AN IMPLICIT WAVELET SPARSE APPROXIMATE INVERSE PRECONDITIONER*

STUART C. HAWKINS† AND KE CHEN†

Abstract. Wavelet based sparse approximate inverse preconditioners are considered for the linear system $Ax = b$. The preconditioners are good sparse approximations to the inverse of A computed by taking advantage of the compression obtained by working in a wavelet basis. When the representation of A in a single scale basis (for example a finite element basis) is available, the formulation presented obviates the need to compute the representation of A in the wavelet basis, and removes the costs associated. Efficient application for both sparse and dense A is considered.

Key words. linear system, preconditioning, sparse approximate inverse, wavelet.

AMS subject classifications. 65F10, 65F50, 65R20

1. Introduction.

We consider the fast solution of the linear system

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

where A is a large $n \times n$ matrix. Such linear systems arise in finite element or finite difference discretisations where typically A is sparse, and in boundary element discretisations where typically A is dense. There is much interest in techniques for iteratively solving (1.1) using Krylov subspace methods such as GMRES [29, 30].

To improve performance of the iterative solver when A is ill conditioned (1.1) may be replaced by the left preconditioned linear system

$$MAx = Mb, \tag{1.2}$$

or the right preconditioned linear system

$$AMy = b. \tag{1.3}$$

To improve performance of the iterative solver we desire $MA \approx I$ in (1.2) or $AM \approx I$ in (1.3). To ensure that matrix vector multiplication with the preconditioner is cheap we require that M be sparse.

Sparse approximate inverse techniques are widely used to produce preconditioners M. Sparse approximate inverses can be produced by iterative techniques, as described by Chow and Saad [13] and Benzi and Tůma [5], but we consider direct techniques based on direct solution of (reduced) least squares problems, as described by Benson and Frederickson [3], Kolotilina and Yeremin [24], Grote and Huckle [22] and Benzi and Tůma [5]. We present our work in the context of right preconditioning, where the aim is to minimise

$$\|AM - I\|_F \tag{1.4}$$

subject to a prescribed sparsity pattern for M. A sparse approximate inverse for left preconditioning is computed by minimising $\|MA - I\|_F$. However, $\|MA - I\|_F = \|A^T M^T - I\|_F$ and so the theory and application is the same as for right preconditioning, though it looks more complicated.

*This work was supported by EPSRC research grant number GR/R22315/01.

†Department of Mathematical Sciences, University of Liverpool, Liverpool, UK.

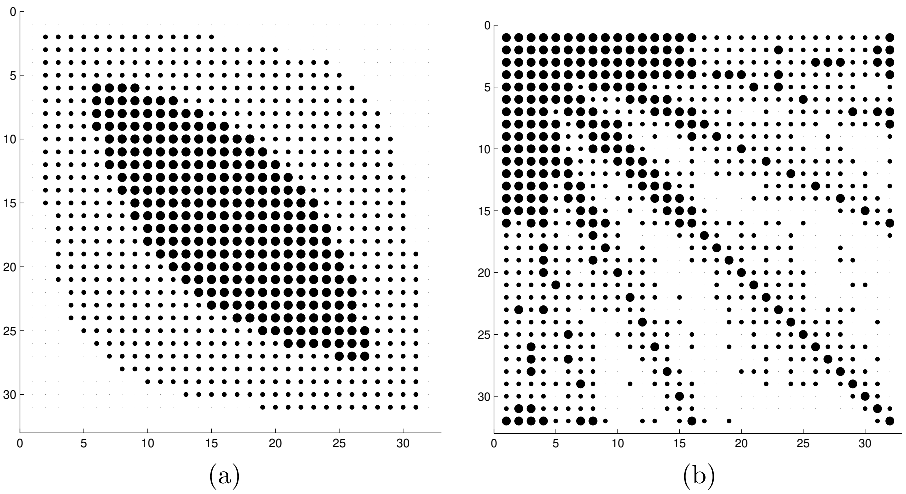


FIG. 1.1. *(a) Visualisation of the inverse of a discretisation of the Laplacian with* $n = 32$. *(b) Visualisation of the same matrix represented in the Daubechies 4 wavelet basis. Radius of dots is based on matrix entry magnitudes.*

To minimise $\|AM - I\|_F$ note that $\|AM - I\|_F^2 = \sum_{j=1}^{n} \|Am_j - e_j\|_2^2$ where m_j is the jth column of M. Thus minimising $\|AM - I\|_F$ reduces to solving n independent least squares problems. This is easy to implement in parallel.

Good approximations to A^{-1} can be obtained from sparse approximate inverse techniques when A^{-1} is sparse, but not when A^{-1} is dense. Sparse matrices need not have a sparse inverse. For example, consider the inverse of the matrix

$$A = n^2 \begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix}$$

obtained by finite difference discretisation of the Laplacian. It can be seen in Figure 1.1 (a) that A^{-1} is dense.

Chen [10] and Carpentieri, Duff and Giraud [7] discuss sparse approximate inverse techniques for dense matrices arising from boundary element discretisations. The idea is to split $A = S + F$ where S is sparse and F is dense. Then A can be preconditioned with a sparse approximate inverse of S. The choice of S is crucial here.

When A is a discretisation of a pseudo-differential operator one can view A^{-1} as the discretisation of an integral operator whose kernel is the Green's function associated with A. A^{-1} is typically dense but representation in a wavelet basis compresses A^{-1} when the Green's function is smooth. If, additionally, the Green's function is smooth away from the diagonal then A^{-1} has a special *finger pattern* in the wavelet basis. Off diagonal fingers result from interactions between wavelets at different scales. This pattern can be seen in Figure 1.1 (b), which is a visualisation of the matrix in Figure 1.1 (a) represented in the Daubechies 4 [17] wavelet basis.

The compression of the matrix A^{-1} in the wavelet basis allows it to be well approximated by a sparse matrix. Furthermore, the finger pattern of the large entries is predictable. Formally, denote representation in the wavelet basis with $\widetilde{\cdot}$ so that, in the wavelet basis, (1.1) becomes

$$\widetilde{A}\widetilde{x} = \widetilde{b}. \tag{1.5}$$

One can approximate $\widetilde{A^{-1}}$ with a finger patterned sparse approximate inverse of $\widetilde{A}$ because $\widetilde{A^{-1}} = \widetilde{A}^{-1}$.

Chan, Tang and Wan [9] use this sparse approximate inverse of $\widetilde{A}$ as a preconditioner for (1.5) but restrict the pattern of the sparse approximate inverse to block diagonal. This simplifies the computation but interactions between wavelets at different scales are omitted in the calculation of the preconditioner. This is in line with the theory of Dahmen and Kunoth [16], who show that diagonal preconditioning yields a condition number that is independent of the number of unknowns.

However, as observed by Cohen and Masson [15], sometimes diagonal preconditioning is not sufficient for efficient iterative solution. Cohen and Masson [15] compute finger patterned sparse approximate inverses for (1.5) in the slightly different context where (1.5) is obtained directly from the underlying partial differential equation (PDE) or boundary integral equation (BIE) with discretisation using wavelet basis functions. Use of the full finger pattern for the sparse approximate inverse means that all interactions between scales are included in this preconditioner. This makes the preconditioner more robust but can sometimes make the computation of the sparse approximate inverse prohibitively expensive because a small number of dense columns or rows can be present in the finger pattern.

Bridson and Tang [6] compute a sparse approximate inverse of $\widetilde{A}$ where the sparsity pattern of the approximate inverse is determined adaptively. The adaptive sparse approximate inverse algorithm does not require the entries of $\widetilde{A}$ to be available, and allows second generation wavelets to be used efficiently. However, Chow [12] notes that adaptive sparse approximate inverse algorithms can be more expensive to implement than sparse approximate inverse algorithms for which the pattern of the approximate inverse is known in advance.

We remark that, outside the sparse approximate inverse framework, Chan and Chen [8] proposed a recursive preconditioner that does not require $\widetilde{A}$ to be computed.

We present a new preconditioner that takes advantage of the structure possessed by A^{-1} when it is represented in the wavelet basis. Like Cohen and Masson [15] we incorporate all of the interactions between wavelets at different scales, which is sometimes necessary to produce a robust preconditioner. We describe strategies that overcome the computational difficulties presented by the subsequent sparsity pattern of the preconditioner. Many of our computations involve A rather than $\widetilde{A}$ because our formulation applies the discrete wavelet transform implicitly. When A is sparse it is often sparser than $\widetilde{A}$ and working with A rather than $\widetilde{A}$ reduces the cost of computing the sparse approximate inverse. The cost of computing $\widetilde{A}$ is also removed.

