APPLIED AND COMPUTATIONAL HARMONIC ANALYSIS **5,** 156–201 (1998)
ARTICLE NO. HA970227

LU Factorization of Non-standard Forms and Direct Multiresolution Solvers

For self-adjoint strictly elliptic operators the multiresolution LU factorization requires only $O((0\log e)^2 r N)$ operations. Combined with $O(N)$ procedures of multiresolution forward and back substitutions, it yields a fast direct multiresolution solver. We also describe direct methods for solving matrix equations and demonstrate how to construct the inverse in $O(N)$ operations (up to a fixed but arbitrary accuracy). We present several numerical examples which illustrate

if the factorized matrix does not admit the same sparse structure as the original matrix. Techniques such as graph theory have been developed to minimize the generation of so-called fill-ins [8]. Here we demonstrate that the specialized structure of NS-forms may be preserved during factorization, leading to efficient factorization algorithms using sparse data structures.

We begin by describing a factorization procedure for the NS-form. The factorization of the NS-form is superficially similar to the standard LU factorization but, in fact, is distinct in several significant ways. First, it is an approximate factorization where the accuracy, ε, is finite but arbitrary. Second, the factorization of NS-forms requires $O(N r(0 \log \varepsilon)^2)$ operations for operators arising from strictly elliptic problems. Combined with $O(N r(0 \log \varepsilon))$ procedures of ''multiscale'' forward and back substitutions, it yields a direct multiresolution solver. Third, the actual LU factorization is performed on well-conditioned matrices even if the original matrix (arising from a strictly elliptic problem) has a large condition number. Using the multiresolution solver, we also construct the inverse in $O(N(0 \log \varepsilon)^2))$ operations. We note that in problems where the choice of the size of the matrix and of accuracy are connected, typically $\varepsilon \} N^{0 a}$, a 0.

Our direct multiresolution solver presents an alternative to an iterative multigrid approach. In fact, the algorithms of this paper may be viewed as a ''direct multigrid,'' without V and W cycles (or, more precisely within this terminology, with a single V cycle). The absence of cycles of the usual full multigrid methods (for references see [9]) is easy to explain since we generate (within computational accuracy) a linear system for the exact projection of the solution on coarse scales. Once such a system is solved, there is no need to revisit that scale again. Thus, the algorithms of this paper provide a connection between multigrid methods and classical techniques of Gaussian elimination and LU decomposition. In this role, our approach provides a systematic algebraic structure to multiresolution computations (what we call multiresolution linear algebra), and we go to some length to provide details of such algebraic operations.

We recall that the non-standard form is not an ordinary matrix. The NS-form has a multiresolution structure, and the usual operations such as multiplication of a vector by the NS-form or multiplication of NS-forms are different from the standard matrix–vector and matrix–matrix multiplications. The remarkable feature of the non-standard form is the decoupling it achieves among the scales.

The outline of the paper is as follows: in Sections 2 and 3 we introduce multiresolution analysis and the notion of the non-standard form which serve as a foundation for the algorithms presented in the paper. We do so without specific reference to the properties of wavelets (e.g., number of vanishing moments, size of the support, etc). This allows us to describe the algebraic structure of the algorithms without considering specific bases. On the other hand, the sparsity of the non-standard form for a given accuracy, and thus the operation count of the algorithms, does depend on the choice of the basis. We discuss the issues of sparsity separately in Section 5.

In Section 4 we describe multiresolution LU factorization. In particular, we describe in Section 4.2 the procedure for computing lower and upper NS-forms where the NS-form of an operator has been precomputed. In Section 4.3 we describe how to construct the lower and upper NS-forms directly from the original operator, without precomput-

ing the NS-form; such a procedure is computationally more efficient. Finally, in Section 4.4, we present an alternative version of the factorization procedure which may be useful during partial pivoting (although pivoting is needed for matrices outside the class for which we prove that the multiresolution LU factorization is sparse. We discuss partial pivoting in the Appendix).

In Section 6 we show how the factorization procedure may be incorporated into a fast direct solver for linear systems of equations. Toward that end, we describe the algorithms of multiresolution forward and backward substitution. We then describe in Section 7S481the

instead of (2.1). In numerical realizations the subspace $\mathbf{V}_0$ is always of a finite dimension.

Let T be an operator

$$T: \mathbf{L}^2(\mathbf{R}^d) \to \mathbf{L}^2(\mathbf{R}^d). \tag{2.3}$$

By defining projection operators on the subspace $\mathbf{V}_j$, $j \in \mathbf{Z}$,

$$P_j: \mathbf{L}^2(\mathbf{R}^d) \to \mathbf{V}_j, \tag{2.4}$$

and expanding T in a ''telescopic'' series, we obtain

$$T = \sum_{j=0}^{n} (Q_j T Q_j + Q_j T P_j + P_j T Q_j) + P_n T P_n, \tag{2.5}$$

where $Q_j = P_{j-1} - P_j$ is the projection operator on the subspace $\mathbf{W}_j$,

$$Q_j: \mathbf{L}^2(\mathbf{R}^d) \to \mathbf{W}_j. \tag{2.6}$$

If the scale $j = 0$ is the finest scale, then

$$T_0 = \sum_{j=1}^{n} (Q_j T Q_j + Q_j T P_j + P_j T Q_j) + P_n T P_n, \tag{2.7}$$

where $T = T_0 = P_0 T P_0$ is a discretization of T on the finest scale. Expansions (2.5) and (2.7) decompose T into a sum of contributions from different scales.

2.1.1. The Non-standard Form

The NS-form introduced in [1] is a representation of an operator T as a chain of triplets $T = \{\{A_j, B_j, C_j\}_{0 \le j \le n}, T_n\}$, where operators A_j, B_j, C_j (as well as T_j) are defined as

$$A_j = Q_j T Q_j, \tag{2.8a}$$

$$B_j = Q_j T P_j, \tag{2.8b}$$

$$C_j = P_j T Q_j, \tag{2.8c}$$

$$T_j = P$$

FIG. 1. Organization of the non-standard form of a matrix. The submatrices A_j, B_j, and C_j, j 1, 2, 3, and T_3 are the only non-zero submatrices.

$$A_j: \mathbf{W}_j \text{ r } \mathbf{W}_j, \tag{2.10a}$$

$$B_j: \mathbf{V}_j \text{ r } \mathbf{W}_j, \tag{2.10b}$$

$$C_j: \mathbf{W}_j \text{ r } \mathbf{V}_j, \tag{2.10c}$$

whereas operators T_j operate on subspaces $\mathbf{V}_j$,

$$T_j: \mathbf{V}_j \text{ r } \mathbf{V}_j. \tag{2.11}$$

The wavelet transform recursively represents operators T_j as

$$\begin{pmatrix} A_{j/1} & B_{j/1} \\ C_{j/1} & T_{j/1} \end{pmatrix}, \tag{2.12}$$

