Adaptive Wavelet Bases for Image Compression

Ronald A. DeVore

Abstract. Wavelets are being suggested as a platform for various tasks in image processing. We shall consider several issues that arise in the application of wavelets to image compression. Our main emphasis will be on the choice of wavelet basis which are best suited for compression algorithms. Some results of numerical experiments will be presented. Finally, we suggest some alternatives to the usual wavelet bases to be used in the design of compression algorithms.

§1. Introduction

The application of wavelet decompositions to image processing brings forward several questions on just how wavelets should be implemented. We shall restrict our attention to the application of wavelets to image compression, and we shall discuss some questions ensuant to this application. Our main point of emphasis will be the question of which wavelet basis is most suitable for the design of compression algorithms. In partial answer to this question, we shall suggest two types of bases for further analysis.

This paper reflects recent research on wavelet decompositions and image processing that the author has undertaken with several collaborators. While their names do not appear as authors of the present paper, their influence is certainly present in this writeup. Foremost to mention is Brad Lucier who has collaborated with the author on most of his work in image processing. During the past year, a working seminar at the University of South Carolina uncovered many of the new ideas presented here.

Motivation for some of the questions on image processing discussed here came from data transmission problems considered by the Partnership in Computational Science (PICS) consortium. PICS is concerned with real time transmission and visualization of large data sets arising from the numerical modeling of groundwater flow. I am very thankful to Dick Ewing and Bob Sharpley for the many interesting discussions about this problem.

Wavelets, Images and Surface Fitting
P. J. Laurent, A. Le Méhauté, and L. L. Schumaker (eds.), pp. 197–219.

ISBN 1-56881-040-7.

§2. Wavelets and Multiresolution Analysis

Wavelet functions give an efficient method for the representation of functions and images. Several properties of wavelet decompositions suggest their superiority to more traditional decompositions such as the Fourier decompositions. These include the fast computation of wavelet coefficients and the fast numerical reconstruction of functions from these coefficients; the localness of the functions in the wavelet basis; the efficient encoding of smoothness and other properties of functions in the wavelet coefficients; and the discretization of many function norms in terms of wavelet coefficients. The latter property for example allows the simple solution of extremal problems, relevant to numerical applications, to be accomplished in terms of wavelet coefficients (see e.g. [10]).

Given a multivariate function g defined on $\mathbb{R}^d$, we use the notation

$$g_{j,k} := g(2^k \cdot - j), \quad \tilde{g}_{j,k} := 2^{kd/2} g_{j,k}, \quad j \in \mathbb{Z}^d, k \in \mathbb{Z},$$

to denote its shifted dilates and its L_2-normalized shifted dilates. For the purposes of this paper, we shall say that a univariate function ψ is a *wavelet* if its L_2-normalized shifted dilates $\tilde{\psi}_{j,k}$, $j,k \in \mathbb{Z}$, are a stable basis for $L_2(\mathbb{R})$. That is, each $f \in L_2(\mathbb{R})$ should have the unique representation

$$f = \sum_{j,k \in \mathbb{Z}} c_{j,k}(f) \tilde{\psi}_{j,k} \tag{2.1}$$

and

$$C_1 \sum_{j,k \in \mathbb{Z}} |c_{j,k}(f)|^2 \le \|f\|^2_{L_2(\mathbb{R})} \le C_2 \sum_{j,k \in \mathbb{Z}} |c_{j,k}(f)|^2$$

with absolute constants C_1, C_2. Here the $c_{j,k}$ are coefficient functionals defined on $L_2(\mathbb{R})$. If the functions $\tilde{\psi}_{j,k}$ form a complete orthonormal system for $L_2(\mathbb{R})$, then we say that ψ is an orthogonal wavelet.

The usual method for constructing wavelets is by multiresolution. In the univariate case, we begin with a function ϕ whose shifts $\phi(\cdot - j)$, $j \in \mathbb{Z}$, are $L_2(\mathbb{R})$-stable and we form the shift-invariant space $S := S(\phi)$ which is by definition the closure in $L_2(\mathbb{R})$ of all finite linear combinations $\sum_j c_j \phi(\cdot - j)$. Then each function $s \in S(\phi)$ has the representation

$$s = \sum_{j \in \mathbb{Z}} c_j(s) \phi(\cdot - j)$$

with

$$C_1 \sum_{j \in \mathbb{Z}} |c_j(s)|^2 \le \|s\|^2_{L_2(\mathbb{R})} \le C_2 \sum_{j \in \mathbb{Z}} |c_j(s)|^2.$$

From the space S, we form by dilation the spaces $S^k := \{S(2^k \cdot) : S \in S\}$. We assume that the spaces S^k are nested: $S^k \subset S^{k+1}$, $k \in \mathbb{Z}$. This imposes a severe restriction on ϕ since it says that $\phi \in S^1$, and therefore

$$\phi(x) = \sum_{j \in \mathbb{Z}} a_j \phi(2x - j), \quad x \in \mathbb{R}, \tag{2.2}$$

for some sequence $(a_j)_{j\in\mathbb{Z}}$ from $\ell_2(\mathbb{Z})$. This is called the *refinement equation* for ϕ. For applications in image processing it is desirable that the mask coefficients a_j, $j \in \mathbb{Z}$, are finite in number.

To construct wavelet functions ψ, we take a projector P which maps $L_2(\mathbb{R})$ onto $\mathcal{S}$ and we let $Q := I - P$ which is also a projector from $L_2(\mathbb{R})$ onto its range. We denote by W the range of $\mathcal{S}^1$ under Q. The space W is called the *wavelet space*. In the case that P is the orthogonal projector onto $\mathcal{S}$ which takes $f \in L_2(\mathbb{R})$ into its best L_2-approximation from $\mathcal{S}$, then W is the orthogonal complement of $\mathcal{S}$ in $\mathcal{S}^1$, i.e. $W = \mathcal{S}^1 \ominus \mathcal{S}$. In general W is the complement of $\mathcal{S}$ in $\mathcal{S}^1$ with respect to the projector P and we have

$$\mathcal{S}^1 = \mathcal{S} + W$$

in the sense that each $f \in \mathcal{S}^1$ has the unique decomposition

$$f = s + w, \quad s = Pf, \quad w = Qf.$$

We obtain wavelet functions by finding generators ψ for W. That is, we should have $W = \mathcal{S}(\psi)$. We shall restrict our discussions in this paper to the following special wavelets.

Haar wavelet. If we take $\phi := \chi_{[0,1]}$, then $\mathcal{S}(\phi)$ is the space of piecewise constant $L_2(\mathbb{R})$-functions with integer breakpoints. With P the orthogonal projector, the Haar function $\psi := \chi_{[0,1/2]} - \chi_{(1/2,1]}$ is a generator of the wavelet space W. The Haar function is an orthogonal wavelet.

Daubechies wavelets. Daubechies has found a class of orthogonal wavelets which have compact support and can have an arbitrarily high smoothness. These wavelets are found by discovering suitable refinement coefficients (2.2). For each r, Daubechies has shown the existence of refinement coefficients $a_j = a_j(r)$, such that the corresponding multiresolution generator $\phi = \phi_r$ has compact support and has orthogonal shifts. The number of nonzero coefficients in the refinement mask is $2r$. Then using the orthogonal projector P onto $\mathcal{S}$, the corresponding orthogonal wavelet $\psi = \mathcal{D}_r$ is also of compact support. The smoothness of the $\mathcal{D}_r$ increase with r (they have roughly $r/2$ order of smoothness in $C(\mathbb{R})$; see [3] for a detailed discussion of smoothness of the Daubechies and the biorthogonal wavelets that follow.

Biorthogonal wavelets. Cohen, Daubechies, and Feauveau [1] (see also Vetterli and Herley [10]) have given a general procedure to construct certain nonorthogonal wavelets with compact support and high smoothness. Their construction begins with two refineable functions ϕ and η which are in duality:

$$\int_{\mathbb{R}} \phi(x)\eta(x-j)\,dx = \delta(j), \quad j \in \mathbb{Z},$$

with δ the Kronecker delta. Under certain conditions, multiresolution generates from this pair a wavelet ψ whose dual functionals $c_{j,k}$ in the wavelet

decomposition (2.1) are given by

$$c_{j,k}(f) = \int_{\mathbb{R}} f\mu_{j,k}\,dx$$

for a suitable function μ. The wavelet function ψ and the dual function μ are obtained in a simple way from ϕ, η and the refinement masks (see (2.2)) for ϕ and η (see [3], Chapter 8, but note that our notation is different from that given there).