A typical discrete wavelet transform is applied to the vector $x \in \mathbb{R}^n$ as follows. For simplicity we assume $n = 2^N$ for some $N \in \mathbb{N}$, but the theory in this paper applies whenever $n = s^N p$ for some $N, s, p \in \mathbb{N}$. Let $h_0, h_1, \ldots, h_{m-1}$ and $g_0, g_1, \ldots, g_{m-1}$ be the low pass and high pass filter coefficients of the transform. Then the level $L \leq N$ discrete wavelet transform of x is the vector $\widetilde{x} = (s^L, d^L, d^{L-1}, \ldots, d^1)$ where

$$s^0 = x, \tag{1.6}$$

$$s_j^k = \sum_{l=0}^{m-1} h_l s^{k-1}_{\langle l+2j-1\rangle_{n/2^{k-1}}}, \qquad j = 1, 2, \ldots, n/2^k, \tag{1.7}$$

$$d_j^k = \sum_{l=0}^{m-1} g_l s^{k-1}_{\langle l+2j-1 \rangle_{n/2^{k-1}}}, \qquad j = 1, 2, \ldots, n/2^k, \tag{1.8}$$

where $\langle a \rangle_b$ denotes $c+1$ where c is the remainder of $a-1$, modulo b. For example, $\langle 256 \rangle_{256} = 256$ and $\langle 257 \rangle_{256} = 1$. When $L = N$ we say that $\widetilde{x}$ is fully transformed. The discrete wavelet transform of a matrix is obtained by transforming each row of the matrix, and then transforming each column of the result.

Equation (1.7) can be rewritten

$$s^k = U_k s^{k-1}$$

where

$$U_k = \begin{bmatrix} h_0 & h_1 & h_2 & \ldots & h_{m-1} & & & \\ & & h_0 & h_1 & h_2 & \ldots & h_{m-1} & \\ & & & \vdots & & & & \\ h_2 & \ldots & h_{m-1} & & & & h_0 & h_1 \end{bmatrix}$$

is an $n/2^k \times n/2^{k-1}$ matrix. In the same way (1.8) can be rewritten

$$d^k = V_k s^{k-1}$$

where

$$V_k = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & g_{m-1} & & & \\ & & g_0 & g_1 & g_2 & \cdots & g_{m-1} & \\ & & & \vdots & & & & \\ g_2 & \cdots & g_{m-1} & & & & g_0 & g_1 \end{bmatrix}$$

is an $n/2^k \times n/2^{k-1}$ matrix. When $m - 1 > n/2^{k-1}$ the entries wrap around, for example, when $m = 4$ the 1×2 level N matrices are

$$\begin{aligned} U_N &= \begin{bmatrix} h_0 + h_2 & h_1 + h_3 \end{bmatrix}, \\ V_N &= \begin{bmatrix} g_0 + g_2 & g_1 + g_3 \end{bmatrix}. \end{aligned}$$

The level L discrete wavelet transform can be written

$$\widetilde{x} = W^T x = W_L^T W_{L-1}^T \ldots W_1^T x \tag{1.9}$$

with

$$W_k^T = \left[\begin{array}{c|c} \begin{matrix} U_k \\ V_k \end{matrix} & 0 \\ \hline 0 & I_{n-n/2^{k-1}} \end{array}\right].$$

The paper is structured as follows. In §2 we review the wavelet sparse approximate inverse preconditioner of Chan, Tang and Wan [9]. In §3 we introduce our new preconditioner that avoids transforming A. In §4 we discuss how to implement our new algorithm and consider how to compute the entries of the wavelet basis vectors, which are required to implement the new method. We end the section with some comments on the relative costs of computing the Chan, Tang and Wan [9] preconditioner and the new preconditioner. In §5 we consider modifications to these preconditioners for the case when A is dense. In §6 we present some theory on the conditioning of the new preconditioned iteration matrix. In §7 we present some numerical experiments. We finish with some conclusions in §8.

2. Wavelet Based Sparse Approximate Inverse. Cohen and Masson [15] show that, for a wide class of PDE problems, A^{-1} is sparse in the wavelet basis. Noting this we review the preconditioning method of Chan, Tang and Wan [9].

Let W be as in (1.9). Since $\widetilde{A^{-1}} = W^T A^{-1} W = (W^T A W)^{-1} = \widetilde{A}^{-1}$, Chan, Tang and Wan [9] propose a sparse approximate inverse preconditioner $\widetilde{M}$ for $\widetilde{A}$ computed by solving a minimisation problem analogous to (1.4), that is, by minimising $\|\widetilde{A}\widetilde{M} - I\|_F$. Cohen and Masson [15] established that, for a wide class of pseudo-differential equations, both $\widetilde{A}$ and $\widetilde{A}^{-1}$ have a finger pattern. Thus a sparse approximate inverse of $\widetilde{A}$ is justified and a finger pattern for $\widetilde{M}$ can usually be prescribed. The Chan, Tang and Wan [9] algorithm is described by Algorithm 1.

Algorithm 1.

1. *Compute* $\widetilde{A} = W^T A W$, $\widetilde{b} = W^T b$.
2. *Compute* $\widetilde{M}$, *a right sparse approximate inverse of* $\widetilde{A}$.
3. *Solve* $\widetilde{A}\,\widetilde{M}\widetilde{y} = \widetilde{b}$.
4. *Compute* $x = W\widetilde{M}\,\widetilde{y}$.

Step 1 involves computing $\widetilde{A}$, an operation that has complexity $O(n)$ depending on the level of the transform, the method of applying the transform, and the sparsity of A. The computation of $\widetilde{A}$ is a significant part of the total cost of implementing Algorithm 1.

When A is sparse the matrix vector multiplications with $\widetilde{A}$ involved in Step 3 should be applied in factored form with $W^T A W$ rather than with $\widetilde{A}$ itself.

Finger patterned matrices, though sparse, can have a small number of dense columns and rows. As noted by Gould and Scott [20] the presence of dense columns in $\widetilde{M}$ require that care be taken in its computation. Reducing the sparsity pattern of $\widetilde{M}$ to block diagonal, as described by Chan, Tang and Wan [9], is one way to solve this problem but relinquishes robustness because interactions between scales are disregarded. In this paper we present a new and cheaper way of using wavelets to give a sparse preconditioner with a predictable pattern but without computing $\widetilde{A}$. Interactions between wavelets at different scales are included.

3. A New Implicit Wavelet Sparse Approximate Inverse Preconditioner. We describe a new wavelet sparse approximate inverse preconditioner that is closely related to the preconditioner in Algorithm 1 but the requirement that $\widetilde{A}$ be available is removed and the wavelet transform is applied implicitly. This is facilitated by changing the way in which the sparse approximate inverse is computed. Additionally we use the whole finger pattern for the preconditioner to ensure that all interactions between scales are included.

Observe that

$$\begin{aligned}
\|\widetilde{A}\,\widetilde{M} - I\|_F &= \|W^T A W W^T M W - I\|_F \\
&= \|W^T A M W - I\|_F \\
&= \|W W^T A M W - W\|_F \\
&= \|A\widehat{M} - W\|_F
\end{aligned}$$

where $\widetilde{M} = W^T M W$ and $\widehat{M} = MW = W\widetilde{M}$. It is not difficult to see that $\widehat{M}$ has a predictable sparsity pattern, because $\widetilde{M}$ has a predictable sparsity pattern. Minimising $\|A\widehat{M} - W\|_F$ reduces to n independent least squares problems in the same way as minimising $\|AM - I\|_F$. These observations lead to a new algorithm.

Algorithm 2.

