A class of algorithms is introduced for the rapid numerical application of a class of linear operators to arbitrary vectors. Previously published schemes of this type utilize detailed analytical information about the operators being applied, and are specific to extremely narrow classes of matrices. In contrast, the methods presented here are based on the recently developed theory of wavelets, and are applicable to all Calderon-Zygmund operators. The algorithms of this paper require order $O(n)$ or $O(n \cdot log(n))$ operations to apply an $n \times n$ matrix to a vector (depending on the particular operator and the version of the algorithm being used), and our numerical experiments indicate that many previously intractable problems become manageable with the techniques presented here.

Fast Wavelet Transforms and Numerical Algorithms I

G. Beylkin[1,2], R. Coifman[2] , V. Rokhlin[3]

Research Report YALEU/DCS/RR-696
August 1989. Revised December 1989.

[1] Permanent address: Schlumberger-Doll Research, Ridgefield, CT 06877
[2] Research partially supported by ONR grant N00014-88-K0020
[3] Research partially supported by ONR grants N00014-89-J1527, N00014-86-K0310, and IBM grant P00038486.
Keywords: *Integral Operators, Wavelets, Calderon-Zygmund Operators, Numerical Methods*

FAST WAVELET TRANSFORMS AND NUMERICAL ALGORITHMS I.

G. Beylkin[1,2] , R. Coifman[2] , V. Rokhlin[3]

Yale University
New Haven, Connecticut 06520

I Introduction

The purpose of this paper is to introduce a class of numerical algorithms designed for rapid application of dense matrices (or integral operators) to vectors. As is well-known, applying directly a dense $N \times N-$ matrix to a vector requires roughly N^2 operations, and this simple fact is a cause of serious difficulties encountered in large-scale computations. For example, the main reason for the limited use of integral equations as a numerical tool in large-scale computations is that they normally lead to dense systems of linear algebraic equations, and the latter have to be solved, either directly or iteratively. Most iterative methods for the solution of systems of linear equations involve the application of the matrix of the system to a sequence of recursively generated vectors, which tends to be prohibitively expensive for large-scale problems. The situation is even worse if a direct solver for the linear system is used, since such solvers normally require $O(N^3)$ operations. As a result, in most areas of computational mathematics dense matrices are simply avoided whenever possible. For example, finite difference and finite element methods can be viewed as devices for reducing a partial differential equation to a sparse linear system. In this case, the cost of sparsity is the inherently high condition number of the resulting matrices.

For translation invariant operators, the problem of excessive cost of applying (or inverting) the dense matrices has been met by the Fast Fourier Transform (FFT) and related algorithms (fast convolution schemes, etc.). These methods use algebraic properties of a matrix to apply it to a vector in order $N \log(N)$ operations. Such schemes are exact in exact arithmetic, and are fragile in the sense that they depend on the exact algebraic properties of the operator for their applicability. A more recent group of fast algorithms [1, 2, 5, 9] uses explicit analytical properties of specific operators to rapidly apply them to arbitrary vectors. The algorithms in this group are approximate in exact arithmetic (though they are capable of producing any prescribed accuracy), do not require that the operators in question be translation invariant, and are considerably more adaptable than the algorithms based on the FFT and its variants.

In this paper, we introduce a radical generalization of the algorithms of [1, 2, 5, 9]. We describe a method for the fast numerical application to arbitrary vectors of a wide variety of operators. The method normally requires order $O(N)$ operations, and is directly applicable

[1]Permanent address: Schlumberger-Doll Research, Ridgefield, CT 06877

[2]Research partially supported by ONR grant N00014-88-K0020

[3]Research partially supported by ONR grants N00014-89-J1527, N00014-86-K0310 and IBM grant P00038436

to all Calderon-Zygmund and pseudo-differential operators. While each of the algorithms of [1, 2, 5, 9] addresses a particular operator and uses an analytical technique specifically tailored to it, we introduce several numerical tools applicable in all of these (and many other) situations. The algorithms presented here are meant to be a general tool similar to FFT. However, they do not require that the operator be translation invariant, and are approximate in exact arithmetic, though they achieve any prescribed finite accuracy. In addition, the techniques of this paper generalize to certain classes of multi-linear transformations (see Section 4.6 below).

We use a class of orthonormal "wavelet" bases generalizing the Haar functions and originally introduced by Stromberg [10] and Meyer [7]. The specific wavelet basis functions used in this paper were constructed by I. Daubechies [4], and are remarkably well adapted to numerical calculations. In these bases (for a given accuracy) integral operators satisfying certain analytical estimates have a band-diagonal form, and can be applied to arbitrary functions in a 'fast' manner. In particular, Dirichlet and Neumann boundary value problems for certain elliptic partial differential equations can be solved in order N calculations, where N is the number of nodes in the discretization of the boundary of the region. Other applications include an $O(N \log(N))$ algorithm for the evaluation of Legendre series, and similar schemes (comparable in speed to FFT in the same dimensions) for other special function expansions. In general, the scheme of this paper can be viewed as a method for the conversion (whenever regularity permits) of dense matrices to a sparse form.

Once the sparse form of the matrix is obtained, applying it to an arbitrary vector is an order $O(N)$ procedure, while the construction of the sparse form in general requires $O(N^2)$ operations. On the other hand, if the structure of the singularities of the matrix is known *a priori* (as for Green's functions of elliptic operators or for Calderon-Zygmund operators) the compression of the operator to a banded form is an order $O(N)$ procedure. The non-zero entries of the resulting compressed matrix mimic the structure of the singularities of the original kernel.

Effectively, this paper provides two schemes for the numerical evaluation of integral operators. The first is a straightforward realization ("standard form") of the matrix of the operator in the wavelet basis. This scheme is an order $N \log(N)$ procedure (even for such simple operators as multiplication by a function). While this straightforward realization of the matrix is a useful numerical tool in itself, its range of applicability is significantly extended by the second scheme, which we describe in this paper in more detail. This realization ("non-standard form") leads to an order N scheme. The estimates for the latter follow from the more subtle analysis of the proof of the "$T(1)$ theorem" of David and Journé (see [3]). We also present two numerical examples showing that our algorithms can be useful for certain operators which are outside the class for which we provide proofs. The paper is organized as follows. In Section II we use the well-known Haar basis to describe a simplified version of the algorithm. In Section III we summarize the relevant facts from the theory of wavelets. Section IV contains an analysis of a class of integral operators for which we obtain an order N algorithm, and a description of a version of the algorithm for bilinear operators. Section V contains a detailed description and a complexity analysis of the scheme. Finally, in Section VI we present several numerical

applications.

Generalizations to higher dimensions and numerical operator calculus containing $O(N \log(N))$ implementations of pseudodifferential operators and their inverses will appear in a sequel to this paper.

II The algorithm in the Haar system

The Haar functions $h_{j,k}$ with integer indices j and k are defined by [1]

$$h_{j,k}(x) = \begin{cases} 2^{-j/2} & \text{for } 2^j(k-1) < x < 2^j(k-1/2) \\ -2^{-j/2} & \text{for } 2^j(k-1/2) \le x < 2^j k \\ 0 & \text{elsewhere.} \end{cases} \tag{2.1}$$

Clearly, the Haar function $h_{j,k}(x)$ is supported in the dyadic interval $I_{j,k}$

$$I_{j,k} = [2^j(k-1), 2^j k]. \tag{2.2}$$

We will use the notation $h_{j,k}(x) = h_{I_{j,k}}(x) = h_I(x) = 2^{-j/2}h(2^{-j}x - k + 1)$, where $h(x) = h_{0,1}(x)$. We index the Haar functions by dyadic intervals $I_{j,k}$ and observe that the system $h_{I_{j,k}}(x)$ forms an orthonormal basis of $L^2(\mathbf{R})$ (see, for example, [8]).

We also introduce the normalized characteristic function $\chi_{I_{j,k}}(x)$

$$\chi_{I_{j,k}}(x) = \begin{cases} |I_{j,k}|^{-1/2} & \text{for } x \in I_{j,k} \\ 0 & \text{elsewhere,} \end{cases} \tag{2.3}$$

where $|I_{j,k}|$ denotes the length of $I_{j,k}$, and will use the notation $\chi_{j,k} = \chi_{I_{j,k}}$.

Given a function $f \in L^2(\mathbf{R})$ and an interval $I \subset \mathbf{R}$, we define its Haar coefficient d_I of f

$$d_I = \int_{-\infty}^{+\infty} f(x) h_I(x) dx, \tag{2.4}$$

and " average" s_I of f on I as

$$s_I = \int_{-\infty}^{+\infty} f(x) \chi_I(x) dx, \tag{2.5}$$

and observe that

[1] We define the basis so that the dyadic scale with the index j is finer than the scale with index $j+1$. This choice of indexing is convenient for numerical applications.

$$d_I = (s_{I'} - s_{I''})\frac{1}{\sqrt{2}}, \tag{2.6}$$

where I' and I'' are the left and the right halves of I.

To obtain a numerical method for calculating the Haar coefficients of a function we proceed as follows. Suppose we are given $N = 2^n$ "samples" of a function, which can for simplicity be thought of as values of scaled averages

$$s_k^0 = 2^{n/2} \int_{2^{-n}(k-1)}^{2^{-n}k} f(x)dx, \tag{2.7}$$

of f on intervals of length 2^{-n}. We then get the Haar coefficients for the intervals of length 2^{-n+1} via (2.6), and obtain the coefficients

$$d_k^1 = \frac{1}{\sqrt{2}}(s_{2k-1}^0 - s_{2k}^0). \tag{2.8}$$

We also compute the 'averages'

$$s_k^1 = \frac{1}{\sqrt{2}}(s_{2k-1}^0 + s_{2k}^0) \tag{2.9}$$

on the intervals of length 2^{-n+1}. Repeating this procedure, we obtain the Haar coefficients

$$d_k^{j+1} = \frac{1}{\sqrt{2}}(s_{2k-1}^j - s_{2k}^j) \tag{2.10}$$

and averages

$$s_k^{j+1} = \frac{1}{\sqrt{2}}(s_{2k-1}^j + s_{2k}^j) \tag{2.11}$$

for $j = 0, \ldots, n-1$ and $k = 1, \ldots, 2^{n-j-1}$. This is illustrated by the pyramid scheme

$$\begin{array}{ccccccccc} \{s_k^0\} & \longrightarrow & \{s_k^1\} & \longrightarrow & \{s_k^2\} & \longrightarrow & \{s_k^3\} & \cdots \\ & \searrow & & \searrow & & \searrow & & \\ & & \{d_k^1\} & & \{d_k^2\} & & \{d_k^3\} & \cdots \end{array} \tag{2.12}$$

It is easy to see that evaluating the whole set of coefficients d_I, s_I in (2.12) requires $2(N-1)$ additions and $2N$ multiplications.

In two dimensions, there are two natural ways to construct Haar systems. The first is simply the tensor product $h_{I\times J} = h_I \otimes h_J$, so that each basis function $h_{I\times J}$ is supported on the rectangle $I \times J$. The second basis is defined by associating three basis functions: $h_I(x)h_{I'}(y)$, $h_I(x)\chi_{I'}(y)$, and $\chi_I(x)h_{I'}(y)$ to each square $I \times I'$, where I and I' are two dyadic intervals of the same length.

We consider an integral operator

$$T(f)(x) = \int K(x,y)f(y)dy, \tag{2.13}$$

and expand its kernel (formally) as a function of two variables in the two-dimensional Haar series

$$K(x,y) = \sum_{I,I'} \alpha_{II'} h_I(x)h_{I'}(y) + \sum_{I,I'} \beta_{II'} h_I(x)\chi_{I'}(y) + \sum_{I,I'} \gamma_{II'}\chi_I(x)h_{I'}(y), \tag{2.14}$$

where the sum extends over all dyadic squares $I \times I'$ with $|I| = |I'|$, and where

$$\alpha_{II'} = \int\int K(x,y)h_I(x)h_{I'}(y)dxdy, \tag{2.15}$$

$$\beta_{II'} = \int\int K(x,y)h_I(x)\chi_{I'}(y)dxdy, \tag{2.16}$$

and

$$\gamma_{II'} = \int\int K(x,y)\chi_I(x)h_{I'}(y)dxdy. \tag{2.17}$$

When $I = I_{j,k}, I' = I_{j,k'}$ (see (2.2)), we will also use the notation

$$\alpha^j_{k,k'} = \alpha_{I_{j,k},I_{j,k'}}, \tag{2.18}$$

$$\beta^j_{k,k'} = \beta_{I_{j,k},I_{j,k'}}, \tag{2.19}$$

$$\gamma^j_{k,k'} = \gamma_{I_{j,k},I_{j,k'}}, \tag{2.20}$$

defining the matrices $\alpha^j = \{\alpha^j_{i,l}\}$, $\beta^j = \{\beta^j_{i,l}\}$, $\gamma^j = \{\gamma^j_{i,l}\}$, with $i,l = 1,2\ldots,2^{n-j}$. Substituting (2.14) into (2.13), we obtain

$$T(f)(x) = \sum_I h_I(x) \sum_{I'} \alpha_{II'}\, d_{I'} + \sum_I h_I(x) \sum_{I'} \beta_{II'}\, s_{I'} + \sum_I \chi_I(x) \sum_{I'} \gamma_{II'}\, d_{I'} \tag{2.21}$$

(recall that in each of the sums in (2.21) I and I' always have the same length).

To discretize (2.21), we define projection operators

$$P_j f = \sum_{|I|=2^{j-n}} \langle f, \chi_I\rangle \chi_I, \qquad j = 0,\ldots,n \tag{2.22}$$

and approximate T by

$$T \sim T_0 = P_0 T P_0, \tag{2.23}$$

where P_0 is the projection operator on the finest scale. An alternative derivation of (2.21) consists of expanding T_0 in a 'telescopic' series

$$T_0 = P_0TP_0 = \sum_{j=1}^{n}(P_{j-1}TP_{j-1} - P_jTP_j) + P_nTP_n$$

$$= \sum_{j=1}^{n}[(P_{j-1} - P_j)T(P_{j-1} - P_j) + (P_{j-1} - P_j)TP_j + P_jT(P_{j-1} - P_j)] + P_nTP_n. \tag{2.24}$$

Defining the operators Q_j with $j = 1, 2, \ldots, n$, by the formula

$$Q_j = P_{j-1} - P_j, \tag{2.25}$$

we can rewrite (2.24) in the form

$$T_0 = \sum_{j=1}^{n}(Q_jTQ_j + Q_jTP_j + P_jTQ_j) + P_nTP_n. \tag{2.26}$$

The latter can be viewed as a decomposition of the operator T into a sum of contributions from different scales. Comparing (2.14) and (2.26), we observe that while the term P_nTP_n (or its equivalent) is absent in (2.14), it appears in (2.26) to compensate for the finite number of scales.

Observation 2.1. Clearly, expression (2.21) can be viewed as a scheme for the numerical application of the operator T to arbitrary functions. To be more specific, given a function f, we start with discretizing it into samples $s_k^0, k = 1, 2, \ldots, N$, which are then converted into a vector $\tilde{f} \in R^{2N-2}$ consisting of all coefficients s_k^j, d_k^j and ordered as follows

$$\tilde{f} = (d_1^1, d_2^1, \ldots, d_{N/2}^1, s_1^1, s_2^1, \ldots, s_{N/2}^1, d_1^2, d_2^2, \ldots, d_{N/4}^2, s_1^2, s_2^2, \ldots, s_{N/4}^2, \ldots, d_1^n, s_1^n). \tag{2.27}$$

Then, we construct the matrices $\alpha^j, \beta^j, \gamma^j$ for $j = 1, 2, \ldots, n$ (see (2.15) - (2.20) and Observation 3.2) corresponding to the operator T, and evaluate the vectors $\hat{s}^j = \{\hat{s}_k^j\}$, $\hat{d}^j = \{\hat{d}_k^j\}$ via the formulae

$$\hat{d}^j = \alpha^j(d^j) + \beta^j(s^j) \tag{2.28}$$

$$\hat{s}^j = \gamma^j(d^j), \tag{2.29}$$

where $d^j = \{d_k^j\}, s^j = \{s_k^j\}, k = 1, 2, \ldots, 2^{n-j}$, with $j = 1, \ldots, n$. Finally, we define an approximation T_0^N to T_0 by the formula

$$T_0^N(f)(x) = \sum_{j=1}^{n}\sum_{k=1}^{2^{n-j}}(\hat{d}_k^j\ h_{j,k}(x) + \hat{s}_k^j\ \chi_{j,k}(x)). \tag{2.30}$$

Clearly, $T_0^N(f)$ is a restriction of the operator T_0 in (2.23) on a finite-dimensional subspace of $L^2(\mathbf{R})$. A rapid procedure for the numerical evaluation of the operator T_0^N is described (in a more general situation) is Section III below.