We shall only be interested in special cases of the biorthogonal wavelets discussed in Chapter 8 of [3]. They are indexed on two integer parameters k, ℓ. The integers k and ℓ relate to the number of coefficients in the refinement equations and indicate that the wavelet function ψ is a spline function of order k (degree $k-1$) with breakpoints at the half integers and that the dual wavelet μ has ℓ vanishing moments. A discussion of these wavelets and graphs of typical examples can be found in Chapter 8 of [3].

§3. Bivariate Wavelets

Images are bivariate. The usual method for representing bivariate functions with wavelets is as follows. From the univariate scaling function ϕ and the corresponding wavelet ψ, we form the set Ψ consisting of the three functions

$$\phi(x)\psi(y) \quad \psi(x)\phi(y), \quad \psi(x)\psi(y). \tag{3.1}$$

The functions

$$\tilde{\psi}_{j,k}, \quad \psi \in \Psi \tag{3.2}$$

then form a stable basis for $L_2(\mathbb{R}^2)$. This can be viewed as multiresolution analysis for the shift invariant spaces $S(\varphi)$ generated by the tensor product $\varphi(x,y) := \phi(x)\phi(y)$ of the univariate function ϕ with itself.

There is another natural bivariate wavelet basis which can be formed from the univariate wavelet ψ; it consists of the tensor products

$$\tilde{\psi}_{j,k}(x)\tilde{\psi}_{j',k'}(y), \quad j,j',k,k' \in \mathbb{Z}. \tag{3.3}$$

This is sometimes called the *standard basis* in wavelet literature. We shall call it the *hyperbolic basis* to make clear its distinction from the basis in (3.2).

One should note that in the hyperbolic basis, the dilation parameters 2^k and $2^{k'}$ are different in different coordinate directions. The result is that the function $\tilde{\psi}_{j,k}(x)\tilde{\psi}_{j',k'}(y)$ has support associated to the rectangle $[j2^{-k}, (j+1)2^{-k}] \times [j'2^{-k'}, (j'+1)2^{-k'}]$. The linear space H^n spanned by the functions $\tilde{\psi}_{j,k}(x)\tilde{\psi}_{j',k'}(y)$ with $2^k 2^{k'} \le 2^n$ is analogous to the Fourier space spanned by the exponentials $e_k(x)e_{k'}(y)$ with $|kk|' \le 2^n$ and $e_k(x) := e^{ikx}$. This latter region of points (k,k') in $\mathbb{R}^2$ is called a *hyperbolic cross.*

Approximation from the hyperbolic spaces H^n is quite different then approximation from S^n. While the approximation from the S^n is governed by the usual ways of measuring smoothness (derivatives and moduli of smoothness and Besov spaces), the hyperbolic approximation is governed by certain mixed derivatives and a new modulus of smoothness (see [7]).

§4. Wavelet Representation of Grey Scale Images

We shall discuss here only the case of grey scale images; color images can be handled by using a vector of coefficients but interesting questions arise as to the best choice of representing the three dimensional space of colors. It will also be convenient to restrict our attention to square images with the number of pixel values $N = 2^{2m}$. (Some comments are made later about efficient ways to handle images of different dimensions). Thus, for us, a digitized grey scale image $\mathcal{I}$ can be considered as an array of pixels values p_j, $j = (j_1, j_2)$, with each p_j an integer between 0 (black) and 255 (white) depicting the level of grey scale intensity. It is sometimes useful to view the digitized image as arising from an intensity function F defined on the unit square $\Omega := [0,1]^2$ with the pixel value p_j the average of F over the square with lower left corner $2^{-m}(j_1, j_2)$ and side length 2^{-m}.

In order to utilize wavelet decompositions in image processing, we need a way to go from the image $\mathcal{I}$ (i.e: the set of pixel values (p_j)) to a wavelet decomposition. We shall first discuss this for the usual wavelet bases (3.1). Image representation with hyperbolic wavelet bases is discussed in §6.

We fix a univariate generating function ϕ and its corresponding wavelet ψ. We recall that the usual wavelet basis generated by the shifted dilates of the three functions (3.1) can be obtained by multiresolution with respect to the tensor product φ of ϕ with itself: $\varphi(x,y) := \phi(x)\phi(y)$.

To obtain a wavelet decomposition for the image $\mathcal{I}$, we begin by assigning coefficients c_j, $j \in \mathbb{Z}^2$, and represent $\mathcal{I}$ at the fine dyadic level m as

$$\mathcal{I} \sim \sum_{j \in \mathbb{Z}^2} c_j \varphi_{j,m}(x,y). \tag{4.1}$$

We shall discuss just how to assign the (c_j) in a moment. We next use a change of basis to transform from the representation (4.1) to the wavelet representation

$$I \sim \sum_{j \in \mathbb{Z}^2} c_j \varphi_{j,m}(x,y) = \sum_{j \in \mathbb{Z}^2} d_{j,0}\phi_{j,0} + \sum_{k=0}^{m-1} \sum_{j \in \mathbb{Z}^2} \sum_{\psi \in \Psi} c_{j,k}\psi_{j,k} \tag{4.2}$$

with Ψ the set of the three functions in (3.1).

It is important to note that this change of basis is performed by the Fast Wavelet Transform (see e.g. [3] or [8]) and all computations are made in terms of the refinement coefficients. If, as we assume, these coefficients are finite in number, then the change of basis requires at most $O(N)$ operations with $N = 2^{2m}$ the number of pixels. Similarly, one easily changes from a wavelet representation (4.2) to the representation (4.1) at the fine level using rewrite rules again given by the refinement coefficients. Again these operations require only $O(N)$ computations.

4.1. Assignment of wavelet coefficients

We next discuss the assignment of the coefficients c_j. A naive assignment is to simply take $c_j := p_j$ wherever the p_j are defined and to take $c_j := 0$ otherwise. This works perfectly well for example for the Haar wavelets because the computation of the coefficients $c_{j,k}$ in (4.2) corresponding to functions which do not vanish identically on the unit square never involves any coefficients which have been set to zero.

When employing more general bivariate wavelets, there are functions $\psi_{j,k}$ appearing in (4.2) whose support intersects both Ω and its complement. The coefficient $c_{j,k}$ of such a function, when computed by the Fast Wavelet Transform, will involve pixel values which have been set to zero. The subsequent application of compression strategies such as thresholding and quantization (as discussed in the next section) will generally create artifacts in the compressed image due to the coefficients set to zero.

The appearance of artifacts is illustrated by Figure 1 where the left image of Lena was compressed using Daubechies wavelets $\mathcal{D}_4$. The black lines appearing near the boundary are artifacts caused by defining extended coeficients to be 0.

There is another explanation of the deficiency in simply setting coefficients to be 0 in the representation (4.1). Such an assignment has a debilitating effect on the smoothness of the function (4.1) representing $\mathcal{I}$. This can be seen by the fact that wavelet coefficients (in the representation (4.2)) corresponding to functions $\psi_{j,k}$ supported near the boundary will be artificially large. Subsequent application of compression strategies processes these large coefficients. For example, thresholding will keep some of these coefficients but will discard others in the compressed image.

To avoid the creation of artifacts, we need to assign coefficients c_j so that the function in (4.1) is smooth. This can be accomplished by a smooth extension of the pixel values outside of the unit square. We first need to understand how many coefficients c_j this involves. If $\psi_{j,0}$ is any wavelet function appearing at the coarsest level in the decomposition (4.1), its coefficient (computed by the Fast Wavelet Transform) involves all c_j for which the support of $\phi_{j,m}$ is contained in the support of $\psi_{j,0}$. Thus, this involves c_j corresponding to j that are $O(2^m)$ pixel rows from the boundary. Thus, there are $O(2^{2m})$ coefficients which would need to be assigned new values.