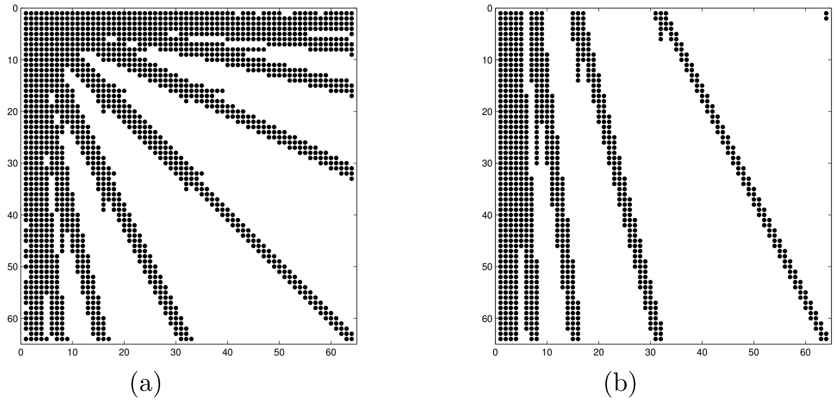


FIG. 3.1. *Visualisations of (a) a finger patterned matrix and (b) a one-sided finger patterned matrix.*

1. *Compute $\widehat{M}$, the minimiser of $\|A\widehat{M} - W\|_F$ subject to the prescribed sparsity pattern for $\widehat{M}$.*
2. *Solve $W^T A\widehat{M}\widetilde{y} = W^T b$.*
3. *Compute $x = \widehat{M}\widetilde{y}$.*

$\widehat{M}$ is an approximation to A^{-1} represented from the wavelet basis to the standard, single scale basis. We establish in Theorem 6.1 that this matrix possesses a band pattern like that shown in Figure 3.1 (b) and which is analogous to the finger pattern shown in Figure 3.1 (a). In fact W possesses such a band pattern and our practical experience has shown that the pattern of W provides a suitable pattern for $\widehat{M}$.

In fact we have implicitly applied a one sided wavelet transform to M, compared with the two sided wavelet transform of M in the Chan, Tang and Wan [9] algorithm. In this way we achieve compression and structure in A^{-1} and like the Cohen and Masson [15] preconditioner, our preconditioner includes interactions between different scales. The one sided application of the wavelet transform provides less compression than the two sided application but our practical experience is that the compression achieved is sufficient.

In general the preconditioner $\widehat{M}$ is unsymmetric, even when A is symmetric. Symmetrisation techniques such as the one presented for Algorithm 1 by Chan, Tang and Wan [9] are precluded because the nonsymmetric pattern of $\widehat{M}$ is dictated by the decay of the inverse of A and the properties of the wavelet basis. Though in general it is more expensive to solve unsymmetric systems than symmetric systems, use of this new strategy for symmetric systems is justified because construction of the preconditioner in Algorithm 2 is cheaper than construction of the preconditioner in Algorithm 1.

4. Implementation Details. We now discuss how to solve in practice the least squares problem in Algorithm 2. Analysis of the least squares problem reveals two important considerations for practical implementation of the algorithm. We show how to address these considerations by modifying slightly how the preconditioner is constructed, and by finding cheaper ways of computing the wavelet matrix W. We finish with some comments about the complexity of computing the matrices $\widetilde{A}$ and W.

Consider the problem of minimising $\|A\widehat{M} - W\|_F$ at Step 1 of Algorithm 2. Let

S_j be the set of indices of the nonzero entries of $\widehat{m}_j$ where $\widehat{m}_j$ is the jth column of $\widehat{M}$. The classical approach to solving the minimisation problem is to observe that

$$\|A\widehat{m}_j - w_j\|_2 = \|A(:,S_j)\,\widehat{m}_j(S_j) - w_j\|_2$$

and

$$\text{argmin}\|A(:,S_j)\,\widehat{m}_j(S_j) - w_j\|_2 = \text{argmin}\|A(T_j,S_j)\,\widehat{m}_j(S_j) - w_j(T_j)\|_2$$

where T_j indexes the non-zero rows of $A(:,S_j)$ and w_j is the jth column of W. With the reduced QR factorisation $A(T_j,S_j) = QR$ we have

$$\|A(T_j,S_j)\,\widehat{m}_j(S_j) - w_j(T_j)\|_2 = \|QR\,\widehat{m}_j(S_j) - w_j(T_j)\|_2, \tag{4.1}$$

which can be minimised by solving $R\,\widehat{m}_j(S_j) = Q^T w_j(T_j)$ by back substitution.

We make two observations. Firstly, the entries $w_j(T_j)$ must be known. Secondly, the principal cost of minimising (4.1) is that of computing the QR factorisation of $A(T_j,S_j)$. This requires $O(|T_j||S_j|^2)$ arithmetic operations, becoming $O(|S_j|^3)$ when A is banded because then $|T_j| \sim |S_j|$. Algorithm 2 becomes impractical in its basic form when $L = N$ because then $|S_j| = n$ for some columns j. The following modifications address the second observation.

1. Use a wavelet transform of level $L < N$. Note $\max_{j=1,n} |S_j| \approx 2^L$.
2. Compute $\widehat{m}_j$ by minimising (4.1) whenever $|S_j|/n > \rho$ for some parameter $\rho \in (0,1)$. Otherwise compute $\widehat{m}_j$ by solving $A\widehat{m}_j = w_j$ using a small number of GMRES steps.
3. Restrict S_j. This approach is used by Chan, Tang and Wan [9]. We do not use this approach because we aim to include the off diagonal finger pattern into our preconditioner.

The ability to compute the necessary entries of W is crucial. Many discrete wavelet transforms are applied level by level, with

$$W = W_1 W_2 \dots W_{L-1} W_L.$$

The matrices W_i represent single levels of the transform and are often readily available, although W may not be. Computation of W by multiplying the factors W_i is possible because W and its factors are sparse.

More generally, the kth column of W can be computed by inverse-transforming the kth Euclidean vector e_k using any transform algorithm. Only $O(L)$ wavelets need to be computed in this way because the wavelets at each level are translations of each other—thus only one wavelet at each level must be computed.

It remains to comment on the relative costs of Algorithms 1 and 2. Sparse approximate inverse and setup are considered. In computing the sparse approximate inverse in Algorithm 1, the restricted pattern for $\widetilde{M}$ yields $|S_j| = c$ for some constant c. However $|T_j|$ is difficult to estimate because it depends on the pattern of $\widetilde{A}$. Nevertheless, one can bound the costs of computing $\widetilde{M}$ by $O(n^2c^2)$. A realistic estimate is $O(nc^2)$ because there are few dense columns of $\widetilde{A}$.

In Algorithm 2, recall that $|T_j| \sim |S_j|$ when A is banded. The cost of computing $\widehat{M}$ is then bounded by $O(n[\max_{j=1,n}|S_j|]^3)$. Note that $\max_{j=1,n}|S_j| \approx 2^L$, yielding a cost $O(n2^{3L})$. When L is kept constant the cost becomes linear in n.

Setup for Algorithm 1 involves computing $\widetilde{A}$. Setup for Algorithm 2 involves computing W. The former requires wavelet forward-transforming $2n$ vectors. The latter requires inverse wavelet transforming $O(L) \leq O(\log_2 n)$ vectors. W may be stored and reused for other problems of the same size. We will show that $\widetilde{A}$ can be computed in $O(12mn\alpha)$ arithmetic operations when A has bandwidth α, and W can be computed in $O(4m(m-1)n)$ arithmetic operations. $\widetilde{A}$ and W may be computed naively by multiplication of sparse matrices. This is easily implemented but introduces a computational overhead, although the arithmetic operation estimates will be realised. Computation of $\widetilde{A}$ remains more expensive than that of W.

Algorithm 3 is the typical pyramid algorithm for the forward discrete wavelet transform. Let $\alpha \in \mathbb{N}$ be large compared with $m/2$. Then the complexity of transforming a vector with α nonzeros is approximately $O(2m \times 2\alpha[1-1/2^L])$ and the transform of the vector contains approximately $2\alpha[1-1/2^L]$ nonzeros. It follows that transforming a banded matrix with α nonzeros in each column involves transforming n columns each with α nonzeros and then n rows each with $2\alpha[1-1/2^L]$ nonzeros. The complexity of this transformation is approximately

$$\begin{aligned}&O\left(2mn \times \left[2\alpha[1-1/2^L] + 2\left(2\alpha[1-1/2^L]\right)[1-1/2^L]\right]\right)\\ \approx\ &O(12mn\alpha).\end{aligned}$$

Algorithm 4 is the typical pyramid algorithm for the inverse discrete wavelet transform. The complexity of inverse transforming the Euclidean vector whose nonzero entry appears in s^l or d^l is approximately $O(2m(m-1)2^l)$. Thus the complexity of transforming L Euclidean vectors whose nonzeros appear in $s^1, s^2, \ldots, s^L$ respectively is approximately

$$\begin{aligned}&O\left(\sum_{l=1}^{L} 2m(m-1)2^l\right)\\ \approx\ &O(4m(m-1)2^L).\end{aligned}$$

When $n = 2^L$ the second estimate becomes $O(4m(m-1)n)$ and we see that computing $\widetilde{A}$ is approximately $12\alpha/4(m-1)$ times more expensive than computing W.

