APPROXIMATE AND INCOMPLETE FACTORIZATIONS

TONY F. CHAN * * AND HENK A. VAN DER VORST †

Abstract. In this chapter, we give a brief overview of a particular class of preconditioners known as incomplete factorizations. They can be thought of as approximating the exact LU factorization of a given matrix A (e.g. computed via Gaussian elimination) by disallowing certain fill-ins. As opposed to other PDE-based preconditioners such as multigrid and domain decomposition, this class of preconditioners are primarily algebraic in nature and can in principle be applied to any sparse matrices. When applied to PDE problems, they are usually not *optimal* in the sense that the condition number of the preconditioned system will grow as the mesh size h is reduced, although usually at a slower rate than for the unpreconditioned system. On the other hand, they are often quite robust with respect to other more algebraic features of the problem such as rough and anisotropic coefficients and strong convection terms.

We will describe the basic ILU and (modified) MILU preconditioners. Then we will review briefly several variants: more fills, relaxed ILU, shifted ILU, ILQ, as well as block and multilevel variants. We will also touch on a related class of approximate factorization methods which arise more directly from approximating a partial differential operator by a product of simpler operators.

Finally, we will discuss parallelization aspects, including re-ordering, series expansion and domain decomposition techniques. Generally, this class of preconditioner does not possess a high degree of parallelism in its original form. Re-ordering and approximations by truncating certain series expansion will increase the parallelism, but usually with a deterioration in convergence rate. Domain decomposition offers a compromise.

1. Introduction and General References.

The general problem of finding a preconditioner for a linear system $Ax = b$ is to find a matrix M (the *preconditioner*) with the properties that M is a good approximation to A in some sense and that the system $Mx = b$ is much easier to solve than the original system. In other words, we are looking for a *nearby* problem which is *easier* to solve than the given one. The idea is to then solve this nearby problem repeatedly (with appropriately chosen right-hand-side vectors b) in such a way that the solution to the original problem can be obtained in the limit. Mathematically, we will solve the *preconditioned system* $M^{-1}Ax = M^{-1}b$ (or the symmetric version $(M^{-1/2}AM^{-1/2})(M^{1/2}x) = M^{-1/2}b$ when both A and M are symmetric positive definite) by standard iterative methods such as the conjugate gradient method, in which only the actions of A and M^{-1} are needed. Typically, the convergence rate is governed by the condition number of the preconditioned system $\kappa(M^{-1}A)$. The fundamental tradeoff is to choose M to reduce the condition number without increasing the cost of solving systems with M.

The choice of M varies from purely "black box" algebraic techniques which can be applied to general matrices to "problem dependent" preconditioners which exploits

* Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90024-1555. E-mail: chan@math.ucla.edu. The work of this author was partially supported by the National Science Foundation under contract ASC 92-01266, the Army Research Office under contracts DAAL03-91-G-0150 and DAAL03-91-C-0047 (Univ. Tenn. subcontract ORA4466.04 Amendment 1), and ONR under contract ONR-N00014-92-J-1890.

† Mathematics Institute, University of Utrecht, Utrecht, The Netherlands. Email: vorst@math.ruu.nl.

special features of a particular problem class. Obviously, the more special features we know and can exploit in a problem, the better we can construct a preconditioner for it. However, there is still a practical need for preconditioning techniques that can be applied to a general matrix. Incomplete factorization preconditioners are among the candidates for this. They can be thought of as modifications of Gaussian Elimination in which certain fill-ins are disallowed in order to ensure that the action of M^{-1} is inexpensive. These preconditioners can also be viewed as a bridge between direct methods such as Gaussian Elimination and classical relaxation methods such as Jacobi, Gauss-Seidel and SOR.

Before we begin the technical discussions, we shall first give some historical references. The earliest mention of these methods were in the 1960's — Buleev [27] and Oliphant [81, 82]. These authors proposed them as methods for solving discretizations of partial differential equations. The incomplete factorization methods fall in the more general class of matrix splitting techniques considered by Varga [100, 101], who provided a convergence analysis for Stieltjes and M-matrices. Evans [59] was believed to be the first to introduce the term *preconditioning* and he also considered the use of sparse LU factors as preconditioners. Methods of this type were of particular interest to researchers in the field of oil reservoir simulation. In particular, Stone [89] and Dupont-Kendall-Rachford [53] proposed approximate factorizations method for elliptic problems and gave some of the earliest theoretical convergence rate results. Beauwens [20] showed that the method in [53] is actually equivalent to the method of Buleev [27]. Wosnicki [108] applied these methods to solve diffusion equations arising from nuclear reactor simulations. He also introduced very useful matrix notations for the analysis and further developments along this line were made by Beauwens [19]. Axelsson [2] considered these methods as generalized relaxation methods. Meijerink and Van der Vorst [75] considered these methods as incomplete factorizations and they proved the existence of ILU preconditioners for M-matrices. Gustafsson [63] proposed a modified version of the ILU preconditioner with improved spectral properties. Finally, the paper of Kershaw [69] provided convincing numerical experiments to show the effectiveness of these methods.

Besides these historical references, there are several more easily accessible references which provide an introduction to this class of methods. Among these are the following books: Axelsson-Barker [8], Ortega [83], Golub and Van Loan [62], Dongarra, Duff, Sorensen and Van der Vorst [50], Pommerell [85], Barrett et al [16], Hackbusch [65] and Axelsson [7]. The following papers provide brief reviews: Axelsson [3, 4], Beauwens [21] and Il'in [67]. For more recent research, see the collections of papers in [61, 66]. The paper by Demmel, Heath and van der Vorst [43] contains a recent survey of parallel aspects of incomplete factorization methods.

2. Point ILU and MILU.

2.1. Point ILU. The basic idea in the point ILU preconditioner is to modify Gaussian elimination to allow fill-ins at only a restricted set of positions in the LU

Algorithm ILU for an n by n matrix A (following [7]):

for $r := 1$ step 1 until $n-1$ do
  $d := 1/a_{r,r}$
  for $i := (r+1)$ step 1 until n do
    if $(i,r) \in S$ then
      $e := d a_{i,r}$; $a_{i,r} := e$;
      for $j := (r+1)$ step 1 until n do
        if $(i,j) \in S$ and $(r,j) \in S$ then
          $a_{i,j} := a_{i,j} - e a_{r,j}$
        end if
      end (j-loop)
    end if
  end (i-loop)
end (r-loop)

FIG. 1. *Algorithm ILU for a general matrix A*

factors. Let the allowable fill-in positions be given by the index set S, i.e.

$$l_{i,j} = 0 \quad \text{if} \quad j > i \quad \text{or} \quad (i,j) \notin S; \quad u_{i,j} = 0 \quad \text{if} \quad i > j \quad \text{or} \quad (i,j) \notin S. \tag{1}$$

A commonly used strategy is to define S by:

$$S = \{(i,j) | \quad a_{i,j} \neq 0\}. \tag{2}$$

That is, the only non-zeros allowed in the LU factors are those for which the corresponding entries in A are non-zero. Let the preconditioner M be defined by the product of the resulting LU factors, i.e. $M = LU$. For M to be a good preconditioner, it must be a good approximation to A in some measure. A typical strategy is to require the entries of M to match those of A on the set S:

$$m_{i,j} = a_{i,j} \quad \text{if} \quad (i,j) \in S. \tag{3}$$

Even though the conditions (1) and (3) together are sufficient (for certain classes of matrices) to determine the non-zero entries of L and U directly, it is more natural and simpler to compute these entries based on a simple modification of the Gaussian elimination algorithm; see Fig. 2.1. The main difference from the usual Gaussian elimination algorithm is in the inner-most j-loop where an update to $a_{i,j}$ is computed only if it is allowed by the constraint set S.