In order to avoid the computational expense of directly defining c_j for all of these j, Brad Lucier has suggested the following multiscale extension strategy. At the finest level, we define coefficient values c_j only for those j such that the support of $\phi_{j,m}$ is contained in the support of some $\psi_{j,m-1}$ which intersects Ω. This guarantees that the wavelet coefficients $c_{j,m-1}$ computed at the dyadic level $m-1$ will not be polluted by a poor definition of the c_j. This is also true for the coefficients $d_{j,m-1}$ of the functions $\phi_{j,m-1}$ which represent the projection (of the image) onto the space $\mathcal{S}^{m-1}$. However, we need to extend the coefficients $d_{j,m-1}$ in order to retain our fidelity at the next dyadic level. That is, we need to extend them to all values j such that

$d_{j,m-1}$ contributes to the computation of one of the coefficients of $\psi_{j,m-2}$ or $\phi_{j,m-2}$. We continue in this way, extending coefficients as needed.

The advantage of the multilevel extension is that at each level we need define only $O(2^k)$ new coefficients, hence $O(2^m)$ coefficients in total, a savings over the $O(2^{2m})$ coefficients which would need to be defined if the extension only occured at the finest level.

We have not discussed what form the extension operator should take. There are several possibilities such as some variant of the Whitney extension operators (see e.g. [8]) which preserve polynomials of a fixed degree. We have found that simply preserving constants performs well in compression algorithms.

The right image in Figure 1 shows Lena after compression when a multilevel extension of coefficients (based on extending constant functions exactly) has been used.

There are other approaches to handling boundary effects in wavelet based image processing. One of these is to modify the wavelet basis to be orthogonal on Ω. We refer the reader to the paper of Cohen, Daubechies, and Vial [2] or the book of Daubechies [3] for a more complete discussion of this subject.

Naive extension Multiscale extension

Fig. 1. 50-1 Compression with $\mathcal{D}_4$: Naive extension (L); Multiscale extension (R).

§5. Image Compression Algorithms

Some of the main issues confronted in the design of image compression algorithms are:

- choice of wavelet basis;
- choice of metric to measure compression error;
- compression strategy: linear versus nonlinear;
- smoothness of images;

- encoding of compressed wavelet coefficients;
- computational efficiency.

While the choice of wavelet basis is the main topic of the present paper, it will be convenient to postpone its discussion until some of the other issues have been vented.

5.1. Choice of compression metric

The criteria used to evaluate compressed images should depend on the intended application. For example, producing compressed images for video are quite a different matter from producing them for medical imagery. In the former, the human visual system would be the ultimate judge of quality, while retention of diagnostic fidelity is the main issue in the latter. In some applications, image compression is done as a preprocessor to other image processing tasks such as feature classification or object recognition. Compression can then have the effect of reducing the complexity of the postprocessing task and enhancing the features important for the postprocessing application. For example, some compression strategies can be viewed as removing noise from images.

To carry out a sensible mathematical analysis of compression, it is desireable to quantify in mathematical terms the compression criteria. One way of doing this is to attempt to find a metric defined on images to measure the error between the original and the compressed image. The metric chosen should model the goals of the compression algorithm.

Most metrics used for measuring compression error arise by considering images as functions defined on a square or rectangle Ω. Even here there is some ambiguity in how to interpret the image as a function. One possibility is to consider the image as the piecewise constant function given by its pixel representation. Another is to consider the image as the function which is its wavelet representation. Still another possibility is to think of the image as a function defined on a continuum and its pixel values as samples obtained by averaging over small squares (the function F introduced earlier). Fortunately, the decision of how to view the image as a function has no essential effect on the design and analysis of compression algorithms.

For the purposes of the following discussion of compression algorithms it will be convenient to think of an image $\mathcal{I}$ as its wavelet representation, i.e. as the function appearing on the right side of (4.1) (or equivalently (4.2)).

The most frequently used compression metric is the L_2 norm. The reason for this seems to be that it is easiest to describe optimal compression strategies with this choice since we can compute compression error exactly in terms of pixel values or wavelet coefficients (in the orthogonal wavelet case). On the other hand there is much evidence to indicate that this may not always be the best choice given the intended application of compression.

For example, to produce visually pleasing compressed images, it seems more reasonable to try to model the human visual system. Experiments of this type in the psychological community suggest that the L_1-norm models

better some aspect of the human visual system (see for example the discussion in [4]).

While other possibilities could be considered, we shall assume that our compression metric comes from one of the L_p-(quasi)norms, $0 < p \leq \infty$.

5.2. Compression strategies: linear and nonlinear

We suppose that we have chosen a compression metric that models our compression criteria; we assume that this is one of the L_p-metrics $0 < p \leq \infty$. We also assume that we have chosen the wavelet basis we shall use to represent our image. We obtain compression by retaining some of these terms in the wavelet decomposition (4.2) of the image and deleting others. But what exactly should be our strategy for the retention of terms.

For notational convenience, we shall assume that all terms corresponding to $\phi_{j,0}$ will be retained in the compressed image; in any case, there are very few of them.

We want to draw the distinction between two classes of compression algorithms: linear and nonlinear. In *linear algorithms*, one chooses a dyadic level $0 \leq K \leq m$ and retains in the compressed image $\tilde{I}$ all terms in (4.2) coresponding to $k < K$. We say that this is a linear algorithm because we can view the compressed image as an approximation to I from the linear space $\mathcal{S}^K = \mathcal{S}^K(\phi)$.

If the L_2-projector was used in the construction of the wavelet ψ, then the linear compression strategy is optimal with respect to compression in the L_2-metric in the sense that once we have decided to approximate the image I by functions from the linear space $\mathcal{S}^K$ then our strategy for choosing the approximation is optimal. Indeed, the portion of the wavelet decomposition (4.2) corresponding to the values of $k < K$ gives the L_2 projection of $\mathcal{I}$ on the space $\mathcal{S}^K$ and is therefore the best L_2 approximation from that space. It is also important to note that in linear methods the compression strategy (the terms that we retain) does not depend on the image.

In *nonlinear compression methods*, we allow the terms that we retain in the compressed image $\tilde{I}$ to depend on the image I. Thus, typically some but not necessarily all terms are chosen from each dyadic level. If n is the total number of terms to be retained, this can be thought of as approximation from the nonlinear manifold consisting of all functions g which are a linear combination of at most n of the functions appearing on the right side of (4.2).

There is one case where it is particularly simple to describe the optimal nonlinear compression strategy, namely, if ϕ and ψ have orthogonal shifts and the compression is done in the metric L_2. If we fix the number n of terms we wish to retain in the compressed image (which obviously is related to the rate of compression), then the optimal strategy is to retain the n terms in (4.2) for which

$$\|c_{j,k}\psi_{j,k}\|_{L_2} = |2^{-k}c_{j,k}|\|\psi\|_{L_2}$$

is largest (with any criteria used to break ties).

In numerical implementation, to avoid the time expensive sorting, the following modification of this strategy is implemented. We fix a positive number $\epsilon > 0$ which will measure the quality of the compressed image. We retain in the compressed image $\tilde{I}$ all terms of (4.2) for which

$$|2^{-k}c_{j,k}| > \epsilon. \tag{5.2.1}$$

The selection (5.2.1) is called *thresholding*. The smaller the value of ϵ, the more terms that are retained in the compressed image and the smaller the compression error.

A disadvantage of thresholding is that one does not know in advance the relationship between the choice of ϵ and the compression ratio. One possible way to circumvent this difficulty is to adaptively choose ϵ. One begins with a large value of ϵ so that no coefficients satisfy the threshold criteria (5.2.1). Given in hand any value of ϵ, we can replace ϵ by $\epsilon/2$ and add to our compressed image the additional terms that satisfy (5.2.1). We proceed until we have the desired quality in the compressed image or the desired compression ratio. The buildup of the compressed image in this way is progressive since we only add to the previous compressed image and do not have to redo any previous calculations. A discussion of progressive transmission for nonlinear based wavelet compression and their encoding can be found in Shapiro [12].

There is another important variant in the nonlinear strategy. Rather than using the entire coefficient $c_{j,k}$ in the terms that are retained, we replace it by an approximation $\bar{c}_{j,k}$ which satisfies

$$2^{-k}|c_{j,k} - \bar{c}_{j,k}| \leq \epsilon. \tag{5.2.2}$$

This is called *quantization* and is implemented numerically by retaining only the leading bits of the coefficient $c_{j,k}$. The effect of the L_2-quantization strategy (5.2.2) is to retain one less bit in the coefficients as we move from one dyadic level to a finer dyadic level.