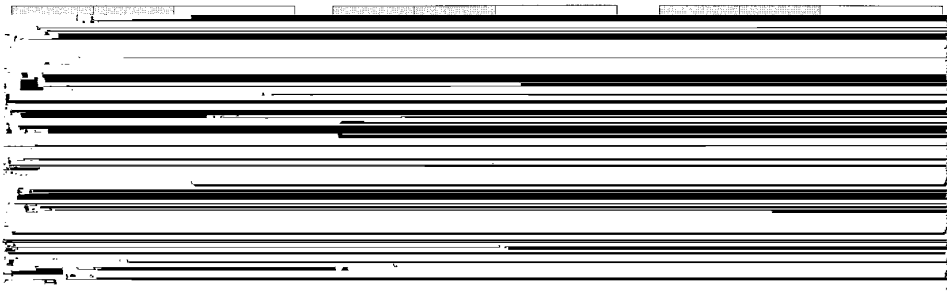
which is a mapping

$$\begin{pmatrix} A_{j/1} & B_{j/1} \\ C_{j/1} & T_{j/1} \end{pmatrix}: \mathbf{W}_{j/1} \; ! \; \mathbf{V}_{j/1} \text{ r } \mathbf{W}_{j/1} \; ! \; \mathbf{V}_{j/1}, \tag{2.13}$$

where $\mathbf{V}_j$ $\mathbf{W}_{j/1}$! $\mathbf{V}_{j/1}$. If the number of scales is finite, then we obtain T_0 $\{\{A_j, B_j, C_j\}_{1 \le j \le n}, T_n\}$, and the blocks of the NS-form are organized as blocks of a matrix shown in Fig. 1.

We note that for d ¢ 2 the blocks of the NS-form have additional structure. From now on we will assume that d 1, although many considerations are essentially the same for d ¢ 2. We will defer additional remarks about dimensions d ¢ 2 to Section 10.

Remark 2.1. Since projection operators involve subsampling, equalities like that in (2.9) may appear inconsistent if we compare sizes of blocks in Fig. 1. For example, the size of blocks A_3, B_3, C_3, and T_3 is not the same as that of T_2. The transition from operator notation as in (2.9) to a matrix notation involves combining the blocks A_3, B_3, C_3, and T_3 as in (2.12), and performing one step of the two-dimensional



$$f_0 \quad \sum_{j\ 1}^{n} Q_j f \ / \ P_n f, \tag{2.16}$$

and obtain the wavelet expansion $f_0 \quad \{\{d_j\}_{1 \,\mathrm{i}\, j \,\mathrm{i}\, n}, s_n\}$, where

$$d_j \quad Q_j f \ \hat{} \ \mathbf{W}_{\mathbf{j}}, \tag{2.17a}$$

$$s_j \quad P_j f \ \hat{} \ \mathbf{V}$$

3.1. Matrix Operations and Projections

The matrix operations at each scale involve blocks of the original NS-form *and* blocks obtained via projections. As an example, in the matrix–matrix multiplication of two NS-forms, $\hat{T}_0$ and $\tilde{T}_0$, we compute $A_j \quad \check{A}_j \;/\; A\!V_j$, where $\check{A}_j \quad \hat{A}_j\tilde{A}_j \;/\; \hat{B}_j\tilde{C}_j$ is computed from the original NS-forms, whereas $A\!V_j$ is a projection onto scale j, from scale $j \;0\; 1$. Relations of this type show how projections and matrix operations are combined at each scale. Such relations define the necessary algebraic operations for a given algorithm, and we will refer to them as the governing equations for that algorithm.

On each scale, the multiresolution algorithms produce intermediate matrices or vectors, some of which are projected to other scales. To describe the projection procedure, let us consider some intermediate vectors $\{\check{s}_j\}_{1 \,\mathrm{i}\, j \,\mathrm{i}\, n}$. We need to project all vectors $\check{s}_j$ on subspaces $\mathbf{W}_k$, $k \quad j, \ldots, n$. Instead of projecting each part of the vector separately, we combine contributions on a given scale before applying the projection operator. To illustrate, let us assume that projections have been completed up to scale k

be an extended wavelet representation for f_0. In what follows we define the *multiresolution product* of T_0 and f_0 as

$$g_0 \quad T_0 \mathsf{r} f_0, \tag{3.5}$$

where g_0 is the same vector obtained using matrix–vector multiplication with the usual representations of T_0 and f_0.

In order to derive the necessary matrix operations for (3.5), we write a telescopic series,

$$T_0 f_0 \;0\; T_n f_n \quad \sum_{j\;1}^{n} (P_{j\,0\,1} T_0 P_{j\,0\,1})(P_{j\,0\,1} f_0) \;0\; (P_j T_0 P_j)(P_j f_0). \tag{3.6}$$

Since $P_{j\,0\,1} \quad P_j \;/\; Q_j$ we obtain

$$T_0 f_0 \;0\; T_n f_n \quad \sum_{j\;1}^{n} (Q_j T_0 Q_j)(Q_j f_0) \;/\; (Q_j T_0 P_j)(P_j f_0) \;/\; (P_j T_0 Q_j)(Q_j f_0), \tag{3.7}$$

or, using (2.8) and (2.17),

$$T_0 f_0 \quad \sum_{j\;1}^{n} (A_j d\!H_j \;/\; B_j s\!I_j \;/\; C_j d\!H_j)$$

$$s_{j-1} = s_{V_{j-1}} \quad d_{U_j} = s_{V_j} \tag{3.13}$$

via (3.2), and by forming the sum

$$d_j \quad d_{Z_j} = d_{U_j}. \tag{3.14}$$

At the last scale we compute

$$s_n \quad s_{a_n} = s_{V_n}. \tag{3.15}$$

Equations (3.14) and (3.15) are the governing equations for multiresolution matrix vector multiplication, while (3.13) is the corresponding projection equation. The following algorithm summarizes the process:

ALGORITHM 3.1 (Application of NS-Forms to Vectors). *Given the* NS-*form* T_0 $\{\{A_j, B_j, C_j\}_{1 \le j \le n}, T_n\}$, *and the extended wavelet representation* f_0 $\{d_{H_j}, \tilde{s}_j\}_{1 \le j \le n}$, *we compute the multiresolution product* $T_0 f_0$ b_0 *where* b_0 $\{\{d_j\}_{1 \le j \le n}, s_n\}$ *using the following steps:*

1. *Initialization: set* d_{U_j} s_{V_0} 0.
2. *For* j 1, 2, . . . , n, *compute*
   (a) d_{Z_j} *and* $\check{s}_j$ *via* (3.11),
   (b) d_{U_j} *and* s_{V_j} $(j - 1)$ *via* (3.13),
   (c) d_j *via* (3.14).
3. *At the last scale compute* s_n *via* (3.15).