After the completion of the algorithm, the incomplete LU factors are stored in the corresponding lower and upper triangular parts of the array A. It can be shown that the computed LU factors satisfy (3). These LU factors can be used in subsequent applications of the preconditioner; there is no need to explicitly form the product $M = LU$.

Remark 1: If all fill-ins are allowed, i.e. if S consists of all possible index pairs, then the above algorithm is simply the usual Gaussian elimination algorithm.

Remark 2: The incomplete LU factors of A are *not* the same as those obtained by setting those entries of the full LU factors which do not belong to S. The following example is from p. 211 of Ortega [83]: The Cholesky factor of

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 2 \end{pmatrix}$$

is

$$L = 2^{-1/2} \begin{pmatrix} 2 & & \\ 1 & 3^{1/2} & \\ 1 & -3^{-1/2} & 2^{3/2}3^{-1/2} \end{pmatrix}, \tag{4}$$

whereas the incomplete Cholesky algorithm (the obvious analog of the ILU algorithm given earlier) gives:

$$L = 2^{-1/2} \begin{pmatrix} 2 & & \\ 1 & 3^{1/2} & \\ 1 & 0 & 3^{1/2} \end{pmatrix}$$

which is different from setting the $(3,2)$ element of (4) to zero.

2.2. Point MILU. While the ILU preconditioner works quite well for many problems, it does not perform well for some PDE problems. For example, when ILU is applied to elliptic problems, its asymptotic (as the mesh size h becomes small) convergence rate is no better than that of the unpreconditioned A. This was already observed by Dupont, Kendall and Rachford [53] for elliptic PDEs and they proposed a simple modification which dramatically improves the performance as h tends to zero. We shall next describe the generalization of this *modified* ILU (MILU) preconditioner to a general matrix A due to Gustafsson [63].

The basic idea is extremely simple: in the condition (3) for ILU, the condition $m_{i,i} = a_{i,i}$ is removed and a new row sum condition is added. That is, (3) is replaced by:

$$\sum_{j=1}^{n} m_{i,j} = \sum_{j=1}^{n} a_{i,j} \quad \forall i \quad \text{and} \quad m_{i,j} = a_{i,j} \quad \text{if} \quad i \neq j \quad \text{and} \quad (i,j) \in S. \tag{5}$$

Again, for certain classes of matrices, the conditions (5) and (1) are sufficient to determine the LU factors in MILU directly. However, in practice it is easier to compute these LU factors by a simple modification of the ILU algorithm: instead of dropping the disallowed fill-ins in the ILU algorithm, these terms are added to the main diagonal of the same row; see Fig. 2.2. Again, it can be shown that the computed LU factors satisfy (5).

Note only the j-loop is different from the ILU algorithm. Again, the LU factors are stored in the lower and upper triangular parts of the array A respectively.

Algorithm MILU for an n by n matrix A

```
for r := 1 step 1 until n − 1 do
    d := 1/a_{r,r}
    for i := (r + 1) step 1 until n do
        if (i, r) ∈ S then
          e := a_{i,r} d; a_{i,r} := e;
          for j := (r + 1) step 1 until n do
              if (r, j) ∈ S then
                if (i, j) ∈ S then
                  a_{i,j} := a_{i,j} − e a_{r,j}
                else
                  a_{i,i} := a_{i,i} − e a_{r,j}
                end if
              end if
          end (j-loop)
        end if
    end (i-loop)
end (r-loop)
```

FIG. 2. *Algorithm MILU for a general matrix A*

2.3. ILU and MILU for Difference Operators. In the last two sections, we have given the ILU and MILU algorithms in a form that can be applied to general matrices. It turns out the algorithms can be simplified quite a bit if the matrix A correspond to a finite difference operator. We shall show now how to do this, following the derivations given in Dupont, Kendall and Rachford [53].

It is more natural to work with difference stencils instead of the corresponding matrices for this purpose. Thus let A correspond to the following 2D 5-point difference stencil in the usual *natural row-ise or column-wise ordering*:

$$A = \begin{pmatrix} & & t_{i,j} & & \\ & & | & & \\ c_{i,j} & --- & a_{i,j} & --- & b_{i,j} \\ & & | & & \\ & & s_{i,j} & & \end{pmatrix}.$$

The entries in the above stencil are exactly the non-zero elements on the row of the matrix A corresponding to the (i, j)-th grid point. In what follows, it is more convenient to think of computing an incomplete $LD^{-1}U$ factorization of A of a special form, where L is lower triangular, D is diagonal and U is upper triangular. Since the LU factors must have a sparsity pattern given by the condition (1), in the natural ordering (or in any ordering where the points corresponding to $c_{i,j}$ and $s_{i,j}$ are numbered before those corresponding to $b_{i,j}$ and $t_{i,j}$) they correspond to the following difference stencils:

$$L = \begin{pmatrix} & & \\ & & \\ f_{i,j} & --- & d_{i,j} \\ & & | \\ & & g_{i,j} \end{pmatrix}, D^{-1} = \begin{pmatrix} \\ \\ d_{i,j}^{-1} \\ \\ \\ \end{pmatrix}, U = \begin{pmatrix} p_{i,j} & & \\ | & & \\ d_{i,j} & --- & q_{i,j} \\ & & \\ & & \end{pmatrix}.$$

By multiplying the stencils out (or equivalently by multiplying the corresponding matrices), it can be verified that the product $M = LD^{-1}U$ corresponds the following stencil (i.e. the non-zero entries in the (i,j)-th row of M):

$$\begin{pmatrix} f_{i,j}p_{i-1,j}/d_{i-1,j} & --- & p_{i,j} & & \\ | & & | & & \\ f_{i,j} & --- & d_{i,j} + f_{i,j}q_{i-1,j}/d_{i-1,j} + g_{i,j}p_{i,j-1}/d_{i,j-1} & --- & q_{i,j} \\ & & | & & | \\ & & g_{i,j} & --- & g_{i,j}q_{i,j-1}/d_{i,j-1} \end{pmatrix}.$$