If in place of the L_2-metric we use the L_p-metric, $p \neq 2$, or if we use non-orthogonal wavelets, then it is not so clear what should be the strategy for thresholding or quantization. The following strategy was derived in [5] and shown to be optimal in a sense that will be described in the next subsection. If we consider the compression problem as retaining the best n terms of the wavelet decomposition of the image so as to minimize L_p compression error then the strategy put forth in [5] is to choose the n terms for which

$$\|c_{j,k}\psi_{j,k}\|_p = |2^{-2k/p}c_{j,k}|\|\psi\|_{L_2} \tag{5.2.3}$$

are largest. Again this leads to corresponding thresholding and quantization strategies obtained from (5.2.1) and (5.2.2) by replacing 2^{-k} by $2^{-2k/p}$. For example, the quantization strategy when using the L_1 error metric is to choose two less bits as we move from one dyadic level to the next finer dyadic level.

5.3. Smoothness of images

A common way of making comparisons between compression algorithms is to test their performance on certain standardized images. This obviously has its limitations. Another possibility for comparing performance of compression algorithms is to try to classify images according to their compressibility. Compression is a form of approximation and we know from such questions in approximation theory that the rate of approximation of a function is connected with its smoothness. This leads us to consider the classification of images by their membership in smoothness spaces.

While there are many smoothness spaces, we shall limit our discussion to the Besov spaces. They are robust enough to measure all orders of smoothness in all L_p spaces. As we shall see, there is room for discussion as to whether these are appropriate spaces for measuring the smoothness of images.

For any $0 < s \leq \infty$ and $\alpha > 0$, we let $B_q^\alpha(L_s)$ denote the *Besov space* of smoothness order α in L_s with secondary parameter q. We do not give the definition of these spaces here (see e.g. [5] for a definition). One can heuristically view the space $B_q^\alpha(L_s)$ as consisting of the functions f from L_s whose α-th derivative is in L_s (the parameter q gives a more subtle delineation of smoothness). However, note that we allow α to be noninteger and s to be less than one.

We can classify the smoothness of an image by its membership in the Besov spaces. Since Besov space norms are applied to functions, we need to view the image as a function. One possibility is to consider the image as the underlying function F (introduced earlier) whose cell averages gave the pixel values. It is also reasonable to consider either the pixel representation (piecewise constant function) or the wavelet representation as being the function representing the image. We adopt the latter viewpoint in this section. One could argue that the wavelet function representation is always smooth (because it consists of a finite sum of smooth functions $\psi_{j,k}$) and that the image would be in all Besov spaces within the smoothness of the wavelet. But correctly viewed, the wavelet representation is just a partial sum of some larger series and this larger series has some intrinsic smoothness. For the purposes of the following discussion, one should adopt this latter viewpoint.

The value of measuring smoothness of images is that it gives us a way of evaluating the performance of compression algorithms. For example, if the dual functionals for the wavelet basis have r vanishing moments, then standard results in approximation theory (see e.g. [8]) say that the linear method of compression will approximate a bivariate function f to an order $O(n^{-\alpha/2})$ (with n the number of terms retained and $\alpha < r$) in the L_p-metric if and only if f is in the Besov space $B_\infty^\alpha(L_p)$. Hence, we can expect this performance by the compression algorithm only if the image has this smoothness.

A similar result holds for the nonlinear approximation which gave rise to the nonlinear compression algorithm (see [5]). In this case, there is a characterization of the bivariate functions f which have approximation order like $O(n^{-\alpha/2})$ for $\alpha < r$. The function f should be in the Besov space $B_\tau^\alpha(L_\tau)$

with $\tau = (\alpha/2 + 1/p)^{-1}$.

It is interesting to compare the linear and the nonlinear algorithms from this vantage point. To obtain an L_p-error $O(n^{-\alpha/2})$ (with n the number of terms retained), both require smoothness of order α but the linear algorithm measures this smoothness in the space L_p (the same space as the compression error is measured) while the nonlinear algorithm measures the smoothness in L_τ with $\tau = (\alpha/2 + 1/p)^{-1}$. Note that $\tau < p$ and it is more likely for a function to be in the Besov space for nonlinear approximation than in the one for linear approximation. Thus the nonlinear algorithm will compress more images to order $O(n^{-\alpha/2})$, i.e., require less smoothness for the image to be compressed with this accuracy.

For example, consider compression in the L_2 error metric. For compression error of order $O(n^{-\alpha/2})$, the image should have smoothness of order α as measured in L_2. Since images have edges and other discontinuities they cannot be in these Besov spaces for large values of α. For example, the characteristic function of a square is in the Besov space $B^\alpha_\infty(L_2)$ only if $\alpha \le 1/2$, and hence the linear compression algorithm will not give compression error $O(n^{-\alpha/2})$ for any $\alpha > 1/2$. However, in the case of the nonlinear algorithm, the smoothness is measured in L_τ. For example, this same characteristic function of a square is in the space $B^\alpha_\tau(L_\tau)$, $\tau = (\alpha/2+1/2)^{-1}$ provided $\alpha < 1/\tau$, or equivalently if $\alpha < 1$. In other words, images can have higher smoothness orders α when measured in the L_τ spaces and therefore can be compressed more efficiently by the nonlinear algorithms. A similar discussion applies to approximation in the metric L_p.

Choosing the coefficients for which (5.2.3) is largest provides a near optimal strategy for nonlinear approximation in the following sense. For functions in the unit ball of the Besov spaces $B^\alpha_\tau(L_\tau)$, this strategy provides an approximation error $O(n^{-\alpha/2})$. No other compression strategy based on approximation by a sum $\sum_{j,k,\psi\in\Lambda} b_{j,k}\psi_{j,k}$, with $|\Lambda| \le n$, can improve on this error. There are however the constants in the $O(n^{-\alpha/2})$ term about which we are not able to say anything. These constants are of importance in numerical comparisons and it may be that some other strategy could improve on (5.2.3) in these constants.

There is another sense in which the nonlinear strategy based on (5.2.3) is optimal. Given any stable nonlinear approximation scheme, not necessarily based on wavelets, it cannot improve on the error estimate $O(n^{-\alpha/2})$ of (5.2.3) (see [6]). Therefore, if we agree in advance that we want to compress all images from unit ball of the Besov space $B^\alpha_\tau(L_\tau)$) then no stable compression strategy can do better whether it is based on wavelets or some other ideas such as fractals or Fourier expansions.

The above strategy also gives a method for numerically computing the smoothness of an image. One computes the errors from the nonlinear approximation strategy (5.2.3), and graphs them on a log-log scale (error versus number of coefficients). This graph will look linear for a certain range of n (typically for $n \le 30,000$ in the case of 512×512 images). The slope of

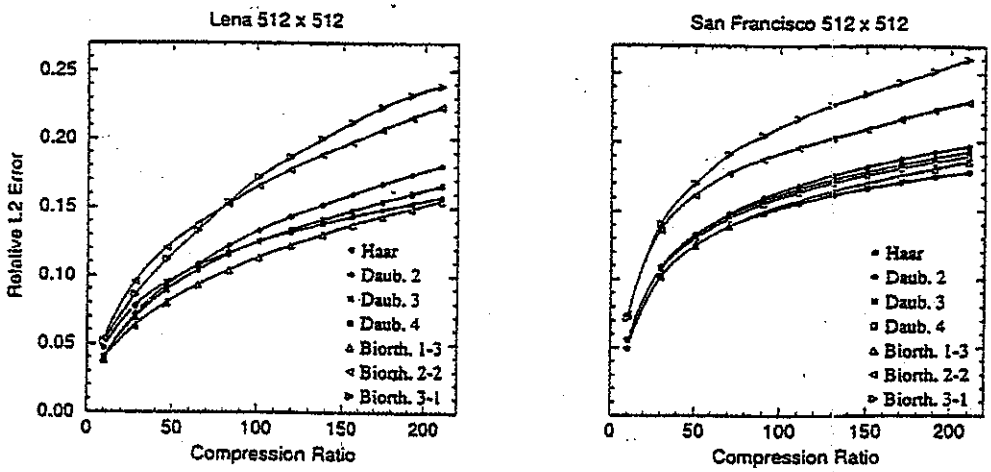

Fig. 2. L_2-error versus compression.

this line gives the largest α for which the image is in $B^\alpha_\tau(L_\tau)$. We refer the reader to [4] for a more detailed discussion of the numerical determination of smoothness of images.

