UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Wavelet Sparse Approximate Inverse Preconditioners

T. F. Chan
W. P. Tang
W. L. Wan

August 1997

CAM Report 97-34

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90095-1555

BIT 37:3 (1997), 001–017

WAVELET SPARSE APPROXIMATE INVERSE PRECONDITIONERS *

T. F. CHAN[1] †, W. P. TANG[2] ‡, and W. L. WAN[1]

[1] *Department of Mathematics, University of California, Los Angeles CA 90095-1555, USA. email: chan@math.ucla.edu, wlwan@math.ucla.edu*

[2] *Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada. email: wptang@yoho.waterloo.edu*

Abstract.

We show how to use wavelet compression ideas to improve the performance of approximate inverse preconditioners. Our main idea is to first transform the inverse of the coefficient matrix into a wavelet basis, before applying standard approximate inverse techniques. In this process, smoothness in the entries of A^{-1} are converted into small wavelet coefficients, thus allowing a more efficient approximate inverse approximation. We shall justify theoretically and numerically that our approach is effective for matrices with smooth inverses.

AMS subject classification: 65F10, 65F35, 65F50, 65Y05, 65Y20.

Key words: Preconditioning, approximate inverses, sparse matrices, wavelet.

1 Introduction.

Consider solving the linear systems:

$$Ax = b, \tag{1.1}$$

where A is large and sparse. There is an increasing interest in using sparse approximate inverse preconditioners for Krylov subspace iterative methods to solve (1.1). On one hand, it possesses a conceptually straightforward parallel implementation. Also, the application of the preconditioner is simply matrix-vector multiply instead of forward and backward solve and it can be done easily in parallel. On the other hand, similar to the incomplete LU factorization (ILU) preconditioners, it is applicable to both general and PDE problems due to its purely algebraic nature. For example, the studies of Grote and Huckle [20] and

*Received September 1996. Revised January 1997.

†Supported by grants from ONR: ONR-N00014-92-J-1890, and the Army Research Office: DAAL-03-91-C-0047 (Univ. of Tenn. subcontract ORA4466.04 Amendment 1 and 2). The first and the third author also acknowledge support from RIACS/NASA Ames NAS 2-96027 and the Alfred P. Sloan Foundation as Doctoral Dissertation Fellows, respectively.

‡The work was supported by the Natural Sciences and Engineering Research Council of Canada, the Information Technology Research Centre (which is funded by the Province of Ontario), and RIACS/NASA Ames NAS 2-96027.

Chow and Saad [10] show that it is robust for a wide range of matrices in the Harwell-Boeing collection.

An approach of computing sparse approximate inverse is described by Benson [2] and Benson and Frederickson [3]. Consider the right preconditioned linear system:

$$AMy = b, \quad x = My, \tag{1.2}$$

where M is a right preconditioner. A sparse approximate inverse preconditioner M is defined as a solution to the following minimization problem:

$$\min_M \|AM - I\|_F^2, \tag{1.3}$$

subject to some constraint on the number and position of the nonzero entries of M. The Frobenius norm is particularly useful for parallel implementation. Notice that

$$\|AM - I\|_F^2 = \sum_{j=1}^{n} \|Am_j - e_j\|_2^2,$$

where m_j and e_j are the jth columns of M and I respectively. Thus solving (1.3) leads to solving n independent least squares problems,

$$\min_{m_j} \|Am_j - e_j\|_2, \qquad j = 1, \ldots, n, \tag{1.4}$$

which can be done in parallel.

Another possibility is to use a weighted Frobenius norm which had been investigated intensively by Kolotilina, Yeremin and others [23, 24, 25]. A complete survey can be found in [1]. In our present paper, however, we shall focus primarily on the Frobenius norm approach. Other constructions of approximate inverse are discussed in [4, 5, 8, 10, 9, 18, 28]. A comparison of approximate inverse preconditioners and ILU(0) on Harwell-Boeing matrices can be found in [19].

In practice, it is desirable to look for sparse solutions of (1.4). However, this poses two difficulties: how to determine the sparsity pattern of M and how to solve (1.4) efficiently. Recently, two main approaches have been suggested. One is discussed by Cosgrove et al [11] and Grote and Huckle [20] and the other is by Chow and Saad [10]. For the former approach, the least squares problems (1.4) are solved by the QR factorization, which may seem costly. But since m_j is sparse, the cost of the QRF can be greatly reduced. Moreover, algorithms can be derived to determine the positions of the nonzero entries adaptively. Similar methods can be found in [21, 22, 23, 24, 25] where the sparsity pattern of M is typically fixed as banded or that of the nonzeros of A.

For Chow and Saad's approach, standard iterative methods (e.g. GMRES) are used to find an approximate solution to $Am_j = e_j$, and a dropping strategy is applied to m_j to control the amount of fill-in. The idea is to let the Krylov subspace build up the sparsity pattern gradually and then the nonzeros entrie are selected automatically by size.

The idea of sparse approximate inverse is based on the assumption that A^{-1} can be approximated by a sparse matrix M. Thus sparse approximate inverses are particularly useful for matrices whose inverse contain only a small number of relatively large entries. For example, if the matrix is banded or diagonal dominant, the inverse entries decay away from the diagonal [7, 14, 15, 16]. However, as mentioned in [11, 10], if we require $\|AM - I\|_1 < 1$, we can always find a sparse matrix A for which the corresponding M has to be structurally dense. Another source of difficulty comes from matrices arising from elliptic PDEs with smooth coefficients, whose inverses may not necessary have enough number of small entries so that sparse approximate inverses are effective. For example, consider the following near tridiagonal symmetric positive definite matrix of size 40×40, derived from an artificial periodic-boundary-like elliptic problem:

$$A = \begin{bmatrix} 2.01 & -1 & & & & & -1 \\ -1 & 2.01 & -1 & & & & \\ & -1 & 2.01 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2.01 & -1 & \\ -1 & & & & -1 & 2.01 & \end{bmatrix}. \tag{1.5}$$

As we can see from Figure 1.1, the inverse entries are all greater than one and hence have no small quantities nor decay. For this kind of matrix, sparse approximate inverses may fail to give any effective sparse approximation.