Now for both ILU and MILU, the conditions (3) and (5) require the off-diagonal non-zero elements of A to match the corresponding ones in M. This leads to the following formulas for the following off-diagonal entries of the LU factors:

$$f_{i,j} = c_{i,j}, \quad g_{i,j} = s_{i,j}, \quad p_{i,j} = t_{i,j} \quad q_{i,j} = b_{i,j}.$$

The only thing remaining is to how to determine $d_{i,j}$ and this is where ILU differs from MILU. For ILU, condition (3) requires that the diagonal entry of A to be equal to the diagonal entry of M. This leads to the following recurrence for $d_{i,j}$:

$$d_{i,j} = a_{i,j} - c_{i,j}b_{i-1,j}/d_{i-1,j} - s_{i,j}t_{i,j-1}/d_{i,j-1}.$$

Note that if we sweep over the computational grid in any order in which i and j are nondecreasing (e.g. in the natural ordering), the values of $d_{i-1,j}$ and $d_{i,j-1}$ will have been computed already by the time it comes to update $d_{i,j}$.

For MILU, the row sum condition in (5) requires that the sum of all the entries in the stencil of A (which corresponds to the entries in a given row of A) matches the sum of all the entries in M for the corresponding row. After a little algebra, this leads to the following recurrence for $d_{i,j}$:

$$d_{i,j} = a_{i,j} - c_{i,j}(b_{i-1,j} + t_{i-1,j})/d_{i-1,j} - s_{i,j}(t_{i,j-1} + b_{i,j-1})/d_{i,j-1}.$$

Remark 1. For application to second order elliptic problems, a small term ch^2 (where c is a constant and h is the mesh size) is often added to the right hand side of the above recurrence for MILU:

$$d_{i,j} = a_{i,j} - c_{i,j}(b_{i-1,j} + t_{i-1,j})/d_{i-1,j} - s_{i,j}(t_{i,j-1} + b_{i,j-1})/d_{i,j-1} + ch^2.$$

It is found that using a small positive value for c improves significantly the asymptotic convergence rate for many elliptic problems[53, 63].

Remark 2. It can easily be seen that the cost of computing the ILU and MILU factors involve only one sweep over the grid at a cost that is proportional to the total number of grid points. Moreover, the application of M^{-1} on a given vector v can be computed via $U^{-1}(L^{-1}v)$ by a forward sweep over the grid (corresponding to L^{-1}) followed by a backward sweep (corresponding to U^{-1}).

Remark 3. Let $A = \tilde{D} + \tilde{L} + \tilde{U}$ be the usual diagonal, strictly lower triangular and strictly upper triangular splitting of A. Then it is easy to see from the stencils for the $LD^{-1}U$ factors that the preconditioner M can be expressed as:

$$M = (D + \tilde{L})D^{-1}(D + \tilde{U}). \tag{6}$$

This form resembles that corresponding to the SSOR preconditioner, which is given by:

$$M_{SSOR} = (\tilde{D}/\omega + \tilde{L})(\tilde{D}/\omega)^{-1}(\tilde{D}/\omega + \tilde{U}).$$

Thus, the only difference between ILU, MILU and SSOR for 5-point difference operators is in the diagonal matrix D (in general $D \neq \tilde{D}$).

2.4. The Eisenstat Trick. Eisenstat [56] made an important observation that a substantial saving can be achieved in the application of M^{-1}, when M can be written in the form (6). The intuitive explanation is that there is a lot of cancellation of terms in the preconditioned matrix $M^{-1}A$ (or its symmetrized version) due to the fact that the off-diagonal entries of both M and A are the same.

To proceed, we first transform the left-preconditioned system $M^{-1}Ax = M^{-1}b$ into an equivalent split-preconditioned form $\tilde{A}\tilde{x} = \tilde{b}$ where

$$\tilde{A} = (D + \tilde{L})^{-1}A(D + \tilde{U})^{-1}, \quad \tilde{x} = (D + \tilde{U})x, \quad \tilde{b} = (D + \tilde{L})^{-1}b.$$

Note that $\tilde{b}$ can be computed once for all and x can be easily recovered from $\tilde{x}$ with very little cost. In solving $\tilde{A}\tilde{x} = \tilde{b}$ by a Krylov subspace method (e.g. the conjugate gradient method or the Chebychev method), it is required to compute the matrix-vector product $\tilde{A}v$ for a given vector v and this is where the saving occurs. To illustrate this, we write:

$$\begin{aligned}
\tilde{A}v &= (D + \tilde{L})^{-1}A(D + \tilde{U})^{-1}v \\
&= (D + \tilde{L})^{-1}[(D + \tilde{L}) + (\tilde{D} - 2D) + (D + \tilde{U})](D + \tilde{U})^{-1}v \\
&= [(D + \tilde{U})^{-1}v + (D + \tilde{L})^{-1}(\tilde{D} - 2D)(D + \tilde{U})^{-1}v + (D + \tilde{L})^{-1}v],
\end{aligned}$$

which can be computed efficiently by the following algorithm:

Eisenstat trick for evaluating $\tilde{A}v$:

1. Compute $y = (D + \tilde{U})^{-1}v$.
2. Compute $z = (\tilde{D} - 2D)y$.
3. Compute $w = (D + \tilde{L})^{-1}(z + v)$.
4. Then $\tilde{A}v = y + w$.

Note that the number of floating point operations is about the same as the number of nonzeros in A, i.e. the cost of just applying A to a vector v. In other words, the preconditioner does not cost substantial extra work.

2.5. Some Theoretical Results. We summarize here some of the basic known theoretical results for the ILU and MILU preconditioners. For further results and details, we refer the reader to [7], [65] and references therein. This is still an active area of research and we do not attempt to survey the current literature.

First is the question of *existence* of the ILU and MILU factorizations. That is, are the conditions (1) and (3) for ILU and (1) and (5) for MILU sufficient to guarantee the existence of the corresponding incomplete factorizations? It is easy to see that a sufficient condition is that Algorithm ILU and Algorithm MILU given earlier in this section do not encounter a zero pivot (i.e. $a_{r,r} = 0$.) It turns out that this can only be guaranteed for certain special classes of matrices. Specifically, Meijerink and Van der Vorst [75] proved that the ILU factorization for arbitrary fill patterns exists for M-matrices (i.e. matrices A with $a_{i,j} \leq 0, i \neq j$, and $A^{-1} > 0$ componentwise). Moreover, they proved that the splitting $A = M - N$ is *regular*, i.e. $\rho(M^{-1}N) < 1$, which implies that the fixed point iteration $x_{i+1} = M^{-1}(b - Ax_i)$ is convergent. They also proved that the incomplete decomposition (for M-matrices) without pivoting is al least as stable as complete decomposition without pivoting. Varga, Saff and Mehrman [102] extended the existence theory to cover the more general class of H-matrices.