Further discussion of linear and nonlinear compression algorithms can be found in [4]. They show the overall superiority of the nonlinear algorithms. For the remainder of this paper, we shall limit our discussion to nonlinear compression.

5.4. Choice of wavelet: fixed basis

There is the interesting question of whether any particular wavelet is preferable over another in compression algorithms. We have analyzed several wavelets for this purpose. In order to isolate the question of choice of wavelet basis, we have made two restrictions in the analysis that follows. First, in order to not have the compression strategy play a role in this analysis, we have chosen to measure the compression error in the L_2-metric. In this case, when using orthogonal wavelets, we know the optimal compression strategy for thresholding or quantization (we use only thresholding in the analysis that follows). Secondly, our analysis considers compression error (in the L_2-metric) versus the number of wavelet coefficients. A more accurate analysis of the performance of a compression algorithm would study the compression error versus the number of bits needed to encode the compressed image. However the latter would depend on the encoding of the wavelet coefficients (discussed in the last section). The best encoding could very well depend on the wavelet that is utilized in the compression algorithm. To eliminate this dependence on encoding we have simply counted coefficients.

We have tested several images with quite similar results. Figure 2 gives

a graphical presentation of our results for two particular images (Lena and San Francisco) that were compressed using Haar, Daubechies ($\mathcal{D}_2$, $\mathcal{D}_3$, $\mathcal{D}_4$) and biorthogonal ((1,3), (2,2), and (3,1)) bivariate bases (3.1). Here are some conclusions that can be drawn from these results.

There seems to be little evidence that smoother wavelets perform better in compression. For example, at high rates of compression, Haar performed better than the smoother Daubechies wavelets. At lower rates of compression, the smoother Daubechies wavelets $\mathcal{D}_r$, $r = 3, 4$, perform slightly worse than $\mathcal{D}_2$. The biorthogonal $(1, 3)$ wavelets which are piecewise constant perform better in all ranges than any of the other wavelet bases.

It seems that a more important issue is the playoff between the support of the coefficients functionals and the number of its vanishing moments. If the coefficient functional has large support then the dominant discontinuities in the image effect many wavelet coefficients. In low compression, the large coefficients from the dominant discontinuities are small in number compared to the number of terms retained and the deciding factor in the compression rate are the coefficients corresponding to smoother parts of the image. These coefficients are smaller because the coefficient functionals have many vanishing moments. For high compression however, the coefficients corresponding to the dominant discontinuities are the deciding factor.

Overall, the biorthogonal spline based wavelets performed better than the Daubechies wavelets. An explanation put forward for this by others is the symmetry in the biorthogonal wavelets. We should also reiterate the advantage of certain of the biorthogonal wavelets in that the computation of the wavelet decomposition can be done in integer arithmetic with shifts.

§6. Hyperbolic Bases

The use of hyperbolic bases in image compression brings forward new questions as to how compression algorithms should be implemented with these wavelets. As with the standard wavelet basis, one can divide the compression question into two parts corresponding to linear methods (approximation from linear spaces) and nonlinear methods. There is a theory (see [7]) for linear methods of compression based on hyperbolic wavelets that is similar in spirit to that given in §5.2. We do not want to discuss in detail these results but mention only a couple highlights. This theory introduces new moduli of smoothness based on certain mixed differences and defines new smoothness spaces in a similar way to the Besov spaces but using the new moduli of smoothness in place of the usual moduli of smoothness. It is shown then that these new smoothness spaces replace the roles of the Besov spaces when characterizing rates of approximation by the linear spaces H^n.

Unfortunately, there is not yet a corresponding theory for the nonlinear compression problem using hyperbolic wavelets except for special case when the error is measured in L_2. One of the reasons for this is that there is no characterization of the approximation spaces for nonlinear approximation using hyperbolic wavelets. This however does not prevent the experimentation

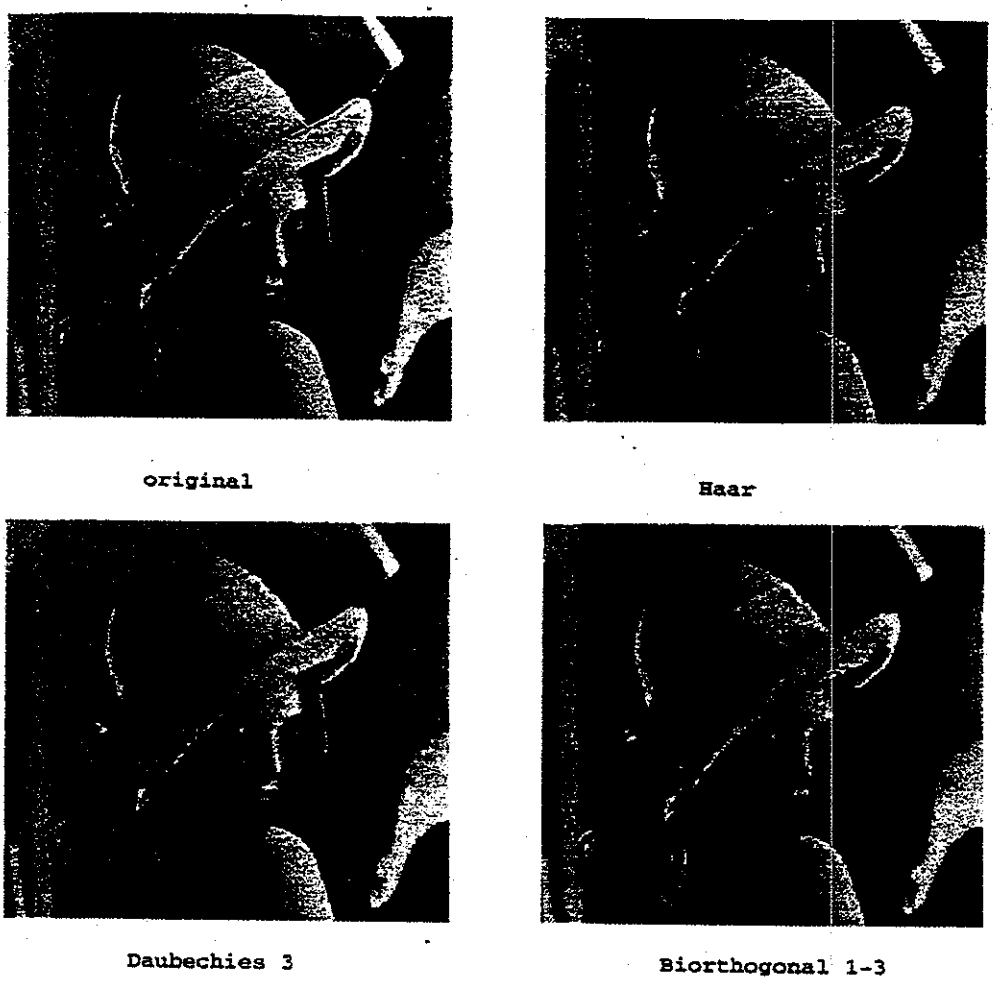

Fig. 3. 60-1 Compression.

with hyperbolic wavelet in image compression. The following is an initial report on some of our experience with hyperbolic wavelets.

We first discuss the representation of digitized images by hyperbolic wavelet bases. In order to simplify the following discussion by not having to deal with extensions near the boundary of the unit square (as discussed in §4), we shall limit our discussion to the case of the hyperbolic Haar basis. Using ideas similar to that presented in §4, we can derive expansions of images using other hyperbolic basis.