In the following algorithm we use operators $\hat{T}_j$ and $\tilde{T}_j$. We note that the algorithm requires only a band around the diagonal of their matrices, even though these operators are generally not sparse. We will consider sparsity of the blocks of NS-forms separately and here present a formal derivation of the algorithm for multiplication of NS-forms following [4].

In order to derive the necessary matrix operations, we again write a telescopic series,

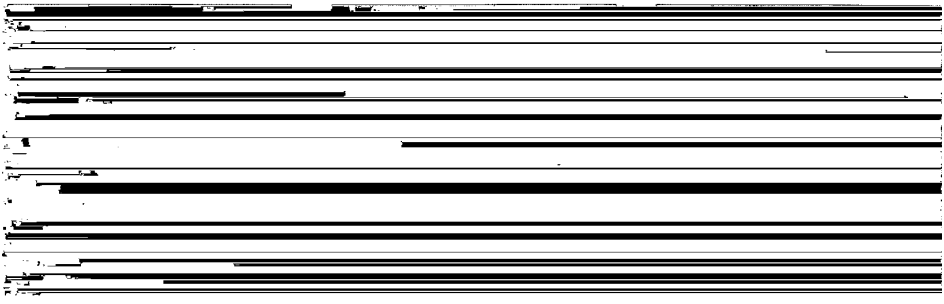


FIG. 3. Organization of the multiresolution LU factorization.

4. MULTIRESOLUTION LU FACTORIZATION

We now describe the main tool in our approach, LU factorization with respect to the multiresolution product (r) introduced in Section 3.3. Given an NS-form T_0 $\{\{A_j, B_j, C_j\}_{1 \le j \le n}, T_n\}$, our goal is to compute the NS-forms $\hat{T}_0$ and $\tilde{T}_0$, such that

$$T_0 \quad \hat{T}_0 \, r \, \tilde{T}_0, \tag{4.1}$$

where $\hat{T}_0$ and $\tilde{T}_0$ are analogous to lower and upper triangular matrices. In defining lower and upper triangular NS-forms, we require that $\hat{B}_j \quad \tilde{C}_j \quad 0$ for $j \quad 1, 2, \ldots, n$; that blocks $\{\{\hat{A}_j\}_{1 \le j \le n}, \hat{T}_n\}$ are lower triangular; and that blocks $\{\{\tilde{A}_j\}_{1 \le j \le n}, \tilde{T}_n\}$ are upper triangular, as shown in Fig. 3. We call $\hat{T}_0 \quad \{\{\hat{A}_j, \hat{C}_j\}_{1 \le j \le n}, \hat{T}$

for $j = 1, 2, \ldots, n$, and from (3.22) and (3.25),

$$T_n = C_n B_n / T_n T_n / T_n. \tag{4.3}$$

The operators A_j, B_j, C_j are computed by first setting $A_1 = B_1 = C_1 = T_1 = 0$, and then, for $j = 1, 2, \ldots, n$, computing the projections as in (3.4),

$$C_{j-1} B_{j-1} / T_{j-1} = A_j / B_j / C_j / T_j. \tag{4.4}$$

4.2. Factorization

Let us now assume that T_0 is given and obtain a recurrence relation that permits us to compute the lower and upper NS-forms $\hat{T}_0$ and $\tilde{T}_0$. According to (4.2) and (4.3), the blocks of the NS-forms $\hat{T}_0$ and $\tilde{T}_0$ satisfy the relations

$$A_j A_j = A_j \; 0 \; A_j, \tag{4.5a}$$

$$A_j B_j = B_j \; 0 \; B_j, \tag{4.5b}$$

$$C_j A_j = C_j \; 0 \; C_j, \tag{4.5c}$$

on all scales $j = 1, 2, \ldots, n$, and on the last scale,

$$T_n T_n = T_n \; 0 \; T_n \; 0 \; C_n B_n. \tag{4.6}$$

Equations (4.5) and (4.6) are the governing equations for multiresolution LU factorization, and (4.4) is the corresponding projection equation. Comparing (4.5) and (4.6) to (4.2) and (4.3) we note that for the purposes of finding $\hat{A}_j$,

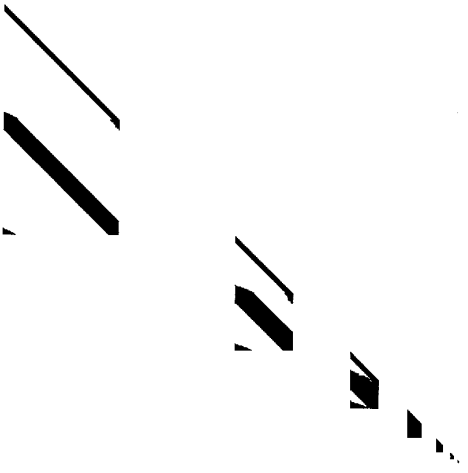


FIG. 4. The lower NS-form for Example 1 of Section 9. All entries whose absolute values are larger than 10^{-07} are shown in black.

To proceed, we compute the LU factorization at scale k. We note that the governing equations for this procedure are the same as those in Section (4.2), and thus, factoriza-

Let us organize the blocks $(A_j \ 0 \ A_j^V)$, $(B_j \ 0 \ B_j^V)$, $(C_j \ 0 \ C_j^V)$, and $(T_j \ 0 \ T_j^V)$, as in (2.12)), and compute a partial LU factorization,

$$\begin{bmatrix} A_j & 0 \\ C_j & I \end{bmatrix}$$

of the matrix and the selection of accuracy are connected and, thus, c cannot be considered a constant. We prefer not to mix the choices of size and accuracy in our estimates so as to encompass a wider range of applications, and this remark should be sufficient to avoid any confusion.

Finally, we note that in this paper we provide only an outline of the proofs and refer to a companion paper [13] for additional details.

5.1. Compression of Operators

The compression of operators, or, in other words, the construction of their sparse representations in orthonormal bases, has been proposed in [1]. The standard and non-standard forms of operators described in [1] may be viewed as compression schemes for a wide class of operators frequently encountered in analysis and applications, namely, Calderón–Zygmund and pseudo-differential operators.