Algorithm 3. *Forward Transform*

1. *Let* $s^0 = x$.
2. *For level* $k = 1, 2, \ldots, L$
   (a) *Let* $s_j^k = \sum_{l=0}^{m-1} h_l s^{k-1}_{\langle l+2j-1\rangle_{n/2^{k-1}}}, \qquad j = 1, 2, \ldots, n/2^k.$
   (b) *Let* $d_j^k = \sum_{l=0}^{m-1} g_l s^{k-1}_{\langle l+2j-1\rangle_{n/2^{k-1}}}, \qquad j = 1, 2, \ldots, n/2^k.$
3. *Let* $\widetilde{x} = (s^L, d^L, d^{L-1}, \ldots, d^1)$.

Algorithm 4. *Inverse Transform*

1. *Let* $(s^L, d^L, d^{L-1}, \ldots, d^1) = \widetilde{x}$.
2. *For level* $k = L, L-1, \ldots, 1$
   (a) *Let* $s_j^{k-1} = \sum_{l=0}^{m/2-1} h_{2l} s^k_{\langle (j+1)/2-l\rangle n/2^k} + \sum_{l=0}^{m/2-1} g_{2l} d^k_{\langle (j+1)/2-l\rangle n/2^k}$
   for $j = 1, 3, \ldots, n/2^{k-1} - 1$.
   (b) *Let* $s_j^{k-1} = \sum_{l=0}^{m/2-1} h_{2l+1} s^k_{\langle j/2-l\rangle n/2^k} + \sum_{l=0}^{m/2-1} g_{2l+1} d^k_{\langle j/2-l\rangle n/2^k}$
   for $j = 2, 4, \ldots, n/2^{k-1}$.
3. *Let* $x = s^0$.

5. Dense Problems. Although our primary purpose is to precondition sparse problems arising from PDEs, we consider in this section the application of Algorithms 1 and 2 to dense problems where A is either fully available, or is partially available, for example, using the Fast Multipole Method [21, 27].

Consider the general problem of finding M that minimises $\|BM - C\|_F$ subject to a specified sparsity pattern for M. When $B = \widetilde{A}$ and $C = I$ this corresponds to the minimisation problem at Step 2 of Algorithm 1. When $B = A$ and $C = W$ this corresponds to the minimisation problem at Step 1 of Algorithm 2.

Let S_j index the non-zero entries of m_j, where m_j is the jth column of M. A generalisation of the argument in §4 shows that the minimisation problem reduces to solving n problems of the form $R\,m_j(S_j) = Q^T c_j(T_j)$ where we have the QR factorisation $B(T_j, S_j) = QR$, where T_j indexes the non-zero rows of $B(:, S_j)$, and where c_j is the jth column of C.

The principal task in solving each least squares problem is the QR factorisation of the $t \times s = |T_j| \times |S_j|$ matrix $B(T_j, S_j)$. This operation has complexity $O(ts^2)$. Note that s depends upon the chosen sparsity pattern for M, and that t, loosely speaking, depends upon the sparsity of B.

When A is dense then $\widetilde{A}$ in Algorithm 1 is also dense. However, if A is also smooth then $\widetilde{A}$ will have a structure containing many small entries. Computation of a column of the sparse approximate inverse of a dense matrix using the least squares approach has complexity $O(ns^2)$ and for all but the smallest s is too expensive. Computation using the GMRES approach (for example, when the column is dense) is also expensive because it requires repeated matrix vector multiplication with a dense matrix.

We consider two adaptations to the method of computing the sparse approximate inverse. Write $B = S + F$ where S is sparse and F is dense.

1. B is replaced by S in the least squares minimisation problems. Thus we minimise $\|SM - C\|_F$ instead of $\|BM - C\|_F$.

2. Dense columns of M are computed by solving $Sm_j = c_j$ instead of $Bm_j = c_j$ using GMRES.

We propose that (2) is always necessary for computing dense columns, but that (1) is required only when $s = |S_j|$ is not small.

In both cases the idea is to let S be a good sparse approximation to B so that M, which approximates S^{-1}, is a good preconditioner for B. A similar approach is taken by Chen [10] who shows that if the continuous operator $\mathcal{S}$ underlying S is bounded and the continuous operator $\mathcal{F}$ underlying F is compact then $\mathcal{S}^{-1}(\mathcal{S}+\mathcal{F}) = \mathcal{I}+\mathcal{S}^{-1}\mathcal{F}$ and $\mathcal{S}^{-1}\mathcal{F}$ is compact. One expects the corresponding matrix $S^{-1}(S+F)$ to have the clustered eigenvalues associated with the discretisation of such an operator.

The choice of S and F is crucial. Carpentieri et al. [7] propose the following strategies.

1. *Algebraic.* S is obtained from B by discarding entries whose absolute value falls below a specified threshold.

2. *Topological.* When B is obtained by a finite element discretisation, S is obtained by discarding those entries b_{ij} of B whose corresponding nodes i and j are not (at some given level) mesh neighbours.

3. *Geometric.* When B is obtained by a finite element discretisation, S is obtained by discarding those entries of b_{ij} of B whose corresponding nodes i and j are separated by more than a specified distance.

For Algorithm 1 we propose Strategy 1 because Strategies 2 and 3 do not directly apply since the entries in $\widetilde{A}$ do not directly relate to connections between nodes in a mesh. However, analogous strategies to 2 and 3 can be developed by replacing relationships between nodes with relationships between supports of wavelet basis functions, see for example Hawkins and Chen [23].

For Algorithm 2 one could apply any of these strategies. In some circumstances details of the mesh may be available but the entries of A are not explicitly known. Then $\widetilde{A}$ cannot easily be computed and our method is particularly useful. Under this context Carpentieri et al. [7] propose Strategies 2 and 3. This situation arises, for example, in particle simulations or integral equation formulations for Laplace's equation when they are solved using the Fast Multipole Method [21, 27]. Here matrix vector multiplication with A can be performed cheaply and to a specified accuracy but A is not computed explicitly. One way to obtain columns of A for the purposes of computing a least squares sparse approximate inverse is to multiply Euclidean vectors by A. This naive approach may be very expensive, because all columns of A will be needed, though they will not be needed or stored simultaneously when the preconditioner is sparse. A better way to compute the sparse approximate inverse in this case might be to use Algorithm 2 with Adaptation 1, replacing A in the sparse approximate inverse computations by a sparse approximation S to A. The entries of S might be obtained using only near field contributions associated with the singularity.

6. Theoretical Details. In this section we justify the banded pattern chosen for $\widehat{M}$ and present a theoretical bound on $\|W^T A\widehat{M} - I\|_F$. This provides a measure of the conditioning of the iteration matrix $W^T A\widehat{M}$.

The following theorem establishes that $A^{-1}W$, which we approximate by $\widehat{M}$, has the banded pattern illustrated in Figure 3.1 (b). Let $\{\psi_i\}_{i=1}^n$ be a wavelet basis in L_2 with m vanishing moments. Such properties of wavelets are discussed fully by Strang and Nguyen [31]. Let I_i denote the support of ψ_i, and suppose that $\|\psi_i\|_2 = 1$. Let $\{h_i\}_{i=1}^n$ be a single scale basis in L_2, with $\bar{I}_i$ denoting the support of h_i, and with $\|h_i\|_2 = 1$.