For this discussion, we let $\phi = \chi_{[0,1]}$ and let ψ be the univariate Haar function. We start with the representation (4.1) of the image $\mathcal{I}$ as

$$\mathcal{I} \sim \sum_{j \in \mathbb{Z}^2} c_j \varphi_{j,m}(x,y). \tag{6.1}$$

with $\varphi(x,y) := \phi(x)\phi(y)$ and c_j defined to be the given pixel values p_j in

case the support of $\varphi_{j,m}$ is contained in Ω and zero otherwise. The univariate functions $\phi_{s,m}$, $0 \le s < 2^m$, are a linear combination of $\phi_{0,0}$ and the univariate Haar functions ψ_{j_1,k_1}, $0 \le j_1 < 2^{k_1}$. Making this change of basis gives the hyperbolic Haar representation

$$\mathcal{I} \sim \sum_{j,k \in \mathbb{Z}^2} c_{j,k} \eta_{j,k}(x,y) \tag{6.2}$$

with $j = (j_1, j_2)$ and $k = (k_1, k_2)$ and

$$\eta_{j,k}(x,y) := \psi_{j_1,k_1}(x)\psi_{j_2,k_2}(y)$$

with the convention that $\psi_{0,0}$ is replaced by $\phi_{0,0}$ and $c_{j,k} = 0$ unless the support of $\eta_{j,k}$ is contained in Ω. As is discussed in the following section, the coefficients in the representation (6.2) can be found from the univariate Fast Wavelet Transform.

Given the hyperbolic wavelet representation (6.2) (or its analogue in the non-Haar case) of an image, we can implement compression strategies to threshold or quantize the coefficients $c_{j,k}$. We have implemented the analogue of the nonlinear L_2 compression strategy (5.2.1) for various hyperbolic wavelets. Figure 4 gives a graphical representation of the results of our analysis of two hyperbolic bases (Haar and Daubechies $\mathcal{D}_3$) versus the usual wavelet bases. We again display L_2-compression error versus the number of coefficients using only thresholding. The results are representative of our usual experience that for some images (such as San Francisco), the hyperbolic wavelet bases performed slightly better than the usual wavelet basis while on other (for example Lena) the opposite is true. It seems therefore that the choice whether to use the standard wavelet basis or the hyperbolic basis in compression depends on the image to be compressed.

§7. Adaptive Bases for Image Compression

The advantages of hyperbolic bases in compression led us to consider the possibility of adaptively chosing a wavelet basis to depend on the image being compressed. To discuss this approach, we suppose that ϕ is a univariate function satisfying the refinement equation. Suppose further that ϕ has orthogonal shifts and that ψ is a univariate orthogonal wavelet derived from ϕ by multiresolution. We have seen two bivariate orthogonal bases which can be derived from ϕ and ψ; namely the usual wavelet bivariate basis given by the shifted dilates of the three functions in (3.1) and secondly the hyperbolic basis given in (3.3). We can consider these as special cases of more general orthogonal bases constructed from ϕ and ψ.

To describe these general orthogonal bases, it is convenient to introduce the following notation. If $I = [j2^{-k}, (j+1)2^{-k}]$ is a dyadic interval, we let $\phi_I(x) := |I|^{-1/2}\phi(2^k x - j)$ denote the L_2 normalized shifted dilate of ϕ corresponding to the interval I. We use a similar notation for ψ. The interval I is associated to the support of ϕ_I and ψ_I. In fact, in the special case that

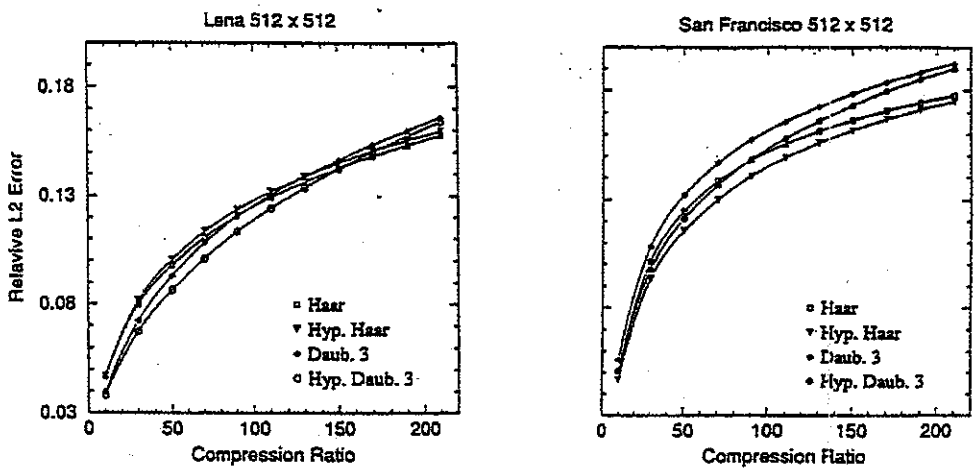

Fig. 4. L_2 error versus compression for hyperbolic and regular bases.

$\phi = \chi_{[0,1]}$ and ψ is the Haar function these are exactly the supports of ϕ_I and ψ_I.

We consider the totality of function

$$\phi_I(x)\phi_J(y), \quad \phi_I(x)\psi_J(y), \quad \psi_I(x)\phi_J(y), \quad \psi_I(x)\psi_J(y), \tag{7.1}$$

where I and J are dyadic intervals. The two wavelet bases already mentioned are only special cases of orthogonal bases for $L_2(\Omega)$, $\Omega := [0,1]^2$, that can be selected from the totality of functions in (7.1). We mention, in the special case of the Haar wavelets, some other orthogonal basis useful in partitioning (segmenting) the image. Starting with any finite partition $\mathcal{R}$ of the unit square into dyadic rectangles $R = I \times J$, then the functions $\phi_I(x)\phi_J(y)$ can be completed into an orthogonal bases for $L_2(\Omega)$ by adjoining a regular or hyperbolic Haar bases for each of these rectangles.

Given an image $\mathcal{I}$, we would like to select an orthogonal basis from the functions (7.1) which is particularly good for compression or some other image processing application. To keep the discussion simple, we shall only discuss here the case where $\phi = \chi_{[0,1]}$ and ψ is the Haar wavelet. Similar ideas can be applied to other pairs ϕ, ψ.

Suppose that $\mathcal{I}$ is a given image of dimension $2^m \times 2^m$. Let $\mathcal{D}_k$ denote the collection of dyadic subintervals of $[0,1]$ of sidelength 2^{-k} and let $\mathcal{D} = \cup_{k=0}^m \mathcal{D}_k$. We begin with the representation

$$\mathcal{I} \sim \sum_{I,J \in \mathcal{D}_m} p_{I,J}\phi_I(x)\phi_J(y) := f_{\mathcal{I}}, \tag{7.2}$$

where $p_{I,J}$ is the pixel value associated with the square $I \times J$. This is our usual representation (4.1) of the image. The function $f_{\mathcal{I}}$ is the piecewise constant function which takes the corresponding pixel values on the squares $I \times J$.