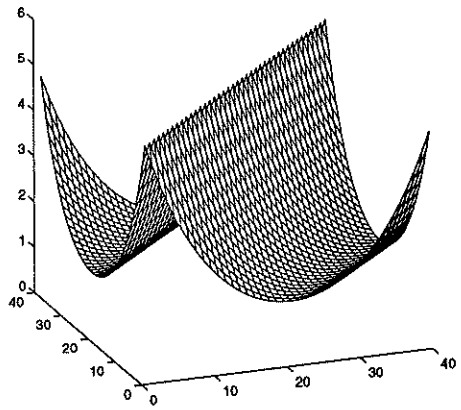

Figure 1.1: Mesh plot of A^{-1}.

Our main idea in this paper is to transform A to a new basis in which A^{-1} has a sparse approximation. For example, the inverse of a discrete elliptic operator (corresponding to the discrete Green's function [26]) typically possesses some smoothness which can be converted into small wavelet coefficients. Specifically, we apply a wavelet transform to compress the piecewise smooth entries of A^{-1} and then apply the standard techniques (e.g. Grote and Huckle's implementa-

tion) to construct a sparse approximate inverse. The use of wavelets to solve integral and differential equations can also be found in [6, 8, 17, 18].

We should also mention that factorized approximate inverse technique is another approach to deal with the case where the inverse contains only a few number of small entries. See [4, 5, 23, 24, 25] for details.

We also remark that even if A^{-1} has decay away from the diagonal (e.g. if A is the Laplace operator), the rate of decay may not be enough for the approximate inverse to have optimal convergence, in the sense that the number of iterations for convergence is independent of the mesh size. This is verified numerically in Table 1.1, where SPAI is the sparse approximate inverse given by Grote and Huckle's implementation [20]. The number in the bracket is the maximum allowable size of the residual norm of each column. In general, the smaller the number, the better (but also the denser) the approximate inverse is. We shall show numerically in Section 6 that our approach provides a means to improve the optimality of sparse approximate inverse preconditioners for solving elliptic PDEs. In fact, we shall show in Section 4 that our wavelet approximate inverse preconditioner is closely related to the well-known hierarchical basis preconditioner.

Table 1.1: Convergence of GMRES(20) where A = 2D Laplacian.

h	No. of GMRES(20) iter			No. of nonzeros in precond.		
	SPAI(0.4)	SPAI(0.2)	ILU(0)	SPAI(0.4)	SPAI(0.2)	ILU(0)
1/8	16	10	9	208	696	288
1/16	29	17	14	1040	3640	1216
1/32	67	37	25	4624	16440	4992
1/64	160	63	57	19472	69688	20224

The outline of the paper is as follows. In Section 2, we show how to adopt the wavelet transform in the least squares approach to solve (1.4). In Section 3, we justify theoretically that a smooth inverse has better decay in the wavelet basis. We also make an interesting connection between our wavelet based preconditioner and the classical hierarchical basis preconditioner. In Section 5, we estimate the extra cost for the wavelet transform and discuss the implementation issues of how to simplify the algorithm. In Section 6, we present several numerical examples to compare the various methods. Finally, we make some conclusions in Section 7.

We would like to remark that the purpose of this paper is not to present an ultimate algorithm for solving linear systems. Rather, we address a limitation of the standard approximate inverse technique and propose a way to extend its applicability. The main emphasis is on how to model the approximate inverse appropriately in order to solve a certain class of problems, e.g. matrices with piecewise smooth inverse. Many algorithmic variants are in fact possible but have not yet been fully explored. It is hoped that this paper will lead to further development and improvement of algorithms.

2 Fast wavelet based approximate inverse.

Since we are only interested in the application of wavelets to construct an approximate inverse, we only mention a few features of wavelets. See e.g. [13] for a more detail description of wavelets.

Given an orthogonal wavelet function in the continuous space, there corresponds an orthogonal matrix W that transforms vectors from the standard basis to the wavelet basis. Furthermore, if v is a vector of smoothly varying numbers (with possibly local singularities), its wavelet representation, $\tilde{v} = Wv$, will have mostly small entries. We can also represent two dimension transforms by W. Let A be a matrix in the standard basis. Then $\tilde{A} = WAW^T$ is the representation of A in the wavelet basis. This wavelet representation $\tilde{A}$ is also called the standard form of A [6].

Assuming A^{-1} is piecewise smooth, our idea is to apply a wavelet transform to compress A^{-1} and then use it as a preconditioner. At first glance, this seems impossible since we do not even have A^{-1}. Our trick is the observation that

$$\widetilde{A^{-1}} \equiv WA^{-1}W^T = (WAW^T)^{-1} = \tilde{A}^{-1},$$

where W is an orthogonal wavelet transform matrix. Therefore we can first transform A to its wavelet basis representation $\tilde{A}$ and then apply, for example, Grote and Huckle's method to find an approximate inverse for $\tilde{A}$, which is the preconditioner that we want to compute. In other words, we do not need to form A^{-1} but are still able to compute its transform.