Theorem 6.1. *Suppose that the Green's function $K(x, y)$ satisfies*

$$\left|\frac{\partial^m K}{\partial x^m}(x, y)\right| \le \frac{C_m}{|x-y|^{m+1}}, \quad \textit{for } x \ne y \tag{6.1}$$

for some constant C_m. Let $\widehat{N}$ be the discrete operator of $K(x, y)$ from the wavelet basis $\{\psi_j\}_{j=1}^n$ to the single scale basis $\{h_i\}_{i=1}^n$. Then

$$|\widehat{N}_{ij}| \le b(I_j, \bar{I}_i)$$

with

$$b(I_j, \bar{I}_i) = C|I_j|^{(2m+1)/2} \frac{1}{\operatorname{dist}(I_j, \bar{I}_i)^{m+1}}$$

for some constant C.

The proof of Theorem 6.1 is given in the Appendix.

Choose $\epsilon > 0$ and let $S = \{(i, j) \ : \ |b(I_j, \bar{I}_i)| > \epsilon\}$. Denote by $\widehat{N}^S$ the matrix obtained by cutting $\widehat{N}$ to the pattern S. Then

$$\begin{aligned} \max_{(i,j)} |\widehat{N}^S_{ij} - \widehat{N}_{ij}| &\le \max_{(i,j)\notin S} |b(I_j, \bar{I}_i)| \\ &\le \epsilon. \end{aligned}$$

THEOREM 6.2. *If we choose S as the sparsity pattern for $\widehat{M}$ then*

$$\|W^T A\widehat{M} - I\|_F \leq n\epsilon \|A\|_F.$$

Proof. Denote the jth column of $\widehat{N}^S$ by $\widehat{n}_j^S$. Similarly denote the jth column of $\widehat{M}$ by $\widehat{m}_j$ and the jth column of W by w_j.

Then

$$\begin{aligned}
\|W^T A\widehat{M} - I\|_F^2 &= \|A\widehat{M} - W\|_F^2 \\
&= \sum_{j=1}^{n} \|A\widehat{m}_j - w_j\|_2^2 \\
&\leq \sum_{j=1}^{n} \|A\widehat{n}_j^S - w_j\|_2^2 \\
&= \|A\widehat{N}^S - W\|_F^2 \\
&\leq \|\widehat{N}^S - A^{-1}W\|_F^2 \|A\|_F^2 \\
&\leq \|\widehat{N}^S - \widehat{N}\|_F^2 \|A\|_F^2 \\
&\leq n^2\epsilon^2 \|A\|_F^2. \qquad \square
\end{aligned}$$

Remarks 6.3. These results give a guide to the effectiveness of $\widehat{M}$ for small ϵ. The eigenvalues of the preconditioned matrix will have a desirable distribution when ϵ is small, because they will be clustered close to 1.

Theorem 6.2 suggests that one should choose ϵ sufficiently small, so that the pattern S leads to a sufficiently small bound $\|W^T A\widehat{M} - I\|_F^2$ for the preconditioner $\widehat{M}$. In practice, good results can be obtained when ϵ is large.

A similar bound for $\|\widetilde{A}\widetilde{M} - I\|_F$ is given by Chan, Tang and Wan [9].

7. Numerical Results. We test variants of Algorithms 1 and 2 with GMRES [29, 30] for a range of finite-difference discretisations of one, two and three dimensional model PDEs, for a range of sparse off-the-shelf matrices obtained from the Matrix Market [25] and representing finite difference and finite element discretisations of two and three dimensional problems, and for a dense problem arising from a boundary element discretisation of a boundary integral equation. The test problems are,

1D Laplacian One dimensional Poisson's equation

$$u_{xx} = \sin 2\pi x, \quad x \in [0,1]$$

with a Dirichlet boundary condition discretised with central differences.

2D Laplacian Two dimensional Poisson's equation

$$u_{xx} + u_{yy} = -100x^2, \quad (x,y) \in [0,1]^2$$

with a Dirichlet boundary condition discretised with central differences.

2D Disc Two dimensional PDE with discontinuous coefficients

$$(a(x,y)u_x)_x + (b(x,y)u_y)_y + u_x + u_y = \sin(\pi xy), \quad (x,y) \in [0,1]^2$$

with

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

with a Dirichlet boundary condition discretised with central differences.

3D Laplacian Three dimensional Poisson's equation

$$u_{xx} + u_{yy} + u_{zz} = -100x^2, \quad (x,y,z) \in [0,1]^3$$

with a Dirichlet boundary condition discretised with central differences.

3D PDE Three dimensional PDE with variable coefficients

$$\big((1+x^2)u_x\big)_x + ((1+y)u_y)_y + \big((2-z^2)u_z\big)_z = -100x^2, \quad (x,y) \in [0,1]^2$$

with a Dirichlet boundary condition discretised with mixed forward and backward differences.

Sherman 3 Matrix Market [25] problem arising from finite difference discretisation of a three dimensional oil reservoir model.

Sherman 4 Matrix Market [25] problem arising from finite difference discretisation of a three dimensional oil reservoir model.

Watt 1 Matrix Market [25] problem arising in petroleum engineering.

RDB2048L Matrix Market [25] problem arising from finite difference discretisation of a reaction diffusion Brusselator model.

NONSYMA Two dimensional PDE with variable coefficients

$$-\epsilon\Delta u + \frac{\partial}{\partial x}(e^{xy}u) + \frac{\partial}{\partial y}(e^{-xy}u) = g, \quad \epsilon = \frac{1}{1000}, \quad (x,y) \in [0,1]^2$$

with a Dirichlet boundary condition discretised with central differences. The right hand side is chosen so that the solution vector is $(1,1,\ldots,1)^T$. This is a widely used test problem, see for example [4, 28].

NONSYMB Two dimensional PDE with variable coefficients

$$-\epsilon\Delta u + \frac{\partial}{\partial x}(e^{xy}u) + \frac{\partial}{\partial y}(e^{xy}u) = h, \quad \epsilon = \frac{1}{1000}, \quad (x,y) \in [0,1]^2$$

with a Dirichlet boundary condition discretised with central differences. The right hand side is chosen so that the solution vector is $(1,1,\ldots,1)^T$.

NONSYMC Two dimensional PDE with variable coefficients

$$-\Delta u + 2P_1 u_x + 2P_2 u_y = f, \quad P_1 = P_2 = 50, \quad (x,y) \in [0,1]^2$$

with a Dirichlet boundary condition discretised with central differences. the right hand side is chosen following Elman [18] so that the solution is $u(x,y) = xe^{xy}\sin(\pi x)\sin(\pi y)$.

BIE Dense matrix arising from a boundary element discretisation of the boundary integral operator $-\frac{1}{2}I + M_k + iN_k$ arising in the Burton and Miller formulation for the exterior Neumann Helmholtz problem [2, 11].

Daubechies 4 wavelets [17] were used for all problems. For the BIE problem we additionally used multiwavelets [1]. Our method can be applied with other types of orthogonal wavelet and is easily generalised to work with biorthogonal wavelets [14], or second generation wavelets [32]. In all experiments we used a wavelet transform of the same dimension as the underlying PDE problem. Wavelet transforms in two and three dimensions are easily obtained using tensor products. The Sherman 3 and Sherman 4 problems have a dimension whose number of unknowns is not divisible by two, and thus not amenable to wavelet transform. We can still use wavelets if we use a Schur complement method to extract a subsystem involving an appropriately sized submatrix. For the Sherman 3 and Sherman 4 problems we solve systems involving appropriate submatrices but do not implement the full Schur complement to obtain the right hand side or the solution to the original system. Except where stated otherwise results were obtained using software written in Fortran and C++ running on a 1GHz Intel Pentium 3 machine.

The New preconditioner is of sparse approximate inverse type. As such, it complements incomplete factorisation based preconditioners [29]. For illustration we include results for an ILUT preconditioner [29]. The fill-in parameter for the ILUT preconditioner is chosen so that the total number of nonzeros in the incomplete LU factors is comparable with the number of nonzeros in the New preconditioner. Incomplete factorisation based preconditioning is known to be very effective for M-matrices [26] but can fail for some nonsymmetric problems [4, 18, 28, 29]. For test problems NONSYMA, NONSYMB and NONSYMC we compute the ILUT preconditioner after applying a symmetric reverse Cuthill-McKee reordering of the matrix. Benzi, Szyld and van Duin [4] show that for nonsymmetric problems such reorderings dramatically improve the effectiveness of incomplete factorisation based preconditioners. Note that reordering is not required for the New preconditioner.