Let us briefly state the results of [1]. Although in what follows we consider matrix representations of these operators, we note that any appropriate discretization procedure may be used, such as the Nyström method or the method of moments. In such cases the wavelet transform is simply a linear algebra tool.

Given an NS-form T_0, the operators A_j, B_j, C_j, T_j are represented by the matrices a^j, b^j, g^j, s^j, where

$$\mathrm{a}^j_{k,k'} = \int\int K(x, y)\, \mathrm{c}_{j,k}(x)\, \mathrm{c}_{j,k'}(y)\, dxdy, \tag{5.1a}$$

$$\mathrm{b}^j_{k,k'} = \int\int K(x, y)\, \mathrm{c}_{j,k}(x)\, \mathrm{f}_{j,k'}(y)\, dxdy, \tag{5.1b}$$

$$\mathrm{g}^j_{k,k'}$$

The right-hand side of (5.3) is small whenever either I

FIG. 5. A matrix representing the NS-form of the matrix of Example 1. All entries whose absolute values are larger than 10^{07} are shown in black.

$$\|T_0^{N,B} - T_0^N\| \le \frac{C}{B^M} \log_2 N, \tag{5.12}$$

where C is a constant determined by the kernel K. In most numerical applications, the accuracy ε of calculations is fixed, and the parameters of the algorithm (in our case, the bandwidth 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 $\|T_0^{N,B} - T_0^N\| \le C/B^M \log_2 N \le \varepsilon$, or, equivalently, $B \ge (C/\varepsilon \log_2 N)^{1/M}$.

The estimate (5.12) is sufficient for practical purposes. It is possible, however, to obtain

$$\|T_0^{N,B} - T_0^N\| \le \frac{C}{B^M} \tag{5.13}$$

instead of (5.12) (see [1]).

Finally we note that strictly elliptic operators and their Green's functions are compressible in the wavelet bases and the decay of the elements of the blocks of the nonstandard forms away from the diagonal may be controlled by choosing appropriate number of vanishing moments of the wavelet.

As an illustration, we display in Fig. 5 the NS-form of the matrix in Example 1 of Section 9. Using periodized wavelets with 6 vanishing moments and setting to zero all entries whose absolute values are smaller than 10^{07}, we display the remaining

The condition number $k \quad \|T^{-1}\| \, \|T\|$ may be viewed as an amplification factor for the relative error. Thus, (5.14) implies that if thresholding has been performed using $\|dT\| \leq \varepsilon \|T\|$, then the relative error of the solution will not exceed $k\varepsilon$.

Let us denote by $\hat{T}_\varepsilon$ and $\tilde{T}_\varepsilon$ approximations to $\hat{T}$ and $\tilde{T}$ obtained by setting all entries that are less than ε to zero, and assuming (without a loss of generality) $\|\hat{T}\| \quad \|\tilde{T}\| \quad 1$. We obtain $\|\hat{T} - \hat{T}_\varepsilon\| \leq \varepsilon$, $\|\tilde{T} - \tilde{T}_\varepsilon\| \leq \varepsilon$, and therefore, $\|\hat{T}\tilde{T} - (\hat{T}_\varepsilon\tilde{T}_\varepsilon)_\varepsilon\| \leq \varepsilon / \varepsilon(1 / \varepsilon) / \varepsilon(1 / \varepsilon)^2$. The right side is dominated by 3ε. Thus, for truncating the factors $\hat{T}$ and $\tilde{T}$, we use a threshold which is one-third of the threshold used for T.

5.2. Compression of Lower and Upper NS-Forms

In Section 4.4 we introduced the recursively defined reduced operator

$$R_j \quad T_j - (TU$$

From Theorem 5.3 it follows that entries of the blocks A_{R_j}, B_{R_j}, and C_{R_j} have the same rate of decay as blocks of the original NS-form. Similar to Theorem 5.1, Theorem 5.3 does not give sharp estimates for the constants. We provide numerical examples in Section 9 to show that the constant in the decay estimate in Theorem 5.3 is not significantly different from that in Theorem 5.1 since the sparsity (after applying accuracy cutoff) of the multiresolution LU factors is almost the same as that of the original NS-form.

In order to prove that all multiresolution LU factors are sparse, we note that both A_{R_j} and $A_{R_j}^{01}$ have a fast rate of decay away from the diagonal as stated in theorems of this section. Let us now select an ε, the desired accuracy, and truncate both A_{R_j} and $A_{R_j}^{01}$ independently so that the error (in the operator norm) is ε. Using results in [16] (i.e., Proposition 2.1 therein), we observe that the banded bi-infinite matrix A^{01} has banded Cholesky factors (or LU factors). Since computing B_{R_j} and C_{R_j} involves the product of two banded matrices, we conclude that all blocks of NS-forms in the multiresolution LU factorization are banded for a given accuracy ε.

We note that in actual computations truncation is performed by restricting computations to a band (which is selected to accommodate all the entries up to a certain size) and that matrices are finite rather than bi-infinite.

Combining previous results, we obtain

THEOREM 5.5. *Let us assume that the operator T and the wavelet basis satisfy conditions of Theorems* 5.1 *and* 5.3. *Let T_0 be the projection of T on the subspace $\mathbf{V}_0$.*

For such operators the NS*-form of T_0, T*

M is the number of vanishing moments of the wavelet basis). Thus, coefficients of the scaling function at coarser scales may be computed without the wavelet transform using a quadrature formula. The simplest (one-point) quadrature formula is obtained if we require $f(x)$ to have M 0 1 (shifted) vanishing moments. Using such wavelets, the coefficients $s_{i,l}$ at any scale may be approximated by

$$s^{j}_{i,l} \quad 2^{j} s^{0}_{2^{j}(i\,0\,1\,/\,t)\,0\,t\,/\,1,2^{j}(l\,0\,1\,/\,t)\,0\,t\,/\,1} \tag{5.20}$$

where s^0 is the original matrix and t is the shift parameter,

$$t \quad \int_{0}^{\infty} f(x)x\,dx. \tag{5.21}$$

The shift parameter t may be chosen to be an integer. We note, however, that for any wavelet basis a quadrature formula (with M terms) may be constructed and used instead of (5.20) (see [1]).

We describe now how to use quadrature formulas in constructing the banded NS-form. Let us begin by filling the matrix for T_0 within a band of width $2w$. We then compute the banded approximate operators A_1, B_1, C_1, T_1 using the wavelet transform. We note that

1. *Compute entries of* T_0 *within a band of width* $2w$.
2. *For* $j = 1, 2, \ldots, n$,
   (a) *compute* A_j, B_j, C_j, *and* T_j *via* (5.19),
   (b) *extend the band of* T_j *to* $2w$ *via* (5.20).