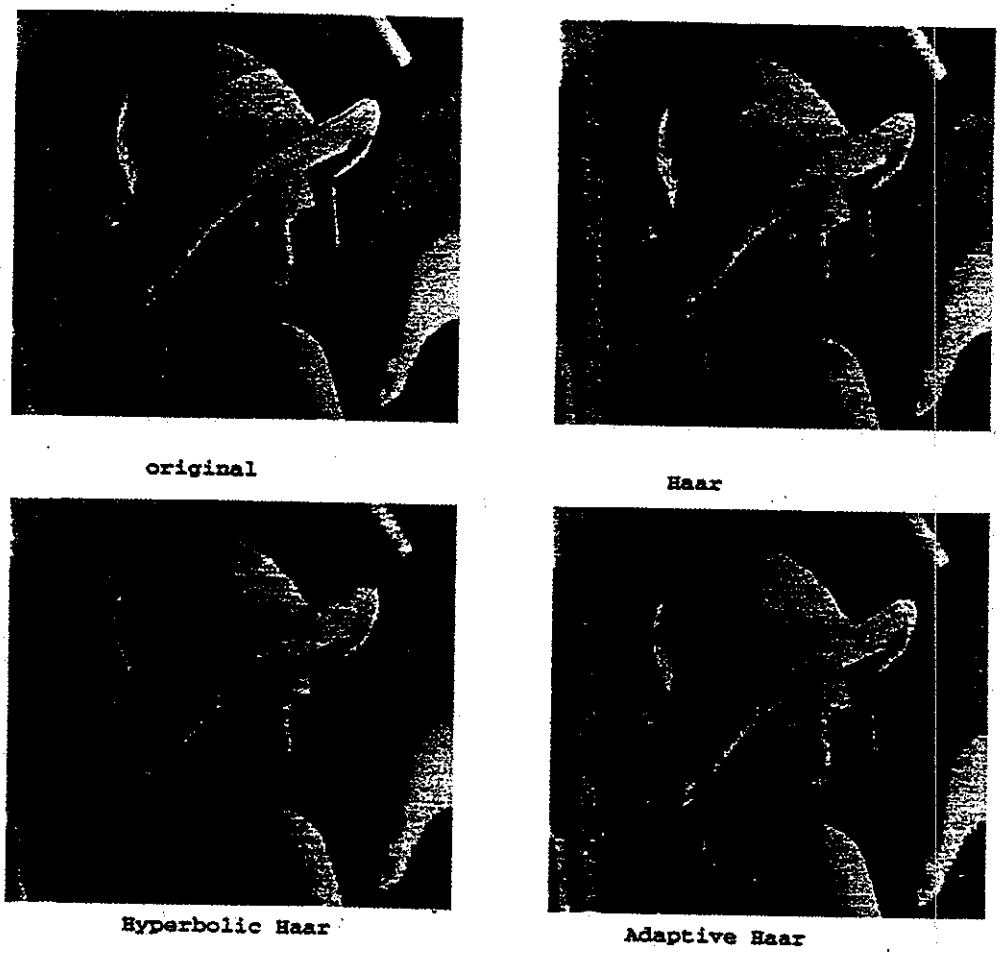

Fig. 5. 120-1 compression of Lena.

We first want to observe that we can efficiently compute the inner products of $f_{\mathcal{I}}$ with respect to all of the functions of (7.1) when $I, J \in \mathcal{D}$. Let r_j denote the univariate piecewise constant function

$$r_j(x) = \sum_{I \in \mathcal{D}_m} p_{I,J} \phi_I(x),$$

where $J = [j2^{-m}, (j+1)2^{-m}]$. Then r_j corresponds to the j-th row of pixel values. We can use the fast one dimensional Haar Transform to calculate the inner products

$$\int_{\mathbb{R}} r_j \phi_I \, dx, \quad \int_{\mathbb{R}} r_j \psi_I \, dx, \quad I \in \mathcal{D}. \tag{7.3}$$

Note that for each fixed j, the fast one dimensional Haar Transform while devised to calculate the integrals in (7.3) with respect to the functions ψ_I, $I \in \mathcal{D}$, computes in the process the integrals with respect to the ϕ_I as well. We replace the j-th row of pixel values by the integrals (7.3). We then repeat the application of the fast one dimensional Haar transform to the column vectors of the resulting matrix. This results in the computation of the inner products

$$\begin{aligned} &\int_{\mathbb{R}^2} f_{\mathcal{I}}\phi_I(x)\phi_J(y)\,dx\,dy, \quad \int_{\mathbb{R}^2} f_{\mathcal{I}}\phi_I(x)\psi_J(y)\,dx\,dy, \\ &\int_{\mathbb{R}^2} f_{\mathcal{I}}\psi_I(x)\phi_J(y)\,dx\,dy, \quad \int_{\mathbb{R}^2} f_{\mathcal{I}}\psi_I(x)\psi_J(y)\,dx\,dy. \end{aligned} \tag{7.4}$$

The number of computations needed to compute the inner products in (7.4) is thus $O(N)$ with N the number of pixel values and all computations can be done in integer arithmetic with shifts.

Suppose now that we wish to find a best orthogonal basis for the compression of a given image from among the functions (7.1). When the compression error is measured in the L_2 metric this is particularly simple to describe. We fix the number of coefficients n that we shall retain in the compressed image. The best orthogonal basis (which depends on the image) is the set of n functions taken from (7.1) which are mutually orthogonal and which have the largest sum of squares for its coefficients (7.4).

While it is a simple matter to describe the desired best basis, it is by no means simple to compute it. There are two difficulties. First of all it is not easy to decide which collections of n functions taken from (7.1) produce mutually orthogonal functions. Secondly the number of such bases is enormous and would challenge even the largest of computers when n is large. Thus, we desire numerically efficient algorithms which can select a "good" (in place of best) basis (for the purposes of compression).

A naive algorithm is the following. We order the coefficients (7.4) according to size (with ties broken in an arbitrary way). We select the function (7.1) corresponding to the largest coefficient and place it in our basis. We then consider the next largest coefficient. We check if its corresponding function is orthogonal to the function already in our basis. If so, we place it in our basis. Otherwise, we proceed to the next largest coefficient and continue in this manner. We stop when we have chosen n functions.

This naive algorithm does not give the best basis. In fact, it is not clear that one can prove anything quantitative about this basis selection. On the other hand, this basis performs well in image compression. For example, Figures 5 and 6 compares the compression of two standard images with the adaptive Haar basis with the fixed regular and hyperbolic Haar bases. A computation of compression error versus the number of coefficients retained given in Figure 7 shows about a 2-1 gain in compression with the adaptive basis selection versus fixed bases. That is, a given L_2-error can be achieved in the adaptive bases selection with about one half the coefficients needed with a fixed basis.

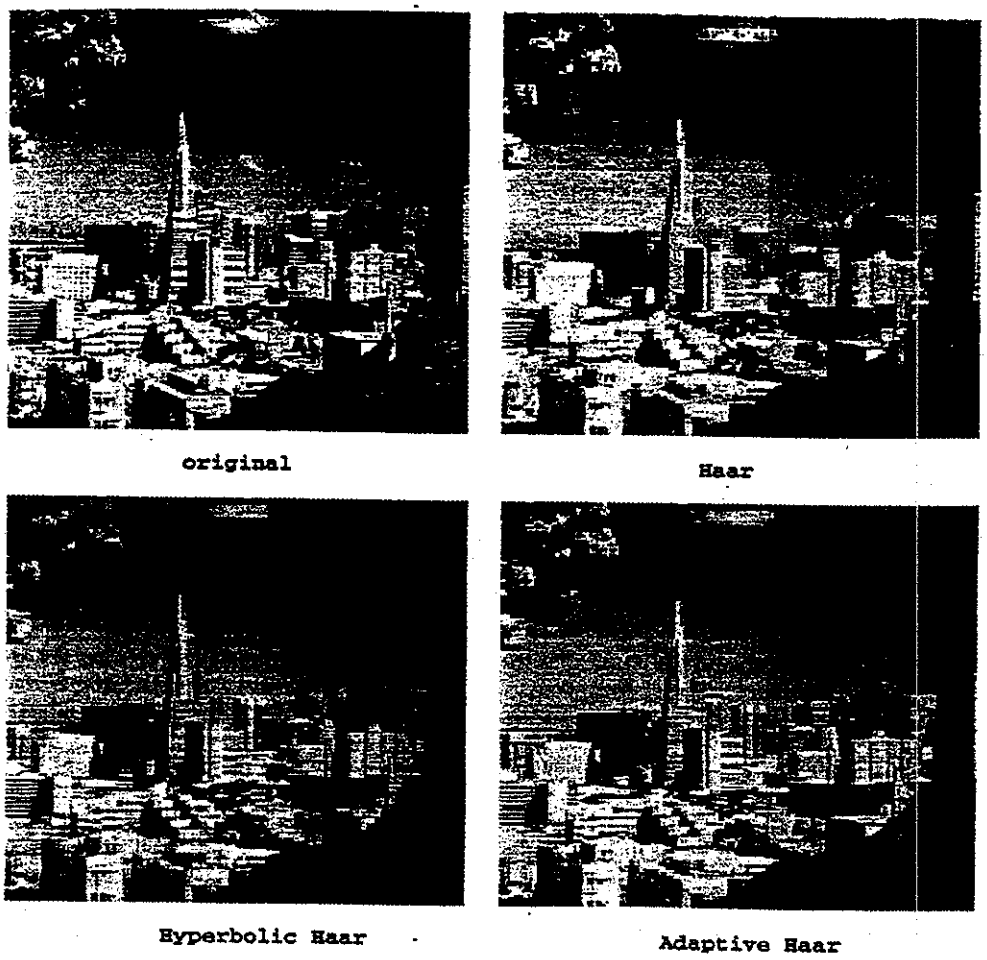

Fig. 6. 120-1 compression of San Francisco.

