UCLA

COMPUTATIONAL AND APPLIED MATHEMATICS

Sparse Approximate Inverse Smoother for Multi-Grid

Wei-Pai Tang
W. L. Wan

March 1998

CAM Report 98-18

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

SPARSE APPROXIMATE INVERSE SMOOTHER FOR MULTI-GRID

WEI-PAI TANG * AND W. L. WAN †

Abstract. Various forms of sparse approximate inverses (SAI) have been shown to be useful techniques for preconditioning. Their potential usefulness in a parallel environment has motivated much interest in recent years. However, the capability of an approximate inverse in eliminating the local error has not yet been fully exploited in multi-grid algorithms. A careful examination of the iterative matrices of these approximate inverses indicates their superiority in smoothing the high frequency error in addition to their inherent parallelism. We propose a new class of sparse approximate inverse smoothers in this paper and present their analytic smoothing factors for constant coefficient PDEs. The several distinctive features to make this technique special are:

- By adjusting the quality of the approximate inverse, the smoothing factor can be improved accordingly. For hard problems, this is useful.
- In contrast to the ordering sensitivity of other smoothing techniques, this technique is ordering independent.
- In general, the sequential performance of many superior parallel algorithms is not very competitive. This technique is useful both in parallel and sequential computations.

Our theoretical and numerical results have demonstrated the effectiveness of this new technique.

Key words. Sparse approximate inverse, smoother, multi-grid.

AMS subject classifications. 15A09, 15A23, 65F10, 65F50, 65Y05,

1. Introduction.

The effectiveness of multi-grid is based on two main components: *smoothing* and *coarse grid correction*. The smoothing process, usually carried out by a relaxation method, damps away the high frequency error components. The coarse grid correction process, carried out by an interpolation, approximates the low frequency components on the coarser grids. The idea is that the combination of the two will result in a significant error reduction independent of the problem size [19] and hence lead to an efficient solution procedure.

Relaxation methods such as Richardson, Jacobi and Gauss-Seidel are oftenly used as smoothers for multi-grid, although other iterative methods, for instance, ILU [20, 37] and (preconditioned) conjugate gradient [1, 6], or even ODE solvers [23] such as Runge Kutta methods, have also been used for specific problems. The relaxation methods are particularly useful for multi-grid since they are simple, and all have the common property of removing the high frequency error components [4, 19].

In practice, Gauss-Seidel is usually the most effective smoother among other relaxation methods. A drawback, however, is that Gauss-Seidel is a very sequential algorithm. A parallel version may be obtained by a special ordering of the unknowns, for example, red-black ordering for the five-point stencil operator on a square grid. The Jacobi method, on the other hand, is a very parallel method, but its smoothing efficiency is not as good as Gauss-Seidel. It is also well known that Gauss-Seidel smoothers do not work well for anisotropic problems and discontinuous coefficient problems. In this paper, we propose a new class of smoothers derived from sparse

* Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. This work was supported by the Natural Sciences and Engineering Research Council of Canada, by the Information Technology Research Centre, which is funded by the Province of Ontario and by NASA under contract NAS 2-96027 between NASA and the Universities Space Research Association (USRA).

† Department of Mathematics, University of California, Los Angeles, CA 90095-1555, USA. This work is supported by the grant under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W07405-Eng-48, and by the Alfred P. Sloan Foundation as a Doctoral Dissertation Fellow.

approximate inverse preconditioners. It has a smoothing efficiency similar to Gauss-Seidel and it is independent of ordering. Moreover, for hard problems, we can improve the smoothing efficiency by adaptively adjusting the quality of the approximate inverse, for instance, by adding more nonzeros. Consequently, this new technique is more robust than Gauss-Seidel smoothers. Our numerical testing verifies this statement.

We remark that Huckle [22] independently experimented with a sparse approximate inverse smoother for multi-grid by modifying the standard Gauss-Seidel iteration. Specifically, instead of using the exact inverse of the lower triangular matrix, he used a sparse approximate inverse of it computed by the techniques described in [17]. In our approach, we do not restrict ourselves to Gauss-Seidel only. Indeed, we replace the Gauss-Seidel smoother completely by a sparse approximate inverse smoother. The resulting multi-grid is efficient, and we have more flexibility in improving the smoothing quality for hard problems.

In Section 2, we describe the construction of the sparse approximate inverse smoother. In Section 3, we analyze the smoothing property of the proposed smoother analytically and numerically for constant coefficient PDEs. Some techniques to improve the smoothing character for the anisotropic problems are presented. Finally, in Section 4, we show the effectiveness of the sparse approximate inverse as a smoother for multi-grid by a variety of problems including anisotropic problems, discontinuous coefficient problems and unstructured grid problems.