In Table 7.1 we present timings and iteration counts for solving the test problems using GMRES. Convergence histories for some of these problems are given in Figures 7.1, 7.2 and 7.3. The New preconditioner is Algorithm 2 implemented with a wavelet transform of level $L < N$. The choice of L varies between problems. For each of the finite difference problems we fix L, though N varies. The New preconditioner is very effective and gives substantial improvement over Algorithm 1 and over no preconditioning. In most cases the fewest iterations are required with ILUT preconditioning. Algorithm 1 and New compute and explicitly store the preconditioner, and application of the preconditioner at each iteration of GMRES involves only matrix-vector multiplication. In contrast, application of the ILUT preconditioner involves triangular solves, which are difficult to parallelise.

In Table 7.3 we present setup times and number of nonzeros for the preconditioning strategies. The setup time for Algorithm 1 includes the time t_{DWT} taken to compute $\widetilde{A}$ and the time t_{SPAI} taken to compute the sparse approximate inverse. The setup time for New includes the time t_{DWT} taken to compute W and the time t_{SPAI} taken to compute the sparse approximate inverse. In almost every case the setup time for New is less than that for Algorithm 1. The setup time t_{FACT} for ILUT is less than that for the other preconditioners. However, the sparse approximate inverse components of Algorithm 1 and New are easy to implement in parallel. The discrete wavelet transform can also be implemented in parallel [19]. In contrast, the incomplete factorisation in ILUT is difficult to implement in parallel.

Remarks 7.1.

1. *In §4 we showed that the sparse approximate inverse and discrete wavelet transform components of New have linear complexity in n. This linear complexity can be observed in Table 7.3.*
2. *Most of the setup time for New is the time taken to compute the sparse*

approximate inverse in serial. However this task is easily parallelised, especially when A is banded, because then load balancing is easy even though the number of nonzeros in each column of $\widehat{M}$ varies. We expect the time taken to compute the sparse approximate inverse to be of order n/p when there are p processors used in parallel.

3. *Improved iteration counts and t_{GMRES} can be achieved by increasing the level L of the wavelet transform. Corresponding increases in t_{SETUP} can in some part be alleviated by using GMRES to compute dense rows of the preconditioner.*

4. *The two and three dimensional problems yield better conditioned matrices than the one dimensional problems because the ill-conditioning of the matrix grows with decreasing mesh size rather than the size of the matrix. In fact the condition number for the d dimensional Laplacian problem grows like $n^{2/d}$.*

The effects of varying L can be seen in Table 7.2 where we present results obtained for the 1D Laplacian with $L = 4$, and $L = N$. When $L = N$ the preconditioner has some full columns so that this strategy is impractical and we include it only for comparison. This strategy can be made more practical by computing column m_j using GMRES whenever $|S_j|/n > 1/2$. Results are presented for this modification.

In Table 7.4 we present results for the dense problem BIE. The matrix is complex but the theory is the same as for the real case. These experiments were performed in Matlab using the Daubechies 4 wavelets [17] and also a basis of super-compact orthogonal multiwavelets [1]. Algorithm 2 is implemented with the adaptations described in §5 for dense problems. The variants of Algorithm 2 required the fewest iterations. Using the multi-wavelets led to fewer iterations than using the Daubechies 4 wavelets. We believe this is due to the higher compression that can be achieved with the multi-wavelets, which have more vanishing moments than the Daubechies 4 wavelets.

8. Conclusion. We have introduced a new wavelet sparse approximate inverse preconditioner. The new preconditioner, in common with the preconditioners of Chan, Tang and Wan [9] and Cohen and Masson [15], takes advantage of the compression and structure created by working in a wavelet basis. Like the Cohen and Masson [15] preconditioner the new preconditioner takes account of interactions between scales, which is considered essential for robustness. However, the new preconditioner works with A rather than with the wavelet transform of A and this makes the computation of the sparse approximate inverse easier because, firstly, the wavelet transform of A need not be computed, and secondly, when A is sparse it is often sparser than its wavelet transform. We have considered efficient application of the new preconditioner and the Chan, Tang and Wan [9] preconditioner to sparse and dense matrices.

To facilitate easy computation of our preconditioner we have (a) considered part wavelet transforms and developed a mixed least squares/GMRES sparse approximate inverse algorithm, and (b) considered efficient ways of computing the wavelet basis matrix W.

The new preconditioning strategy has been tested on a range of sparse and dense linear systems arising from discretisation of partial differential equations and boundary integral equations. In all cases the preconditioner has been shown to be very useful in reducing the number of GMRES iterations required to solve the linear system.

Appendix. Compression Theorem.

We now state the proof of Theorem 6.1.

Proof. We begin by expanding K as a Taylor series in x about $x_0 = \inf I_j$ to give

$$K(x,y) = K(x_0,y) + (x-x_0)\frac{\partial K}{\partial x}(x_0,y) + \ldots + \frac{(x-x_0)^{m-1}}{(m-1)!}\frac{\partial^{m-1} K}{\partial x^{m-1}}(x_0,y) + R(x,y)$$

where the remainder

$$R(x,y) = \frac{(x-x_0)^m}{m!} \; \frac{\partial^m K}{\partial x^m}(\xi, y)$$

for some $\xi \in I_j$. All terms in the Taylor series, except for the remainder term, are orthogonal to ψ_j because of the vanishing moments properties of the wavelets.

Now

$$\widehat{N}_{ij} = \int_{\bar{I}_i} \int_{I_j} K(x,y) \; \psi_j(x) \; dx \; h_i(y) \; dy,$$

and

$$\begin{aligned}
\int_{I_j} K(x,y) \; \psi_j(x) \; dx &= \int_{I_j} R(x,y) \; \psi_j(x) \; dx \\
&= \int_{I_j} \frac{(x-x_0)^m}{m!} \; \frac{\partial^m K}{\partial x^m}(\xi,y) \; \psi_j(x) \; dx \\
&= \frac{\partial^m K}{\partial x^m}(\xi,y) \; \int_{I_j} \frac{(x-x_0)^m}{m!} \; \psi_j(x) \; dx.
\end{aligned}$$

It follows that

$$\begin{aligned}
&\int_{\bar{I}_i} \int_{I_j} K(x,y) \; \psi_j(x) \; dx \; h_i(y) \; dy \\
&= \int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y) \; \int_{I_j} \frac{(x-x_0)^m}{m!} \; \psi_j(x) \; dx \; h_i(y) \; dy \\
&= \left(\int_{I_j} \frac{(x-x_0)^m}{m!} \; \psi_j(x) \; dx \right) \; \int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y) \; h_i(y) \; dx.
\end{aligned}$$

The Cauchy-Schwarz Inequality gives

$$\begin{aligned}
&\left| \int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y) \; h_i(y) \; dy \right| \\
&\leq \left(\int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y)^2 \; dy \right)^{1/2} \; \left(\int_{\bar{I}_i} h_i(y)^2 \; dy \right)^{1/2} \\
&\leq \left(\int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y)^2 \; dy \right)^{1/2} \; \|h_i\|_2 \\
&= \left(\int_{\bar{I}_i} \frac{\partial^m K}{\partial x^m}(\xi,y)^2 \; dy \right)^{1/2} \\
&\leq |\bar{I}_i|^{1/2} \sup_{y \in \bar{I}_i} \left| \frac{\partial^m K}{\partial x^m}(\xi,y) \right|.
\end{aligned}$$

A similar argument shows

$$\left| \int_{I_j} \frac{(x-x_0)^m}{m!} \; \psi_j(x) \; dx \right| \leq \frac{|I_j|^{(2m+1)/2}}{m! \; (2m+1)^{1/2}}.$$

Combining these gives

$$\begin{aligned}\left|\int_{\bar{I}_i}\int_{I_j} K(x,y)\ \psi_j(x)\ dx\ h_i(y)\ dy\right| &\le \frac{|I_j|^{(2m+1)/2}}{m!\ (2m+1)^{1/2}}\ |\bar{I}_i|^{1/2} \sup_{y\in\bar{I}_i}\left|\frac{\partial^m K}{\partial x^m}(\xi,y)\right| \\ &\le \frac{|I_j|^{(2m+1)/2}}{m!\ (2m+1)^{1/2}}\ |\bar{I}_i|^{1/2}\frac{C_m}{\mathrm{dist}(I_j,\bar{I}_i)^{m+1}} \\ &\le C|I_j|^{(2m+1)/2}\frac{1}{\mathrm{dist}(I_j,\bar{I}_i)^{m+1}}\end{aligned}$$

where

$$C = C_m\frac{|\bar{I}_i|^{1/2}}{m!\ (2m+1)^{1/2}}.$$

Here we make use of the fact that the functions $\{h_i\}$ are a fixed scale basis and so $|\bar{I}_i|$ is constant. ∎

Acknowledgements. We thank M. Benzi and two anonymous referees, whose comments and suggestions greatly improved this paper.