We shall next show how we adopt the wavelet transform in the least squares approach. Consider equation (1.3) again. Let W be an orthogonal wavelet transform matrix, i.e. $\tilde{x} = Wx$ is the vector x in the wavelet basis. (Note that W can be a 1-level or full $\log_2 n$-level wavelet transform matrix.) Then

$$\begin{aligned} \min_M \|AM - I\|_F &= \min_M \|WAW^TWMW^T - I\|_F \\ &= \min_{\tilde{M}} \|\tilde{A}\tilde{M} - I\|_F, \end{aligned} \tag{2.1}$$

where $\tilde{A} = WAW^T$ and $\tilde{M} = WMW^T$ are the representations of A and M in the wavelet basis respectively. Thus, our n least squares problems become

$$\min_{\tilde{m}_j} \|\tilde{A}\tilde{m}_j - e_j\|_2, \qquad j = 1, 2, \ldots, n. \tag{2.2}$$

Note that $\tilde{A}$ is sparse (but probably denser than A) since A is. Because of the wavelet basis representation, if M is piecewise smooth, we would expect $\tilde{M}$, neglecting small entries, to be sparse too. Therefore, the sparse solution of (2.2) would hopefully give rise to a more effective approximate inverse than the original approach without the wavelet transform. We shall justify our claim numerically in Section 6.

We summarize our algorithm as follows:

Wavelet Based Approximate Inverse Algorithm

(a) Wavelet transform A to get $\tilde{A} = WAW^T$.

(b) Apply a standard approximate inverse algorithm (e.g. SPAI) to $\tilde{A}$ to obtain $\tilde{M}$.

(c) Use $\tilde{M}$ as preconditioner to solve: $\tilde{A}\tilde{x} = \tilde{b}$, where $\tilde{b} = Wb$.

(d) Apply backward wavelet transform to $\tilde{x}$ to obtain $x = W^T\tilde{x}$.

It should be noted that if we know the sparsity pattern of the large entries of $\tilde{M}$ *a priori*, then it is more efficient and also much simpler to use that pattern to solve the least squares problems (2.2) instead of using Grote and Huckle's adaptive approach.

3 Theoretical aspects.

Our wavelet based approximate inverse relies on the ability of wavelets to change (local) smoothness to small wavelet coefficients. In this section, we shall combine the classical result of Beylkin et al [6] and our construction to derive a residual estimate for our preconditioner.

In the discussion below, we shall follow the notation in [6]. We list some useful definitions which will be used later. Define the set of dyadic intervals on [0,1] by:

$$\mathcal{I} = \{[2^{-j}k, 2^{-j}(k+1)] : 0 \le k \le 2^j - 1, 0 \le j \le \log_2 n\}.$$

Let $I_{jk} = [2^{-j}k, 2^{-j}(k+1)] \in \mathcal{I}$. Then $|I_{jk}|$ =length of I_{jk} is defined as: $2^{-j}(k+1) - 2^{-j}k = 2^{-j}$.

In order to bound the size of the elements of $\tilde{A}^{-1}$ in the wavelet basis, a sufficient condition is the following smoothness assumptions on the Green's function $G(x,y)$:

$$|G(x,y)| \le \frac{1}{|x-y|}, \tag{3.1}$$

$$|\partial_x^m G(x,y)| + |\partial_y^m G(x,y)| \le \frac{C_m}{|x-y|^{m+1}}, \tag{3.2}$$

for some $m \ge 1$ and $C_m \ge 0$.

The following is a classical result of Beylkin et al [6] on the estimate of integral operator.

THEOREM 3.1. *Suppose the Green's function $G(x,y)$ satisfies the smoothness assumptions* (3.1) *and* (3.2). *Let $\tilde{A}^{-1}$ be the discrete operator of $G(x,y)$ in the wavelet basis. Then the (k,l)th entry of $\tilde{A}^{-1}$ is bounded by:*

$$(\tilde{A}^{-1})_{k,l} \le C_m \left(\frac{|I_k|}{|I_l|}\right)^{\frac{1}{2}} \left(\frac{|I_k|}{d(I_k, I_l)}\right)^{m+1},$$

where $I_k, I_l \in \mathcal{I}$, $|I_k| \le |I_l|$ and $d(I_k, I_l)$ = distance between I_k and I_l.

From the above bound, we can see that the length and position of I_k and I_l determine the size of $(\tilde{A}^{-1})_{k,l}$. By the definition of dyadic intervals and as mentioned in [6], $d(I_k, I_l)$ is equal or close to 0 at $O(n \log_2 n)$ locations only. In other words, effectively $\tilde{A}^{-1}$ has only $O(n \log_2 n)$ elements for large enough n.

With this result, we are able to estimate the quality of our approximate inverse. Let $\epsilon > 0$ be given. Define a sparsity pattern $\mathcal{S}$ to be:

$$\mathcal{S} = \{(k,l) : (\tilde{A}^{-1})_{k,l} \geq \epsilon\}.$$

Due to Theorem 3.1, the number of elements in $\mathcal{S} = O(n \log_2 n)$. We have the following estimate.

THEOREM 3.2. *If we choose $\mathcal{S}$ as our sparsity pattern, then*

$$\|AM - I\|_F \leq n \|A\|_F \epsilon. \tag{3.3}$$

PROOF. We first define an intermediate matrix $\tilde{N}$ which is essentially the truncation of $\tilde{A}^{-1}$ by:

$$(\tilde{N})_{i,j} = \begin{cases} (\tilde{A}^{-1})_{i,j} & (i,j) \in \mathcal{S}, \\ 0 & \text{otherwise}, \end{cases}$$

and denote the jth column of $\tilde{N}$ by $\tilde{n}_j$. The inequality (3.3) is a direct consequence of (2.1), (2.2), the definition of least squares solution and the definition of $\tilde{N}$ and is derived as follows:

$$\begin{aligned} \|AM - I\|_F^2 = \|\tilde{A}\tilde{M} - I\|_F^2 &= \sum_{j=1}^{n} \|\tilde{A}\tilde{m}_j - e_j\|_2^2 \\ &\leq \sum_{j=1}^{n} \|\tilde{A}\tilde{n}_j - e_j\|_2^2 \\ &= \|\tilde{A}\tilde{N} - I\|_F^2 \\ &= \|\tilde{A}(\tilde{N} - \tilde{A}^{-1})\|_F^2 \\ &\leq \|A\|_F^2 \|\tilde{N} - \tilde{A}^{-1}\|_F^2 \\ &\leq n^2 \|A\|_F^2 \epsilon^2. \end{aligned}$$

□

REMARK 3.1. A similar bound can also be found in [20]. Our result is different in the sense that we have an $O(n \log_2 n)$ estimate for the number of nonzeros of the computational approximate inverse $\tilde{M}$ while that [20] does not have such a bound.