2. SAI smoother. Various techniques have been proposed for an effective sparse approximate inverse preconditioner [2, 3, 8, 10, 15, 17, 26, 27, 34]. However, the goal of constructing an effective smoother is actually very different from finding a good preconditioner. For a powerful preconditioner, the capability of removing the lower frequency error is essential. In contrast, a good smoother should damp the high frequency errors effectively. In this respect, much of the weakness [34] of sparse approximate inverse preconditioners becomes the strength of the SAI smoother. Our new proposal is to explore them to construct a powerful smoother.

Most sparse approximate inverse approaches seek a sparse matrix M so that the error

$$\mathcal{E} = AM - I$$

is minimized in some measure. The sparsity constraint often limits the effectiveness of M as a preconditioner due to the locality of the sparse approximation. The lack of the global approximation has created many difficulties for a powerful preconditioner. Various additional techniques are required to improve the quality of a SAI preconditioner [7, 34]. The requirement for a good smoother, on the other hand, can take advantage of the locality of the sparse approximation. Another interesting consequence is that many of the techniques to improve the quality of a SAI preconditioner are no longer relevant to a good smoother. In this paper, a more primitive form of approximate inverse – *local least-squares approximation* is studied. It turns out that this simple form of approximate inverse performs well as a smoother, and the cost of computing this form of SAI is the cheapest among all others.

A sparse matrix A can be represented by a digraph $\mathcal{G} = (\mathcal{O}, \mathcal{E})$ [13]. For multigrid methods, the graph often is the mesh on the PDEs solution domain. Define $\mathcal{L}_k(o_i)$, the k-level neighbor set of node o_i, the nodes which are a distance $k+1$ or less from o_i. The use of the level concept to define the sparsity pattern for incomplete factorizations is widely used for ILU preconditioning[18, 11]. The 0-level neighbor set

$\mathcal{L}_0(o_i)$ contains all the nodes which directly connect to node o_i. For PDE problems, $\mathcal{L}_0(o_i)$ is the set of the mesh points from the discretization. Similarly, define $\mathcal{W}_k(o_i)$, the k^{th}-wavefront of node o_i, as the set of nodes which are a distance $k+1$ from node o_i. If we assign the values of the elements in a row[1] of inverse to the corresponding location in the graph and the surface plot is the picture of the discrete Green's function at node o_i.

It can be shown that the elements in a discrete Green's function decay in a wavefront fashion for many problems [32]. In particular, $\mathcal{L}_k(o_i)$ includes the k most influential wavefronts. That is the motivation to choose $\mathcal{L}_k(o_i)$ for the sparsity pattern to approximate the discrete Green's function at node o_i. The computation of these locations is also cheap. A good choice of the sparsity pattern has a big impact on the quality of the SAI.

The rectangular submatrix[2]

$$A(\mathcal{L}_k(o_i), \mathcal{L}_l(o_i)) \tag{2.1}$$

is defined as the k,l-level local matrix of node o_i. This matrix takes rows corresponding to nodes $\mathcal{L}_k(o_i)$ and columns corresponding to nodes $\mathcal{L}_l(o_i)$ from A.

We introduce the *(k,l)-level local least-squares approximation* of an inverse as: for each node o_i (row) of the approximation, the least squares solution[3] x of

$$A^T(\mathcal{L}_k(o_i), \mathcal{L}_l(o_i))x = e_{o_i} \tag{2.2}$$

is taken for the nonzero values at the corresponding location $\mathcal{L}_k(o_i)$ of the sparse approximation of the discrete Green's function of node o_i. More precisely, we inject each element of the least squares solution x into a zero vector of size of the matrix A at the corresponding location in $\mathcal{L}_k(o_i)$ and use this sparse row to approximate the row o_i of the inverse A^{-1}. In (3.1), a $(0,1)$-level case is presented. It is clear that, the sparsity pattern of the approximate inverse is related to the locations of $\mathcal{L}_k(o_i)$ and the approximation range with which we are concerned to the locations of $\mathcal{L}_l(o_i)$. A higher level k implies a denser approximation while a higher level l provides a approximation which is good for more neighbors. The $(0,0)$-level approach was used by Benson [2] originally, and was not a good choice for both preconditioner and smoother. Our experiences indicate that the $(0,1)$-level local least-squares approximation is the most cost-effective smoother in general. However, for anisotropic and/or other difficult problems, a higher level SAI smoother is required. This adaptable quality makes the SAI smoother a robust technique.

To compute these local least squares solutions is an easy task and can be implemented in parallel effectively. For any constant coefficient PDE on a regular mesh, we may even use the local least squares solution for one interior grid point for all nodes to save the cost of computations. Our test and analysis indicate that this simple approach does not bring any noticeable penalty in performance. Various techniques can also be adopted to save the cost to compute the SAI. A follow-up paper will discuss these in lengths. For example:

- the shape of the Green's function for different positions inside the solution region does not vary much for a PDE with constant coefficients. When an

[1] Corresponding to the node o_i

[2] The Matlab notation is adopted for extracting a submatrix from a given matrix A

[3] In (2.2), e_{o_i} is a unit basis vector of size $|\mathcal{L}_l(o_i)|$ with one in the o_i position and zeros in the rest of the location in $\mathcal{L}_k(o_i)$. An example is given for (0,1)-level in (3.1)

unstructured mesh is used, we may compute one approximation of the discrete Green's function accurately and use interpolation to obtain the approximation for all other mesh point.