Computational cost. Step 1 is computed in $O(N)$ steps. The cost of Step 2a is $O(wN_j)$, where N_j is the size of $T_{j\,0\,1}$. The cost of Step 2b is $O(wN_j)$. The total computational cost is $O((0 \log e)N)$ since the bandwidth w is proportional to $0 \log e$.

5.4. Fast Reconstruction of NS-Forms

Let us now show how to *reconstruct* a banded version of the operators T_j from an

$$A_j \; 0 \; A_j J \quad L_j U_j, \tag{5.22}$$

and set $\hat{A}_j \quad L_j$ and $\tilde{A}_j \quad U_j$. If the matrix representing $A_j \; 0 \; A_j Y$ is sparse, then the factors in (5.22) are computed using sparse LU factorization, where computations are

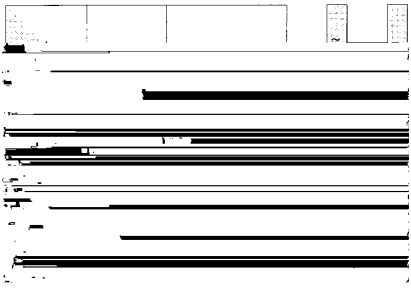


FIG. 6. Organization of multiresolution forward substitution.

Computational cost. Step 2 is computed in $O(wN)$ steps. The cost of Step 3a is $O(wN)$. Step 3b is $O(wN)$. Steps 3c and 3d are $O(w^2N)$. Step 3e is $O(w^2N)$. The total cost of this procedure is $O((0\log e)^2N)$ since the bandwidth is proportional to $0\log e$.

6. SOLUTIONS OF LINEAR ALGEBRAIC EQUATIONS

In this section we combine the results of previous sections and describe a *direct multiresolution solver* for operators described in Section 5. We proceed along the usual lines of using LU factorization for this purpose.

on all scales $j = 1, 2, \ldots, n$, and on the last scale

$$T_n s_n = s_n \oplus s_n \oplus C_n d_n. \tag{6.3}$$

The terms d_j, s_j are computed via the projection equation obtained from (3.13)

$$C_{j-1} d_{j-1} \ominus s_{j-1} = d_j \ominus s_j. \tag{6.4}$$

At a given scale k, Eq. (6.2) is satisfied by first computing the projection of vector $C_{k-1} d_{k-1} \ominus s_{k-1}$ on scale k to obtain d_k and s_k as in (6.4) and then solving (6.2) for d_k using standard forward substitution. On the final scale we compute $\tilde{s}_n$ by solving (6.3).

ALGORITHM 6.1 (Multiresolution Forward Substitution). *Given the lower* NS---

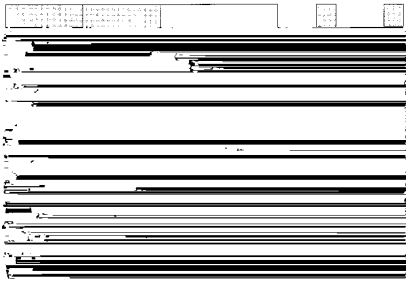


FIG. 7. Organization of multiresolution backward substitution.

on all scales $j = 1, 2, \ldots, n$, and on the last scale,

$$T_n s_n = s_n. \tag{6.6}$$

We note that no projections enter into Eqs. (6.5) and (6.6), since $C_j = 0$ in (3.11b) implies that $\tilde{s}_j = sV = 0$. We show, however, that projections from coarser to finer scales (i.e., reconstructions) will be required throughout the algorithm.

To demonstrate this, let us begin on the coarsest scale, $j = n$, and solve (6.6) for $\tilde{s}_n$ using the usual backward substitution. This completes the procedure for $j = n$. We now assume that Eq. (6.5) has been satisfied for $j = n, \ldots, k + 1$, and *reconstruct* $\tilde{s}_k$,

$$s_k = d_{k+1} + s_{k+1}, \tag{6.7}$$

using the inverse wavelet transform (see Remark 2.3). Next, we compute d_{n+1} by solving (6.5) using the usual backward substitution. We have

ALGORITHM 6.2. (Multiresolution Backward Substitution). *Given the upper* NS-*form* $T_0 = \{\{A_j, B_j\}_{1 \le j \le n}, T_n\}$

for general NS-forms $T_0 \quad \{\{A_j, B_j, C_j\}_{1 \le j \le n} T_n\}$. The procedure is analogous to direct methods based on the usual LU factorization. The only difference is that the product in (6.8) refers to the multiresolution product defined in Section (3.2).

We begin by computing the multiresolution LU factorization $\hat{T}_0 \mathbin{\mathsf{r}} \tilde{T}_0 \quad T_0$ as outlined in Sections 4.2, 4.3, and 4.4. We proceed to solve the system $\hat{T}_0$

FIG. 8. Organization of multiresolution forward substitution for matrices.

$$L_j U_j \quad T_j \; 0 \; C_j B_j \; 0 \; T_j, \tag{7.3}$$

where $L_j \quad \hat{T}_j$ and $U_j \quad \tilde{T}_j$.

Remark 7.1. We note that this procedure requires the multiplication of $\hat{C}_j$ and $\tilde{B}_j$, and the computation of T_j via (4.4). These computations are a part of the LU factorization of T_0 (see, e.g., Step 2a of Algorithm 4.1). The results may be stored as $T_j \; 0 \; \hat{C}_j \tilde{B}_j \; 0 \; T_j$ at each scale j during LU factorization.

ALGORITHM 7.1. (LU Factorization of Blocks T_j). *Given the extended* NS-*form* $T_0 \quad \{A_j, B_j, C_j, T_j\}_{1 \le j \le n}$, and its LU *factors* $\hat{T}_0 \quad \{\{\hat{A}_j, \hat{C}_j\}_{1 \le j \le n}, \hat{T}_n\}$ *and* $\tilde{T}_0 \quad \{\{\tilde{A}_j, \tilde{B}_j,\}_{1 \le j \le n}, \tilde{T}_n\}$, *the lower and upper triangular factors* $\hat{T}_j$ *and* $\tilde{T}_j$ *may be computed using the following steps:*

1. *Initialization: set* $T_j \quad 0$.
2. *For* $j \quad 1, 2, \ldots, n$ *compute*
   (a) $T_j \; (j \; \text{x} \; 1)$ *via* (4.4),
   (b) $\hat{T}_j$ *and* $\tilde{T}_j$ *via* (7.3).