The question of existence of the MILU factorization is more delicate. Generally speaking, the more diagonally dominant the matrix is, the more likely the MILU factorization exists; see Gustafson[64], Beauwens and Quenon [24] and Notay [79]. However, unlike for ILU, existence is not guaranteed for general M-matrices. Moreover, it is highly dependent on the ordering of the unknowns [57, 55]. For a specific existence theorem for 5-point difference operators, see Theorem 8.5.12 in [65].

Besides existence, it is desirable to have estimates of convergence rates. For a general matrix A, this is difficult to obtain. However, some results are known for specific classes of problems. For example, for A corresponding to a 5-point discretization of a second order elliptic problem in the natural ordering, it is proved in [53] that the condition number $\kappa(M^{-1}A) = O(h^{-2})$ for ILU (i.e. no improvement asymptotically from the condition number of A) whereas for MILU with $c > 0$, we have $\kappa(M^{-1}A) = O(h^{-1})$. For certain classes of matrices, Axelsson [6] has given rather accurate bounds on the relevant eigenvalues for matrices preconditioned with the generalized SSOR method (including standard ILU and MILU factorizations). From these, bounds on the number of preconditioned conjugate gradient iteration steps can be derived, which shows the effect of preconditioning.

Although the above condition number estimates lead to bounds on the convergence rate of the preconditioned iterative methods, these bounds can sometimes be too conservative because they do not reveal the *distribution* of the eigenvalues of the preconditioned system, which affects the convergence rate of the conjugate gradient method in a crucial way. For example, even though ILU does not improve the asymptotic condition number for elliptic problems, in many cases ([69, 76, 99]), it improves the eigenvalue distribution favourably with a corresponding reduction in the number of conjugate gradient iterations. Unfortunately, no rigorous theory exists which can predict the preconditioned eigenvalue distribution for ILU and MILU. However, in the case of elliptic problems, a *heuristic* method based on Fourier analysis (analogous to the Von Neumann stability analysis for time dependent PDEs) has been shown to give very good estimates of the eigenvalue distribution and the condition number $\kappa(M^{-1}A)$ for a large class of preconditioners including ILU, MILU and some of their variants. Moreover, the Fourier method can also be used to choose optimal parameters (e.g. the constant c in MILU) in the preconditioner. The method can be applied to matrices A which correspond to constant coefficient elliptic problems. It turns out the stencils of the incomplete LU factors in this case tend to a limiting constant coefficient one as one moves away from the boundary of the domain. Therefore, a Von Neumann type analysis can be applied to the preconditioned system. The reader is referred to [30, 49, 29] for more details.

3. Variants. Many variants of the basic ILU and MILU methods have been proposed in the literature. These variants are designed to either improve the performance or reduce the drawbacks of the basic methods. To describe all of these variants is beyond the scope of this chapter; we shall only describe the ideas of several more popular variants and give citation to the literature where more details can be found.

3.1. More Fill. A natural approach is to allow more fill-ins in the LU factor (i.e. a larger set S) than those allowed by the condition (2). Several possibilities have been proposed.

The most obvious variant is to allow more fills in specific positions in the LU factors, e.g. allowing more non-zero bands in the L and U matrices (i.e. larger stencils) [63, 8, 76].

Another variant is to replace the *drop-by-position* strategy in (2) by a *drop-by-size* one. That is, a fill is dropped if its absolute value is below a certain tolerance. This *drop tolerance* strategy was proposed by [84, 110, 11]. For applications to fluid flow problems, see [109, 40, 41].

A slightly different approach is to allow more than one *level* of fills [63, 76]. The level of a particular fill is defined recursively. Fills caused by original entries of A are defined to have level 1. The level of other fills are defined to be one higher than the level of the entries which contribute to it. The usual strategy is to keep only a small number of levels of fills; the intuition being that the higher level fills are not as important as the lower level ones.

3.2. Relaxed ILU. Even though MILU produces a better asymptotic condition number bound than ILU for elliptic problems, in practice MILU does not always perform better than ILU. Part of the reason is the favourable eigenvalue distribution for ILU and part of it has to do with the higher sensitivity of MILU to round-off errors [94]. This provides motivation for an *interpolated* version between ILU and MILU, first proposed in [10, 1]. Namely, in the MILU algorithm, the update of $a_{i,i}$ in the inner-most loop is replaced by:

$$a_{i,i} := a_{i,i} - \omega e a_{r,j},$$

where $0 \le \omega \le 1$ is a user specified relaxation parameter. Obviously, $\omega = 0$ and 1 correspond to ILU and MILU respectively. It has been observed empirically in [94], and verified using the Fourier analysis method [29], that a value of $\omega = 1 - ch^2$ gives the best results. The optimal value of c can be estimated and is related to the optimal value of c in the DKR method in [53] (see Remark 1 in Sec. 2.3). Van der Vorst [98] suggested to use the simple rule of $\omega = .95$ in practice.

Van der Vorst [93] suggested relaxed ILU methods for nonsymmetric A's and this has been improved and extended by Elman [58] who also studied the stability properties of these methods. Notay [80] gave strategies for choosing $\omega = \omega_{i,j}$ dynamically in order to improve the robustness and performance for anisotropic problems.

3.3. Shifted ILU. Manteuffel [74] considered ILU factorizations of diagonally shifted matrices. He proved the following two results:

1. If A is symmetric positive definite, then there exists a constant $\alpha > 0$, such that the ILU factorization of $A + \alpha I$ exists. This result shows that even though the ILU factorization of an SPD matrix is not guaranteed to exist, a suitably positively shifted version is.
2. If A is an M-matrix, then there exists a constant $\alpha < 0$ such that the ILU factorization of $A + \alpha I$ exists and is "close" to the MILU factorization of A in some sense.

3.4. ILQ. To avoid some of the inherent instability problems for ILU and MILU when applied to general non-SPD matrices (inherited from the fact that they derive from Gaussian elimination *without* pivoting), Saad [87] proposed to compute instead an incomplete $A = LQ$ factorization, where L is still lower triangular but Q is an incomplete version of an orthogonal matrix. One method is to compute Q by a Gram-Schmidt process applied to the rows of A, with an appropriate drop-tolerance strategy imposed to achieve sparsity in L. Since LL^T is the Cholesky factorization of AA^T in the *complete* LQ factorization of A, Saad suggested to applied a CG process to the preconditioned normal equations $L^{-1}AA^TL^{-T}y = L^{-1}b$, where $x = A^TL^{-T}y$. Experimental results can be found in Saad [87] and Donato [48].