- For variable coefficient PDEs, the use of one local least squares solution for several of its neighbors is feasible.
- For anisotropic problems, a higher level SAI smoother is required. However, the cost of the local least-squares problems grows rapidly. A prior-drop technique [34] can significantly reduce the computations.

3. Smoothing factor analysis. We present an analysis of the smoothing factors for constant coefficient PDE on a two dimensional rectangular region in this section.

Consider the following constant coefficient second order elliptic PDE

$$\begin{aligned} w_1 U_{xx} + w_2 U_{yy} + sU_x + tU_y &= f(x,y), \qquad (x,y) \in \Omega, \\ U|_\Gamma &= g(x,y) \end{aligned}$$

on an $m \times n$ regular grid; the resulting matrix is a 5-diagonal matrix A of size $(m \times n)^2$, using the conventional finite difference or finite volume method. Assume the 5-point stencil of the discretization is

$$bu_{i+1,j} + du_{i,j+1} - au_{i,j} + cu_{i-1,j} + eu_{i,j-1} = h^2 f_{i,j}, \qquad 1 \le i \le m, \quad 1 \le j \le n$$

The $(0,1)$-level local least squares problem for an interior grid point is:

$$\begin{pmatrix} a & & & & c \\ & a & & & d \\ & & a & & b \\ & & & a & e \\ c & d & b & e & a \\ c & & & & \\ d & c & & & \\ & d & & & \\ & b & d & & \\ & & b & & \\ & & e & b & \\ & & & e & \\ e & & & c & \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.1}$$

The solution x_i is used for the stencil of our simplified $(0,1)$-level SAI smoother. Notice that the $(0,1)$-level local least squares solutions near the boundary may different from (3.1). For the analysis and cost-saving purpose, we use the constant stencil for all grid node for the simplified local least-squares SAI smoother. The resulting matrix of this smoother is also a five-diagonal matrix. Our numerical testing indicates that the impact of this simplification on the smoothing factor is negligible, see Fig. 3.1.

The smoothing factor analysis is based on the following theorem.

THEOREM 3.1. *Given two tridiagonal matrices*

$$B = tridiag(b, a', c)_{n \times n}, \quad C = tridiag(d, a'', e)_{m \times m}$$

The eigenvalues of the matrix

$$B \otimes I_{m \times m} + I_{n \times n} \otimes C \tag{3.2}$$

are

$$a+2\sqrt{bc}\cos\frac{k\pi}{n+1}+2\sqrt{de}\cos\frac{j\pi}{m+1},\quad 1\le k\le n;\quad 1\le j\le m$$

where $a=a'+a''$. The corresponding eigenvectors are

$$\left\{\sin\frac{ks\pi}{n+1}\sin\frac{jt\pi}{m+1}\right\},\quad 1\le s\le n;\ 1\le t\le m.$$

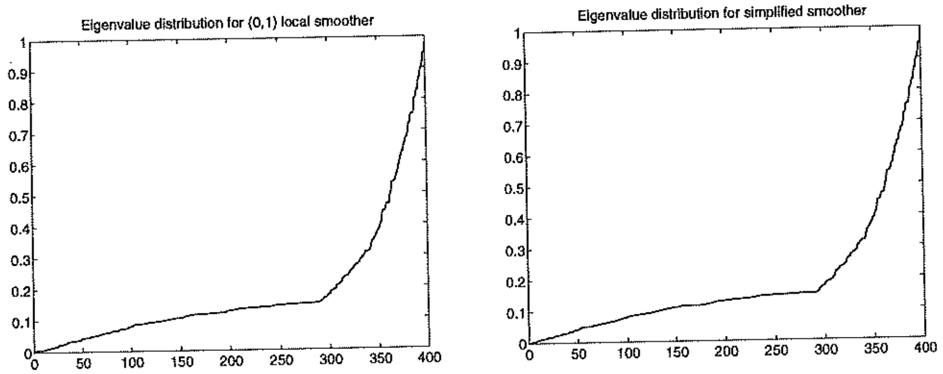
This result was cited in many books and articles in different forms, for example, "Iterative Solutions of Large Linear Systems" by David Young or [31]. Denote S the simplified smoother of matrix A. Notice that S can also be written in form of (3.2).