The result of Theorem 3.2 is primarily of theoretical interest only. First of all, the sparsity pattern $\mathcal{S}$ is not known in general. Besides, $O(n \log_2 n)$ elements for $\tilde{M}$ may be still too dense for practical purposes. Furthermore, because of the special finger-like distribution of the nonzero elements of $\tilde{M}$ given by $\mathcal{S}$,

the amount of computation for solving the least squares problems may differ substantially from column to column. Thus in our implementation, we only choose a subset of $\mathcal{S}$ which corresponds to those entries near the main diagonal. We find that the preconditioning quality is still promising, as will be shown in Section 6. The implementation details will be discussed in Section 5.

4 Connection to hierarchical basis preconditioner.

Because of the hierarchical structure of wavelets, there is a natural connection between our wavelet approximate inverse and the hierarchical basis preconditioner [27, 31]. Our wavelet approximate inverse, denoted by M^{WAI}, can be considered as an approximation to the transformed A^{-1}. That is,

$$\begin{aligned} M^{WAI} &= W^T \tilde{M} W, \\ \tilde{M} &= \text{approx}\,(\tilde{A}^{-1}), \end{aligned} \tag{4.1}$$

where $\text{approx}\,(\tilde{A}^{-1})$ is an approximation of $\tilde{A}^{-1}$. In our case, it is given by the solution of the least squares problems (2.2). We can also express the approximate inverse, M^{HB}, corresponding to the hierarchical basis preconditioner in a similar form [27]:

$$\begin{aligned} M^{HB} &= S^T \hat{M} S, \\ \hat{M} &= (\text{approx}\,(\hat{A}))^{-1}, \end{aligned} \tag{4.2}$$

where S^T is the non-orthogonal transformation matrix from the hierarchical basis to the standard basis, $\hat{A} = SAS^T$, and $\text{approx}\,(\hat{A})$ is another approximation of $\hat{A}$, e.g. a coarse grid approximation of A.

These two approximate inverses are similar in that both possess a hierarchical structure. In fact, the hierarchical basis can also be considered as a special kind of wavelet since it consists of a hierarchy of piecewise linear functions and they are precisely the "hat" functions in the wavelet terminology. Our wavelet approximate inverse is more general in the sense that one is allowed to use other kind of wavelets, in particular, the orthogonal wavelets with compact support by Daubechies [12]. On the other hand, one could apply the hierarchical basis transform in a more general domain.

The main difference between the two approximate inverses is the way they approximate the original matrix A. For the approximate inverse given by the hierarchical basis in (4.2), we first approximate $\hat{A}$, e.g. by a block diagonal matrix, and then compute its exact inverse. For our wavelet approximate inverse, we compute $\tilde{A}$ exactly and then approximate its inverse by solving the least squares problems as discussed in Section 2. Typically, the approximate inverse given by the hierarchical basis is block diagonal with zero bandwidth except for the coarsest block while the one by wavelets can have nonzeros anywhere. If we choose the same block diagonal for the nonzeros of $\tilde{M}$, it will reduce to the same form (but probably with different values on the entries) of $\hat{M}$.

Connections of wavelets and hierarchical basis are also made in [29, 30].

5 Complexity and implementation of algorithm.

The naive algorithm in Section 2 needs quite an amount of overhead for doing the wavelet transformation. In this section, we will analyze each step of the algorithm and discuss some implementation issues of how to simplify and speed up the procedures. Meanwhile, we also analyze the sequential complexity of each step. We shall show that it is essentially $O(n)$, except *Step (a)* which requires $O(kn)$ operations, where k is the number of levels of the wavelet transform.

In the following discussion, we assume that the wavelet used is orthogonal and of compact support [12, 13]. Orthogonal wavelets are used so that the formulation developed in Section 2 makes sense. However, one could also use non-orthogonal wavelets anyway. Compact support, on the other hand, is indispensable so that the wavelet transform is only $O(n)$ and $\tilde{A}$ does not become dense.

Step (a). In general, to compute the wavelet transform of a vector requires $O(n)$ operations. Computing $\tilde{A} = WAW^T$ is equivalent to transforming the columns and then the rows of A (or vice versa). Thus it will cost $O(n^2)$ operations. However, since A is sparse, if we assume that there are only $O(1)$ nonzeros in each column and each row, the cost will be reduced to $O(kn)$. In fact, in the parallel implementation, we do not need to form $\tilde{A}$ explicitly in each processor. Notice that solving each least squares problem only requires a few columns of $\tilde{A}$. We just form those columns and the cost will be reduced to $O(n)$.

Step (b). In each level of the transform, we will introduce a fixed amount of nonzero entries. Even though there are only $O(1)$ nonzeros in each column and row of A, there will be $O(k)$ nonzeros in each column and row of $\tilde{A}$. We could choose k so small that the number of nonzeros introduced is acceptable. We may also reduce the cost significantly by taking advantage of the fact that for smooth coefficient problems, the Green's function typically has singularity only at a point. In other words, A^{-1} only has an essential singularity along the diagonal, together with a few additional artificial singularities due to the wrapping of the 2D discrete Green's function to a one dimensional array in each row. Hence it is reasonable to fix the sparsity pattern of $\tilde{M}$ to be block diagonal. In fact, our current implementation already assumes block diagonal structure for $\tilde{M}$. This assumption saves an enormous amount of time used for searching for the next nonzero entries adaptively. By the way, the adaptive searching procedure is usually less amenable to parallel implementation. We may further reduce the cost to $O(n)$ by adopting the concept of *local inverse* in [28].