In the same paper, Saad also suggested an incomplete LU factorization with partial pivoting to improve the numerical stability.

3.5. Multilevel and Re-ordered ILU. The performance of ILU and MILU is dependent on the ordering of the rows and columns of A [52]. Therefore various ordering strategies have been proposed to try to improve the performance. Some of these orderings are designed to improve the degree of parallelism in the application of the preconditioners and we shall discuss these in later sections. Here we describe several orderings which are designed primarily to improve the convergence rate of the preconditioners.

An approach that seems to work well for elliptic problems is to use a *multilevel ordering* of the grid points motivated by the multigrid method. For example, Brand and Heinemann [26] proposed a *repeated red-black* (RRB) ordering. Consider a regular 5-point finite difference grid for which the grid points are colored in the usual red-black

29	*61*	30	*62*	31	*63*	32	*64*
45	25	46	26	47	27	48	28
21	*57*	22	*58*	23	*59*	24	*60*
41	17	42	16	43	19	44	20
13	*53*	14	*54*	15	*55*	16	*56*
37	9	38	10	39	11	40	12
5	*49*	6	*50*	7	*51*	8	*52*
33	1	34	2	35	3	36	4

FIG. 3. *An RRB ordering, stopping after 1 coarse level (in italics)*

1	2	3	4	5	6
7	*28*	8	*29*	9	*30*
10	11	12	13	14	15
16	*31*	17	**36**	18	*32*
19	20	21	22	23	24
25	*33*	26	*34*	27	*35*

FIG. 4. *A hierarchical multigrid ordering*

fashion. The RRB ordering starts with all the red points (these are numbered say using the natural ordering), followed by half of the remaining black points corresponding to alternating horizontal grid lines. It can be easily verified that the remaining black points form a regular grid again with double the mesh spacing as the original one. The process can then be repeated recursively until a sufficiently coarse level is reached on which the usual natural ordering can be applied. Figure 3.5 shows an example of an RRB ordering. An ILU factorization is then performed on the grid according to this ordering of the grid points. Brand and Heineman proposed stopping the ILU factorization when the coarsest level grid has been reached at which point a complete LU factorization is performed. They showed that the resulting ILU preconditioner gives a condition number of order $O(h^{-1/2})$, a significant improvement over the naturally ordered ILU ($O(h^{-2})$) and MILU ($O(h^{-1})$). Ciarlet [37] showed that by choosing appropriately the coarsest level on which the complete LU factorization is performed, an $O(h^{-1/3})$ condition number can be achieved.

Recently, van der Ploeg[92] used a hierarchical multigrid ordering, an example of which is shown in Figure 3.5. The different levels are shown in different typefaces. An ILU factorization with a particular drop-tolerance strategy is then applied to A in this ordering. Numerical results and some analysis show that the condition number and the work per grid point are bounded independent of h.

Bank, Dupont and Yserentant [14] made the interesting observation that their hierarchical basis multigrid (HBMG) preconditioner can be viewed as an ILU method with a particular hierarchical ordering. Based on this observation, recently Bank and

Xu [15] extended the HBMG method to unstructured grids in which the coarse grids are not given as part of the fine grid. The Schur complement matrices on the coarse grids are not computed by a Gaussian elimination algorithm as is done in the usual ILU method, but is instead obtained using the underlying geometry of the mesh.

A related idea is to use ILU as a smoother in multigrid methods; see Wesseling [105, 104], Kettler [70] and Wittum [106].

Clift and Tang [38] proposed a "weighted-graph" ILU method for elliptic problems with anisotropic coefficients and high convection terms. The main ordering strategy is the recursive Cuthill-McGee method [60] but ties are broken using the size of the weights on the grid graph.

3.6. Block ILU and MILU. Block version of classical relaxation methods, such as block SSOR and ADI, have been used extensively for some time and block versions of ILU and MILU can be defined analogously.

For grid problems, the blocks usually correspond to points on grid lines in one coordinate direction, but the methods can be applied to any block structured matrix A. Thus let A be a block tridiagonal matrix:

$$A = tridiag(B_{i-1}, A_i, B_i^T).$$

The block LDU factorization of A can be written as:

$$A = (D + L)D^{-1}(D + L^T),$$

where

$$L = bidiag(B_i, 0), \quad \text{and} \quad D = diag(D_i),$$

and the diagonal blocks D_i satisfy the following recurrence:

$$D_1 = A_1, \quad D_i = A_i - B_{i-1}D_{i-1}^{-1}B_{i-1}^T.$$

In the complete block LU factorization, all the D_i's are full matrices. The main idea in the incomplete version is to modify the above recurrence in order to keep the D_i's sparse. For example, in the INV method due to Concus, Golub and Meurant [39], the term D_{i-1}^{-1} is replaced by its main tridiagonal part (it turns out this can be computed efficiently if D_{i-1} is itself tridiagonal). Since for 5-point operators, the B_i's are diagonal and the A_i's are tridiagonal, it follows from the recurrence for D_i that all the D_i's generated by the incomplete factorization remain tridiagonal. In [39], the authors also proposed the MINV method, which is anaologous to the MILU method, by modifying the diagonal entries of the INV tridiagonal approximation to D_{i-1}^{-1} so that the row sum is preserved. A Fourier analysis [34] shows that the condition number bounds for INV and MINV are respectively $O(h^{-2})$ and $O(h^{-1})$ and that the eigenvalue distributions are much better than ILU and MILU. Extensive experiments in [39] show that these block methods perform extremely well for 2D elliptic problems. Unfortunately, the natural extensions of these methods to 3D do not work as well. The reason may be that in 3D the D_i's are themselves block tridiagonal (corresponding to a 2D problem) and therefore the INV or MINV approximation can only be applied after an extra level of recursion.

Similar block methods have also been proposed by Axelsson [4, 5], Beauwens and Bouzid [23], Axelsson and Polman [12], Axelsson and Eijkhout [9], and Magolu [72, 73].

Instead of approximating D_i by approximating D_{i-1}^{-1} by its main tridiagonal part, Axelsson and Polman [13] determined an approximation for D_i implicitly by requiring that $D_i v_i = A_i - B_{i-1} D_{i-1}^{-1} B_{i-1}^T v_i$ on a set of suitably chosen vectors v_i which are chosen to be smooth grid functions (e.g. constant and linear). It is interesting to note that this idea of *probing* was used independently to construct interface preconditioners in domain decomposition methods by Chan and Resasco [35]. For a more recent account, see Chan and Mathew [33] and Axelsson [7].

The probing idea can be combined with the multigrid idea. Wittum [107] did this by choosing the v_i's to correspond to different frequencies. More recently, Chan and Vassilevski [36] considered a coarse grid approximation to D_{i-1}^{-1}.

