.A. DEVORE, Nonlinear approximation, Acta Numerica, A. Iserles (Ed.), Cambridge University Press, Cambridge, 1998, p. 51–150.

[13] M.N. DO and M. VETTERLI, The contourlet transform: An efficient directional multiresolution image representation, *IEEE Trans. Image Process.* **14** (2005) 2091–2106.

[14] D.L. DONOHO, Wedgelets: Nearly minimax estimation of edges, *Ann. Stat.* **27** (1999), 859–897.

[15] K. GUO and D. LABATE, Optimally sparse multidimensional representation using shearlets, *SIAM J. Math. Anal.* **39** (2007), 298–318.

[16] K. GUO, W.-Q. LIM, D. LABATE, G. WEISS, and E. WILSON, Wavelets with composite dilations, *Electr. res. Announc. of AMS* **10** (2004), 78–87.

[17] A. HARTEN, Discrete multiresolution analysis and generalized wavelets, *J. Applied Num. Math.* **12** (1993), 153–193.

[18] A. HARTEN, Multiresolution representation of data: general framework, *SIAM J. Numer. Anal.* **33** (1996), 1205–1256.

[19] L. JACQUES, J.-P. ANTOINE, Multiselective pyramidal decomposition of images: wavelets with adaptive angular selectivity, *Int. J. Wavelets Multiresolut. Inf. Process.* **5** (2007), 785–814.

[20] E. Le PENNEC, and S. MALLAT, Bandelet image approximation and compression, *Multiscale Model. Simul.* **4** (2005), 992–1039.

[21] S. MALLAT, A wavelet tour of signal processing, Academic Press, San Diego, 1999.

[22] S. MALLAT, Geometrical grouplets, *Appl. Comput. Harmon. Anal.* (2009), to appear.

[23] B. MATEI, Méthods multirésolution no-linéaires - Applications au traitement d'image, PhD dissertation, Université Paris VI, 2002.

[24] B. MATEI, Smoothness characterization and stability in nonlinear multiscale framework: theoretical results, *Asymptotic Anal.* **41** (2005), 277–309.

[25] D.D. PO and M.N. DO, Directional multiscale modeling of images using the contourlet transform, *IEEE Trans. Image Process.* **15** (2006), 1610–1620.

[26] R. SHUKLA, P.L. DRAGOTTI, M.N. DO, and M. VETTERLI, Rate-distortion optimized tree structured compression algorithms for piecewise smooth images, *IEEE Trans. Image Process.* **14** (2005), 343–359.

[27] M.B. WAKIN, J.K. ROMBERG, H. CHOI, and R.G. BARANIUK, Wavelet-domain approximation and compression of piecewise smooth images, *IEEE Trans. Image Process.* **15** (2006), 1071–108.