Step (c). When we solve $\tilde{A}\tilde{x} = \tilde{b}$ by some iterative method, we need to perform $\tilde{A}$ times a vector. If we do it directly, the cost will be $O(kn)$ as $\tilde{A}$ has $O(kn)$ nonzeros in each row. Note that

$$\tilde{A}v = W(A(W^T v)).$$

If we first backward transform v, apply A and then transform it back, the overall process will only be $O(n)$.

6 Numerical results.

In this section, we compare our preconditioner with SPAI (by Grote and Huckle) and ILU(0). We choose several matrices that come from different elliptic PDEs. The inverses of all these matrices are piecewise smooth and the singularities are clustered around the diagonal. For efficiency, instead of applying SPAI to solve for $\tilde{M}$ adaptively in step (a) of our algorithm, we specify a block diagonal structure *a priori* for $\tilde{M}$ and then solve (2.2) by the QRF as discussed in Section 5. In all the tests, we use the compact support wavelet D_4 by I. Daubechies [12, 13]. We apply 6 levels of wavelet transform to matrices of order 1024 and 8 levels of transform to matrices of order 4096. Note that the number of levels is arbitrary. One could use different numbers in different situations.

We apply these preconditioners to GMRES(20). The initial guess was $x_0 = 0, r_0 = b$ and the stopping criterion was $\|r_n\|/\|r_0\| < 10^{-6}$. All the experiments were done in MATLAB in double precision.

We remark that GMRES was used because the matrices we are interested in are, in general, nonsymmetric and hence we do not adapt to using the conjugate gradient method even if A happens to be symmetric positive definite (Examples 6.1 and 6.2). Nevertheless, it is true that our algorithm does not, in general, preserve symmetry. However, the conjugate gradient method can still be used if we replace $\tilde{M}$ by $(\tilde{M} + \tilde{M}^T)/2$ in the case $\tilde{M}$ is known to be positive definite. We also remark that another approach is to use factorized approximate inverses as described in [4, 23, 24, 25] for symmetric problems.

EXAMPLE 6.1. We use two simple 1D matrices to show the benefit from using wavelet transforms. The first one is a slightly modified artificial matrix in (1.5) where the diagonal entries are changed from 2.01 to 2.00001 and the size is 1024×1024. The second matrix is the 1D Laplacian operator derived from the following:

$$u''(x) = f(x), \quad \text{in } (0,1), u(0) = 0, \quad u'(1) = 0.$$

Neumann boundary condition at $x = 1$ is used so that there is no decay in the Green's function near the boundary. The bandwidth is $0, 0, 5, 5, 5, 5$ for the 1st to 6th level of the block diagonal structure of $\tilde{M}$ respectively.

EXAMPLE 6.2. In this example, A is the 2D Laplacian operator with size 1024×1024 and 4096×4096. For $n = 1024$, we choose the bandwidth of $\tilde{M}$ as before. For $n = 4096$, we choose the bandwidth equal to $0, 0, 0, 0, 5, 5, 5, 5$ for the 1st to 8th level of the block diagonals respectively.

EXAMPLE 6.3. We try to solve something more complicated than the Laplace equation but still having a piecewise smooth inverse. Consider the following PDE with variable coefficients,

$$((1 + x^2)u_x)_x + u_{yy} + (\tan y)^2 u_y = -100x^2.$$

We solve the 32×32 and 64×64 grid cases. The bandwidth of the block diagonal of $\tilde{M}$ is the same as before.

EXAMPLE 6.4. In this case, A comes from a PDE of helical spring:

$$u_{xx} + u_{yy} + 3(5 - y)^{-1} u_x - 2G\lambda = 0,$$

where G and λ are some constants. Same setting as before.

EXAMPLE 6.5. Finally, we show an example where our wavelet preconditioner does not work. The matrix A comes from a discontinuous coefficients PDE:

$$(a(x,y)u_x)_x + (b(x,y)u_y)_y + u_x + u_y = \sin(\pi xy),$$

where the coefficients $a(x,y)$ and $b(x,y)$ are defined as:

$$a(x,y) = b(x,y) = \begin{cases} 10^{-3} & (x,y) \in [0,0.5] \times [0.5,1] \\ 10^{3} & (x,y) \in [0.5,1] \times [0,0.5] \\ 1 & \text{otherwise.} \end{cases}$$

The bandwidth is chosen to be $5, 5, 10, 10, 15, 15$ to make the number of nonzeros comparable to that of SPAI(0.2). Such modification is made so that sparsity is not a factor for the failure.

Table 6.1: Number of GMRES(20) iterations.

Ex.	n	Wavelet SPAI	SPAI(0.4)	SPAI(0.2)	ILU(0)	No prec.
1a	1024	32	>200	>200	3	>200
1b	1024	71	>200	>200	1	>200
2	1024	26	62	34	28	116
	4096	47	160	63	57	>200
3	1024	26	100	40	34	>200
	4096	66	>200	129	93	>200
4	1024	26	84	36	31	183
	4096	68	>200	126	89	>200
5	1024	>200	64	34	21	>200

Table 6.2: Number of nonzero in approximate inverse.

Example	n	Wavelet SPAI	SPAI(0.4)	SPAI(0.2)	ILU(0)
1a	1024	3544	5120	21504	3072
1b	1024	3544	5121	25425	3072
2	1024	3544	4624	16440	4992
	4096	6616	19472	69688	20224
3	1024	3544	4514	17260	4992
	4096	6616	18618	73936	20224
4	1024	3544	4624	16387	4992
	4096	6616	19472	69628	20224
5	1024	13464	5677	18952	4992

The convergence of GMRES(20) with different preconditioners in each example is shown in Figures 6.1–6.4 and is summarized in Table 6.1. In Example 6.1, we can see that SPAI(0.4) and SPAI(0.2) converge very slowly in this somewhat

Table 6.3: Operation count estimate. The count for ILU(0) is normalized to 1.