THEOREM 3.2. *The eigenvalues of the iterative matrix $I-SA$ are*

$$1-\Bigg(\Bigg((x_5+2\sqrt{x_1x_3}\cos\frac{n+1-k}{n+1}\pi+2\sqrt{x_2x_4}\cos\frac{m+1-j}{m+1}\pi\Bigg)$$
$$\Bigg(a+2\sqrt{bc}\cos\frac{k\pi}{n+1}+2\sqrt{de}\cos\frac{j\pi}{m+1}\Bigg)\Bigg)\Bigg)$$

where $1\le k\le n$; $1\le j\le m$.

This analysis can also be generalized to three dimensional problems. For the model problem on a 20×20 rectangular grid, we present the plots of the eigenvalue distributions for both the (0,1)-level local least squares smoother and its simplified smoother. The eigenvalue computation of the former is based on the Matlab *eig* function. The two distributions are so close so that we can not put the two pictures in one frame and to identify the difference. In Fig. 3.1, we present the eigenvalue distribution from the Matlab computation in (a), while the analytic solutions for the simplified smoother in (b).



(a) $(0,1)$-level local smoother

(b) Simplified smoother

FIG. 3.1. *Comparison of the eigenvalue distributions.*

For the single-direction anisotropic problem

$$\begin{aligned}100U_{xx}+U_{yy}&=f(x,y),\qquad (x,y)\in\Omega,\\ U|_\Gamma&=g(x,y),\end{aligned}\tag{3.3}$$

the noticeable difference in eigenvalue distributions between the simplified SAI and the local least-square SAI can be identified in the plot. However, the difference in distribution doesn't have any noticeable impact on the smoothing factor. Fig. 3.2 is the comparison of the two eigenvalue distributions. The solid line shows the eigenvalues for the simplified smoother.

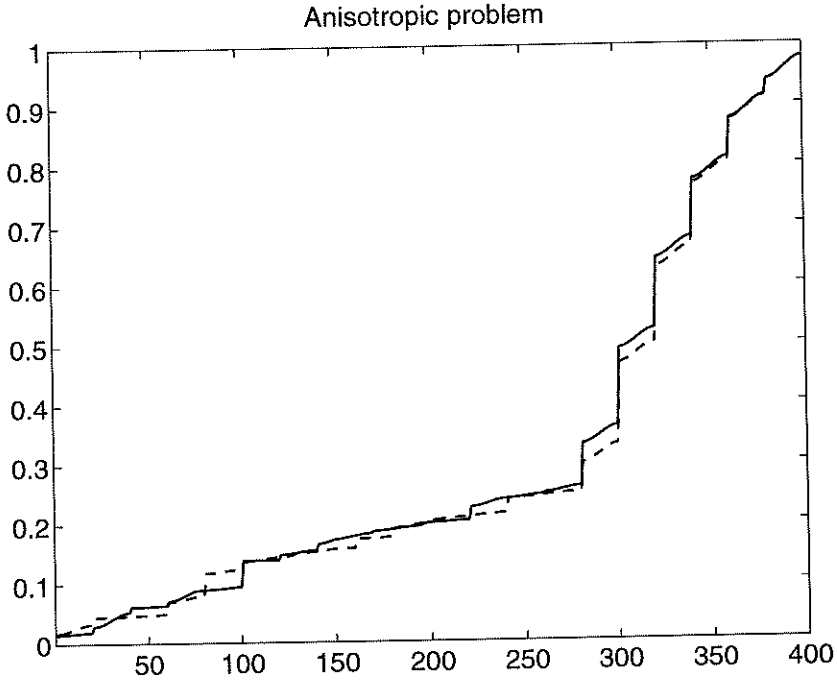


FIG. 3.2. *Comparison of the eigenvalue distributions.*

4. **Numerical results.** In this section, we demonstrate the effectiveness of the proposed SAI smoother. In all the examples, Dirichlet boundary conditions are used. In the multi-grid procedure, a V-cycle is used with one pre-smoothing and one post-smoothing, unless otherwise stated. Linear interpolation is used for structured grid problems and a specialized energy-minimizing interpolation [35, 36] is used for unstructured grid problems. The number of multi-grid levels is such that the coarsest grid is 3×3 for structured square grid problems, and four levels for general unstructured grid problems. The iteration was terminated when the relative residual norm was less than 10^{-8}. We are using zeros as the initial guess in all cases. Actually, we may report better numbers of iterations if a random initial guess is used. The results are summarized in the form of a table where the number of V-cycles and the average convergence rate are shown.

Example 1: We compare the performance of different Gauss-Seidel and SAI smoothers by solving the Laplace equation on a 33×33 squared grid. The results are shown in Table 4.1. We see that SAI is slightly better than GS in addition to the fact that it is much more easily parallelized. In this simple case, SAI is slightly worse than GS(rb) maybe because of the special geometry. We will see in Example 4 that SAI performs better for unstructured grid problems. As described in Section 2, SAI(1 pt) is the low cost implementation version of SAI for constant coefficient PDEs, and they essentially perform the same.