Computational cost. We note that operations in this algorithm are performed on dense matrices. However, since we require only a banded version of the operators, we may perform computations in a fast manner. Step 2a requires the projection of a banded matrix on a coarser scale and may be computed in $O(wN)$ steps using the fast algorithm described in Section 5.3. We also require the matrix multiplication of C_j and B_j which requires $O(w^2N)$ operations, where w is the matrix bandwidth. In Step 2b we compute the *LU* factorization of a banded operator, which requires $O(w^2N)$ operations. The total computational cost is $O((\; 0 \log \; \text{e})^2$

$$A_j Q A_j^H$$

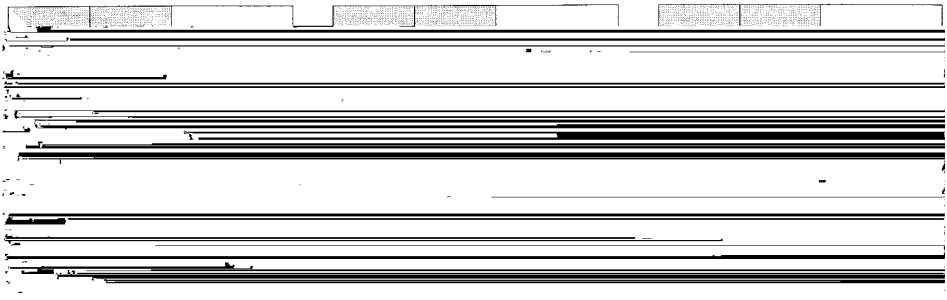


FIG. 9. Organization of multiresolution backward substitution for matrices.

$$A_j Q_j A_{j+1} \quad A_j \; 0 \; B_j Q_j C_{j+1}, \tag{7.7a}$$

$$A_j Q_j B$$

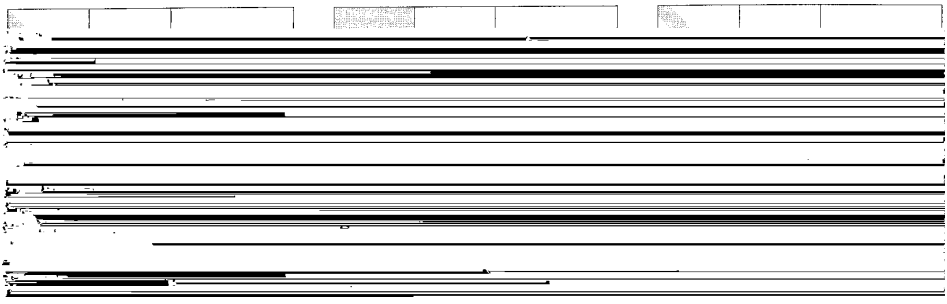


FIG. 10. Organization of multiresolution forward substitution for finding the inverse.

7.4. Multiresolution Direct Solver for Matrix Equations

We now consider the solution to matrix equations of the type in (7.1) for general NS-forms T_0 $\{\{A_j, B_j, C_j\}_{1 \le j \le n} T_n\}$. This procedure is completely analogous to the multiresolution direct methods developed for linear systems in Section 6.3.

We begin by computing the multiresolution LU factorization $\hat{T}_0 \mathrm{r} \tilde{T}_0$ T_0 as outlined in Sections 4.2, 4.3, and 4.4. In addition, we store the banded versions of the blocks $\hat{T}_j$ and $\tilde{T}_j$ as described in Section 7.1. We then solve the system $\hat{T}_0 \mathrm{r} Y_0$ B_0 for Y_0 using multiresolution forward substitution for matrices, as described in Section 7.2. Given Y_0, we may obtain X_0 by solving $\tilde{T}_0 \mathrm{r} X_0$ Y_0 using multiresolution backward substitution for matrices, as described in Section 7.3.

Each of these algorithms is $O((0 \log \mathrm{e})^2 N)$, where operators satisfy the conditions in Section 5.2. We thus have a direct, $O((0 \log \mathrm{e})^2 N)$ method for computing solutions to matrix equations.

7.5. Computing the Inverse Operator

Let us describe the algorithm to compute the inverse operator in greater detail. Given the NS-form $\hat{T}_0$ $\{\{\hat{A}_j, \hat{B}_j, \hat{C}_j\}_{1 \le j \le n} \hat{T}_n\}$, we seek to find the inverse NS-form T_0^{01} $\{\{\tilde{A}_j, \tilde{B}_j, C$

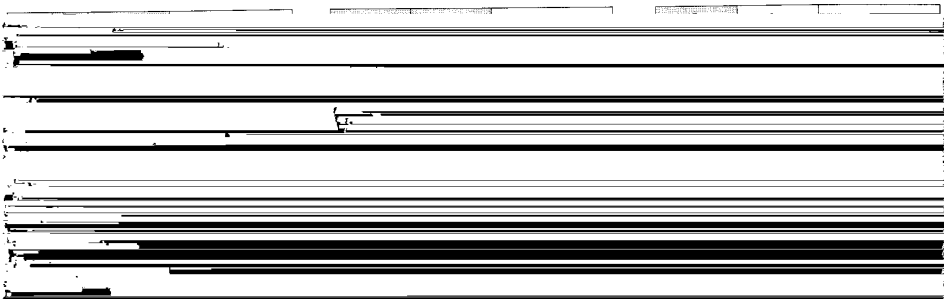


FIG. 11. Organization of multiresolution backward substitution for finding the inverse.

. . . , n, and no projections need be computed. Thus, operations on each scale may be performed independently. From (7.4) and (7.5) we obtain the simplified equations

$$A_j Q_j A_j^H = I_j, \tag{7.11a}$$

$$T_j Q_j C_j^H$$

S-forms may be obtained by first constructing the NS-form and then converting it to an S-form using the following algorithm:

ALGORITHM 8.1. (Computing the S-Form from the NS-Form). *Given the* NS-*form* $T_0 \quad \{\{A_j, B_j, C_j\}_{1 \le j \le n}, T_n\}$, *the* S-*form may be obtained at each scale j using the following steps:*

1. *Recursively apply the wavelet transform to each row in* B_j *for* $k \quad j, j / 1, \ldots, n$.
2. *Recursively apply the wavelet transform to each column in* C_j *for* $k \quad j, j / 1, \ldots, n$.
3. *Place the blocks* $\{A_j, B_j, C_j\}$ *in the space occupied by* $T_{j\,0\,1}$. (*This step returns the system to its original dimension.*)

We now use Algorithm 8.1 to construct the S-form of lower and upper NS-forms. We have

THEOREM 8.1. *Let* $T_0 \quad \hat{T}_0 \mathrm{r} \tilde{T}_0$ *be the multiresolution LU factorization of the NS-form for* T_0, *and let* $T_0 \quad LU$ *be the standard LU factorization of the S-form for* T_0. *Then* $\hat{T}_0$ *is the NS-form of L, and* $\tilde{T}_0$ *is the NS-form of U. Therefore, sparsity of lower and upper NS-forms implies sparsity of lower and upper factors of the standard form.*

Remark 8.1. We note that since S-form representations also admit a sparse structure, standard LU factorization may be used to compute the factors L and U. However, such approach is less efficient than that of using the NS-forms. The loss of efficiency may be clearly seen from Theorem 8.1. Single bands of $\hat{C}_j$ and $\tilde{B}_j$ are expanded into several bands to account for interaction between scales, thus reducing the sparsity of corresponding matrices by a significant factor.