Example	n	Wavelet SPAI	SPAI(0.4)	SPAI(0.2)	ILU(0)
1a	1024	10	>72	>111	1
1b	1024	72	>215	>362	1
2	1024	0.95	2.65	2.02	1
	4096	0.78	2.79	1.54	1
3	1024	0.73	2.90	1.63	1
	4096	0.63	>2.12	1.98	1
4	1024	0.80	2.68	1.58	1
	4096	0.68	2.23	1.97	1
5	1024	>12	3.11	2.33	1

artificial but illustrating case. On the other hand, the wavelet based preconditioner converges much faster. This shows the advantage of wavelet transform in the case where A^{-1} is smooth with singularity only along the diagonal. We do not show the convergence of ILU(0) since it only takes 3 iterations to converge. This is exceptional because of the special near tridiagonal structure of A. Table 6.2 shows the number of nonzeros for each preconditioner. The wavelet based preconditioner requires much less amount of memory than SPAI does.

In Examples 6.2–6.4, the wavelet based preconditioner is most efficient in terms of convergence and storage. Although the convergence of the wavelet based preconditioner still depends on the mesh size (Figure 6.5), the dependence is less than that of ILU(0) and much less than that of SPAI. However, we would like to point out that this comparison is very rough since the preconditioners SPAI and ILU(0) take up many more nonzeros.

Besides rapid convergence, we can also see a tremendous gain in storage for

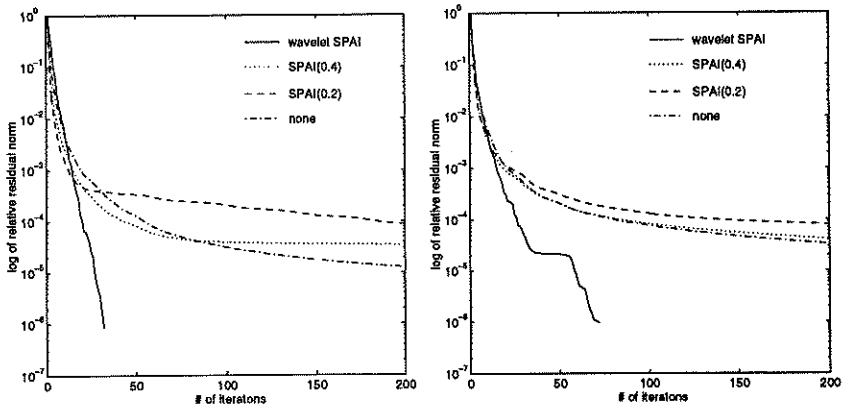

Figure 6.1: Convergence behavior of GMRES(20) where (a) A is an artificial matrix (b) A is 1D Laplacian with Dirichlet and Neumann boundary conditions.

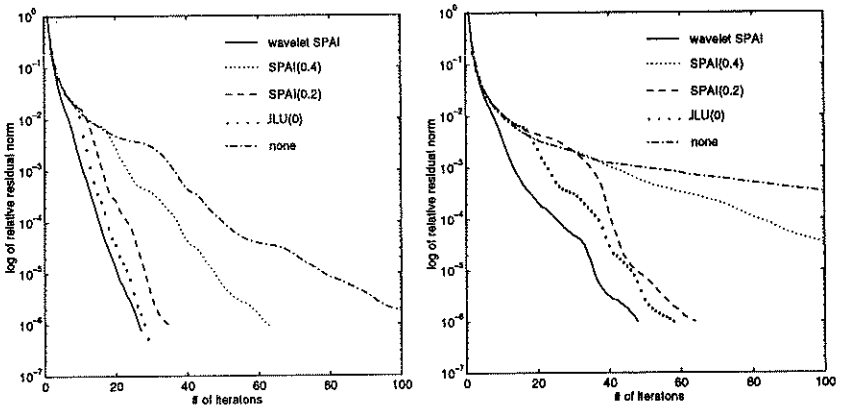

Figure 6.2: Convergence behavior of GMRES(20) where A is 2D Laplacian with size (a) $n = 1024$ (b) $n = 4096$.

the wavelet based preconditioner as n increases. This gain essentially comes from the wavelet compression. The larger n is, the more compression we can get. It is because the effect of singularity becomes less and less prominent as the singularity is only located along the diagonal.

Table 6.3 gives a comparison of the total operation counts for each method. The count estimate consists of the number of GMRES(20) iteration, the cost of matrix-vector multiply, application of the preconditioner and the number of inner products/saxpy operations. Since the number of inner products/saxpy operations depends on the iteration number, on the average, our operation counts

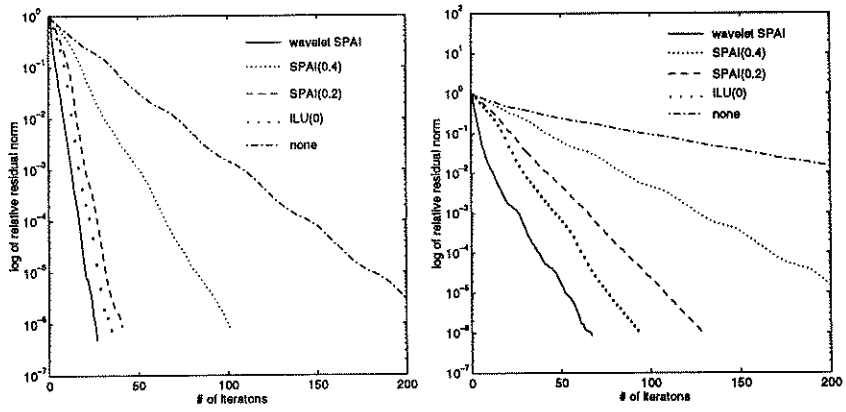

Figure 6.3: Convergence behavior of GMRES(20) where A is variable coefficient operator with size (a) $n = 1024$ (b) $n = 4096$.