It is convenient to organize the matrices $\alpha^j, \beta^j, \gamma^j$ with $j = 1, 2 \ldots, n$ into a single matrix, depicted in Figure 1, and for reasons that will become clear in Section IV, the matrix in Figure 1 will be referred to as the non-standard form of the operator T, while (2.14) will be referred to as the "non-standard" representation of T (note that the (2.14) is *not* the matrix realization of the operator T_0 in the Haar basis).

III Wavelets with vanishing moments and associated quadratures

3.1. Wavelets with vanishing moments. Though the Haar system leads to simple algorithms, it is not very useful in actual calculations, since the decay of $\alpha_{II'}, \beta_{II'}, \gamma_{II'}$ away from diagonal is not sufficiently fast (see below). To have a faster decay, it is necessary to use a basis in which the elements have several vanishing moments. In our algorithms, we use the orthonormal bases of compactly supported wavelets constructed by I. Daubechies [4] following the work of Y. Meyer [7] and S. Mallat [6]. We now describe these orthonormal bases.

Consider functions ψ and φ (corresponding to h and χ in Section II), which satisfy the following relations:

$$\varphi(x) = \sum_{k=0}^{2M-1} h_{k+1} \varphi(2x - k), \tag{3.1}$$

$$\psi(x) = \sum_{k=0}^{2M-1} g_{k+1} \varphi(2x - k), \tag{3.2}$$

where

$$g_k = (-1)^{k-1} h_{2M-k+1}, \qquad k = 1, \ldots, 2M \tag{3.3}$$

and

$$\int \varphi(x) dx = 1. \tag{3.4}$$

The coefficients $\{h_k\}_{k=1}^{k=2M}$ are chosen so that the functions

$$\psi_{j,k}(x) = 2^{-j/2} \psi(2^{-j} x - k + 1), \tag{3.5}$$

where j and k are integers, form an orthonormal basis and, in addition, the function ψ has M vanishing moments

$$\int \psi(x)x^m dx = 0, \qquad m = 0, \ldots, M-1. \tag{3.6}$$

We will also need the notation

$$\varphi_{j,k}(x) = 2^{-j/2}\varphi(2^{-j}x - k + 1). \tag{3.7}$$

Note that the Haar system is a particular case of (3.1)-(3.6) with $M = 1$ and $h_1 = h_2 = \frac{1}{\sqrt{2}}$, $\varphi = \chi$ and $\psi = h$, and that the expansion (2.14)-(2.17) and the nonstandard form in (2.26) in Section II can be rewritten in any wavelet basis by simply replacing functions χ and h by φ and ψ respectively.

Remark 3.1. Several classes of functions φ, ψ have been constructed in recent years, and we refer the reader to [4] for a detailed description of some of those.

Remark 3.2. Unlike the Haar basis, the functions φ_I, φ_J can have overlapping supports for $J \neq I$. As a result, the pyramid structure (2.12) 'spills out' of the interval $[1, N]$ on which the structure is originally defined. Therefore, it is technically convenient to replace the original structure with a periodic one with period N. This is equivalent to replacing the original wavelet basis with its periodized version (see [8]).

3.2. Wavelet-based quadratures. In the preceeding subsection, we introduce a procedure for calculating the coefficients s_k^j, d_k^j for all $j \geq 1, k = 1, 2, \ldots, N$, given the coefficients s_k^0 for $k = 1, 2, \ldots, N$. In this subsection, we introduce a set of quadrature formulae for the efficient evaluation of the coefficients s_k^0 corresponding to smooth functions f. The simplest class of procedures of this kind is obtained under the assumption that there exists a real constant τ_M such that the function φ satisfies the condition

$$\int \varphi(x + \tau_M)x^m \, dx = 0, \quad \text{for} \qquad m = 1, 2, \ldots, M-1, \tag{3.8}$$

$$\int \varphi(x) \, dx = 1, \tag{3.9}$$

i.e. that the first $M-1$ 'shifted' moments of φ are equal to zero, while its integral is equal to 1. Recalling the definition of s_k^0

$$s_k^0 = 2^{\frac{n}{2}} \int f(x) \, \varphi(2^n \, x - k + 1) \, dx = 2^{\frac{n}{2}} \int f(x + 2^{-n}(k-1)) \, \varphi(2^n x) \, dx, \tag{3.10}$$

expanding f into a Taylor series around $2^{-n}(k - 1 + \tau_M)$, and using (3.8), we obtain

$$s_k^0 = 2^{\frac{n}{2}} \int f(x + 2^{-n}(k-1))\, \varphi(2^n x)\, dx = 2^{-\frac{n}{2}} f(2^{-n}(k - 1 + \tau_M)) + O(2^{-n(M+\frac{1}{2})}). \quad (3.11)$$

In effect, (3.11), is a one-point quadrature formula for the evaluation of s_k^0. Applying the same calculation to s_k^j with $j \geq 1$, we easily obtain

$$s_k^j = 2^{\frac{-n+j}{2}} f(2^{-n+j}(k - 1 + \tau_M)) + O(2^{-(n-j)(M+\frac{1}{2})}), \quad (3.12)$$

which turns out to be extremely useful for the rapid evaluation of the coefficients of compressed forms of matrices (see Section IV below).

Though the compactly supported wavelets found in [4] do not satisfy the condition (3.8), a slight variation of the procedure described there produces a basis satisfying (3.8), in addition to (3.1) - (3.6). Coefficients of the filters $\{h_k\}$ corresponding to $M = 2, 4, 6$ and appropriate choices of τ_M can be found in Appendix A, and we would like to thank I. Daubechies for providing them to us.

It turns out that the filters in Table 1 are 50% longer that those in the original wavelets found in [4], given the same order M. Therefore, it might be desirable to adapt the numerical scheme so that the 'shorter' wavelets could be used. Such an adaptation (by means of appropriately designed quadrature formulae for the evaluation of the integrals (3.10)) is presented in the Appendix B.

Remark 3.3. We do not discuss in this paper wavelet-based quadrature formulae for the evaluation of singular integrals, since such schemes tend to be problem-specific. Note, however, that for all integrable kernels quadrature formulae of the type developed in this paper are adequate with minor modifications.

3.3. Fast wavelet transform. For the rest of this section, we treat the procedures being discussed as linear transformations in R^N, viewed as the Euclidean space of all periodic sequences with the period N.

Replacing the Haar basis with a basis of wavelets with vanishing moments, and assuming that the coefficients $s_k^0, k = 1, 2, \ldots, N$ are given, we replace the expressions (2.8) - (2.11) with the formulae and

$$s_k^j = \sum_{n=1}^{n=2M} h_n s_{n+2k-2}^{j-1}, \quad (3.13)$$

$$d_k^j = \sum_{n=1}^{n=2M} g_n s_{n+2k-2}^{j-1}, \quad (3.14)$$

where s_k^j and d_k^j are viewed as periodic sequences with the period 2^{n-j} (see also Remark 3.2 above). As is shown in [4], the formulae (3.13) and (3.14) define an orthogonal mapping $O_j : R^{2^{n-j+1}} \to R^{2^{n-j+1}}$, converting the coefficients s_k^{j-1} with $k = 1, 2, \ldots, 2^{n-j+1}$ into the coefficients s_k^j, d_k^j with $k = 1, 2, \ldots, 2^{n-j}$, and the inverse of O_j is given by the formulae

$$
\begin{aligned}
s_{2n}^{j-1} &= \sum_{k=1}^{k=M} h_{2k} s_{n-k+1}^{j} + \sum_{k=1}^{k=M} g_{2k} d_{n-k+1}^{j}, \\
s_{2n-1}^{j-1} &= \sum_{k=1}^{k=M} h_{2k-1} s_{n-k+1}^{j} + \sum_{k=1}^{k=M} g_{2k-1} d_{n-k+1}^{j}.
\end{aligned}
\tag{3.15}
$$

Obviously, given a function f of the form

$$
f(x) = \sum_{k=1}^{2^{n-j}} s_k^j \, 2^{(n-j)/2} \, \varphi(2^{n-j}x - (k-1)) + \sum_{k=1}^{2^{n-j}} d_k^j \, 2^{(n-j)/2} \, \psi(2^{n-j}x - (k-1)), \tag{3.16}
$$

it can be expressed in the form

$$
f(x) = \sum_{l=1}^{2^{n-j+1}} s_l^{j-1} \, 2^{(n-j+1)/2} \, \varphi(2^{n-j+1}x - (l-1)), \tag{3.17}
$$

with $s_l^{j-1}, l = 1, 2, \ldots 2^{n-j+1}$ given by (3.15).

Observation 3.1. Given the coefficients $s_k^0, k = 1, 2, \ldots, N$, recursive application of the formulae (3.13), (3.14) yields a numerical procedure for evaluating the coefficients s_k^j, d_k^j for all $j = 1, 2, \ldots, n$, $k = 1, 2, \ldots, 2^{n-j}$, with a cost proportional to N. Similarly, given the values d_k^j for all $j = 1, 2, \ldots, n$, $k = 1, 2, \ldots, 2^{n-j}$, and s_1^n (note that the vector s^n contains only one element) we can reconstruct the coefficients s_k^0 for all $k = 1, 2, \ldots, N$ by using (3.15) recursively for $j = n, n-1, \ldots, 0$. The cost of the latter procedure is also $O(N)$. Finally, given an expansion of the form

$$
f(x) = \sum_{j=0}^{n} \sum_{k=1}^{2^{n-j}} s_k^j \, 2^{(n-j)/2} \, \varphi(2^{n-j}x - (k-1)) + \sum_{j=0}^{n} \sum_{k=1}^{2^{n-j}} d_k^j \, 2^{(n-j)/2} \, \psi(2^{n-j}x - (k-1)), \tag{3.18}
$$

it costs $O(N)$ to evaluate all coefficients $s_k^0, k = 1, 2, \ldots, N$ by the recursive application of the formula (3.17) with $j = n, n-1, \ldots, 0$.

Observation 3.2. It is easy to see that the entries of the matrices $\alpha^j, \beta^j, \gamma^j$ with $j = 1, 2 \ldots, n$, are the coefficients of the two-dimensional wavelet expansion of the function $K(x, y)$, and can be obtained by a two-dimensional version of the pyramid scheme (2.12), (3.13), (3.14). Indeed, the definitions (2.15)- (2.17) of these coefficients can be rewritten in the form

$$
\alpha_{i,l}^j = 2^{-j} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K(x, y) \, \psi(2^{-j}x - (i-1)) \, \psi(2^{-j}y - (l-1)) \, dxdy, \tag{3.19}
$$

$$
\beta_{i,l}^j = 2^{-j} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K(x, y) \, \psi(2^{-j}x - (i-1)) \, \varphi(2^{-j}y - (l-1)) \, dxdy, \tag{3.20}
$$

$$\gamma_{i,l}^{j} = 2^{-j} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K(x,y)\ \varphi(2^{-j}x - (i-1))\ \psi(2^{-j}y - (l-1))\ dxdy, \tag{3.21}$$

and we will define an additional set of coefficients $s_{i,l}^{j}$ by the formula

$$s_{i,l}^{j} = 2^{-j} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K(x,y)\ \varphi(2^{-j}x - (i-1))\ \varphi(2^{-j}y - (l-1))\ dxdy. \tag{3.22}$$

Now, given a set of coefficients $s_{i,l}^{0}$ with $i,l = 1,2,\ldots,N$, repeated application of the formulae (3.13), (3.14) produces

$$\alpha_{i,l}^{j} = \sum_{k,m=1}^{2M} g_k g_m s_{k+2i-2,m+2l-2}^{j-1}, \tag{3.23}$$

$$\beta_{i,l}^{j} = \sum_{k,m=1}^{2M} g_k h_m s_{k+2i-2,m+2l-2}^{j-1}, \tag{3.24}$$

$$\gamma_{i,l}^{j} = \sum_{k,m=1}^{2M} h_k g_m s_{k+2i-2,m+2l-2}^{j-1}, \tag{3.25}$$

$$s_{i,l}^{j} = \sum_{k,m=1}^{2M} h_k h_m s_{k+2i-2,m+2l-2}^{j-1}, \tag{3.26}$$

with $i,l = 1,2,\ldots,2^{n-j}$, $j = 1,2,\ldots,n$. Clearly, formulae (3.23) - (3.26) are a two-dimensional version of the pyramid scheme (2.12), and provide an order N^2 scheme for the evaluation of the elements of all matrices $\alpha^j, \beta^j, \gamma^j$ with $j = 1,2\ldots,n$.

IV Integral operators and accuracy estimates

4.1. Non-standard form of integral operators. In order to describe methods for 'compression' of integral operators, we restrict our attention to several specific classes of operators frequently encountered in analysis. In particular, we give exact estimates for pseudo-differential and Calderon-Zygmund operators.

We start with several simple observations. The non-standard form of a kernel $K(x,y)$ is obtained by evaluating the expressions

$$\alpha_{II'} = \int\int K(x,y)\ \psi_I(x)\ \psi_{I'}(y)\ dxdy, \tag{4.1}$$

$$\beta_{II'} = \int\int K(x,y)\ \psi_I(x)\ \varphi_{I'}(y)\ dxdy, \tag{4.2}$$

and

$$\gamma_{II'} = \int\int K(x,y)\ \varphi_I(x)\ \psi_{I'}(y)\ dxdy. \tag{4.3}$$

(see Figure 1). Suppose now that K is smooth on the square $I\times I' \in [0,N]\times[0,N]$. Expanding K into a Taylor series around the center of $I\times I'$, combining (3.6) with (4.1) - (4.3) and remembering that the functions $\psi_I, \psi_{I'}$ are supported on the intervals I, I' respectively, we obtain the estimate

$$\mid \alpha_{II'} \mid + \mid \beta_{II'} \mid + \mid \gamma_{II'} \mid \leq C \mid I \mid^{M+1} \sup_{(x,y)\in I\times I'} \sum_j \mid \frac{\partial^M}{\partial x^j\ \partial y^{M-j}} K(x,y) \mid . \tag{4.4}$$

Obviously, the right-hand side of (4.4) is small whenever either $\mid I \mid$ or the derivatives involved are small, and we use this fact to 'compress' matrices of integral operators by converting them to the non-standard form, and discarding the coefficients that are smaller than a chosen threshold.