3.7. PDE-based Approximate Factored Schemes. Besides the above mentioned matrix-based or grid-based incomplete factorization methods, there is another class of related PDE-based approximate factorization methods for solving time dependent PDEs. These so-called approximate factored marching schemes are used extensively in CFD, starting with the influential works of Beam and Warming [18] and Jameson and Turkel [68]. We shall describe these only briefly here.

Consider the PDE:

$$u_t + f(u)_x + g(u)_y = f.$$

Most temporal implicit schemes for solving it can be written in the semi-discrete form:

$$(I + \delta t(\partial_x A + \partial_y B))\delta u^n = RHS,$$

where A, B are Jacobians matrices of f, g.

An LU factored schemes is an approximation to the left-hand-side of the above equation:

$$(I + \delta t(\partial_x^- A_1 + \partial_y^- B_1))(I + \delta t(\partial_x^+ A_2 + \partial_y^+ B_2))\delta u^n = RHS,$$

where $A = A_1 + A_2, B = B_1 + B_2$ are appropriately chosen flux-splittings.

Note that after the above scheme is discretized using one-sided differences, the two factored terms on the left-hand-side give rise to lower and upper triangular matrices respectively. Hence the same issues concerning the data structure and parallelization for ILU and MILU apply to these factored schemes as well.

Finally, these factored schemes are often used by themselves or as smoothers for multigrid methods but typically not as preconditioners, and therefore convergence of the basic iteration often leads to stability limit on the time step δt.

6	7	8	9	10	11
5	6	7	8	9	10
4	5	6	7	8	9
3	4	5	6	7	8
2	3	4	5	6	7
1	2	3	4	5	6

FIG. 5. *The Hyperplane ordering*

4. Parallel Aspects. In this section, we discuss some approaches that have been proposed in the literature to efficiently implement the incomplete factorization methods on parallel computers. A shorter survey can also be found in [43].

There are two general approaches for parallelizing numerical methods:

1. extract maximal parallelism from a method which works well on sequential computer, *without* changing its numerical properties,
2. modify or approximate a good sequential method to increase the parallelism available, thus possibly degrading its numerical properties.

There is a fundamental difficulty when applying the above general principles to incomplete factorization methods. The major parallel bottleneck lies in the backsolves involving the triangular LU factors. The same bottleneck arises in computing the LU factors themselves but this occurs only once in the beginning of the iteration. Even though these backsolves possess some degree of parallelism which can be exploited, this is often not sufficient to efficiently exploit many parallel architectures, especially massively parallel ones. On the other hand, modifying or approximating the sequential method in order to increase the amount of parallelism invariably leads to slower convergence rates. This should not be too surprising; it is just an instance of the fundamental trade-off between parallelism (which prefers locality) and fast convergence rate (which prefers global dependence) which governs many genuinely globally coupled systems (e.g. elliptic PDEs) [28]. The goal is to make the right trade-off for a given architecture/algorithm configuration.

There are three basic methodologies to extract or increase the parallelism in ILU methods: re-ordering, series expansions and domain decomposition. We shall briefly discuss them next. We shall mostly use a 5-point difference operator and point ILU as our example. We note that the issue of parallelization are the same for both ILU and MILU since the data dependence are identical.

4.1. Re-ordering. The Hyperplane Ordering: Assume that the ILU factors have been computed using the *natural ordering* on a rectangular grid. Consider now the task of computing the product $y = L^{-1}v$. The goal is to find an ordering with which the components of y can be computed with maximal parallel efficiency. The fundamental data dependence is such that the computation at a grid point cannot commence until its south and west neighbors have been computed. Hence, the *hyperplane ordering* illustrated in Figure 5 exposes the maximal parallelism in this computation. The grid points with the same numbering (a hyperplane) can be all computed simultaneously in parallel, provided all the grid points with lower numbers have been updated already. Thus the computation can proceed from the lower left corner towards the upper right corner in sequences of hyperplanes. It is important to note that this hyperplane ordering is *equivalent* to the natural ordering in the sense that they produce the same result y. A similar ordering can be used for computing $U^{-1}v$, proceeding from upper right corner to the lower left.

It is obvious that if one generalizes this idea to a $d-$dimensional grid with n grid points in each direction, we can extract $O(n^{d-1})$ degree of parallelism. If the number of parallel processors is of the same order, then one can achieve good parallel efficiency. Thus, for a fixed number of processors, higher dimensional problems are easier to parallelise. On the other hand, potential degradation in performance can be caused by memory addressing with unequal stride for $d > 2$ and cache problems with the indirect addressing needed to access data on the hyperplanes when the grid is stored as a 2D array.

The use of hyperplane ordering has been investigated by Radicati and Vitaletti [86] for the IBM 3090. Numerical experiments on the CM2 can be found in Berryman et al [25], Chan, Kuo and Tong [32], and Tong [91]. Results for the Alliant FX can be found in [50]. The performance on the CM2 is very poor for 2D problems. The hyperplane ordering for 3D has been described in detail for vector computers in van der Vorst [98]. Results which show very high efficiency for the Cyber 205 (a notorious machine for indirect addressing) are reported in Schlichting and van der Vorst [88]. Results for the NEC SX2, Hitachi S810 and the Fujitsu VP200 can be found in van der Vorst [96]. Recently, Barszcz et al [17] gave a data mapping for the hyperplane ordering in 3D useful for distributed memory architectures and compare the performance on an 8-processor Cray Y-MP, a 128 processor Intel iPSC/860 and a 32K processor CM-2.

Multi-color Orderings: Since the degree of parallelism for ILU methods are limited in the natural ordering, a popular alternative is to use orderings that are designed to be more parallel. However, it must be emphasized that most of these orderings are *not* equivalent to the natural ordering, in the sense that the ILU factors computed using these are generally different from those generated using the natural ordering. Thus, the goal is to tradeoff the relatively fast and well-understood convergence rate of the natural ordering for orderings with a high degree of parallelism.

An example is the well-known red-black ordering for 5-point stencils in 2D. Because the red points depend only on the black points but not on each other, they can all be updated in parallel. Thus the degree of parallelism is $n^2/2$, a substantial increase from $O(n)$ for the natural ordering. However, since the data dependence are completely local and there is no global sharing of information, the convergence rate is poor. In fact, Kuo and Chan [71] proved that the condition number of the preconditioned system in the red-black ordering is only about 1/4 that of the *unpreconditioned* system for ILU, MILU and SSOR, with no asymptotic improvement as h tends to zero. These results are verified by experiments by Tong [90] on the CM2 which show that the increase in parallelism is far outweighted by the degradation in convergence rate on this machine.