Smoothers	Iteration	Conv. Rate
GS	14	0.25
GS(rb)	10	0.16
SAI	12	0.19
SAI(1 pt)	13	0.21

TABLE 4.1

Laplace equation. GS: Gauss-Seidel with natural ordering, GS(rb) Gauss-Seidel with red-black ordering, SAI: sparse approximate inverse smoother, SAI(1 pt): SAI generated by a single interior point (see Section 2).

Example 2: Table 4.2 shows that result for a discontinuous coefficient problem with a square interface on a 33 × 33 squared grid. The jump across the interface is 10^4. We get a similar result as before. However, this time SAI(1 pt) does not work since the coefficient is not constant anymore, and in fact it changes enormously across the interface.

Smoothers	Iteration	Conv. Rate
GS	16	0.30
GS(rb)	13	0.22
SAI	13	0.22
SAI(1 pt)	*	*

TABLE 4.2

*A square interface problem. GS: Gauss-Seidel with natural ordering, GS(rb) Gauss-Seidel with red-black ordering, SAI: sparse approximate inverse smoother, SAI(1 pt): SAI generated by a single interior point (see Section 2). * indicates convergence took more than 50 iterations.*

Example 3: In this example, we present a number of variable coefficient problems including a helical spring problem, a viscosity PDE problem, and a discontinuous coefficient problem where the interfaces now consist of a horizontal and a vertical line; see [7] for details of each problem. Table 4.3 shows that SAI may converge while the other two do not.

Problems	Iteration			Conv. Rate		
	GS	GS(rb)	SAI	GS	GS(rb)	SAI
Variable coeff.	10	8	8	0.14	0.08	0.08
Helical spring	9	7	7	0.11	0.06	0.07
Viscosity	9	7	7	0.11	0.06	0.07
Discont. coeff.	*	*	22	*	*	0.40

TABLE 4.3

*Variable coefficient problems. GS: Gauss-Seidel with natural ordering, GS(rb) Gauss-Seidel with red-black ordering, SAI: sparse approximate inverse smoother. * indicates convergence took more than 50 iterations.*

Example 4: We show the effects of unstructured grids (Fig. 4.1) on the smoothers by solving the Laplace equation. In this case, red-black ordering is not defined. Instead, we use a *generalized* red-black ordering for general sparse matrices, which is essentially a greedy coloring algorithm. We still denote the resulting Gauss-Seidel by GS(rb). Table 4.4 shows that SAI is slightly better than GS(rb), which is better than GS.

Grids	Iteration			Conv. Rate		
	GS	GS(rb)	SAI	GS	GS(rb)	SAI
airfoil1	16	15	14	0.35	0.32	0.29
airfoil2	15	14	13	0.40	0.39	0.35
airfoil3	27	26	27	0.61	0.59	0.61
square	16	13	11	0.36	0.28	0.20
parc	18	11	10	0.36	0.18	0.16
spiral	12	9	8	0.22	0.11	0.08

TABLE 4.4

Laplace equation on different unstructured grids. GS: Gauss-Seidel with natural ordering, GS(rb) Gauss-Seidel with generalized red-black ordering, SAI: sparse approximate inverse smoother.

Example 5: We show how SAI smoother can improve the convergence of multi-grid for solving anisotropic coefficient PDEs. We consider two problems. The first problem is the single direction anisotropic problem in (3.3). The second problem has a more sophisticated anisotropy structure in both the x and y direction.

$$a(x,y)u_{xx} + b(x,y)u_{yy} = 1,$$

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

$$a(x,y) = \begin{cases} 100 & (x,y) \in [0,0.5] \times [0,0.5] \text{ or } [0.5,1] \times [0.5,1] \\ 1 & \text{otherwise.} \end{cases}$$

$$b(x,y) = \begin{cases} 100 & (x,y) \in [0,0.5] \times [0.5,1] \text{ or } [0.5,1] \times [0,0.5] \\ 1 & \text{otherwise.} \end{cases}$$

The results are shown in Table 4.5 and 4.6. The first three rows show the results of problem 1 on a 32×32, 64×64 and 128×128 grid, respectively. Similarly, row 4 and 5 show the results of problem 2. The last 4 rows show the results of problem 1 on unstructured grids.

As it is well known, multi-grid with Gauss-Seidel smoother is very slow, except for the first two unstructured grids. Similar results are also obtained for Gauss-Seidel with red-black ordering, and hence its results are omitted. For problem 1 on a squared grid, a usual technique to improve the multi-grid convergence is to use block relaxation methods. As indicated in the table, block Jacobi is quite effective for small grids, but eventually slows down for bigger grids. Moreover, it is costly to invert each block. The previous (0,1)-level SAI smoother is not very effective in this case. We improve the performance by using higher level SAI smoothers (Section 2). For higher levels, however, the approximate inverse is much denser. We control the amount of fill-in by dropping small elements. $SAI(k,\epsilon)$ denotes the smoother with $(k, k+1)$-level and element whose absolute values below ϵ are dropped at the end. We may also drop small elements in the matrix A before we compute the approximate inverse. $SAI(\epsilon_1, k, \epsilon_2)$ indicates that elements of size $< \epsilon_1$ are dropped in the matrix A, and elements of size $< \epsilon_2$ are dropped in the the approximate inverse.