To be more specific, consider pseudo-differential operators and Calderon-Zygmund operators. These classes of operators are given by integral or distributional kernels that are smooth away from the diagonal, and the case of Calderon-Zygmund operators is particularly simple. These operators have kernels $K(x,y)$ which satisfy the estimates

$$|K(x,y)| \leq \frac{1}{|x-y|}, \tag{4.5}$$

$$|\partial_x^M K(x,y)| + |\partial_y^M K(x,y)| \leq \frac{C_0}{|x-y|^{1+M}} \tag{4.6}$$

for some $M \geq 1$. To illustrate the use of the estimates (4.6) for the compression of operators, consider the simplest case of $M = 1$, so that

$$\beta_{II'} = \int\int K(x,y) h_I(x) \chi_{I'}(y) dxdy, \tag{4.7}$$

where we assume that the distance between I and I' is greater than than $\mid I \mid$. Since

$$\int h_{I'}\ dy = 0, \tag{4.8}$$

we have

$$\begin{aligned} \mid \beta_{II'} \mid &\leq \mid \int\int (K(x,y) - K(x,y_{I'}) h_I(x) \chi_{I'}(y) dxdy \mid \\ &\leq 2\frac{\mid I \mid^2}{\mid x_I - y_{I'} \mid^2}, \end{aligned} \tag{4.9}$$

where x_I denotes the center of the interval I. In other words, the coefficient $\beta_{II'}$ decays quadratically as a function of the distance between the intervals I, I', and for sufficiently large N and finite precision of calculations, most of the matrix can be discarded, leaving only a band around the diagonal. However, algorithms using the above estimates (with $M = 1$) tend to be quite inefficient, due to the slow decay of the matrix elements with their distance from the diagonal. The following simple proposition generalizes the estimate (4.9) for the case of larger M, and provides an analytical tool for efficient numerical compression of a wide class of operators.

Proposition 4.1. Suppose that in the expansion (2.14), the wavelets φ, ψ satisfy the conditions (3.1) - (3.3), and (3.6). Then for any kernel K satisfying the conditions (4.6), the coefficients $\alpha^j_{i,l}, \beta^j_{i,l}, \gamma^j_{i,l}$ in the non-standard form (see (2.18) - (2.20) and Figure 1) satisfy the estimate

$$|\alpha^j_{i,l}| + |\beta^j_{i,l}| + |\gamma^j_{i,l}| \leq \frac{C_M}{1 + |i-l|^{M+1}}, \tag{4.10}$$

for all

$$| \, i - l \, | \geq 2M. \tag{4.11}$$

Remark 4.1. For most numerical applications, the estimate (4.10) is quite adequate, as long as the singularity of K is integrable across each row and each column (see the following section). To obtain a more subtle analysis of the operator T_0 (see (2.23) above) and correspondingly tighter estimates, some of the ideas arising in the proof of the '$T(1)$' theorem of David and Journé are required. We discuss these issues in more detail in Section 4.5 below.

Similar considerations apply in the case of pseudo-differential operators. Let T be a pseudo-differential operator with symbol $\sigma(x, \xi)$ defined by the formula

$$T(f)(x) = \sigma(x, D)f = \int e^{ix\xi} \, \sigma(x, \xi) \hat{f}(\xi) \, d\xi = \int K(x, y) f(y) \, dy, \tag{4.12}$$

where K is the distributional kernel of T. Assuming that the symbols σ of T and σ^* of T^* satisfy the standard conditions

$$| \, \partial^\alpha_\xi \, \partial^\beta_x \, \sigma(x, \xi) \, | \leq C_{\alpha,\beta} (1 + | \, \xi \, |)^{\lambda - \alpha + \beta} \tag{4.13}$$

$$| \, \partial^\alpha_\xi \, \partial^\beta_x \, \sigma^*(x, \xi) \, | \leq C_{\alpha,\beta} (1 + | \, \xi \, |)^{\lambda - \alpha + \beta}, \tag{4.14}$$

we easily obtain the inequality

$$|\alpha^j_{i,l}| + |\beta^j_{i,l}| + |\gamma^j_{i,l}| \leq \frac{2^{\lambda \, j} \, C_M}{(1 + |i-l|)^{M+1}}, \tag{4.15}$$

for all integer i, l.

Remark 4.2. A simple case of the estimate (4.15) is provided by the operator $T = \frac{d}{dx}$, in which case it is obvious that

$$|\beta_{il}^j| =< \psi_i^j, \frac{d}{dx}\varphi_l^j >= 2^j \int \psi(2^j\ x - (i-1))\ \varphi'(2^j\ x - (l-1))2^j\ dx = 2^j\ \beta_{i-l}, \tag{4.16}$$

where the sequence $\{\beta_i\}$ is defined by the formula

$$\beta_i = \int \psi(x-i)\ \varphi'(x)\ dx, \tag{4.17}$$

provided a sufficiently smooth wavelet $\varphi(x)$ is used.

4.2. Numerical calculations and compression of operators. Suppose now that we approximate the operator T_0^N by the operator $T_0^{N,B}$ obtained from T_0^N by setting to zero all coefficients of matrices $\alpha = \{\alpha_{II'}\}$, $\beta = \{\beta_{II'}\}$, $\gamma = \{\gamma_{II'}\}$ outside of bands of width $B \geq 2M$ around their diagonals. It is easy to see that

$$\| T_0^{N,B} - T_0^N \| \leq \frac{C}{B^M} \log_2(N), \tag{4.18}$$

where C is a constant determined by the kernel K. In other words, the matrices α, β, γ can be approximated by banded matrices $\alpha^B, \beta^B, \gamma^B$ respectively, and the accuracy of the approximation is

$$\frac{C}{B^M} \log_2(N). \tag{4.19}$$

In most numerical applications, the accuracy ε of calculations is fixed, and the parameters of the algorithm (in our case, the band width B and order M) have to be chosen in such a manner that the desired precision of calculations is achieved. If M is fixed, then B has to be such that

$$\| T_0^{N,B} - T_0^N \| \leq \frac{C}{B^M} \log_2(N) \leq \varepsilon, \tag{4.20}$$

or, equivalently,

$$B \geq \log_2(C_M) + \log_2(\frac{1}{\varepsilon}) + \log_2(\log_2(N)). \tag{4.21}$$

In other words, T_0^N has been approximated to precision ε with its truncated version, which can be applied to arbitrary vectors for a cost proportional to $N \log_2(\log_2(N))$, which for all practical purposes does not differ from N. A considerably more detailed investigation (see Remark 4.1 above and Section 4.5 below) permits the estimate (4.21) to be replaced with the estimate

$$B \geq \log_2(C_M) + \log_2(\frac{1}{\varepsilon}), \tag{4.22}$$

making the application of the operator T_0^N to an arbitrary vector with arbitrary fixed accuracy into a procedure of order exactly $O(N)$.

Whenever sufficient analytical information about the operator T is available, the evaluation of those entries in the matrices α, β, γ that are smaller than a given threshold can be avoided altogether, resulting in an $O(N)$ algorithm (see Section V below for a more detailed description of this procedure).

Remark 4.3. Both Proposition 4.1 and the subsequent discussion assume that the kernel K is non-singular everywhere outside the diagonal, on which it is permitted to have integrable singularities. Clearly, it can be generalized to the case when the singularities of K are distributed along a finite number of bands, columns, rows, etc. While the analysis is not considerably complicated by this generalization, the implementation of such a procedure on the computer is significantly more involved (see Section V below).

4.3. Rapid evaluation of the non-standard form of an operator. In this subsection, we construct an efficient procedure for the evaluation of the elements of the non-standard form of an operator T lying within a band of width B around the diagonal. The procedure assumes that T satisfies conditions (4.5), (4.6), of Section IV, and has an operation count proportional to NB (as opposed to the $O(N^2)$ estimate for the general procedure described in Observation 3.2).

To be specific, consider the evaluation of the coefficients $\beta_{i,l}^j$ for all $j = 1, 2, , \ldots, n$, and $| i - l | \leq B$. According to (3.24),

$$\beta_{i,l}^j = \sum_{k,m=1}^{2M} g_k h_m s_{k+2i-2,m+2l-2}^{j-1}, \tag{4.23}$$

which involves the coefficients $s_{i',l'}^{j-1}$ in a band of size $3B$ defined by the condition $| i' - l' | \leq 3B$. Clearly, (3.26), could be used recursively to obtain the required coefficients $s_{i',l'}^{j-1}$, and the resulting procedure would redquire order N^2 operations. We therefore compute the coefficients $s_{i',l'}^{j-1}$ directly by using appropriate quadratures. In particular, the application of the one-point quadrature (3.12) to $K(x, y)$, combined with the estimate (4.6), gives

$$s_{i',l'}^j = 2^{n-j+1} K\left(2^{-n+j-1}(i' - 1 + \tau_M), 2^{-n+j-1}(l' - 1 + \tau_M)\right) + O\left(\frac{1}{| i' - l' |^{M+1}}\right). \tag{4.24}$$

If the wavelets used do not satisfy the moment condition (3.8), more complicated quadratures have to be used (see Appendix B to this paper).

4.4. The standard matrix realization in the wavelet basis. While the evaluation of the operator T via the non-standard form (i.e., via the matrices $\alpha^j, \beta^j, \gamma^j$) is an efficient tool for applying it to arbitrary functions, it is not a representation of T in any basis. There are

obvious advantages to obtaining a mechanism for the compression of operators that is simply a representation of the operator in a suitably chosen basis, even at the cost of certain sacrifices in the speed of calculations (provided that the cost stays $O(N)$ or $O(N\log(N))$). It turns out that simply representing the operator T in the basis of wavelets satisfying the conditions (3.6) results (to any fixed accuracy) in a matrix containing no more than $O(N\log(N))$ non-zero elements. Indeed, the elements of the matrix representing T in this basis are of the form,

$$T_{IJ} = \langle T\psi_I, \psi_J \rangle, \tag{4.25}$$

with I, J all possible pairs of diadic intervals in $\mathbf{R}$, not necessarily such that $|\ I\ | = |\ J\ |$. Combining estimates (4.5), (4.6) with (3.6), we easily see that

$$|\ T_{IJ}\ | \leq C_M \left(\frac{|\ I\ |}{|\ J\ |}\right)^{1/2} \left(\frac{|\ I\ |}{d(I,J)}\right)^{M+1}, \tag{4.26}$$

where C_M is a constant depending on M, K, and the choice of the wavelets, $d(I,J)$ denotes the distance between I, J, and it is assumed that $|\ I\ | \leq |\ J\ |$. It is easy to see that for large N and fixed $\varepsilon \geq 0$, only $O(N\log(N))$ elements of the matrix (4.25) will be greater than ε, and by discarding all elements that are smaller than a predetermined threshold, we compress it to $O(N\log(N))$ elements.

Remark 4.4. A considerably more detailed investigation (see [8]) shows that in fact the number of elements in the compressed matrix is asymptotically proportional to N, as long as the images of the constant function under the mappings T and T^* are smooth. Fortunately, the latter is always the case for pseudo-differential and many other operators.

Numerically, evaluation of the compressed form of the matrix $\{T_{IJ}\}$ starts with the calculation of the coefficients s^0 (see (2.7)) via an appropriately chosen quadrature formula. For example, if the wavelets used satisfy the conditions (3.8), (3.9), the one-point formula (3.10) is quite adequate. Other quadrature formulae for this purpose can be found in Appendix B to this paper. Once the coefficients s^0 have been obtained, the subsequent calculations can be carried out in one of three ways.

1. The naive approach is to construct the full matrix of the operator T in the basis associated with wavelets by following the pyramid (2.12). After that, the elements of the resulting matrix that are smaller than a predetermined threshold, are discarded. Clearly, this scheme requires $O(N^2)$ operations, and does not require any prior knowledge of the structure of T.

2. When the structure of singularities of the kernel K is known, the locations of the coefficients of the matrix $\{T_{IJ}\}$ exceeding the threshold ε can be determined *a priori*. After that, these can be evaluated by simply using appropriate quadrature formulae on each of the supports of the corresponding basis functions. The resulting procedure requires order $O(N\log(N))$ operations

when the operator in question is either Calderon-Zygmund or pseudo-differential, and is easily adaptable to other distributions of singularities of the kernel.

3. The third approach is to start with the non-standard form of the operator T, compress it, and convert the compressed version into the standard form. This simplifies the error analysis of the scheme, enabling one to use the one-point quadratures (3.10). The conversion procedure starts with the formula

$$\beta^j_{k,l} = 2^{-j} \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} K(x,y)\ \psi(2^{-j}x - (k-1))\ dx \right)\ \varphi(2^{-j}y - (l-1))\ dy, \tag{4.27}$$

which is an immediate consequence of (2.16), (2.19). Combining (4.27) with (3.14), we immediately obtain

$$\begin{aligned} T_{IJ} &= 2^{-(2j+1)/2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K(x,y)\ \psi(2^{-j}x - (k-1))\ \psi(2^{-(j+1)}y - (i-1))\ dxdy \\ &= \sum_{l=1}^{2M} g_l \beta^j_{k,l+2i-2}, \end{aligned} \tag{4.28}$$

where $I = I_{j,k}$ and $J = I_{j+1,i}$. Similarly, we define the set of coefficients $\{S_{I,J}\}$ via the formula

$$S_{IJ} = \sum_{l=1}^{2M} h_l \beta^j_{k,l+2i-2}, \tag{4.29}$$

and observe that these are the coefficients s_i^{j+1} in the pyramid scheme (2.12). In general, given the coefficients S_{IJ} on step m (that is, for all pairs (I,J) such that $\mid J \mid = 2^m \mid I \mid$), we move to the next step by applying the formula (4.28) recursively.

Remark 4.5. Clearly, the above procedure amounts to simply applying the pyramid scheme (2.12) each column of the matrix β^j.

4.5. Uniform estimates for discretizations of Calderon-Zygmund operators. As has been observed in Remark 4.1, the estimates (4.10) are adequate for most numerical purposes. However, they can be strengthened in two important respects.

1. The condition (4.11) can be eliminated under a weak cancellation condition (4.30).

2. The condition (4.10) does not by itself guarantee either the boundedness of the operator T, or the uniform (in N) boundedness of its discretizations T_0. In this section, we provide the necessary and sufficient conditions for the boundedness of T, or, equivalently, for the uniform boundedness of its discretizations T_0. This condition is, in fact, a reformulation of the '$T(1)$' theorem of David and Journé.

Uniform boundedness of the matrices α, β, γ. We start by observing that estimates (4.5), (4.6) are not sufficient to conclude that $\alpha^j_{i,\ell}, \beta^j_{i,\ell}, \gamma^j_{i,\ell}$ are bounded for $|i-\ell| \leq 2M$ (for example, consider $K(x,y) = \frac{1}{|x-y|}$). We therefore need to assume that T defines a bounded operator on L^2 or a substantially weaker condition

$$|\int_{I\times J} K(x,y)\, dxdy\, | \leq C|I| \tag{4.30}$$

for all dyadic intervals I (this is the "weak cancellation condition", see [8]). Under this condition and the conditions (4.5), (4.6) Proposition 4.1 can be extended to

$$|\alpha^j_{i,\ell}| + |\beta^j_{i,\ell}| + |\gamma^j_{i,\ell}| \leq \frac{C_M}{1+|i-\ell|^{M+1}} \tag{4.31}$$

for all i, l (see [8]).

Uniform boundedness of the operators T_0. We have seen in (2.26) a decomposition of the operator T_0 into a sum of contributions from the different "scales j". More precisely, the matrices $\alpha^j, \beta^j, \gamma^j$ act on the vector $\{s^j_k\}, \{d^j_k\}$, where d^j are coordinates of the function with respect to the orthogonal set of functions $2^{-j/2}\psi(2^{-j}x - k)$, and the s^j are auxiliary quantities needed to calculate the d^j_k. The remarkable feature of the nonstandard form is the decoupling achieved among the scales j followed by a simple coupling performed in the reconstruction formulas (3.17). (The standard form, by contrast, contains matrix entries reflecting "interactions" between all pairs of scales). In this subsection, we analyze this coupling mechanism in the simple case of the Haar wavelets, in effect reproducing the proof of the '$T(1)$' theorem (see [3]).