REFERENCES

[1] B. Alpert, G. Beylkin, R. Coifman, and V. Rokhlin, *Wavelet-like bases for the fast solution of second-kind integral equations*, SIAM J. Sci. Comput., 14 (1993), pp. 159–184.
[2] S. Amini, P. J. Harris, and D. T. Wilton, *Coupled Boundary and Finite Element Methods for the Solution of the Dynamic Fluid-Structure Interaction Problem*, Springer-Verlag, 1992.
[3] M. W. Benson and P. O. Frederickson, *Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems*, Util. Math., 22 (1982), pp. 127–140.
[4] M. Benzi, D. B. Szyld, and A. van Duin, *Orderings for incomplete factorization preconditioning of nonsymmetric problems*, SIAM J. Sci. Comput., 20 (1999), pp. 1652–1670.
[5] M. Benzi and M. Tůma, *A comparative study of sparse approximate inverse preconditioners*, Appl. Numer. Math., 30 (1999), pp. 305–340.
[6] R. Bridson and W. P. Tang, *Multiresolution approximate inverse preconditioners*, SIAM J. Sci. Comput., 23 (2001), pp. 463–479.
[7] B. Carpentieri, I. S. Duff, and L. Giraud, *Sparse pattern selection strategies for robust Frobenius-norm minimisation preconditioners in electromagnetism*, Numer. Linear Algebra Appl., 7 (2000), pp. 667–685.
[8] T. F. Chan and K. Chen, *On two variants of an algebraic wavelet preconditioner*, SIAM J. Sci. Comput., 24 (2002), pp. 260–283.
[9] T. F. Chan, W. P. Tang, and W. L. Wan, *Wavelet sparse approximate inverse preconditioners*, BIT, 37 (1997), pp. 644–660.
[10] K. Chen, *An analysis of sparse approximate inverse preconditioners for boundary integral equations*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 1058–1078.
[11] K. Chen and P. J. Harris, *Efficient preconditioners for iterative solution of the boundary element equations for the three dimensional helmholtz equation*, Appl. Numer. Math., 36 (2001), pp. 475–489.
[12] E. Chow, *A priori sparsity patterns for parallel sparse approximate inverse preconditioners*, SIAM J. Sci. Comput., 21 (2000), pp. 1804–1822.
[13] E. Chow and Y. Saad, *Approximate inverse preconditioners via sparse-sparse interactions*, SIAM J. Sci. Comput., 19 (1998), pp. 995–1023.
[14] A. Cohen, I. Daubechies, and J.-C. Feauveau, *Biorthogonal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 45 (1992), pp. 485–560.
[15] A. Cohen and R. Masson, *Wavelet methods for second-order elliptic problems, preconditioning and adaptivity*, SIAM J. Sci. Comput., 21 (1999), pp. 1006–1026.
[16] W. Dahmen and A. Kunoth, *Multilevel preconditioning*, Numer. Math., 63 (1992), pp. 315–344.

[17] I. Daubechies, *Orthonormal bases of compactly supported wavelets*, Comm. Pure Appl. Math., XLI (1988), pp. 909–996.

[18] H. C. Elman, *A stability analysis of incomplete ILU factorizations*, Math. Comp., 47 (1986), pp. 191–217.

[19] J. Ford, K. Chen, and N. J. Ford, *Flexible parallelization of fast wavelet transforms*, Parallel Algorithms Appl., 18 (2003), pp. 155–169.

[20] N. I. M. Gould and J. A. Scott, *Sparse approximate-inverse preconditioners using norm-minimisation techniques*, SIAM J. Sci. Comput., 19 (1998), pp. 605–625.

[21] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[22] M. J. Grote and T. Huckle, *Parallel preconditioning with sparse approximate inverses*, SIAM J. Sci. Comput., 18 (1997), pp. 838–853.

[23] S. C. Hawkins and K. Chen, *New wavelet preconditioner for solving boundary integral equations over nonsmooth boundaries*, Int. J. Comput. Math., 81 (2004), pp. 353–360.

[24] L. Y. Kolotilina and A. Y. Yeremin, *Factorised sparse approximate inverse preconditionings I. Theory*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 45–58.

[25] *Matrix market*. http://math.nist.gov/MatrixMarket/.

[26] J. A. Meijerink and H. A. van der Vorst, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp, 31 (1977), pp. 148–162.

[27] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys., 60 (1983), pp. 197–207.

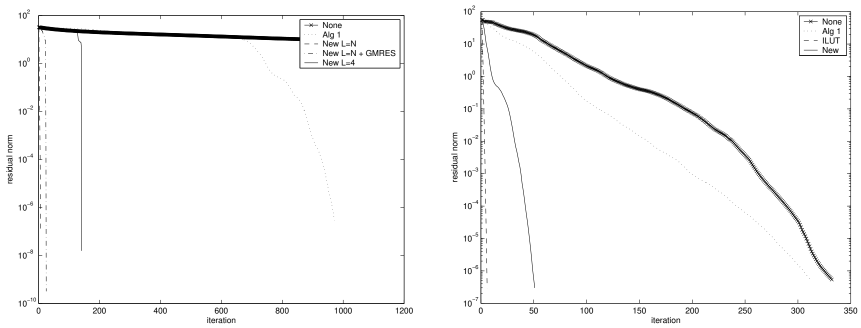
[28] Y. Saad, *Preconditioning techniques for nonsymmetric and indefinite linear systems*, J. Comput Appl. Math., 24 (1988), pp. 89–105.

[29] ———, *Iterative Methods for Sparse Linear Systems*, SIAM, second ed., 2003.

[30] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

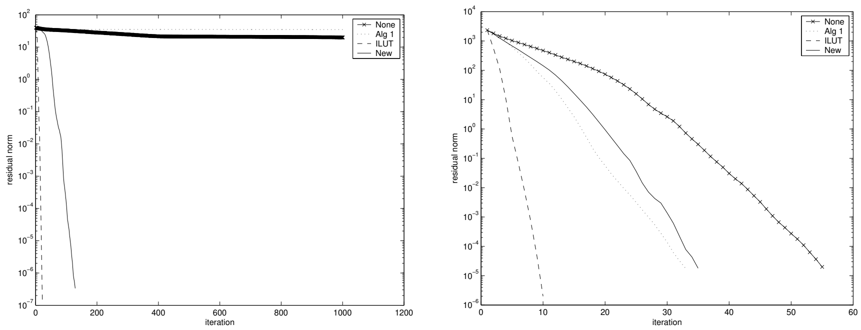
[31] G. Strang and T. Nguyen, *Wavelets and filter banks*, Wellesley–Cambridge Press, 1996.

[32] W. Sweldens, *The lifting scheme: a construction of second generation wavelets*, SIAM J. Math. Anal., 29 (1997), pp. 511–546.