9. NUMERICAL EXAMPLES

In this section we present numerical examples to demonstrate the performance of

FIG. 12. The NS-form of LU factors for Example 1. The factors are combined together as $\hat{T}_0$ & $\tilde{T}_0$ and entries above the threshold 10^{07} are shown in black.

Let us describe organization of the tables. Column 1 indicates the size of the matrix, N. Column 2 contains CPU time t_{fact} required to compute the factored NS-forms $\hat{T}_0$ and $\tilde{T}_0$ using Algorithm 4.2 of Section 4.3. The time t_{fact} includes the matrix fill, wavelet decomposition, and multiresolution LU factorization. Column 3 contains the time t_{sub} necessary to compute the solution of the linear system using sparse multiresolution forward and backward substitution of Section 6. Columns 4 and 5 contain the L_∞ and L_2 errors of the computations. Column 6 contains the compression ratio for T_0, the NS-form of the operator. The compression ratio is defined as the ratio of N^2 to N_s, where N_s is the number of significant entries in the matrix after truncation. Finally, column 7 contains the compression ratio for the lower and upper NS-forms $\hat{T}_0$ and $\tilde{T}_0$. This compression ratio is computed for the matrix obtained by placing both the lower and upper factors into the same matrix (this is how we store the factors). We denote this combination $\hat{T}_0$ & $\tilde{T}_0$ and note that this matrix contains all significant entries of the multiresolution LU factorization. We display the combination $\hat{T}_0$ & $\tilde{T}_0$ for various examples to illustrate the sparsity of the LU factors.

EXAMPLE 1. We consider the matrix

$$A_{ij} \quad \begin{cases} C/\tan(\mathrm{p}(i\ 0\ j)/N), & i \times j \\ 1, & i \quad j, \end{cases} \tag{9.1}$$

where $i, j \quad 1, \ldots, N$ and $C \quad 1/N$. The constant C is chosen so that $A_{ij} \quad 1/(\mathrm{p}(i\ 0\ j))$ for small $i\ 0\ j$. Multiresolution LU factorization was performed using periodized wavelets with six vanishing moments. Operations were restricted to a half-bandwidth of 20, and elements of absolute value less than 10^{07} were truncated. In Fig. 12 we

TABLE 1
Numerical Results for Example 1

	MultiR. LU		Dense LU		Errors		Comp. ratios	
N	t_{fact}	t_{sub}	t_{fact}	t_{sub}	L_∞	L_2	T_0	$\hat{T}_0$ & $\tilde{T}_0$
128	0.46	0.01	0.05	0.01	2.75 1 10^{07}	1.31 1 10^{07}	2.53	2.22
256	1.15	0.02	0.47	0.01	3.50 1 10			

TABLE 2
Numerical Results for Example 2

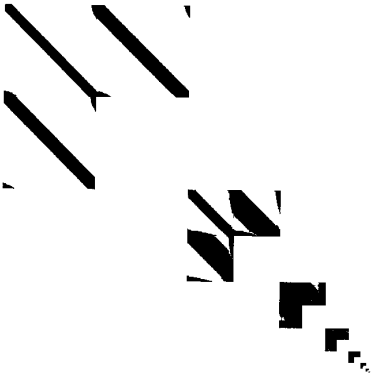


FIG. 14. The NS-form of LU factors for Example 4. The factors are combined together as $\hat{T}_0$ & $\hat{T}$

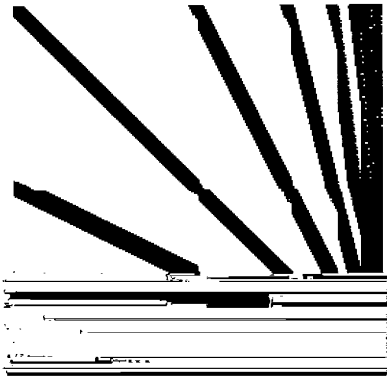


FIG. 15. The S-form of LU factors for Example 5. The factors are combined together as L & U and entries above the threshold 10^{07} are shown in black.

EXAMPLE 5. We compute the S-form of the operator used in Example 1 and then perform standard LU factorization using sparse data structures. All elements of absolute value less than 10^{07} were truncated. We compute the S-form directly (without computing the NS-form first) and it requires $O(N^2)$ operations. We use this example to demonstrate that if NS-forms remain sparse during multiresolution LU factorization (as in Example 1), then the corresponding S-forms will also be sparse (see Section 8 for details). In Fig. 15 we show the truncated matrix L & U which contains the lower and upper triangular matrices produced during LU factorization. From Table 5 we observe that the compression ratio for the S-form is worse than that for the NS-form in Table 1.

EXAMPLE 6. We compute the inverse of the operator used in Example 3 and leave the result in the NS-form. All operations were performed using periodized wavelets with six vanishing moments. Operations were restricted to a half-bandwidth of 10, and elements of absolute value less than 10^{07} were truncated.

Error analysis for this example was performed by computing the solution vector x^* as x^* T_0^{01}

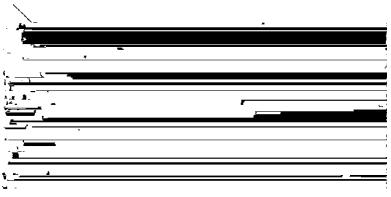


FIG. 16. The NS-form of T_0^{01}, the inverse for operator in Example 6. Entries above the threshold 10^{07} are shown in black. We observe that blocks $\hat{C}$ and $\tilde{B}$ are zero on several scales.