For simplicity, we will restrict our attention to the case where $\alpha = \gamma = 0$, and β satisfies conditions (4.31) (which are essentially equivalent to (4.5), (4.6), (4.31)). In this case, for the Haar wavelets we have

$$T(f)(x) = \sum_I h_I(x) \sum_{I'} \beta_{II'} s_{I'} \tag{4.32}$$

which can be rewritten in the form

$$T(f) = \sum_I h_I(x) \sum_{I'} \beta_{II'}(s_{I'} - s_I) + \sum_I \beta_I s_I h_I(x), \tag{4.33}$$

where

$$\beta_I = \sum_{I'} \beta_{II'} = \frac{1}{|I|^{1/2}} \int\int h_I(x)\, K(x,y)\, dxdy = \langle h_I, \beta(x)\rangle \frac{1}{|I|^{1/2}} \tag{4.34}$$

and

$$\beta(x) = \int K(x,y)\, dy = T(1)(x). \tag{4.35}$$

It is easy to see (by expressing s_I in terms of d_I) that the operator

$$B_1(f) = \sum_I h_I(x) \sum_{I'} \beta_{II'}(s_{I'} - s_I) \tag{4.36}$$

is bounded on L^2 whenever (4.31) is satisfied with $M = 1$. We are left with the "diagonal" operator

$$B_2(f)(x) = \sum_I \beta_I s_I h_I(x), \tag{4.37}$$

$$\beta_I = \frac{1}{|I|^{1/2}} \langle \beta(x), h_I \rangle, \tag{4.38}$$

with

$$s_I = \langle f, \chi_I \rangle. \tag{4.39}$$

Clearly

$$\|B_2(f)\|_2^2 = \sum \beta_I^2 s_I^2. \tag{4.40}$$

If we choose $f = \chi_J$ where J is a dyadic interval we find $s_I = |I|^{1/2}$ for $I \subseteq J$ from which we deduce that a necessary condition for B_2 to define a bounded operator on $L^2(\mathbf{R})$ is given as

$$\sum_{I \subseteq J} |I| \beta_I^2 = \sum_{I \subseteq J} \langle \beta, h_I \rangle^2 \leq c|J| \tag{4.41}$$

but since the h_I for $I \subseteq J$ are orthogonal in $L^2(J)$,

$$\sum_{I \subseteq J} \langle \beta, h_I \rangle^2 = \int_J |\beta(x) - m_J(\beta)|^2, \tag{4.42}$$

with

$$m_J(\beta) = \frac{1}{|J|} \int_J \beta(x) dx. \tag{4.43}$$

Combining (4.41) with (4.42), we obtain

$$\frac{1}{|J|} \int_J |\beta(x) - m_J(\beta)|^2 dx \leq C. \tag{4.44}$$

Expression (4.44) is usually called bounded dyadic mean oscillation condition (BMO) on β, and is necessary for the boundedness of B_2 on L^2. It has been proved by Carleson (see, for example, [8]) that the condition (4.41) is necessary and sufficient for the following inequality to hold

$$\sum \beta_I^2 s_I^2 \leq C \int |f|^2 dx, \; s_I = \langle f, \chi_I \rangle. \tag{4.45}$$

Combining these remarks we obtain

Theorem 4.1 (G. David, J.L. Journé) Suppose that the operator

$$T(f) = \int K(x,y)\, f(y)\, dy \tag{4.46}$$

satisfies the conditions (4.5), (4.6), (4.30). Then the necessary and sufficient condition for T to be bounded on L^2 is that

$$\beta(x) = T(1)(x), \tag{4.47}$$

$$\gamma(x) = T^*(1)(x) \tag{4.48}$$

belong to dyadic $B.M.O.$ i.e. satisfy condition (4.44).

We have shown that the operator T in Theorem 4.1 can be decomposed as a sum of three terms

$$T = B_1 + B_2 + B_3, \tag{4.49}$$

$$B_2(f) = \sum h_I \beta_I s_I, \tag{4.50}$$

$$B_3(f) = \sum \chi_I \gamma_I d_I, \tag{4.51}$$

with $|I|^{1/2}\beta_I = \langle h_I, \beta \rangle$, $\gamma_I = \langle \chi_I, \gamma \rangle$, $\beta = T(1)$, and $\gamma = T^*(1)$.

The principal term B_1, when converted to the standard form, has a band structure with decay rate independent of N. The terms B_2, B_3 are bounded in the standard form only when β, γ are in $B.M.O.$ (see [8]).

4.6. Algorithms for bilinear functionals The terms B_2, B_3 are bilinear transformations in (β, f), (γ, f) respectively. Such "pseudo products" occur frequently as differentials (in the direction β) of non-linear functionals of f (see [3]). In this section, we show that pseudo-products can be implemented in order N operation (or for the same cost as ordinary multiplication). To be specific, we have the following proposition (see [3]).

Proposition 4.2. Let $K(x,y,z)$ satisfy the conditions

$$|K(x,y,z)| \leq \frac{1}{(x-y)^2 + (x-z)^2}, \tag{4.52}$$

$$|\partial_x^M K| + |\partial_y^M K| + |\partial_z^M K| \leq \frac{1}{(|x-y|+|x-z|)^{M+2}}, \tag{4.53}$$

and the bilinear functional $B(f,g)$ be defined by the formula

$$B(f,g)(x) = \int K(x,y,z)\, f(y)\, g(z)\, dydz \tag{4.54}$$

Then the bilinear functional $B(f,g)$ can be applied to a pair of arbitrary functions f, g for a cost proportinal to N, with the proportionality coefficient depending on M and the desired accuracy, and independent of the kernel K.

Following is an outline of an algorithm implementing such a procedure. As in the linear case, we write

$$K(x,y,z) = \sum_{I,Q} \alpha_{I,Q}\psi_I(x)\psi_Q(y,z) + \beta_{I,Q}\psi_I(x)\varphi_Q(y,z) \qquad + \gamma_{I,Q}\varphi_I(x)\psi_Q(y,z) \tag{4.55}$$

where $Q = J \times J'$, $|I| = |J| = |J'|$ and $\psi_Q(y,z)$ is a wavelet basis in two variables (i.e. $\varphi_J(y)\psi_{J'}(z), \psi_J(y)\varphi_{J'}(z), \psi_J(y)\psi_{J'}(z)$) and

$$\varphi_Q(y,z) = \varphi_J(y)\varphi_{J'}(z). \tag{4.56}$$

Substituting in (4.55) into (4.54) we obtain

$$\begin{aligned} B(f,g)(x) &= \sum_I \psi_I(x) \left\{ \sum_{J,J'} \alpha^{(1)}_{I,J,J'} s_J(f) d_{J'}(g) + \alpha^{(2)}_{I,J,J'} d_J(f) s_{J'}(g) + \alpha^{(3)}_{I,J,J'} d_J(f) d_{J'}(g) \right\} \\ &+ \sum_I \psi_I(x) \sum_{J,J'} \beta_{I,J,J'} s_J(f) s_{J'}(g) \qquad (4.57) \\ &+ \sum_I \varphi_I(x) \left\{ \sum_{J,J'} \gamma^{(1)}_{I,J,J'} s_J(f) d_{J'}(g) + \gamma^{(2)}_{I,J,J'} d_J(f) s_{J'}(g) + \gamma^{(3)}_{I,J,J'} d_J(f) d_{J'}(g) \right\} \end{aligned}$$

where $\alpha^{(1)}_{I,J,J'}$, $\alpha^{(2)}_{I,J,J'}$, $\alpha^{(3)}_{I,J,J'}$, $\beta_{I,J,J'}$, and $\gamma^{(1)}_{I,J,J'}$, $\gamma^{(2)}_{I,J,J'}$, $\gamma^{(3)}_{I,J,J'}$ denote the coefficients of the function $K(x,y,z)$ in the three-dimensional wavelet basis. Therefore, combining (4.57) with Observation 3.1, we obtain an order $O(N)$ algorithm for the evaluation of (4.54) on an arbitrary pair of vectors.

It easily follows from the estimates (4.52), (4.53) that

$$|\alpha_{I,J,J'}| + |\beta_{I,J,J'}| + |\gamma_{I,J,J'}| \leq \left(\frac{C|I|}{[dist(I,J) + dist(I,J')]} \right)^{2+M} \tag{4.58}$$

resulting in banded matrices and a "compressed" version having $O(N)$ entries (also, compare (4.58) with (4.10)).

Similar results can be obtained for many classes of non-linear functionals whose differentials satisfy the conditions analogous to (4.52), (4.53).

V Description of the algorithm

In this section, we describe an algorithm for rapid application of a matrix T_0 discretizing an integral operator T to an arbitrary vector. It is assumed that T satisfies the estimates (4.5), (4.6), or the more general conditions described in Remark 4.3. The scheme consists of four steps.

Step 1. Evaluate the coefficients of the matrices $\alpha^j, \beta^j, \gamma^j, j = 1, 2, \ldots, n$ corresponding to T_0 (see (2.18)-(2.20) above), and discard all elements of these matrices whose absolute values are smaller than ε. The remaining number of elements in all matrices $\alpha^j, \beta^j, \gamma^j$ is proportional to N (see estimates (4.21), (4.22)).

Depending on the *a priori* information available about the operator T, one of two procedures are used, as follows.

1. If the *a priori* information is limited to that specified in the Remark 4.3 (i.e. the singularities of K are distributed along a finite number of bands, rows, and columns, but their exact locations are not known), then the extremely simple procedure described in Observation 3.2 is utilized. The resulting cost of this step is $O(N^2)$, and it should only be used when the second scheme (see below) can not be applied.

2. If the operator T satisfies the estimates (4.5), (4.6) for some $M \geq 1$, and the wavelets employed satisfy the condition (3.8), then the more efficient procedure described in Section 4.3 is used. While the implementation of this scheme is somewhat involved, it results in an order $O(N)$ algorithm, and should be used whenever possible.

Step 2. Evaluate the coefficients s_k^j, d_k^j for all $j = 1, 2, \ldots, n$, $k = 1, 2, \ldots, 2^{n-j}$ (see formulae (3.14), (3.14) and Observation 3.1).

Step 3. Apply the matrices $\alpha^j, \beta^j, \gamma^j$ to the vectors s^j, d^j, obtaining the vectors $\hat{s^j}, \hat{d^j}$ for $j = 1, 2, \ldots, n$ (see formulae (2.28), (2.30).

Step 4. Use the vectors $\hat{s^j}, \hat{d^j}$ to evaluate $T_0(f)$ via the formula (3.15) (see Observation 3.1).

Remark 5.1. It is clear that Steps 2 - 4 in the above scheme require order $O(N)$ operations, and that Step 1 requires either order $O(N)$ or $O(N^2)$ operations, depending on the *a priori* information available about the operator T. It turns out, however, that even when Step 1 requires order N operations, it is still the dominant part of the algorithm in terms of the actual operation count. In most applications, a single operator has to be applied to a relatively large number of vectors, and in such cases, it makes sense to produce the non-standard form of the operator T and store it. After that, it can be retrieved and used whenever necessary, for a very small cost (see also Section VI below).

Remark 5.2. In the above procedure, Step 1 requires $O(N^2)$ operations whenever the structure of the operator T is not described by the estimates (4.5), (4.6). Clearly, it is not the only structure of T for which an order $O(N)$ procedure can be constructed. In fact, this can be done for any structure of T described in Remark 4.3, *provided that the location of singularities of T is known a priori.* The data structures required for the construction of such an algorithm are fairly involved, but conceptually the scheme is not substantially different from that described in Section 4.3.

VI Numerical Results

A FORTRAN program has been written implementing the algorithm of the preceeding section, and numerical experiments have been performed on the SUN-3/50 computer equipped with the MC68881 floating-point accelerator. All calculations were performed in three ways: in single precision using the standard (direct) method, in double precision using the algorithm of this paper with the matrices α, β, γ truncated at various thresholds (see Section 4.2 above), and in double precision using the standard method. The latter was used as the standard against which the accuracy of the other two calculations was measured.

We applied the algorithm to a number of operators; the results of six such experiments are presented in this section and summarized in Tables 1-6, and illustrated in Figures 2-9. Column 1 of each of the tables contains the number N of nodes in the discretization of the operator, columns 2, 3 contain CPU times T_s, T_w required by the standard (order $O(N^2)$) and the 'fast' ($O(N)$) schemes to multiply a vector by the resulting discretized matrix respectively, and column 4 contains the CPU T_d time used by our scheme to produce the non-standard form of the operator. Columns 5, 6 contain the L_2 and L_∞ errors of the direct calculation respectively, and columns 7, 8 contain the same information for the result obtained via the algorithm of this paper. Finally, column 9 contains the compression coefficients C_{comp} obtained by our scheme, defined by the ratio of N^2 to the number of non-zero elements in the non-standard form of T. In all cases, the experiments were performed for $N = 64, 128, 256, 512$, and 1024, and in all Figures 2-9, the matrices are depicted for $N = 256$.

Example 1.

In this example, we compress matrices of the form

$$A_{ij} = \begin{cases} \frac{1}{i-j} & i \neq j, \\ 0 & i = j, \end{cases}$$

and convert them to a system of coordinates spanned by wavelets with six first moments equal to zero. Setting to zero all entries in the resulting matrix whose absolute values are smaller than

10^{-7}, we obtain the matrix whose non-zero elements are shown in black in Figure 2. The results of this set of experiments are tabulated in Table 1. The standard form of the operator A with $N = 256$ is depicted in Figure 9.

Example 2.

Here, we compress matrices of the form

$$A_{ij} = \begin{cases} \frac{\log|i-2^{n-1}|-\log|j-2^{n-1}|}{i-j} & i \neq j; i \neq 2^{n-1}; j \neq 2^{n-1} \\ \\ 0 & \text{otherwise} \end{cases}$$

where $i, j = 1, \ldots, N$ and $N = 2^n$.

This matrix is not a convolution and its singularities are more complicated. The decomposition of this matrix using wavelets with six vanishing moments displaying entries above the threshold of 10^{-7} is shown in Figure 3, and the numerical results of these experimants are tabulated in Table 2. In this case, the structure of the singularities of the matrix is not known *a priori*, and its non-standard form was obtained by converting the whole matrix to the wavelet system of coordinates, and discarding the elements that are smaller than the threshold (see Section 4.2). Correspondingly, the cost of constructing the non-standard form of the operator is proportional to N^2 (see column 4 of Table 2). The standard form of the operator A with $N = 256$ is depicted in Figure 10.

Example 3. In this example, we compress and rapidly apply to arbitrary vectors the matrix converting the coefficients of a finite Chebychev expansion into the coefficients of a finite Legendre expansion representing the same polynomial (see [1]). The matrix is given by the formulae

$$A_{ij} = M^N_{2i\,2j}$$

where $i, j = 1, \ldots, N$ and $N = 2^n$ and M^N_{ij} is defined as

$$M^N_{ij} = \begin{cases} \frac{1}{\pi}\Lambda^2(j/2) & \text{if } 0 = i \leq j < N \text{ and } j \text{ is even} \\ \frac{2}{\pi}\Lambda((j-i)/2)\Lambda((j+i)/2) & \text{if } 0 < i \leq j < N \text{ and } i+j \text{ is even} \\ \\ 0 & \text{otherwise,} \end{cases}$$

where $\Lambda(z) = \Gamma(z+1/2)/\Gamma(z+1)$ and $\Gamma(z)$ is the gamma function. Alternatively,

$$A_{ij} = \begin{cases} \frac{1}{\pi}\Lambda^2(j) & \text{if } 0 = i \leq j < N \\ \frac{2}{\pi}\Lambda(j-i)\Lambda(j+i) & \text{if } 0 < i \leq j < N \\ \\ 0 & \text{otherwise.} \end{cases}$$

We used the threshold of 10^{-6} and wavelets with five vanishing moments to obtain the numerical results depicted in Table 3 and Figure 4. As a corollary, we obtain an algorithm for the rapid evaluation of Legendre expansions of the same complexity (and roughly the same actual efficiency) as that described in [1].

Example 4.

Here,

$$A_{ij} = \begin{cases} \log(i-j)^2 & i \neq j \\ \\ 0 & i = j. \end{cases}$$

We use wavelets with six vanishing moments and set to zero everything below 10^{-6}. Table 4 and Figure 5 describe the results of these experiments.

Example 5.