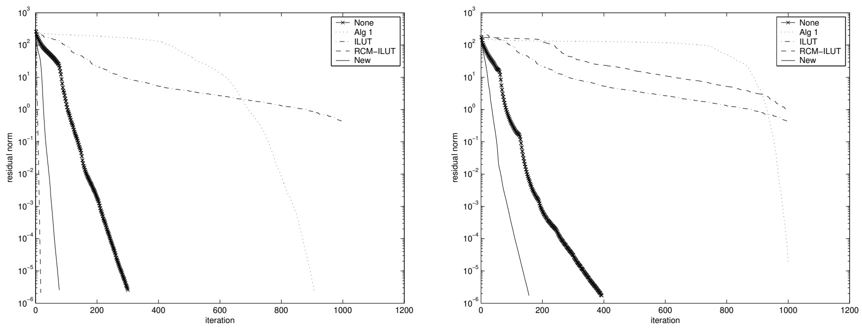
(a) (b)

FIG. 7.1. *(a) Variants of the New preconditioner perform well for the 1D Laplacian. Using cheaper variants such as* $L = 4$ *and* $L = N + \mathrm{GMRES}$ *impairs the performance. (b) GMRES with ILUT preconditioning and New preconditioning converge quickly for the Sherman 3 problem.*



(a) (b)

FIG. 7.2. *(a) GMRES with ILUT preconditioning converges very quickly for the 2D Disc problem [9]. The New preconditioning strategy is also very effective. (b) GMRES with ILUT preconditioning converges very quickly for the 3D Laplacian. This is an easy problem for ILUT but a challenging problem for sparse approximate inverse based preconditioners. Nevertheless the New and Algorithm 1 preconditioners perform well.*



(a) (b)

FIG. 7.3. *The nonsymmetric problems (a) NONSYMB and (b) NONSYMC are challenging for all of the preconditioners. For both problems New is effective but ILUT fails. ILUT can be made effective for NONSYMB by preordering the system with a reverse-Cuthill-McKee ordering (denoted RCM-ILUT).*

Experiment	size	None itns	None t_{GMRES}	ILUT itns	ILUT t_{GMRES}	Alg. 1 itns	Alg. 1 t_{GMRES}	New itns	New t_{GMRES}
1D Laplacian	256	256	0.16	-	-	155	0.11	23	0.00
	512	512	1.36	-	-	282	0.67	40	0.03
	1024	$\times$	$\times$	-	-	520	3.89	74	0.19
	2048	$\times$	$\times$	-	-	970	24.67	140	0.89
2D Laplacian	256	44	0.00	8	0.00	29	0.01	28	0.02
	1024	90	0.09	12	0.02	51	0.19	58	0.16
	4096	179	1.59	23	0.17	95	1.94	118	1.76
2D Disc	256	188	0.08	6	0.01	249	0.31	28	0.00
	1024	645	4.19	11	0.02	$\times$	$\times$	60	0.17
	4096	$\times$	$\times$	21	0.16	$\times$	$\times$	128	1.89
3D Laplacian	512	27	0.01	5	0.00	20	0.09	17	0.08
	4096	54	0.23	9	0.16	32	1.33	34	1.06
3D PDE	512	39	0.02	5	0.01	22	0.09	18	0.08
	4096	79	0.42	9	0.16	35	1.50	38	1.19
Sherman 3	2048*	331	2.56	5	0.03	311	8.27	50	0.73
Sherman 4	1104	127	0.23	57	0.06	115	0.33	92	0.17
	768*	97	0.11	15	0.00	86	0.23	54	0.11
Watt 1	1856	403	3.41	24	0.05	154	1.08	115	0.58
RDB2048L	2048	134	0.53	8	0.09	201	2.53	69	1.09
NONSYMA	1024	299	1.05	$11^{\dagger}$	$0.02^{\dagger}$	672	6.95	80	0.25
NONSYMB	1024	299	1.00	$16^{\dagger}$	$0.03^{\dagger}$	906	11.72	76	0.22
NONSYMC	1024	392	1.73	$\times^{\dagger}$	$\times^{\dagger}$	$\times$	$\times$	155	0.59

Table 7.1

*GMRES iterations "itns" and time "t_{GMRES}" required to reduce the residual norm by a factor of 10^{-8}. A $\times$ denotes that 1000 steps of GMRES were insufficient to achieve this reduction of the residual norm. New denotes a variant of Algorithm 2. All times are measured in seconds. The superscript * denotes that the iteration matrix is a submatrix of the original matrix. ILUT inverts the matrix exactly for the 1D Laplacian. ILUT fails for the nonsymmetric problems NONSYMB and NONSYMC when the lexicographical ordering is used. The superscript † denotes results obtained for a reverse-Cuthill-McKee reordering of the original system.*

size	$L=N$ itns	$L=N$ t_{GMRES}	$L=N$ t_{SETUP}	$L=N$ with GMRES itns	$L=N$ with GMRES t_{GMRES}	$L=N$ with GMRES t_{SETUP}	$L=4$ itns	$L=4$ t_{GMRES}	$L=4$ t_{SETUP}
256	6	0.00	0.38	18	0.00	0.05	23	0.00	0.03
512	6	0.00	2.56	20	0.03	0.34	40	0.03	0.05
1024	6	0.02	23.70	22	0.06	2.81	74	0.19	0.11
2048	6	0.05	210.25	24	0.17	23.20	140	0.89	0.27

Table 7.2

Results from variants of Algorithm 2 applied to the 1D Laplacian problem. The best iteration counts "itns" and times "t_{GMRES}" are obtained when the number of levels of wavelet transform L is large. The setup time "t_{SETUP}" increases with L but this can in some part be alleviated by using GMRES for computing dense columns in the preconditioner. All times are measured in seconds.

Experiment	size	ILUT nnz	ILUT t_{FACT}	Alg. 1 nnz	Alg. 1 t_{DWT}	Alg. 1 t_{SPAI}	New nnz	New t_{DWT}	New t_{SPAI}
1D Laplacian	256	766	0.00	1330	0.00	0.00	3328	0.00	0.03
	512	1534	0.00	2616	0.01	0.03	6656	0.00	0.05
	1024	3070	0.00	5182	0.09	0.06	13312	0.00	0.11
	2048	6142	0.00	10308	0.20	0.22	26624	0.00	0.27
2D Laplacian	256	3896	0.00	1996	0.03	0.05	4096	0.00	0.02
	1024	16024	0.02	6764	0.38	0.39	16384	0.00	0.09
	4096	64856	0.09	24172	2.77	3.95	65536	0.02	0.66
2D Disc	256	3893	0.00	1996	0.03	0.05	4096	0.00	0.01
	1024	16021	0.00	6764	0.38	0.39	16384	0.00	0.09
	4096	64853	0.09	24172	2.75	4.36	65536	0.02	0.64
3D Laplacian	512	30386	0.05	4588	0.45	0.25	32768	0.00	1.63
	4096	256237	1.30	32236	18.38	23.53	262144	0.05	16.02
3D PDE	512	30376	0.05	4588	0.47	0.25	32768	0.00	1.66
	4096	256113	1.31	32236	19.36	24.11	262144	0.03	16.05
Sherman 3	2048*	66978	0.19	13382	4.58	3.13	131072	0.02	4.98
Sherman 4	1104	2698	0.00	7416	0.03	0.08	4416	0.00	0.03
	768*	6709	0.02	5816	0.19	0.28	12288	0.00	0.06
Watt 1	1856	17348	0.02	15114	0.16	0.30	18560	0.00	0.19
RDB2048L	2048	199378	0.25	12076	1.19	1.16	204800	0.05	107.41
NONSYMA	1024	16120†	0.01†	6764	0.38	0.41	16384	0.00	0.09
NONSYMB	1024	16131†	0.01†	6764	0.38	0.41	16384	0.02	0.09
NONSYMC	1024	15984†	0.00†	6764	0.41	0.39	16384	0.00	0.11

TABLE 7.3

*Setup times and numbers of nonzeros in the preconditioner for the experiments in Table 7.1. New denotes a variant of Algorithm 2. Although the setup time t_{FACT} for ILUT is typically less than the setup time $t_{\mathrm{DWT}}+t_{\mathrm{SPAI}}$ for the New strategy, both the discrete wavelet transform and the sparse approximate inverse components of the New strategy can be easily implemented in parallel. The fill-in parameters for ILUT ensure that the number of nonzeros "nnz" in the preconditioner is comparable between ILUT and New. The superscript * denotes that the iteration matrix is a submatrix of the original matrix. The superscript † denotes results obtained for a reverse-Cuthill-McKee reordering of the original system.*

	None	Alg. 1		New $L=N$		New^1 $L=N$		New $L=N-2$	New^2 $L=N$	
	-	D4	A	D4	A	D4	A	D4	D4	A
BIE^3	613	162	166	89	94	108	125	89	75	60
BIE^4	613	162	166	60	67	72	87	60	73	59

TABLE 7.4

Results for a dense problem. The New preconditioning strategies produce large reductions in the number of GMRES iterations required to reduce the residual norm by a factor of 10^{-8}. For New strategies S is ^{3}bandwidth 3, ^{4}bandwidth 7. The superscripts ^{1} and ^{2} denote that dense columns of the preconditioner were computed using GMRES with A and S respectively. For this experiment, which was performed in Matlab, we use (D4) Daubechies 4 wavelets [17] and (A) multiwavelets [1].