The results of this test are shown in Table 6. Column 2 contains the total time required to fill the matrix, compute the multiresolution LU factors, and compute the inverse. Columns 3 and 4 contain the error in the computed solution, and column 5 contains the compression ratio for the inverse operator. We observe that for this example, the time required to compute the inverse is roughly a factor of 2 greater than for computing the LU factorization.

Condition numbers. In Table 7 we present the condition numbers of matrices in four examples and the condition numbers of blocks which are actually factorized during the multiresolution *LU* factorization. The top row shows the condition number of the original matrix of size N 256. In rows 2 through 7 we present the condition number of blocks A_j of the NS-form at different scales j 1, . . . , 7, which are factorized during multiresolution LU factorization.

The second column of Table 7 (Example 2) is most interesting since it shows nearly perfect condition numbers on all scales, whereas the original operator has condition number of $O(N^2)$, where the size of the matrix is N 1 N.

10. GENERALIZATIONS AND CONCLUSIONS

The sparsity of multiresolution LU factorization algorithms does not depend on dimension. This is in a sharp contrast with the usual practice, where LU factorization

TABLE 6
Numerical Results for Example 6

	Run time	Errors		Comp. ratio
N	t_{inv}	L_∞	L_2	T_0^{01}
128	0.55	2.14 1 10^{07}	1.88 1 10^{07}	21.90
256	1.44	2.18 1 10^{07}	2.29 1 10^{07}	74.73
512	3.05	2.60 1 10^{07}	2.04 1 10^{07}	222.34
1024	6.83	1.57 1 10^{07}	1.55 1 10^{07}	615.36
2048	15.79	1.53 1 10^{07}	1.48 1 10^{07}	1572.08

TABLE 7
Condition Numbers for Blocks A_j (N 256)

	Examples			
Scale (j)	1	2	3	4
0[a]	1.41	6641[b]	2.31	1751
1	1.05	2.00	1.00	2.35
2	1.25	3.41	1.00	17.21
3	1.56	3.85	1.00	40.93
4	1.76	3.96	1.00	2.51
5	1.87	3.99	1.00	1.16
6	1.93	4.00	1.01	1.15
7	1.96	4.00	1.14	1.09

[a] Condition number of original matrix.
[b] Null space was removed from matrix.

is not recommended as an efficient approach in problems of dimension two or higher. For example, if we consider the Poisson equation, then LU decomposition is not considered as a practical option since the fill-ins will yield dense LU factors. We emphasize that the off-diagonal decay described in Theorems 5.1 and 5.3 is not specific to dimension one. Thus, multiresolution LU factorization in the wavelet system of coordinates becomes an option in solving elliptic problems in higher dimensions and we plan to demonstrate the multidimensional algorithm in a separate paper.

In multidimensional generalizations it is important to satisfy boundary conditions, and this implies using non-periodized wavelets. We note that the wavelet transform appears only as an orthogonal transformation in our approach and thus multiwavelets [3] or other orthogonal transformations may be used (provided the sparsity is maintained). We forsee future work in this direction, where one would try to optimize the choice of the basis (or coordinate transformation) in an adaptive manner for a given operator.

An additional feature of multiresolution forward and backward substitution algo-

We note that it appears possible to generalize our approach to perform fast multiresolution QR (or LQ) factorization of NS-forms rather than multiresolution LU factorization. The operators for which QR factorization should work have sparse NS-forms, and thus Householder transformations may be described by sparse vectors. We plan to develop this algorithm at a later date. Such an algorithm will have a number of important applications.

Finally, we note that an easy access to inverse operators is very useful in a variety of situations, e.g., preconditioning, low rank updates of inverse operators, etc. In signal processing variants of LU algorithms are used in various linear estimation schemes and we hope that the algorithms of this paper will have an impact on this area as well.

APPENDIX: PIVOTING

In this Appendix we consider linear systems which are not positive-definite. Strictly speaking, our approach is not guaranteed to work for such systems, as may be seen in Example 4. However, some improvement has been observed and we describe the use of partial pivoting with the multiresolution direct methods which have been developed.

1. **Factorization.** We consider the effects of partial pivoting on multiresolution LU factorization described in Section 4. When a row exchange is encountered, we modify the governing equations in (4.5) using a permutation matrix R which contains the pivoting information

$$(RA_jQ_jR^*)(RA_{j-1}) \quad R(A_j \; 0 \; A_j^U), \tag{11.1a}$$

$$(RA_jQ_jR^*)(RB_{j-1}) \quad R(B_j \; 0 \; B_j^U), \tag{11.1b}$$

$$(C_jQ_jR^*)(RA_{j-1}) \quad C_j \; 0 \; C_j^U, \tag{11.1c}$$

and solve for $R\hat{A}_jR^*$, $R\tilde{A}_j$, $R\tilde{B}_j$, and $\hat{C}_jR^*$. To proceed to the next scale, $j \; / \; 1$, we require the decomposition of the term $\hat{C}_j\tilde{B}_j$. We note that this product may be obtained from the relation $(\hat{C}_jR$

denote R_t as the permutation matrix describing the row exchanges in T_j, and let R describe the row exchanges in A_j. We obtain

$$(RA_jQR^*)(RA$$

6. A. Ben-Israel and D. Cohen, On iterative computation of generalized inverses and associate projections, *SIAM J. Numer. Anal.* **3** (1966), 410–419.
7. T. Söderström and G. W. Stewart, On the numerical properties of an iterative method for computing the Moore–Penrose generalized inverse, *SIAM J. Numer. Anal.* **11** (1974), 61–74.
8. I. S. Duff, A. M. Erisman, and J. K. Reid, ''Direct Methods for Sparse Matrices,'' Clarendon Press, Oxford, 1986.
9. Multigrid repository, available at http://na.cs.yale.edu/mgnet/www/mgnet.html.
10. Y. Meyer, Wavelets and operators, *in* ''Analysis at Urbana,'' (N. T. Peck, E. Berkson, and J. Uhl, Eds.), Vol. 1, London Math. Society, Lecture Notes Series 137, Cambridge Univ. Press, Cambridge, UK, 1989.
11. S. Mallat, Multiresolution approximations and wavelet orthonormal bases in $\mathbf{L}^2(\mathbf{R})$, *Trans. Amer. Math. Soc.* **315** (1989), 69–87.
12. M. E. Brewster and G. Beylkin, A multiresolution strategy for numerical homogenization, *Appl. Comput. Harmon. Anal.* **2** (1995), 327–349; PAM Report 187, 1994.
13. G. Beylkin and N. Coult, A multiresolution strategy for homogenization of elliptic PDE's and associated eigenvalue problems, *Appl. Comput. Harmon. Anal.* **5** (1998), 129–155; PAM Report 270, 1996.