In this example,

$$A_{ij} = \begin{cases} \frac{1}{i-j+\frac{1}{2}(\cos ij)} & i \neq j \\ \\ 0 & i = j, \end{cases},$$

and it is easy to see that this operator does not satisfy the condition (4.10). Nonetheless, when a low order version of our scheme is applied to it, the results are quite staisfactory, albeit with an expectedly low accuracy (we used wavelets with two vanishing moments, and set the threshold to 10^{-3}). The results of these numerical experiments can be seen in Figure 7 and Table 5.

Example 6.

Here,

$$A_{ij} = \begin{cases} \frac{i\cos(\log i^2) - j\cos(\log j^2)}{(i-j)^2} & i \neq j \\ \\ 0 & i = j. \end{cases}$$

Like in the preceeding example, the operator being compressed satisfies the condition (4.10) with $M = 1$, and fails to do so for any larger M. Using wavelets with two vanishing moments, and setting the threshold to 10^{-3}, we obtain the results depicted in in Figure 8 and Table 6. Again, the compression rate for this reasonably large threshold is quite satisfactory.

The following observations can be made from Tables 1-6 and Figures 2-7.

1. The CPU times required by the algorithm of this paper to apply the matrix to a vector grow linearly with N, while those for the direct algoriths grow quadratically (as expected).

2. The accuracy of the method is in agreement with the estimates of Section 4, and when the threshold is set to 10^{-6}, the actual accuracies obtained tend to be slightly better than those obtained by the direct calculation in single precision.

3. In many cases, the algorithm becomes more efficient than the direct one by $N = 100$, and by $N = 1000$, the gain is roughly of the factor of 10.

4. Even when the operator fails to satisfy the condition (4.10), the application of the algorithm with a reasonably large threshold and small M leads to satisfactory compression factors.

5. Combining the linear asymptotic CPU time estimate of the algorithm of this paper with the actual timings in Tables 1-6, we observe that whenever the algorithm of this paper is applicable, extremely large-scale problems become tractable, even with relatively modest computing resources.

VII Extensions and Generalizations

7.1. Numerical operator calculus. In this paper, we construct a mechanism for the rapid application to arbitrary vectors of a wide variety of dense matrices. It turns out that in addition to the application of matrices to vectors, our techniques lead to algorithms for the rapid multiplication of operators (or, rather, their standard forms). The asymptotic complexity of the resulting procedure is also proportional to N. When applied recursively, it permits a whole range of matrix functions (polynomials, exponentials, inverses, square roots, etc.) to be evaluated for a cost proportional to N, converting the operator calculus into a competitive numerical tool (as opposed to a purely analytical apparatus it has been). These (and several related) algorithms have been implemented, and are described in a paper currently in preparation.

7.2. Generalizations to higher dimensions. The construction of the present paper is limited to the one-dimensional case, i.e. the integral operators being compressed are assumed to act on $L^2(\mathbf{R})$. Its generalization to problems in higher dimensions is fairly straightforward, and is being implemented. When combined with the Lippman-Schwinger equation, or with the classical pseudo-differential calculus, these techniques should lead to algorithms for the rapid solution of a wide variety of elliptic partial differential equations in regions of complicated shapes, of second kind integral equations in higher-dimensional domains, and of several related problems.

7.3. Non-linear operators. While the present paper discusses the 'compression' of linear and bilinear operators, extensions to multilinear functionals (defined on the functions in one, as well as higher dimensions) is not difficult to obtain. These methods (together with some of their

applications) will be described in a forthcoming paper. The underlying theory can be found in [3].

References

[1] B. Alpert and V. Rokhlin, *A Fast Algorithm for the Evaluation of Legendre Expansions*, Yale University Technical Report, YALEU/DCS/RR-671 (1989).

[2] J. Carrier, L. Greengard and V. Rokhlin *A Fast Adaptive Multipole Algorithm for Particle Simulations*, Yale University Technical Report, YALEU/DCS/RR-496 (1986), SIAM Journal of Scientific and Statistical Computing, 9 (4), 1988.

[3] R. Coifman and Yves Meyer, *Non-linear Harmonic Analysis, Operator Theory and P.D.E.*, Annals of Math Studies, Princeton, 1986, ed. E. Stein.

[4] I. Daubechies, *Orthonormal Bases of Compactly Supported Wavelets*, Comm. Pure, Applied Math, **XL1**, 1988.

[5] L. Greengard and V. Rokhlin, *A Fast Algorithm for Particle Simulations*, Journal of Computational Physics, 73(1), **325**, 1987.

[6] S. Mallat, *Review of Multifrequency Channel Decomposition of Images and Wavelet Models*, Technical Report 412, Robotics Report 178, NYU (1988).

[7] Y. Meyer *Principe d'incertitude, bases hilbertiennes et algébres d'opérateurs*, Séminaire Bourbaki, 1985-86, 662, Astérisque (Socié té Mathématique de France).

[8] Y. Meyer, *Wavelets and Operators*, Analysis at Urbana, vol.1, edited by E.Berkson, N.T.Peck and J.Uhl, London Math. Society, Lecture Notes Series 137, 1989.

[9] S.T. O'Donnel and V. Rokhlin, *A Fast Algorithm for the Numerical Evaluation of Conformal Mappings*, Yale University Technical Report, YALEU/DCS/RR-554 (1987), SIAM Journal of Scientific and Statistical Computing, 1989.

[10] J.O. Stromberg, *A Modified Haar System and Higher Order Spline Systems*, Conference in harmonic analysis in honor of Antoni Zygmund, Wadworth math. series, edited by W. Beckner and al., **II**, 475-493.

Appendix A

The following table contains filter coefficients $\{h_k\}_{k=1}^{k=3M}$ for $M = 2, 4, 6$ for one particular choice of the shift τ. These coefficients have $M - 1$ vanishing moments,

$$M_l = \sum_{k=1}^{k=3M} h_k (k - \tau_M)^l = 0, \quad l = 1, \ldots, M-1$$

where τ_M is the shift, and have been provided to the authors by I. Daubechies (see also Section 3.2 above). For $M = 2$ there are explicit expressions for $\{h_k\}_{k=1}^{k=3M}$, and with $\tau_2 = 5$, they are

$$h_1 = \frac{\sqrt{15}-3}{16\sqrt{2}}, \quad h_2 = \frac{1-\sqrt{15}}{16\sqrt{2}}, \quad h_3 = \frac{3-\sqrt{15}}{8\sqrt{2}},$$

$$h_4 = \frac{\sqrt{15}+3}{8\sqrt{2}}, \quad h_5 = \frac{\sqrt{15}+13}{16\sqrt{2}}, \quad h_6 = \frac{9-\sqrt{15}}{16\sqrt{2}},$$

and for $M = 4, 6$, the coefficients $\{h_k\}$ are presented in the table below.

	k	Coefficients h_k		k	Coefficients h_k
$M = 2$	1	0.038580777747887	$M = 6$	1	-0.0016918510194918
$\tau_2 = 5$	2	-0.12696912539621	$\tau_6 = 8$	2	-0.0034878762198426
	3	-0.077161555495774		3	0.019191160680044
	4	0.60749164138568		4	0.021671094636352
	5	0.74568755893443		5	-0.098507213321468
	6	0.22658426519707		6	-0.056997424478478
				7	0.45678712217269
				8	0.78931940900416
$M = 4$	1	0.0011945726958388		9	0.38055713085151
$\tau_8 = 8$	2	-0.012845579755324		10	-0.070438748794943
	3	0.024804330519353		11	-0.056514193868065
	4	0.050023519962135		12	0.036409962612716
	5	-0.15535722285996		13	0.0087601307091635
	6	-0.071638282295294		14	-0.011194759273835
	7	0.57046500145033		15	-0.0019213354141368
	8	0.75033630585287		16	0.0020413809772660
	9	0.28061165190244		17	0.00044583039753204
	10	-0.0074103835186718		18	-0.00021625727664696
	11	-0.014611552521451			
	12	-0.0013587990591632			

Appendix B.

In this Appendix we construct quadrature formulae using the compactly supported wavelets of [4] which do not satisfy condition (3.8). These quadrature formulae are similar to the quadrature formula (3.12) in that they do not require explicit evaluation of the function $\varphi(x)$ and are completely determined by the filter coefficients $\{h_k\}_{k=1}^{k=2M}$. Our interest in these quadrature formulae stems from the fact that for a given number M of vanishing moments of the basis functions, the wavelets of [4] have the support of length $2M$ compared with $3M$ for the wavelets satisfying condition (3.8). Since our algoritms depend linearly on the size of the support, using wavelets of [4] and quadrature formulae of this appendix makes these algorithms $\approx 50\%$ faster.

We use these quadrature formulae to evaluate the coefficients s_k^j of smooth functions without the pyramid scheme (2.12), where s_k^j are computed via (3.13) for $j = 1, \ldots, n$.

First, we explain how to compute $\{s_k^0\}_{k=1}^{k=N}$. Recalling the definition of s_k^0,

$$s_k^0 = 2^{\frac{n}{2}} \int f(x)\, \varphi(2^n x - k + 1)\, dx = 2^{\frac{n}{2}} \int f(x + 2^{-n}(k-1))\, \varphi(2^n x)\, dx, \tag{7.1}$$

we look for the coefficients $\{c_l\}_{l=0}^{l=M-1}$ such that

$$2^{\frac{n}{2}} \int f(x + 2^{-n}(k-1))\, \varphi(2^n x)\, dx = 2^{-\frac{n}{2}} \sum_{l=0}^{l=M-1} c_l\, f(l + 2^{-n}(k-1)), \tag{7.2}$$

for polynomials of degree less than M. Using (7.2), we arrive at the linear algebraic system for the coefficients c_l,

$$\sum_{l=0}^{l=M-1} l^m\, c_l = \int x^m\, \varphi(x)\, dx, \quad m = 0, 1, \ldots, M-1, \tag{7.3}$$

where the moments of the function $\varphi(x)$ are computed in terms of the filter coefficients $\{h_k\}_{k=1}^{k=2M}$.

Given the coefficients c_l, we obtain the quadrature formula for computing s_k^0,

$$s_k^0 = 2^{-\frac{n}{2}} \sum_{l=0}^{l=M-1} c_l\, f(l + 2^{-n}(k-1)) + O(2^{-n(M+\frac{1}{2})}). \tag{7.4}$$

The moments of the function φ are obtained by differentiating (an appropriate number of times) its Fourier transform $\hat{\varphi}$,

$$\hat{\varphi}(\xi) = (2\pi)^{-1/2} \int dx\, e^{ix\xi}\, \varphi(x). \tag{7.5}$$

and setting $\xi = 0$. The expressions for the moments $\int x^m \varphi(x)\, dx$ in terms of the filter coefficients $\{h_k\}_{k=1}^{k=2M}$ are found using formula for $\hat{\varphi}$ [4],

$$(2\pi)^{1/2}\hat{\varphi}(\xi) = \prod_{j=1}^{\infty} m_0(2^{-j}\xi), \tag{7.6}$$

where

$$m_0(\xi) = 2^{-1/2} \sum_{k=1}^{k=2M} h_k e^{i(k-1)\xi}. \tag{7.7}$$

The moments $\int x^m \varphi(x)\, dx$ are obtained numerically (within the desired accuracy) by recursively generating a sequence of vectors, $\{\mathcal{M}_m^r\}_{m=0}^{m=M-1}$ for $r = 1, 2, \ldots,$

$$\mathcal{M}_m^{r+1} = \sum_{j=0}^{j=m} \begin{pmatrix} m \\ j \end{pmatrix} 2^{m-j(r+1)} \mathcal{M}_{m-j}^r \mathcal{M}_j^1, \tag{7.8}$$

starting with

$$\mathcal{M}_m^1 = 2^{-m-\frac{1}{2}} \sum_{k=1}^{k=2M} h_k (k-1)^m, \quad m = 0, \ldots, M-1. \tag{7.9}$$

Each vector $\{\mathcal{M}_m^r\}_{m=0}^{m=M-1}$ represents M moments of the product in (7.6) with r terms.

We now derive formulae to compute the coefficients s_k^j of smooth functions without the pyramid scheme (2.12). Let us formulate the following

Proposition B1. Let the coefficients s_m^j be those of a smooth function at some scale j. Then

$$s_m^{j+1} = 2^{1/2} \sum_{l=1}^{l=M} q_l s_{2m+2l-3}^j + O(2^{-(n-j)M}), \tag{7.10}$$

is a formula to compute the coefficients s_m^{j+1} at the scale $j+1$ from those at the scale j. The coefficients $\{q_l\}_{l=1}^{l=M}$ in (7.10) are solutions of the linear algebraic system

$$\sum_{l=1}^{l=M} q_l (2l-1)^m = M_m, \qquad m = 0, \ldots, M-1. \tag{7.11}$$

and where M_m are the moments of the coefficients h_k scaled for convenience by $1/H(0) = 2^{-1/2}$,

$$M_m = 2^{-1/2} \sum_{k=1}^{k=2M} h_k k^m, \qquad m = 0, \ldots, M-1. \tag{7.12}$$

Using Proposition B1 we prove the following

Lemma B1. Let the coefficients s_m^j be those of a smooth function at some scale j. Then

$$s_m^{j+r} = 2^{r/2} \sum_{l=1}^{l=M} q_l^r s_{2^r(m+l-2)+1}^j + O(2^{-(n-j-r)M}), \tag{7.13}$$

is a formula to compute the coefficients s_m^{j+r} at the scale $j+r$ from those at the scale j, with $r \geq 1$. The coefficients $\{q_l^r\}_{l=1}^{l=M}$ in (7.13) are obtained by recursively generating the sequence of vectors $\{q_l^1\}_{l=1}^{l=M}, \ldots, \{q_l^r\}_{l=1}^{l=M}$ as solutions of the linear algebraic system

$$\sum_{l=1}^{l=M} q_l^r (2l-1)^m = M_m^r, \qquad m = 0, \ldots, M-1, \tag{7.14}$$

where the sequence of the moments $\{M_m^1 = M_m\}, \{M_m^2\}, \ldots, \{M_m^r\}$ is computed via

$$M_m^{r+1} = \sum_{j=0}^{j=m} \begin{pmatrix} m \\ j \end{pmatrix} M_{m-j} L_j^r, \tag{7.15}$$

where

$$L_j^r = \sum_{l=1}^{l=M} q_l^r (l-1)^j. \tag{7.16}$$

We note that for $r = 1$ (7.13) reduces to (7.10).

Proof of Proposition B1.

Let $H(\xi)$ denote the Fourier transform of the filter with the coefficients $\{h_k\}_{k=1}^{k=2M}$,

$$H(\xi) = \sum_{k=1}^{k=2M} h_k e^{ik\xi}. \tag{7.17}$$

Clearly, the moments M_m in (7.12) can be written as

$$M_m = 2^{-1/2} \left(\frac{1}{i}\partial\right)^m H(\xi)|_{\xi=0}, \qquad m = 0, \ldots, M-1. \tag{7.18}$$

Also, the trigonometric polynomial $H(\xi)$ can always be written as the product,

$$H(\xi) = \tilde{H}(\xi) Q(\xi), \tag{7.19}$$

where we choose Q to be of the form

$$Q(\xi) = \sum_{l=1}^{l=M} q_l e^{i(2l-1)\xi}, \tag{7.20}$$

and $\tilde{H}$ to have zero moments

$$\left(\frac{1}{i}\partial\right)^m \tilde{H}(\xi)|_{\xi=0}, \qquad m = 1, \ldots, M-1. \tag{7.21}$$

By differentiating (7.19) appropriate number of times, setting $\xi = 0$ and using (7.21) we arrive at (7.11). Solving (7.11), we find the coefficients $\{q_l\}_{l=1}^{l=M}$.

Since moments of $\tilde{H}$ vanish, the convolution with the coefficients of the filter $\tilde{H}$ reduces to the one-point quadrature formula of the type in (3.12). Thus applying H reduces to applying Q and scaling the result by $1/H(0) = 2^{-1/2}$. Clearly, there are only M coefficients of Q compared to $2M$ of H, and the particular form of the filter Q (7.20) was chosen so that only every second entry of s_k^j, starting with $k = 1$, is multiplied by a coefficient of the filter Q.