One way to strike a better balance between parallelism and fast convergence is to use

```
 1   5  13   . . . 14   6   2
 9  17   .   . . .     18  10
21        . . . . . . .    22
 .  .  .  .  .  .  .  .  .  .
23   . . . . . . . . . .   24
11  19     . . . . . . 20  12
 3   7  15 . . .   16   8   4
```

FIG. 6. *A Multi-wavefronts Ordering*

```
 1  5  9 13   14 10  6  2
17 21 25 29   30 26 22 18
33 37 41 45   46 42 38 34

35 39 43 47   48 44 40 36
19 23 27 31   32 28 24 20
 3  7 11 15   16 12  8  4
```

FIG. 7. *A 4-front twisted factorization ordering*

more colors; see for example Doi and Lichnewsky [46, 47] for experiments on the Cray 2 and also Doi [44]. In principle, since the different colors are updated sequentially, using more colors decreases the parallelism but increases the global dependence and hence the convergence. The key is to choose the number of colors to match the architecture. For example, Doi and Hoshi [45] used up to 75 colors for a 76^2 grid on the NEC SX-3/14 and achieved 2 Gflops performance, which is much better than that for the hyperplane ordering.

Multi-wavefront Orderings: A different approach to increase parallelism is to use several hyperplane wavefronts to sweep through the grid, the idea being that all wavefronts can be updated in parallel. For example, van der Vorst [98] considered starting wavefronts from each of the four corners in a 2D rectangular grid, as illustrated in Figure 6. Earlier, Meurant [77] and van der Vorst [95] used a similar idea in which the grid is divided into equal parts (e.g. halves or quadrants) and each part is ordered in its own natural ordering. An example is shown in Figure 7. It can be shown that this leads to a *twisted* form of the LU factors in which each is half upper and half lower triangular.

Again, these orderings are not equivalent to the natural orderings and generally the increase in parallelism has to be balanced by the usual deterioration in convergence. Moreover, as pointed out by Eijkhout [55], the MILU method may even break down for some of these orderings (i.e. one of the $d_{i,j}$'s can be zero).

Other Orderings: Many other orderings have been proposed in the literature. For a comprehensive numerical study of the effect of the ordering on the convergence rate of incomplete factorization preconditioned CG methods for model elliptic problems, the reader is referred to the paper by Duff and Meurant [52]. Analysis of some of the orderings was given by Eijkhout [54].

Beauwens [22] defined a class of *S/P consistent* orderings which share some similarities with the wavefront ordering. In addition to being attractive for parallel implementation, these orderings also enhance the convergence behaviour.

4.2. Series Expansions. Instead of using an ordering with more parallelism, a quite different approach to increase the parallelism in the naturally ordered ILU method is to replace it by an *approximation* which can be evaluated more efficiently in parallel.

In order to illustrate this, consider the computation of $(I-L)^{-1}v$, which is needed in applying the preconditioner. Here we have assumed without loss of generality that the diagonal entries of the lower triangular factor has been scaled to unity. It can be easily proved that if the spectral radius $\rho(L)$ satisfies $\rho(L)<1$ and L is n by n strictly lower triangular, then we have the following two finite expansions:

$$\textbf{Neumann Expansion:} \quad (I-L)^{-1}v = (I+L+L^2+...+L^{n-1})v,$$

$$\textbf{Euler Expansion:} \quad (I-L)^{-1}v = (I+L^{2^s})(I+L^{2^{s-1}})...(I+L)v,$$

where $s = \lceil \log_2 n - 1 \rceil$. Note that $L^n = 0$. Each of the terms on the right-hand-side of the above expansions can be evaluated in parallel efficiently because they only involve repeated multiplication of v by sparse matrices. The idea is to then truncate the expansions but keeping enough terms so that the convergence rate is not too adversely affected.

Van der Vorst [97] was the first to use this idea when he applied a truncated Neumann expansion to the diagonal blocks (which correspond to grid lines) in the point ILU factorization in order to increase the degree of vectorization. In the same paper, he also used the Euler expansion (of low order).

Axelsson [3] was the first to propose using a truncated Euler expansion to block ILU. He precomputed the terms L^{2i} so that the backsolves can be done in s matrix-vector products.

Finally, a related method is the class of *polynomial preconditioners* in which A^{-1} is approximated by a low degree polynomial in A, chosen in some optimal manner (not necessarily via the Neumann expansion), see for example Dubois, Greenbaum and Rodrigue [51].

4.3. Domain Decomposition. In this general approach, the physical domain or grid is decomposed into a number of overlapping or non-overlapping subdomains on each of which an independent incomplete factorization can be computed and applied in parallel. The main idea is to obtain more parallelism at the subdomain level rather than at the grid point level. Usually, the interfaces or overlapping region between the subdomains must be treated in a special manner. The advantage of this approach is that it is quite general and can be used with different methods used within different subdomains.

Radicatti and Robert [86] used an algebraic version of this approach by computing ILU factors within overlapping block diagonals of a given matrix A. When applying the preconditioner to a vector v, the values on the overlapped region is averaged from the two values computed from the two overlapping ILU factors.

The twisted factorization of Meurant [77] discussed earlier can be thought of as a special version of this approach with two non-overlapping subdomains. More recently, Meurant [78] and de Sturler [42] used recursive versions of this approach, leading to what they call *repeated twisted factorizations*.

Chan and Goovaert [31] showed that the domain decomposition approach can actually lead to *improved* convergence rates, at least when the number of subdomains is not too large. The reason derives from the well-known divide and conquer effect when applied to methods with superlinear complexity such as ILU: it is more efficient to applied such methods to smaller problems and piece the global solution together.

Recently, Washio and Hayami [103] employed a domain decomposition approach on the NEC SX-3, with special treatment on the interface between the subdomains.

5. Concluding Remarks. In this chapter, we hope we have given the basics of the class of incomplete factorization preconditioners and their efficient parallel implementation. We have also tried to review some of the many extensions and variants that have been proposed in the literature. One of the key points we have emphasized is the trade-off between parallelism and convergence rate. Striking the appropriate balance for a given architecture and problem is the key to an efficient implementation.

In conclusion, we would like to characterize the class of incomplete factorization preconditioners as follows:

1. they are good "black box" preconditioners for general problems without well-understood analytical properties;
2. they are only well-defined for certain classes of matrices;
3. they offer flexibility for the user in choosing between direct methods and relaxation type methods; and
4. they have limited parallelism in their original form but re-ordering, approximations and domain decomposition offer mechanisms for trading off convergence rate for parallelism.

REFERENCES

[1] C. Ashcraft and R. Grimes. On vectorizing incomplete factorizations and SSOR preconditioners. *SIAM J. Sci. Stat. Comput.*, 9:122–151, 1988.

[2] O. Axelsson. A generalized SSOR method. *BIT*, 12:443–467, 1972.