Table 4.5 and 4.6 shows that (3,4)-level SAI performs similarly as block Jacobi, and (4,5)-level SAI performs better. Also, the SAI smoother does not deteriorate if

Grids	Iteration				
	GS	BJ	SAI(3,ϵ_2)	SAI(4,ϵ_2)	SAI(ϵ_1,4,ϵ_2)
problem 1 (32 × 32)	*	8	21	15	15
problem 1 (64 × 64)	*	27	27	20	19
problem 1 (128 × 128)	*	*	37	27	27
problem 2 (32 × 32)	*	*	49	13	13
problem 2 (64 × 64)	*	*	25	20	19
airfoil1	18	–	9	8	8
airfoil2	23	–	12	12	12
parc	46	–	11	9	10
spiral	*	–	18	21	21

TABLE 4.5

Anisotropic problems on different grids. GS: Gauss-Seidel with natural ordering, BJ: Block Jacobi, ϵ_1=2, ϵ_2= 0.0004 for the airfoil2 and spiral grid and 0.0008 for others. See text for the definitions of SAI(3,ϵ_2) and SAI(ϵ_1,4,ϵ_2).

the small elements of A are dropped first. Hence, we may afford to use higher level SAI if we drop small elements beforehand.

For problem 2, because of anisotropy in both directions, block Jacobi does not improve much the multi-grid convergence. The SAI smoothers, which do not require the concept of *direction*, perform as well as in problem 1.

For unstructured grid problems, blocks defined along the direction of the anisotropy no longer exist. Thus we do not test block Jacobi in these problems. Gauss-Seidel does surprisingly well on some of the grids, but still very slow on the others. SAI smoothers are effective on all the grids.

Grids	Conv. Rate				
	GS	BJ	SAI(3,ϵ_2)	SAI(4,ϵ_2)	SAI(ϵ_1,4,ϵ_2)
problem 1 (32 × 32)	*	0.09	0.52	0.40	0.40
problem 1 (64 × 64)	*	0.62	0.51	0.51	0.50
problem 1 (128 × 128)	*	*	0.64	0.53	0.53
problem 2 (32 × 32)	*	*	0.77	0.31	0.31
problem 2 (64 × 64)	*	*	0.59	0.51	0.48
airfoil1	0.38	–	0.10	0.09	0.09
airfoil2	0.61	–	0.30	0.37	0.37
parc	0.71	–	0.19	0.11	0.13
spiral	*	–	0.37	0.47	0.47

TABLE 4.6

Anisotropic problems on different grids. GS: Gauss-Seidel with natural ordering, BJ: Block Jacobi, ϵ_1=2, ϵ_2= 0.0004 for the airfoil2 and spiral grid and 0.0008 for others. See text for the definitions of SAI(3,ϵ_2) and SAI(ϵ_1,4,ϵ_2).

Example 6: Finally, we show that SAI, like other standard smoothers, give rise to a multi-grid method whose convergence rate is independent of the mesh size; see Table 4.7.

Acknowledgments. We would like to thank Tony Chan for his comments and discussions on the development of the sparse approximate inverse smoothers.

Grids	Iteration			Conv. Rate		
	GS	GS(rb)	SAI	GS	GS(rb)	SAI
32×32	14	10	13	0.25	0.16	0.21
64×64	14	11	13	0.25	0.17	0.22
128×128	14	11	13	0.25	0.17	0.22

TABLE 4.7

Laplace equation on grids of different mesh size. GS: Gauss-Seidel with natural ordering, GS(rb) Gauss-Seidel with red-black ordering, SAI: sparse approximate inverse smoother.

REFERENCES

[1] R. E. Bank, and C. C. Douglas, *Sharp Estimates for Multigrid Rates of Convergence with General Smoothing and Acceleration*, SIAM J. Num. Anal., 22:617–633, 1985.

[2] M. W. Benson, *Iterative solution of large scale linear systems*, M. Sc. Thesis, Lakehead University, Thunder Bay, Ontario, 1973.

[3] M. Benzi, C. D. Meyer, and M. Tůma, *A sparse approximate inverse preconditioner for the conjugate gradient method*, SIAM J. Sci. Comput., 17 (1996), pp. 1135–1149.

[4] A. Brandt, *Multi-Level Adaptive Solutions to Boundary-Value Problems*, Math. Comp., 31:333–390, 1977.