Proof of Lemma B1.

Lemma B1 is proved by induction. Since for $r = 1$ (7.13) reduces to (7.10), we have to show that given (7.13), it also holds if r is increased by one.

Let $\tilde{s}_k^j$ be the subsequence consisting of every 2^r entry of s_k^j starting with $k = 1$. Applying filter $\{q_l^r\}_{l=1}^{l=M}$ to s_k^j in (7.13) is equivalent to applying filter P^r to $\tilde{s}_k^j$, where

$$P^r(\xi) = \sum_{l=1}^{l=M} q_l^r e^{i(l-1)\xi}. \tag{7.22}$$

To obtain (7.13), where r is increased by one, we use the quadrature formula (7.10) of Proposition B1. Therefore, the result is obtained by convolving $\tilde{s}_k^j$ with the coefficients of the filter $Q(\xi)P^r(\xi)$, where $Q(\xi)$ is defined in (7.20).

Let us construct a new filter Q^{r+1} by factoring $Q(\xi)P^r(\xi)$ similar to (7.19),

$$Q(\xi)P^r(\xi) = \tilde{\tilde{H}}(\xi)Q^{r+1}(\xi), \tag{7.23}$$

where we chose Q^{r+1} to be of the form

$$Q^{r+1}(\xi) = \sum_{l=1}^{l=M} q_l^{r+1} e^{i(2l-1)\xi}, \tag{7.24}$$

and $\tilde{\tilde{H}}$ to have zero moments

$$\left(\frac{1}{i}\partial\right)^m \tilde{\tilde{H}}(\xi)|_{\xi=0} = 0, \qquad m = 1, \ldots, M-1. \tag{7.25}$$

Again, since moments of $\tilde{H}$ vanish, the convolution with the coefficients of the filter $\tilde{\tilde{H}}$ reduces to scaling the result by $2^{-1/2}$.

To compute moments M_m^{r+1} of $Q(\xi)P^r(\xi)$ we differentiate $Q(\xi)P^r(\xi)$ appropriate number of times, set $\xi = 0$ and arrive at (7.15) and (7.16). To obtain the linear algebraic system (7.14) for the coefficients q_l^{r+1}, we differentiate (7.23) appropriate number of times, set $\xi = 0$ and use (7.25).

Recalling that the filter P^r is applied to the subsequence $\tilde{s}_k^j$, we arrive at (7.13), where r is increased by one.

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.16	7.76	$1.26 \cdot 10^{-7}$	$3.65 \cdot 10^{-7}$	$8.89 \cdot 10^{-8}$	$1.72 \cdot 10^{-7}$	1.39
128	0.48	0.38	32.62	$2.17 \cdot 10^{-7}$	$8.64 \cdot 10^{-7}$	$1.12 \cdot 10^{-7}$	$9.94 \cdot 10^{-7}$	2.22
256	1.92	0.80	96.44	$2.81 \cdot 10^{-7}$	$1.12 \cdot 10^{-6}$	$1.25 \cdot 10^{-7}$	$5.30 \cdot 10^{-7}$	3.93
512	7.68	1.80	252.72	$4.21 \cdot 10^{-7}$	$1.75 \cdot 10^{-6}$	$1.23 \cdot 10^{-7}$	$5.16 \cdot 10^{-7}$	7.33
1024	30.72	3.72	605.74	$6.64 \cdot 10^{-7}$	$3.90 \cdot 10^{-6}$	$1.36 \cdot 10^{-7}$	$5.04 \cdot 10^{-7}$	14.09

Table 1: Numerical results for Example 1

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.16	8.62	$1.87 \cdot 10^{-7}$	$7.53 \cdot 10^{-7}$	$8.24 \cdot 10^{-8}$	$2.87 \cdot 10^{-7}$	1.23
128	0.48	0.34	35.06	$3.18 \cdot 10^{-7}$	$8.62 \cdot 10^{-7}$	$1.14 \cdot 10^{-7}$	$3.79 \cdot 10^{-7}$	2.02
256	1.92	0.84	142.82	$4.30 \cdot 10^{-7}$	$2.03 \cdot 10^{-6}$	$1.33 \cdot 10^{-7}$	$4.72 \cdot 10^{-7}$	3.76
512	7.68	1.72	574.86	$6.63 \cdot 10^{-7}$	$4.42 \cdot 10^{-6}$	$1.44 \cdot 10^{-7}$	$4.80 \cdot 10^{-7}$	7.50
1024	30.72	3.30	2,298.7	$9.25 \cdot 10^{-7}$	$6.06 \cdot 10^{-6}$	$1.71 \cdot 10^{-7}$	$6.77 \cdot 10^{-7}$	15.68

Table 2: Numerical results for Example 2

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.12	10.28	$2.64 \cdot 10^{-7}$	$7.19 \cdot 10^{-7}$	$8.09 \cdot 10^{-7}$	$2.34 \cdot 10^{-6}$	1.73
128	0.48	0.30	42.70	$6.19 \cdot 10^{-7}$	$3.94 \cdot 10^{-6}$	$1.66 \cdot 10^{-6}$	$8.02 \cdot 10^{-6}$	2.89
256	1.92	0.66	133.66	$1.28 \cdot 10^{-6}$	$5.23 \cdot 10^{-6}$	$2.51 \cdot 10^{-6}$	$1.21 \cdot 10^{-5}$	5.18
512	7.68	1.40	344.60	$2.24 \cdot 10^{-6}$	$1.35 \cdot 10^{-5}$	$3.75 \cdot 10^{-6}$	$3.31 \cdot 10^{-5}$	9.70
1024	30.72	2.78	805.90	$4.45 \cdot 10^{-6}$	$2.42 \cdot 10^{-5}$	$6.40 \cdot 10^{-6}$	$9.00 \cdot 10^{-5}$	18.60

Table 3: Numerical results for Example 3

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.14	8.84	$2.22 \cdot 10^{-5}$	$6.31 \cdot 10^{-5}$	$1.13 \cdot 10^{-6}$	$2.33 \cdot 10^{-6}$	1.37
128	0.48	0.34	38.42	$6.23 \cdot 10^{-5}$	$1.62 \cdot 10^{-4}$	$2.07 \cdot 10^{-6}$	$5.19 \cdot 10^{-6}$	2.19
256	1.92	0.84	120.22	$2.11 \cdot 10^{-4}$	$6.99 \cdot 10^{-4}$	$2.99 \cdot 10^{-6}$	$8.46 \cdot 10^{-6}$	3.82
512	7.68	1.76	310.86	$7.90 \cdot 10^{-4}$	$2.47 \cdot 10^{-3}$	$4.08 \cdot 10^{-6}$	$1.23 \cdot 10^{-5}$	7.04
1024	30.72	3.70	736.8	$2.65 \cdot 10^{-3}$	$9.44 \cdot 10^{-3}$	$6.53 \cdot 10^{-6}$	$2.19 \cdot 10^{-5}$	13.43

Table 4: Numerical results for Example 4

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.10	2.84	$1.93 \cdot 10^{-7}$	$5.04 \cdot 10^{-7}$	$1.18 \cdot 10^{-3}$	$3.11 \cdot 10^{-3}$	1.99
128	0.48	0.18	9.00	$2.65 \cdot 10^{-7}$	$9.27 \cdot 10^{-7}$	$1.54 \cdot 10^{-3}$	$4.36 \cdot 10^{-3}$	3.51
256	1.92	0.42	23.62	$3.76 \cdot 10^{-7}$	$1.83 \cdot 10^{-6}$	$2.02 \cdot 10^{-3}$	$8.33 \cdot 10^{-3}$	6.58
512	7.68	0.88	55.62	$4.93 \cdot 10^{-7}$	$2.46 \cdot 10^{-6}$	$3.19 \cdot 10^{-3}$	$3.91 \cdot 10^{-2}$	12.81
1024	30.72	1.74	123.84	$7.53 \cdot 10^{-7}$	$4.78 \cdot 10^{-6}$	$3.99 \cdot 10^{-3}$	$7.57 \cdot 10^{-2}$	25.19

Table 5: Numerical results for Example 5

Input Size (N)	Time T_s	Time T_w	Time T_d	Error of Single Precision Multiplication L_2 - norm	Error of Single Precision Multiplication L_∞ - norm	Error of FWT Multiplication L_2 - norm	Error of FWT Multiplication L_∞ - norm	Compression Coefficient C_{comp}
64	0.12	0.10	4.22	$2.59 \cdot 10^{-7}$	$8.76 \cdot 10^{-7}$	$2.42 \cdot 10^{-3}$	$4.58 \cdot 10^{-3}$	2.37
128	0.48	0.20	16.60	$3.71 \cdot 10^{-7}$	$1.07 \cdot 10^{-6}$	$2.81 \cdot 10^{-3}$	$8.61 \cdot 10^{-3}$	4.13
256	1.92	0.38	66.70	$5.03 \cdot 10^{-7}$	$2.12 \cdot 10^{-6}$	$3.62 \cdot 10^{-3}$	$1.38 \cdot 10^{-2}$	8.25
512	7.68	0.82	263.72	$8.71 \cdot 10^{-7}$	$3.10 \cdot 10^{-6}$	$3.68 \cdot 10^{-3}$	$1.60 \cdot 10^{-2}$	14.80
1024	30.72	1.50	1,107.6	$1.12 \cdot 10^{-6}$	$5.52 \cdot 10^{-6}$	$4.56 \cdot 10^{-3}$	$4.12 \cdot 10^{-2}$	33.07

Table 6: Numerical results for Example 6

FIGURE CAPTIONS

Figure 1.

Representation of the decomposed matrix. Submatrices α, β and γ on different scales are the only nonzero submatrices. In fact, most of the entries of these submatrices can be set to zero given the desired accuracy (see examples in Figures 2-8).

Figure 2.

Entries above the threshold of 10^{-6} of the decomposed matrix of Example 1 are shown black. Note that the width of the bands does not grow with the size of the matrix.

Figure 3.

Entries above the threshold of 10^{-6} of the decomposed matrix of Example 2. Vertical and horizontal bands in the middle of submatrices as well as the diagonal bands are due to the singularities of the kernel (matrix). Note, that in this case the kernel is not a convolution.

Figure 4.

Entries above the threshold of 10^{-6} of the decomposed matrix of Example 3. This matrix is a one of two transition matrices to compute Legendre expansion from Chebyshev expansion.

Figure 5.

Entries above the threshold of 10^{-6} of the decomposed matrix of Example 4.

Figure 6.

Entries of the first column of matrices α and β (on the fine scale) of Example 4. We observe fast decay away from the diagonal. The threshold is 10^{-6}.

Figure 7.

Entries above the threshold of 10^{-3} of the decomposed matrix of Example 5.

Figure 8.

Entries above the threshold of 10^{-3} of the decomposed matrix of Example 6.

Figure 9.

Entries of a compressed standard form for example one, the different bands represent "interactions" between scales.

Figure 10.

Entries of compressed form of example 2.

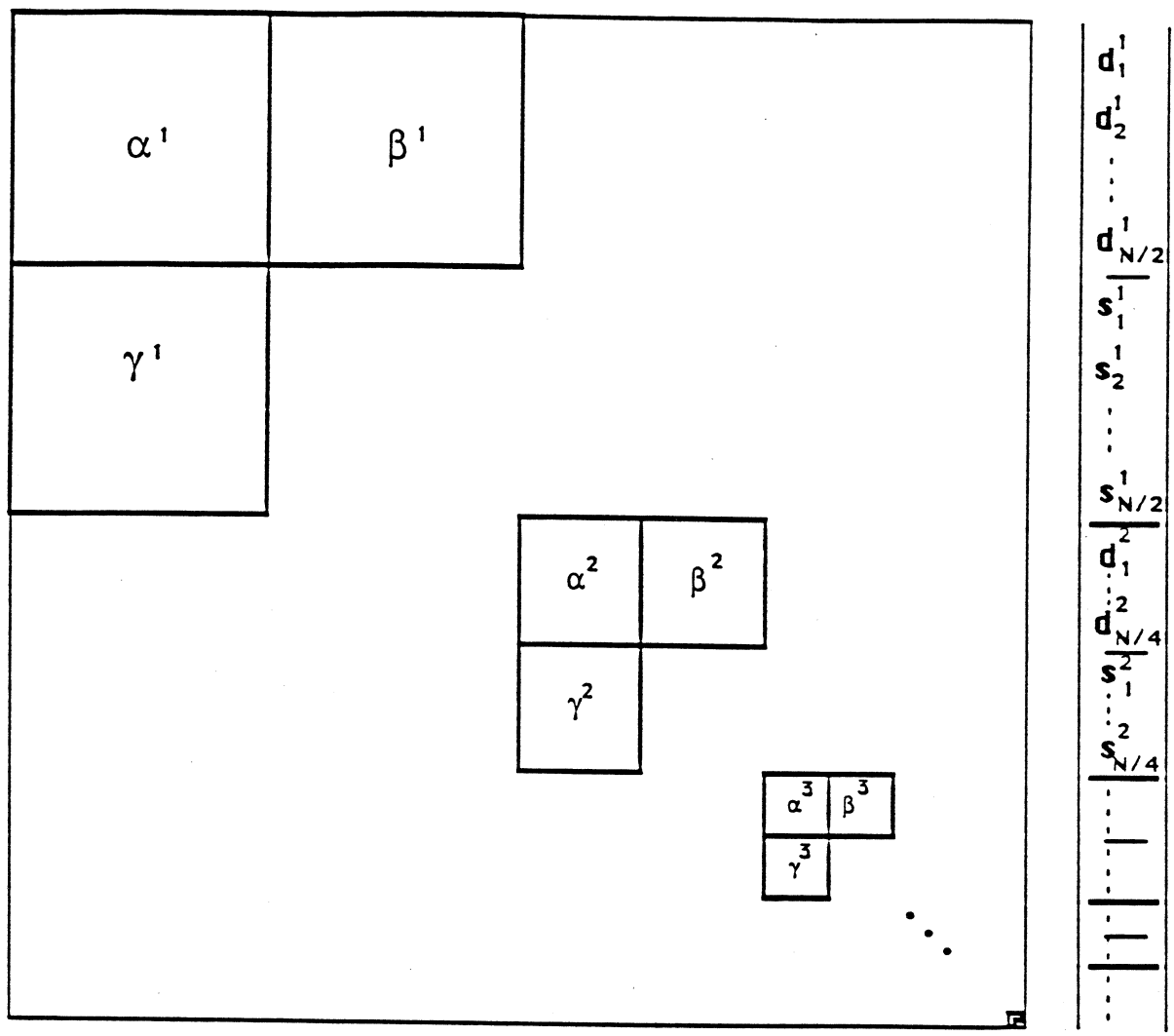

Figure 1

Representation of the decomposed matrix. Submatrices α, β, and γ on different scales are the only non-zero submatrices.