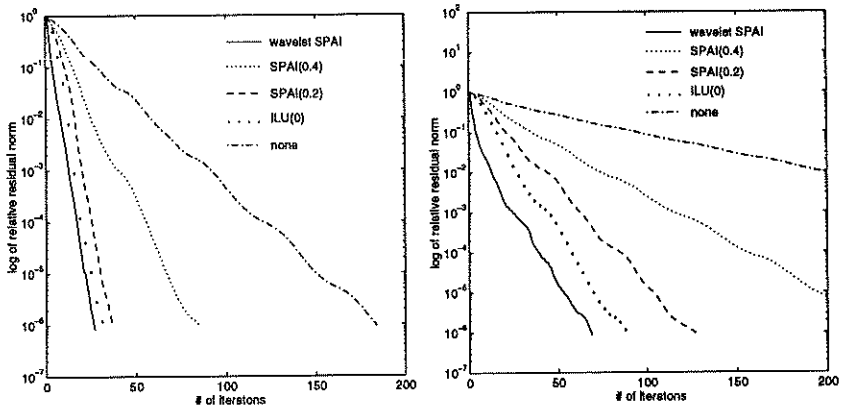

Figure 6.4: Convergence behavior of GMRES(20) where A is Helical spring operator with size (a) $n = 1024$ (b) $n = 4096$.

estimate for one GMRES(20) iteration is:

$$\text{count} = nnz(A) + nnz(M) + 21n,$$

where $nnz(A)$ and $nnz(M)$ are the number of nonzeros of the matrix A and the preconditioner M respectively and n is the size of the matrix. The count for ILU(0) is normalized to one. The wavelet preconditioner shows superior operation counts over all the other methods in Examples 6.2–6.4. In fact, the results are even better when n is larger. ILU(0) is exceptionally good for the 1D problems in Example 6.1 as explained before. Despite that, the wavelet preconditioner still takes much smaller counts than the other two approximate inverses.

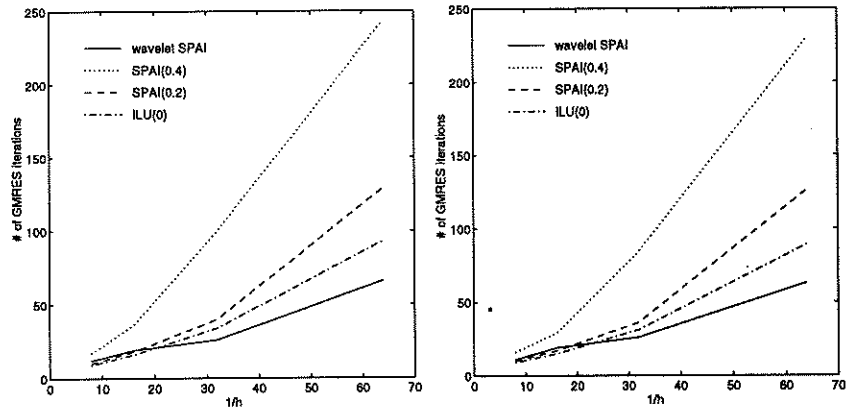

Figure 6.5: Iteration number vs $1/h$, h = mesh size, for (a) variable coefficient (b) Helical spring.

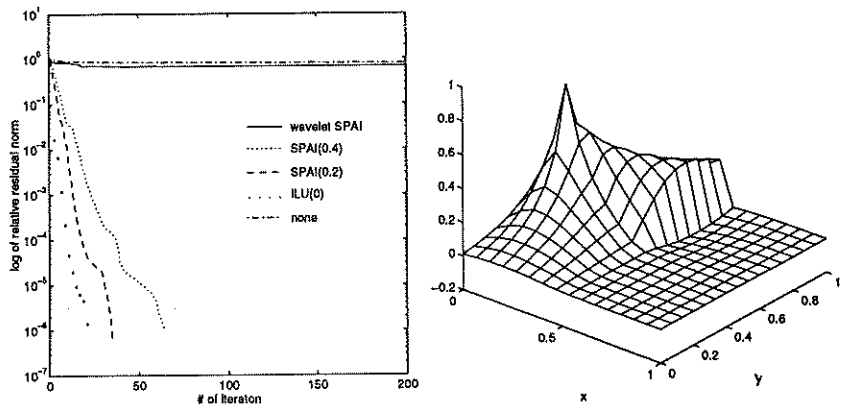

Figure 6.6: (a) Convergence behavior of GMRES(20) where A is discont. coeffs. (b) The Green's function of A at the point $(0.1, 0.5)$

Finally, Figure 6.6 (a) shows that the wavelet based preconditioner does not always work. As mentioned before, we assume that the singularity of the Green's function is only at a point so that the wavelet transformed inverse has large entries near the main diagonal and our implementation can capture those successfully as shown in previous examples. However, for discontinuous coefficient problems, the Green's function has additional singularity along the discontinuities of the coefficients as shown in Figure 6.6 (b). Hence the inverse is not as smooth as before. Thus our block diagonal structure may not completely capture the significant elements of the exact inverse. We should remark that the failure is mainly due to our current implementation. In principle, if we can locate the significant elements by some adaptive procedure (e.g. the one given in [20] and [10]), we should be able to obtain an effective approximate inverse preconditioner. However, such a sophisticated adaptive searching technique is not fully developed yet for this class of problems and further investigation is needed.

7 Conclusion.

We have extended the potential applicability of approximate inverse to a larger class of problems, namely, matrices with piecewise smooth inverses. There are two main factors concerning our preconditioner: choice of basis and sparsity pattern. We have shown that for our block diagonal implementation, the wavelet basis is suitable for matrices with piecewise smooth inverse and singularity mainly along the diagonal. Moreover, a significant amount of storage can be saved. We should remark that other choices of basis are also feasible to solve specific problems, e.g. higher order wavelets, basis derived from multiresolution methods.