[5] R. Bridson, and W.-P. Tang, *Ordering, Anisotropy and Factored Sparse Approximate Inverses*, submitted to *SIAM J. Sci Comput.* .

[6] R. H. Chan, T. F. Chan and W. L. Wan, *Multigrid for Differential-Convolution Problems Arising from Image Processing*, CAM Report 97-20, Department of Mathematics, UCLA, 1997.

[7] T. Chan, W. -P. Tang and W. L. Wan, *Wavelet Sparse Approximate Inverse Preconditioners BIT*, 37, pp. 644–650, 1997.

[8] E. Chow and Y. Saad, *Approximate inverse techniques for block-partitioned matrices. SIAM J. Sci Comput.*, 18, pp. 1657–1675, 1997

[9] S. S. Clift, and W.-P. Tang, *Weighted graph based ordering techniques for preconditioned conjugate gradient methods*, BIT Vol. 35, No. 1, pp. 30-47, 1995 .

[10] J. Cosgrove, J. Diaz, and A. Griewank, *Approximate inverse preconditionings for sparse linear systems*, Intern. J. Computer Math, 44 (1992), pp. 91–110.

[11] D. F. D'Azevedo, P. A. Forsyth and W.-P. Tang, *Towards a cost effective ILU preconditioner with high level fill*, BIT 32, pp. 442-463, 1992.

[12] G. E. Forsythe and W. R. Wasow, *Finite difference methods for partial differential Equations*, Wiley, 1960.

[13] A. George and J. Liu, *Computer solution of large sparse positive definite systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[14] N. I. M. Gould and J. A. Scott, *On approximate-inverse preconditioners*, Tech. Rep. RAL-TR-95-026, June, 1995. Computing and Information System Dept., Atlas Center, Rutherford Appleton Lab., Osfordshire OX11 0QX, England.

[15] M. Grote and H. Simon, *Parallel preconditioning and approximate inverse on the connection machine*, Proceeding of the Scalable High Performance Computing Conference (SHPCC), 1992 Williamsburg, VA, IEEE Computer Science Press, pp. 76-83, 1992.

[16] ———, *Parallel preconditioning and approximate inverses on the connection machine*, in Sixth SIAM Conference on Parallel Processing for Scientific Computing II, R. S. et al, ed., SIAM, 1993, pp. 519–523.

[17] M. Grote and T. Huckle, *Parallel preconditioning with sparse approximate inverses.* SIAM J. Sci. Comput., 18, pp. 838-854, 1997.

[18] I. Gustafsson, A class of first order factorization methods. *BIT*, 18:142-156, 1978.

[19] W. Hackbusch, *Multi-grid Methods and Applications.* Springer-Verlag, Berlin, 1985.

[20] P. W. Hemker, The incomplete LU-decomposition as a relaxation method in multi-grid algorithms. In J. H. Miller, editor, *Boundary and Interior Layers-Computational and Asymptotic Methods, Dublin, Boole Press*, pages 306–311, 1980.

[21] T. Huckle and M. Grote, *A new approach to parallel preconditioning with sparse approximate inverses*, SCCM report, 1994.

[22] T. Huckle, *Sparse approximate inverses and multigrid methods*, Sixth SIAM Conference on Applied Linear Algebra, Snowbird, October29–November 1, 1997.

[23] A. JAMESON, *Solution of the Euler equation for two-dimensional flow by a multigrid method*, Appl. Math. and Comp., 13:327–355, 1983.

[24] I. E. KAPORIN , *New convergence results and preconditioning strategies for the conjugate gradient method*, Num. Lin. Alg. Appl., 1:179–210, 1994.

[25] L. Y. KOLOTILINA , *Explicit preconditioning of systems of linear algebraic equations with dense matrices*, J. Sov., Math, 43:2566–2573, 1988.

[26] 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, Proc of the IMACS International Symposium, Brussels, April 2-4, 1991, R. Beauwens and P. Groen, eds., 1992, pp. 311–312.

[27] L. Y. KOLOTILINA AND A. Y. YEREMIN, *Factorized Sparse Approximate Inverse Preconditionings I. Theory*, SIAM J. Matrix Anal. Appl., 14, 1993, pp. 45-58.

[28] 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, Proc of the IMACS International Symposium, Brussels, April 2-4, 1991, R. Beauwens and P. Groen, eds., 1992, pp. 83–84.

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

[30] J. RICE AND R. BOISVERT, *Solving Elliptic Problems Using ELLPACK*, Spring-Verlag, New York, 1985.

[31] U. SCHUMANN AND R. A. SWEET, *Fast Fourier transforms for direct solution of Poisson's equation with staggered boundary conditions* , J. Comput. Phys. 75(1988), pp. 123-137.

[32] W.-P. TANG, *Schwarz splitting and template operators*, PhD thesis, Stanford University, Computer Science Dept., Stanford, CA94305, 1987.