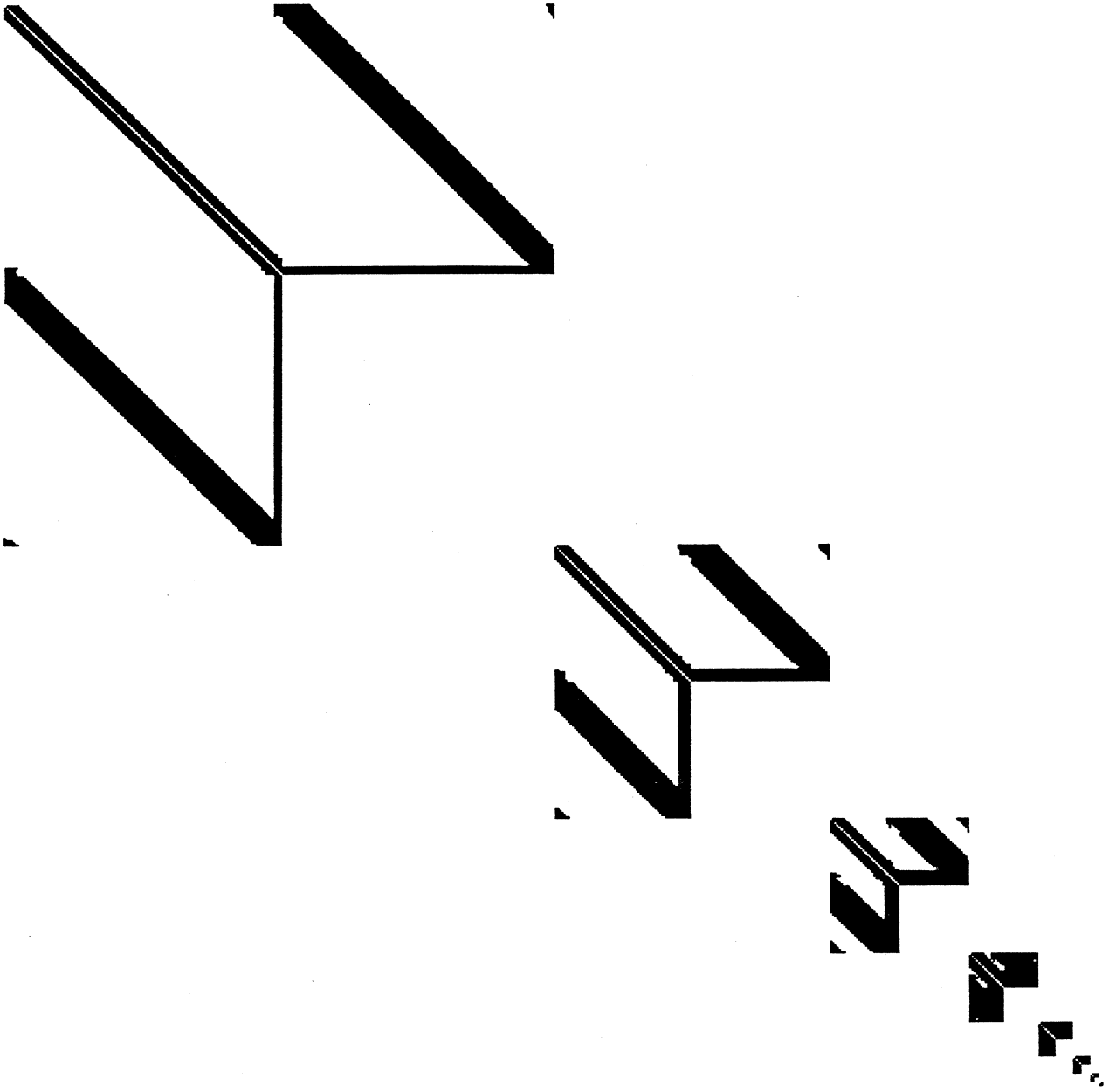

Figure 2

Entries of the decomposed matrix of Example 1 whose absolute values exceed the threshold 10^{-6} are shown in black. Note that the bandwidth does not grow with the size of the matrix.

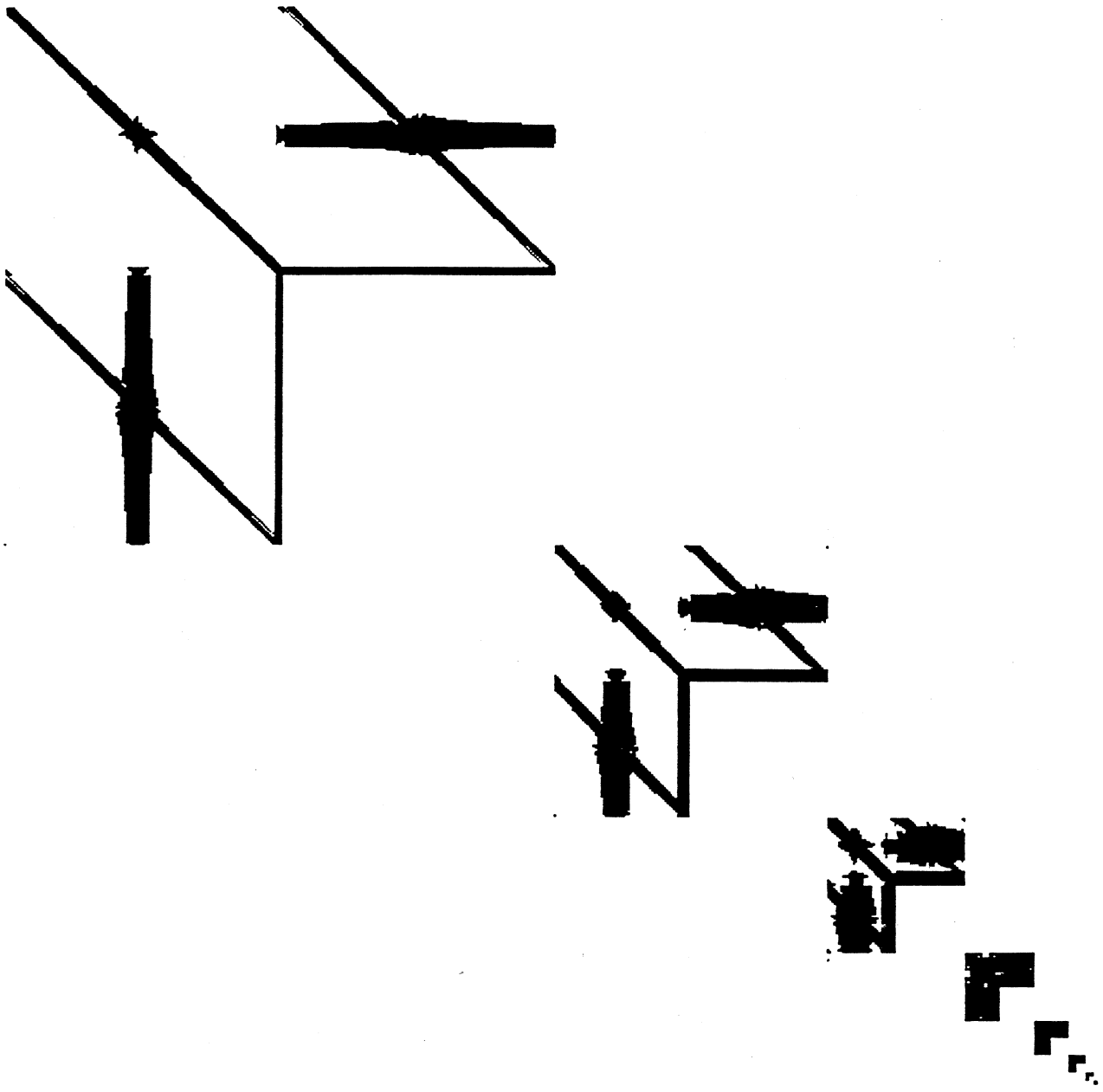

Figure 3

Entries of the decomposed matrix of Example 2 whose absolute values exceed the threshold 10^{-6} are shown in black. Vertical and horizontal bands in the submatrices (as well as the diagonal bands) are due to the singularities of the kernel. Note that in this case, the kernel is not a convolution.

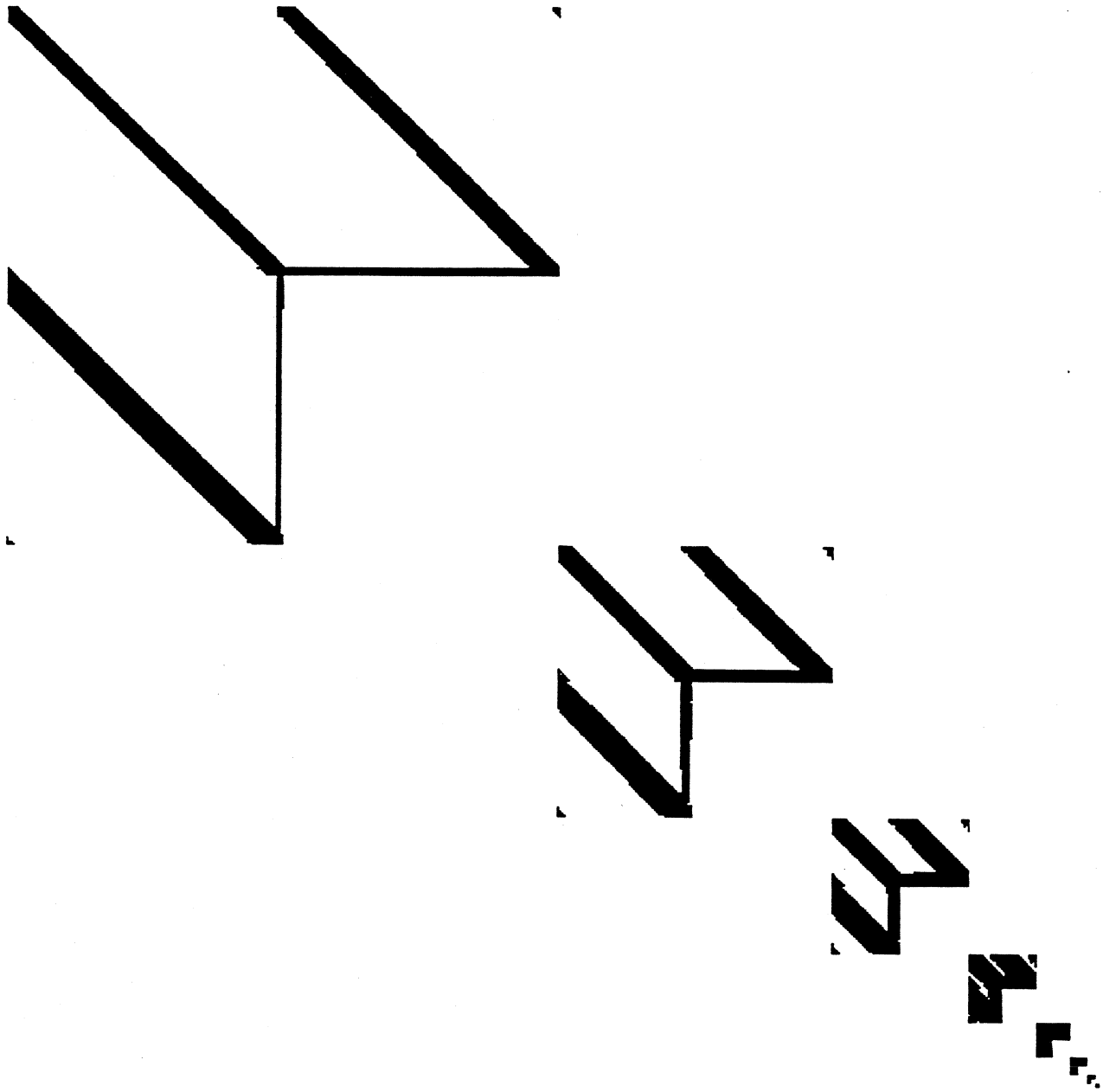

Figure 4

Entries of the decomposed matrix of Example 3 whose absolute values exceed the threshold 10^{-6} are shown in black. This is one of the two transition matrices connecting the coefficients of Legendre and Chebychev series.

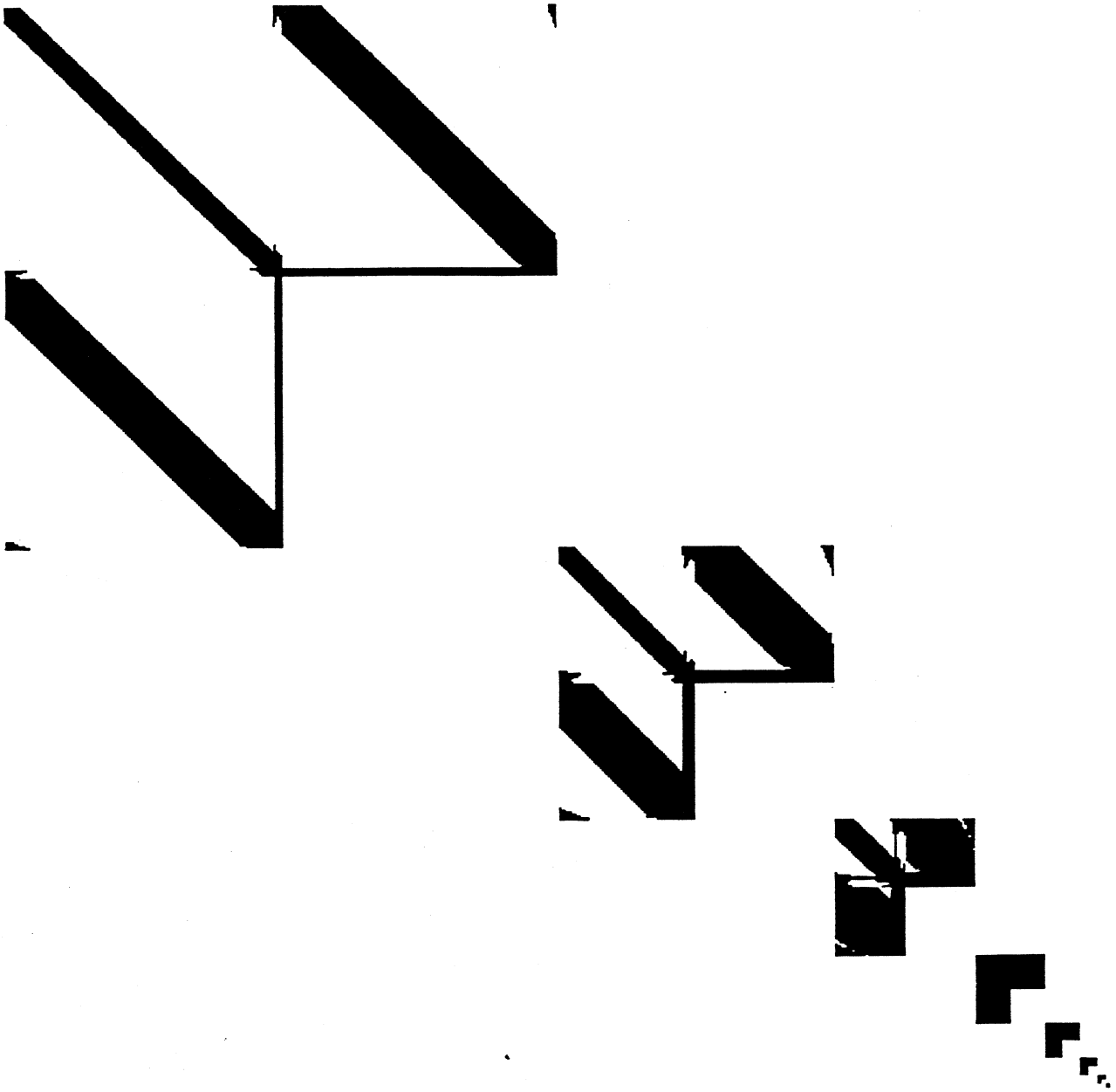

Figure 5

Entries of the decomposed matrix of Example 4 whose absolute values exceed the threshold 10^{-6} are shown in black.

```
LEVEL=1; DIFF #1; COLUMN #1
===========================
 0.41203e+00  0.32050e+00  -.97068e-01  0.25922e-01  -.39495e-02  0.15142e-03
 0.97874e-05  0.14130e-05  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  -.66485e-02  0.26841e-01
 -.97397e-01  0.32056e+00

---------------------
LEVEL=1; DIFF #2; COLUMN #1
===========================
 0.15734e+00  -.18626e+00  -.38867e-01  0.21651e+00  -.12664e+00  0.13501e-01
 0.18219e-02  0.48824e-03  0.17715e-03  0.77125e-04  0.38013e-04  0.20515e-04
 0.11868e-04  0.72555e-05  0.46391e-05  0.30787e-05  0.21085e-05  0.14834e-05
 0.10682e-05  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
 0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.13852e-03  -.14451e-02
 0.84839e-02  -.39957e-01
---------------------
---------------------
ABSOLUTE DEVIATION:
 0.61758e-05
L2-NORM DEVIATION:
 0.20595e-05
---------------------
```

Figure 6

Entries of the first column of the matrices α, β in Example 4 (fine scale). Observe the fast decay of the elements away from the diagonal (the threshold is 10^{-6}).