The naive algorithm is still not numerically efficient. Without modification, the checking of orthogonality which takes place at each step would require $O(j)$ operations with j the number of elements already in the basis. Hence, the total number of operations to construct this basis would be of order $O(N+n^2)$. Fortunately, there is a lot of structure in the basis functions (7.1) which allow for faster checks of orthogonality. For example, if an element $\phi_I\psi_J$ is a member of an orthogonal basis, then no function $\phi_{I'}\psi_J$, with $I \cap I' \neq \emptyset$ can also be a member. Similar ideas exclude many other functions. It is possible to utilize this structure to eliminate many of the remaining functions (7.1) whenever a basis element has been selected. A convenient data structure and fast algorithms for this purpose will be reported on elsewhere. These algorithms construct the same "good" basis with just $O(N + n \log N)$ operations.

We also note that to avoid sorting of coefficients, we could modify the algorithms to be based on thresholding (or quantization).

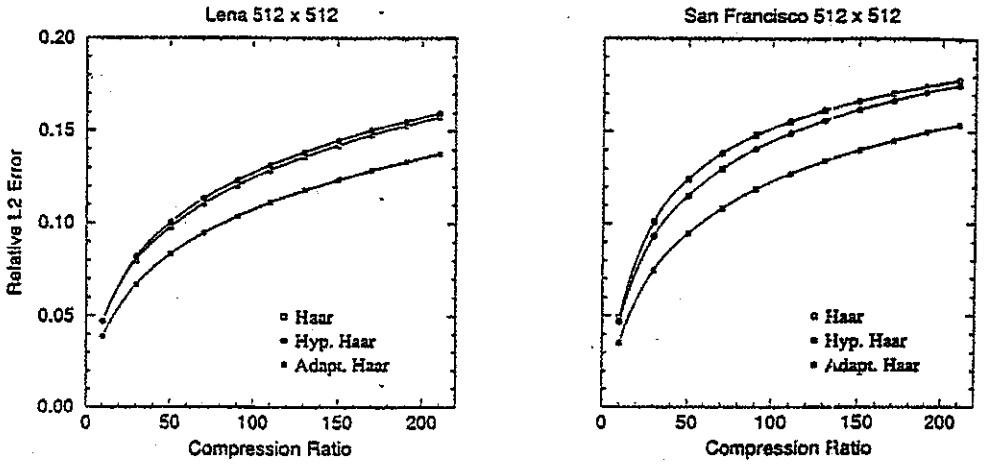

Fig. 7. L_2 error versus compression.

§8. Entropy Encoding of Coefficients

Thus far, we have based our analysis of the performance of a given compression algorithm on the number of coefficients that need to be retained to achieve a given error tolerance. The number of coefficients is not a completely fair assessment. What really counts is the number of bits that must be transmitted in order to reconstruct the compressed image.

If the computation of wavelet coefficients can be done in integer arithmetic with shifts, then each coefficient will have a maximum number of bits (for example, in the case of regular Haar, 10 bits per coefficient suffice for 512×512 images; see the discussion in [4]). If computations are done in real arithmetic, the coefficients will have to be truncated with a truncation error commensurate with the desired compression error (as was already discussed for quantization).

In the case of linear compression algorithms with a fixed wavelet basis, we need only transmit the coefficients in some fixed order, for example, from coarsest level to finest level. An entropy encoder is usually applied to the file of compressed coefficients before transmission. The entropy encoding will add to the compression by a small factor (less than 2 to 1).

In the case of nonlinear compression algorithms, it is not enough to send the coefficients since we must also identify their position. We wish to minimize the number of bits needed to identify this position. Our current algorithms for nonlinear compression methods accomplish this in the following way. We organize all coefficients of the compressed image (even those that are zero) in a file going from the finest to the coarses dyadic levels. Most entries in

this file are zero. We then apply an adaptive arithmetic encoder (in our case a modification of Q-coder developed at IBM) to further compress the file. Arithmetic encoders visualize the string of 0's and 1's to be transmitted as a real number and utilize the probability of occurence of the transmitted symbols (in our case 0 and 1) to devise a new arithmetic representation of this number. When the arithmetic encoder is applied at the back end of a Haar based nonlinear compression algorithm utilizing quantization, the result is roughly six bits per nonzero coefficient.

To be fair then, we should return to our comparison of linear versus nonlinear algorithms and analyze their performance based on the number of bits transmitted. This has been done for example with the Haar wavelets with the result that the nonlinear algorithm is still the clear winner (see the discussion in [4]).

However, all of this analysis is dependent on the entropy encoder that is utilized in conjunction with the wavelet compression algorithm. This adds another level of variability into the decision of which wavelets perform the best for compression. This is why in the present paper, we have chosen to simply count wavelet coefficients. A more correct treatment would analyze pairs consisting of a wavelet compression algorithms followed by an entropy encoder. A good encoder to be used in such a pair should take advantage of the data structure of the wavelet coefficients (multiscale and spatial correlation). There are some encoders of this type. For example, Shapiro [12] uses zero quadtrees to design an encoder for nonlinear wavelet based compression. We should also mention that run length encoding on the zeros followed by a Huffman encoder performed well with biorthogonal wavelets in the recent FBI competition (see the recent article in SIAM news).

New problems arise when applying the adaptive wavelet basis algorithms. In this case the total number of potential coefficients (corresponding to (7.4)) is $4N$ rather than N. Thus, if we again organize all compressed coefficients (zero and otherwise) corresponding to (7.4) into a file than the size of this file will be 4 times larger than in the fixed basis case. Even though the number of nonzero entries will be less than in the fixed basis case (reflecting the better performance of the adaptive basis selection), the encoded file may be larger.

Fortunately, the adaptive basis selection has more structure than is being utilized. As we have remarked earlier, when a certain basis element is chosen in the basis selection, it eliminates many other basis functions from being candidates for future addition to the basis. This means that if a certain coefficient in the matrix of compressed coefficients is nonzero, then many other coefficients are forced to be zero. We are currently investigating how to utilize this information to customize encoders for the adaptive basis compression algorithms.

Acknowledgements. This research was supported by an ONR contract N0014-91-51343 and a DOE contract DE-AC05-84OR21400 Martin Marietta subcontract 19X-SK965C

References

1. Cohen, A., I. Daubechies, and J.C. Feauveau, Biorthogonal bases of compactly supported wavelets,Comm. Pure and Applied Math., **XLV** (1992), 485–560.
2. Cohen, A., I. Daubechies, and P. Vial, Wavelets and fast transform on the interval, preprint, Bell Labs., 1992.
3. Daubechies, I., *Ten Lectures on Wavelets*, CBMS-NSF Regional Copnference Series in Applied Mathematics, vol. 61, SIAM, 1992.
4. DeVore R., B. Jawerth, and B. Lucier, Image compression through wavelet transform coding, IEEE Transaction on Information Theory **38** (1992), 719–747.
5. DeVore, R., B. Jawerth, and V. Popov, Compression of wavelet decompositions, Amer. Jour. Math., **114** (1992), 737–785.
6. DeVore, R., G. Kyriazis, D. Leviatan, and V.M. Tikhomirov, Wavelet compression and nonlinear n-widths, Advances Comp. Math., **1** (1993), 197–214.
7. DeVore, R., S. Konjagin, P. Petrushev, V. Temlyakov, Approximation with hyperbolic wavelets, preprint.
8. DeVore, R., and B. Lucier, Wavelets, Acta Numerica **1** (1992), 1–61.
9. DeVore, R., abd G.G. Lorentz, *Constructive Approximation*, Springer Grundlehren, vol. 303, New York, 1993.
10. DeVore, R., and B. Lucier, Fast wavelet techniques for near-optimal image processing, in *1992 IEEE Military Communications Conference*, IEEE, New York, N.Y., 1129–1135.
11. Vetterli, M. and C. Herley, Wavelets and filter banks: Theory and design, IEEE Trans. ASSP **40** (1992), 2207–2232.
12. J. M. Shapiro, An embedded hierarchial image coder using zerotrees of wavelet coefficients, in *Data Compression Conference*, J.A. Storer and M. Cohn (eds.), IEEE Computer Society Press, Los Alamitos, CA, 1993, 214–223.

Ronald A. DeVore
Department of Mathematics
University of South Carolina
Columbia, SC 29208
devore @math.scarolina.edu