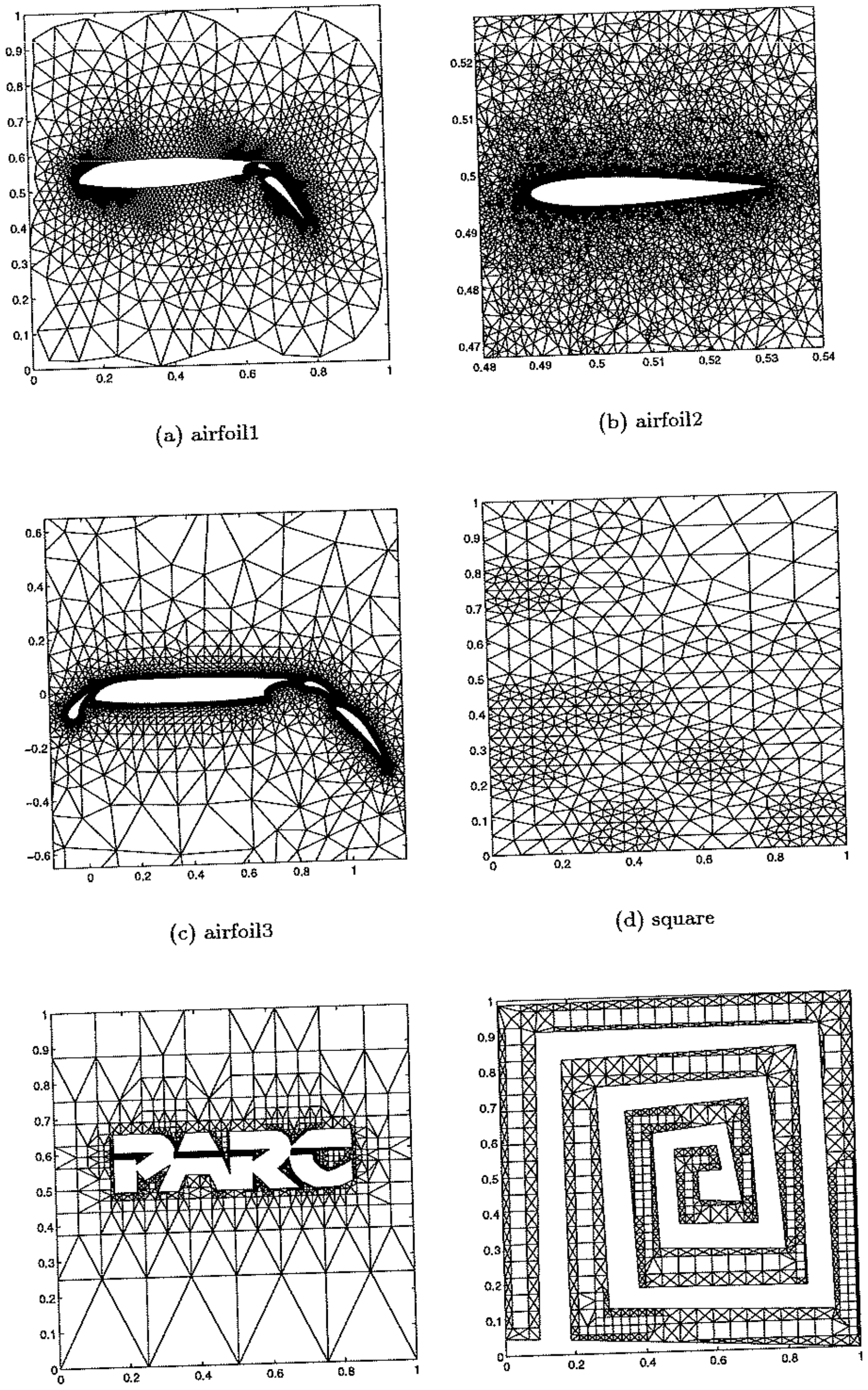
[33] W.-P. TANG, *Template Operators and Exponential Decay*, Dept. of Computer Science, Univ. of Waterloo, CS-90-46, 1990.

[34] W.-P. TANG, *Towards an effective approximate inverse preconditioner*, SIAM J. Sci. Comp., vol. 19, 1998.

[35] W. L. WAN, *An Energy-Minimizing Interpolation for Multigrid*, CAM Report 97-18, Department of Mathematics, UCLA, 1997.

[36] W. L. WAN, T. F. CHAN AND B. SMITH, *An Energy-Minimizing Interpolation for Robust Multigrid*, CAM Report 98-6, Department of Mathematics, UCLA, 1998.

[37] G. WITTUM, *On the robustness of ILU smoothing*, SIAM J. Sci. 1990. Comp., 10:699–717, 1990.



(a) airfoil1

(b) airfoil2

(c) airfoil3

(d) square

(e) parc

(f) spiral

FIG. 4.1. *The unstructured grids for Example 4 and 5.*