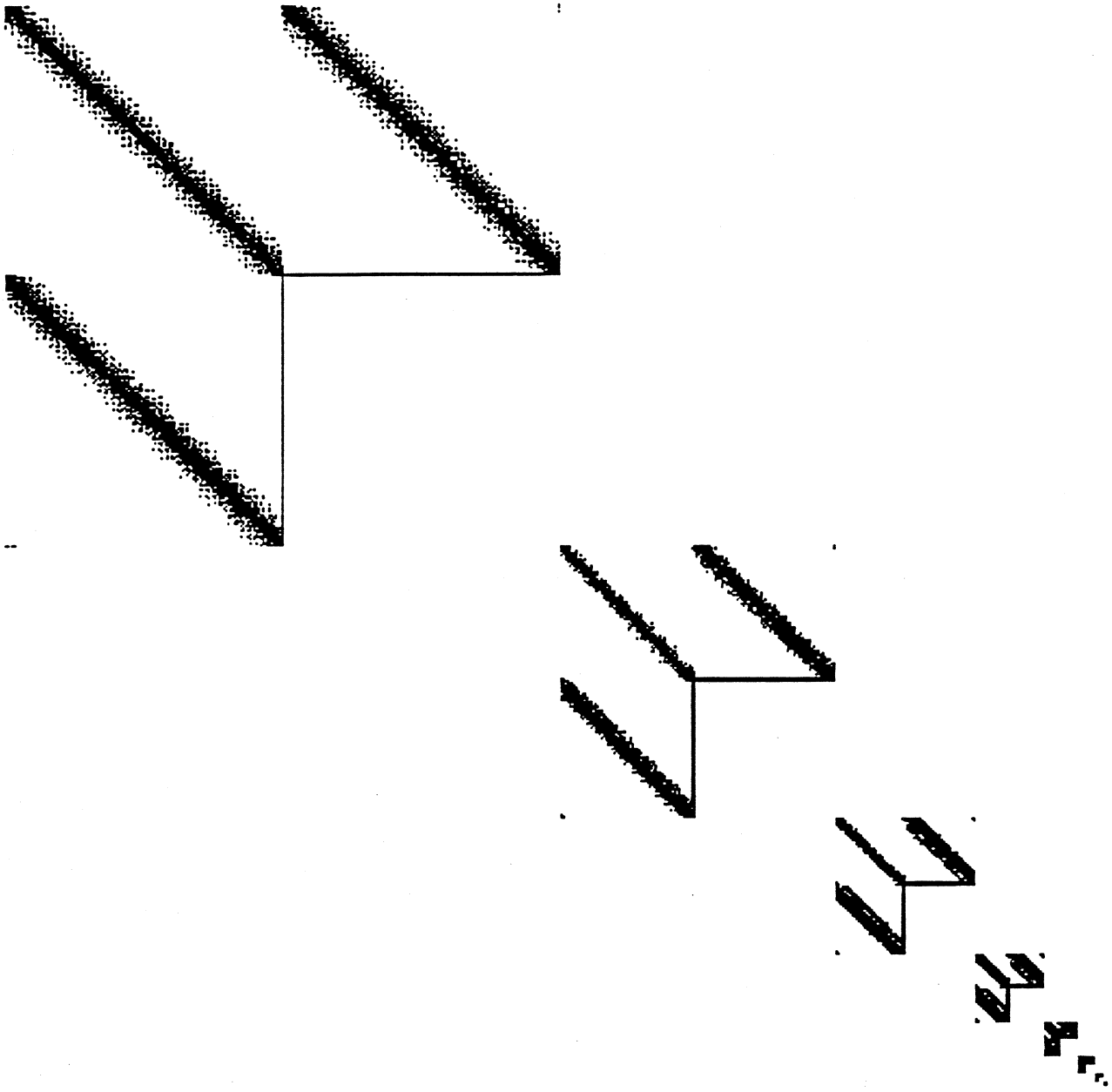

Figure 7

Entries of the decomposed matrix of Example 5 whose absolute values exceed the threshold 10^{-3}.

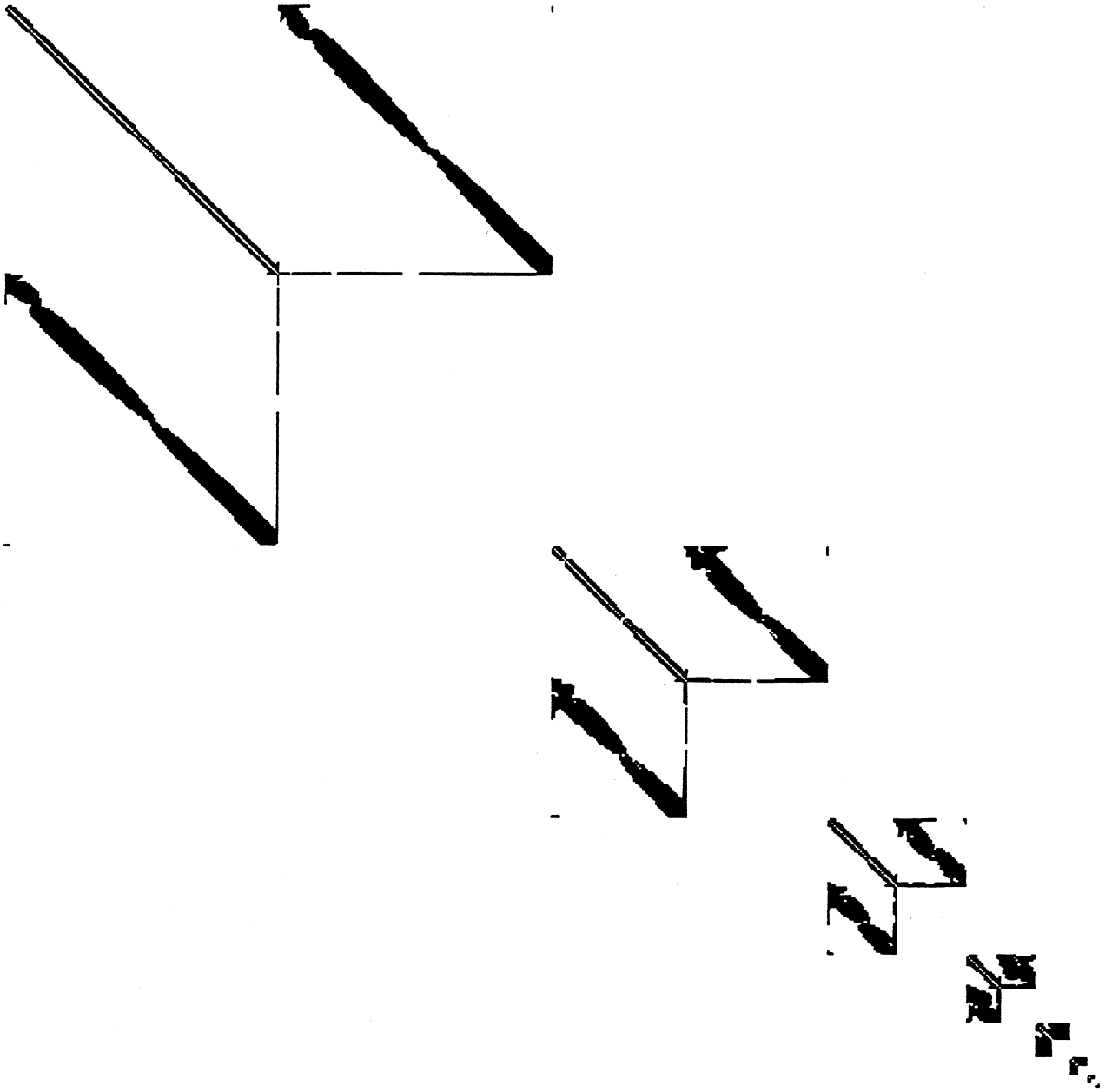

Figure 8

Entries of the decomposed matrix of Example 6 whose absolute values exceed the threshold 10^{-3}

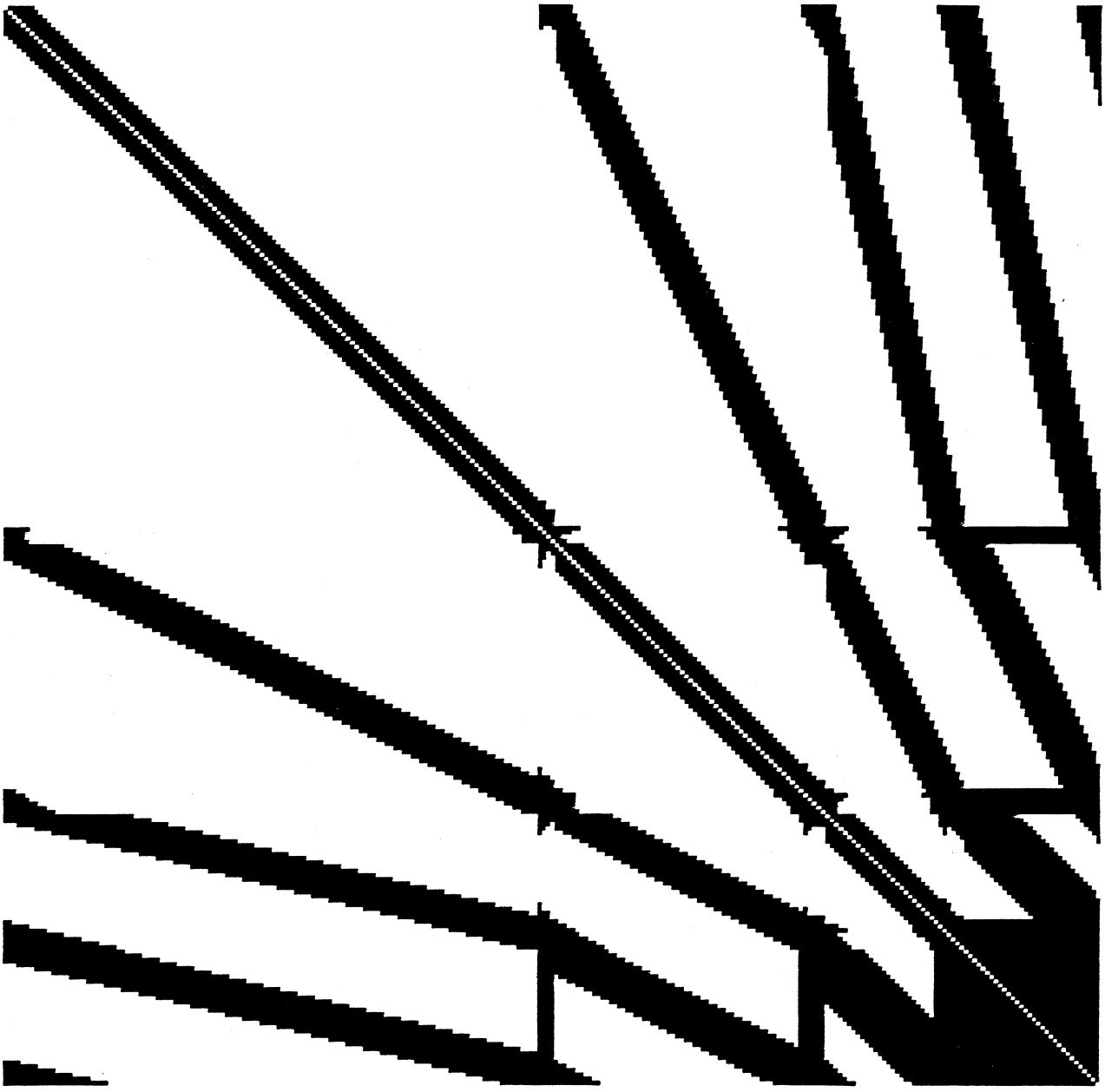

Figure 9

Entries of the compressed 'standard form' in Example 1. The different bands represent 'interaction' between different scales.

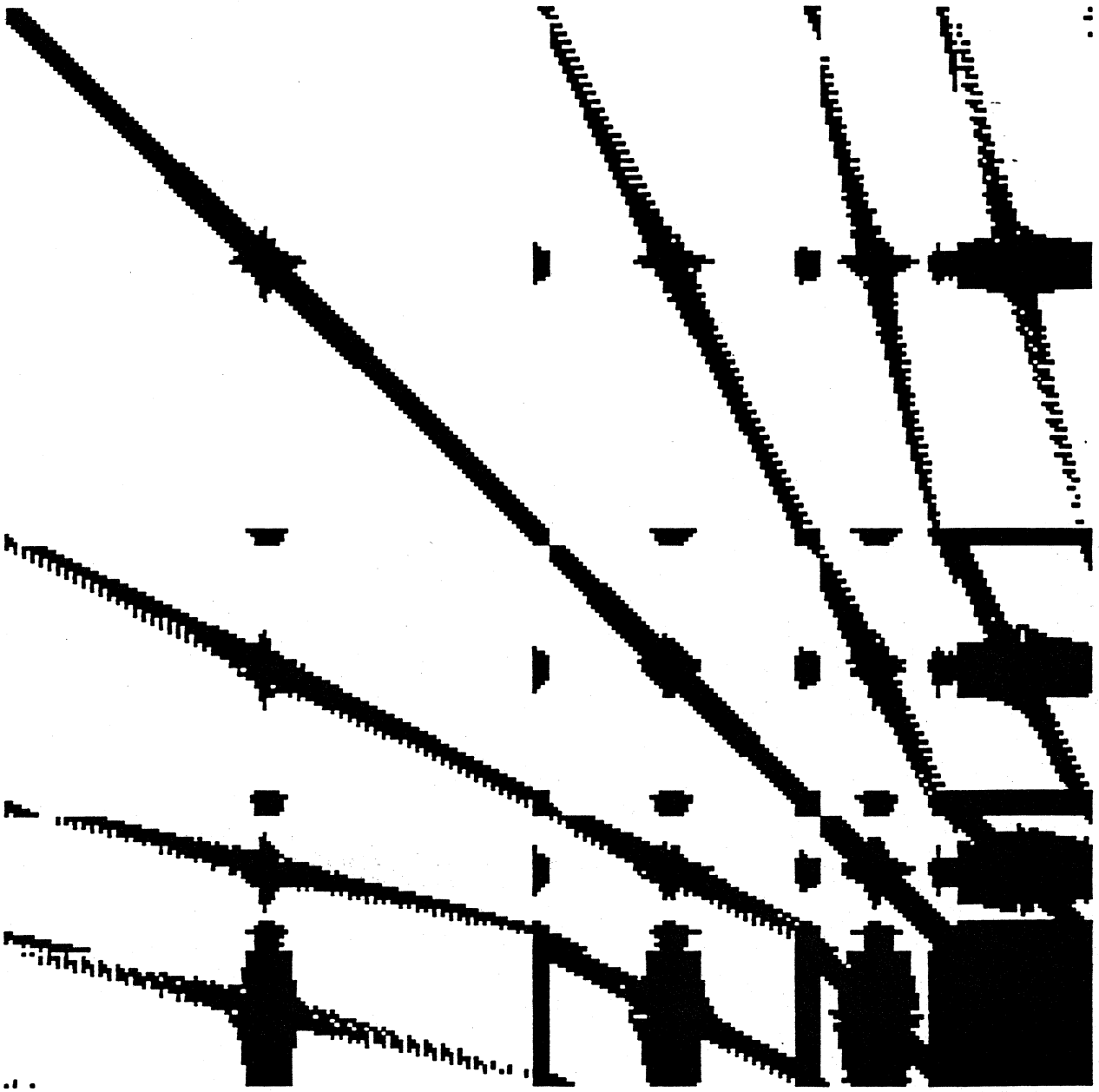

Figure 10

Entries of the compressed 'standard form' in Example 2.