If the essential singularity of A^{-1} is along the diagonal, we have shown that block diagonal structure is sufficient. However, for more general situations, e.g.

discontinuous coefficients, where the singularity is not necessarily near the diagonal, a more sophisticated adaptive searching procedure is needed to locate the sparsity pattern correctly.

Acknowledgement.

We would like to thank Barry Smith for inspiring in us the idea of combining SPAI preconditioners with a wavelet basis and his valuable comments on this paper. We also thank the referees for their helpful comments.

REFERENCES

1. O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, 1994.
2. M. Benson, *Iterative Solution of Large Scale Linear Systems*, Master's thesis, Lakehead University, Thunder Bay, Ontario, 1973.
3. M. Benson and P. Frederickson, *Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems*, Utilitas Math., 22 (1982), pp. 127–140.
4. M. Benzi, C. D. Meyer, and M. Tuma, *A sparse approximate inverse preconditioner for the conjugate gradient method*, SIAM J. Sci. Comput., 17 (1996), pp. 1135–1149.
5. M. Benzi and M. Tuma, *A sparse approximate inverse preconditioner for nonsymmetric linear systems*, Tech. Rep. TR-PA-96-15, CERFACS, 1996. To appear in SIAM J. Sci. Comput., 1997.
6. G. Beylkin, R. Coifman, and V. Rokhlin, *Fast wavelet transforms and numerical algorithms I*, Comm. Pure Appl. Math., 44 (1991), pp. 141–184.
7. C. D. Boor, *Dichotomies for band matrices*, SIAM J. Numer. Anal., 17 (1980), pp. 894–907.
8. F. Canning and J. Scholl, *Diagonal preconditioners for the EFIE using a wavelet basis*, IEEE Trans. Antennas and Propagation, 44 (1996), pp. 1239–1246.
9. E. Chow and Y. Saad, *Approximate inverse techniques for block-partitioned matrices*. 1995. To appear in SIAM J. Sci Comput.
10. E. Chow and Y. Saad, *Approximate inverse preconditioners for general sparse matrices*, Colorado Conference on Iterative Methods, April 5–9, (1994). To appear in SIAM J. Sci. Comput.
11. J. Cosgrove, J. Diaz, and A. Griewank, *Approximate inverse preconditionings for sparse linear systems*, Internat. J. Comput. Math., 44 (1992), pp. 91–110.
12. I. Daubechies, *Orthonormal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 41 (1988), pp. 909–996.
13. I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Series Appl. Math., SIAM, Philadelphia, 1991.
14. S. Demko, *Inverses of band matrices and local convergence of spline projections*, SIAM J. Numer. Anal., 14 (1977), pp. 616–619.
15. S. Demko, W. Moss, and P. Smith, *Decay rates for inverses of band matrices*, Math. Comp., 43 (1984), pp. 491–499.
16. V. Eijkhout and B. Polman, *Decay rates of inverses of banded m-matrices that are near to Toeplitz matrices*, Linear Algebra Applic., 109 (1988), pp. 247–277.

17. B. Engquist, S. Osher, and S. Zhong, *Fast wavelet based algorithms for linear evolution equations*, SIAM J. Sci. Comput., 15 (1994), pp. 755–775.

18. R. Glowinski, A. Rieder, R. Wells, Jr., and X. Zhou, *A wavelet multigrid preconditioner for dirichlet boundary-value problems in general domains*, Tech. Rep. TR93-06, Rice Computational Mathematics Laboratory, 1993.

19. N. I. M. Gould and J. A. Scott, *On approximate-inverse preconditioners*, Tech. Rep. RAL-TR-95-026, The Central Laboratory of the Research Councils, 1995.

20. M. Grote and T. Huckle, *Parallel preconditioning with sparse approximate inverses.* 1995. To appear in SIAM J. Sci. Comput.

21. M. Grote and H. Simon, *Parallel preconditioning and approximate inverses on the connection machine*, in Scalable High Performance Computing Conference (SHPCC), 1992 Williamsburg, VA, IEEE Computer Science Press, 1992, pp. 76–83.

22. M. Grote and H. Simon, *Parallel preconditioning and approximate inverses on the connection machine*, in Sixth SIAM Conference on Parallel Processing for Scientific Computing II, R. S???? et al, eds., SIAM, 1993, pp. 519–523.

23. L. Kolotilina, A. Nikishin, and A. Yeremin, *Factorized sparse approximate inverse (FSAI) preconditionings for solving 3D FE systems on massively parallel computers II*, in Iterative Methods in Linear Algebra, Proceedings of the IMACS International Symposium, Brussels, April 2–4, 1991, R. Beauwens and P. Groen, eds., 1992, pp. 311–312.

24. L. Kolotilina and A. Yeremin, *Factorized sparse approximate inverse preconditionings I. theory*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 45–58.

25. J. Lifshitz, A. Nikishin, and A. Yeremin, *Sparse approximate inverse (FSAI) preconditionings for solving 3D CFD problems on massively parallel computers*, in Iterative Methods in Linear Algebra, Proceedings of the IMACS International Symposium, Brussels, April 2–4, 1991, R. Beauwens and P. Groen, eds., 1992, pp. 83–84.

26. G. Meurant, *A review on the inverse of symmetric tridiagonal and block tridiagonal matrices*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 707–728.

27. E. Ong, *Hierarchical Basis Preconditioners for Second Order Elliptic Problems in Three Dimensions*, Ph.D. thesis, University of Washington, Seattle, 1989.

28. W. P. Tang, *Effective sparse approximate inverse preconditioners.* In preparation, 1995.

29. P. Vassilevski and J. Wang, *Stabilizing the hierarchical basis by approximate wavelets, I: Theory*, Tech. Rep. 95-47, Dept. of Mathematics, UCLA, 1995.

30. P. Vassilevski and J. Wang, *Stabilizing the hierarchical basis by approximate wavelets, II: Implementation and numerical results*, Tech. Rep. 95-48, Dept of Mathematics, UCLA, 1995.

31. H. Yserentant, *On the multilevel splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412.