[3] O. Axelsson. A survey of vectorizable preconditioning methods for large scale finite element matrix problems. In J.G. Verwer, editor, *Colloquium topics in applied numerical analysis.* CWI syllabus 4&5, Amsterdam, 1984.

[4] O. Axelsson. Incomplete block matrix factorization preconditioning methods. the ultimate answer? *J. Comp. Appl. Math.*, 12&13:3–18, 1985.

[5] O. Axelsson. A general incomplete block-matrix factorization method. *Lin. Alg. Appl.*, 74:179–190, 1986.

[6] O. Axelsson. Bounds of eigenvalues of preconditioned matrices. *SIAM J. Matrix Anal. Applic.*, 13:847–862, 1992.

[7] O. Axelsson. *Iterative solution methods.* Cambridge Univ. Press, 1994.

[8] O. Axelsson and A.V. Barker. *Finite element solution of boundary value problems. Theory and computation.* Academic Press, Orlando, Fl., 1984.

[9] O. Axelsson and V. Eijkhout. Vectorizable preconditioners for elliptic difference equations in three space dimensions. *J. Comp. Appl. Math.*, 27:299–321, 1989.

[10] O. Axelsson and G. Lindskog. On the eigenvalue distribution of a class of preconditioning matrices. *Numer. Math.*, 48:479–498, 1986.

[11] O. Axelsson and N. Munksgaard. Analysis of incomplete factorizations with fixed storage allocation. In D. Evans, editor, *Preconditioning Methods – Theory and Applications*, pages 265–293. Gordon and Breach, New York, 1983.

[12] O. Axelsson and B. Polman. On approximate factorization methods for block-matrices suitable for vector and parallel processors. *Lin. Alg. Appl.*, 77:3–26, 1986.

[13] O. Axelsson and B. Polman. A robust preconditioner based on algebraic substructuring and two-level grids. In W. Hackbusch, editor, *Robust multigrid methods. Proc., Kiel, Jan. 1988.*, pages 1–26. Notes on Numerical Fluid Mechanics, Volume 23, Vieweg, Braunschweig, 1988.

[14] R. Bank, T. Dupont, and H. Yserentant. The hierarchical basis multigrid method. *Numer. Math*, 52:427–458, 1988.

[15] R.E. Bank and J. Xu. The hierarchical basis multigrid method and incomplete LU decomposition. Technical report, U.C. San Diego, Dept. of Math., 1994.

[16] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods.* SIAM, Philadelphia, 1994.

[17] E. Barszcz, R. Fatoohi, V. Venkatakrishnan, and S. Weeratunga. Triangular systems for CFD applications on parallel architectures. Technical report, NAS Applied Research Branch, NASA Ames Research Center, 1994.

[18] R.M. Beam and R.F. Warming. An implicit scheme for the compressible Navier-Stokes equations. *AIAA J.*, 16:393–402, 1978.

[19] R. Beauwens. On the analysis of factorization procedures. Technical report, Internal report, Univ. Libre de Bruxelles, 1973.

[20] R. Beauwens. An order relation between factorization iterative methods. Technical report, Internal report, Univ. Libre de Bruxelles, 1973.

[21] R. Beauwens. Factorization iterative methods, M-operators and H-operators. *Numer. Math.*, 31:335–357, 1979.

[22] R. Beauwens. Approximate factorizations with S/P consistently ordered M-factors. *BIT*, 29:658–681, 1989.

[23] R. Beauwens and M. Ben Bouzid. Existence and conditioning properties of sparse approximate block factorizations. *SIAM Numer. Anal.*, 25:941–956, 1988.

[24] R. Beauwens and L. Quenon. Existence criteria for partial matrix factorizations in iterative methods. *SIAM J. Numer. Anal.*, 13:615–643, 1976.

[25] H. Berryman, J. Saltz, W. Gropp, and R. Mirchandaney. Krylov methods preconditioned with incompletely factored matrices on the CM-2. Technical Report 89-54, NASA Langley Research Center, ICASE, Hampton, VA, 1989.

[26] C. Brand and Z.E. Heinemann. A new iterative solution technique for reservoir simulation equations on locally refined grids. *SPE*, (18410), 1989.

[27] N.I. Buleev. A numerical method for the solution of two-dimensional and three-dimensional equations of diffusion. *Math. Sb.*, 51:227–238, 1960.

[28] T.F. Chan. Hierarchical algorithms and architectures for parallel scientific computing. In *Proc. ACM Int'l Conf. Supercomputing, Amsterdam, The Netherlands*, pages 318–329, 1990.

[29] T.F. Chan. Fourier analysis of relaxed incomplete factorization preconditioners. *SIAM J. Sci. Stat. Comput.*, 12:668–680, 1991.

[30] T.F. Chan and H. Elman. Fourier analysis of iterative methods for elliptic problems. *SIAM Review*, 31(1):20–49, 1989.

[31] T.F. Chan and D. Goovaerts. A note on the efficiency of domain decomposed incomplete factorizations. *SIAM J. Sci. Stat. Comp.*, 11(4):794–803, 1990.

[32] T.F. Chan, Jay C.C. Kuo, and C.H. Tong. Parallel elliptic preconditioners: Fourier analysis

and performance on the Connection Machine. *Comp. Phys. Commun.*, 53:237–252, 1989.

[33] T.F. Chan and T. Mathew. The interface probing technique in domain decomposition. *SIAM J. Matrix Anal. Applic.*, 13(1):212–238, 1992.

[34] T.F. Chan and G. Meurant. Fourier analysis of block preconditioners. Technical Report CAM 90-04, Dept. of Math., UCLA, Los Angeles, 1990.

[35] T.F. Chan and D. Resasco. A survey of preconditioners for domain decomposition. Technical Report DCS/RR-414, Dept. of Comp. Sci., Yale Univ., 1985.

[36] T.F. Chan and P. Vassilevski. A framework for block ILU factorizations using block size reduction. *Math. Comp.*, 1994. to appear.

[37] P. Ciarlet Jr. Repeated red-black ordering: a new approach. *Numer. Alg.*, 1994. To appear.

[38] S.S. Clift and W.P. Tang. Weighted graph based ordering techniques for preconditioned conjugate gradient methods. Technical report, Dept. of Comp. Sci., Univ. of Waterloo, May 1994.

[39] P. Concus, G.H. Golub, and G. Meurant. Block preconditioning for the conjugate gradient method. *SIAM J. Sci. Stat. Comput.*, 6:220–252, 1985.

[40] E.F. D'Azevedo, P.A. Forsyth, and W.P. Tang. Drop tolerance preconditioning for incompressible viscous flow. *Int'l J. Comp. Math.*, 44:301–312, 1992.

[41] E.F. D'Azevedo, P.A. Forsyth, and W.P. Tang. Ordering methods for preconditioned conjugate gradient methods applied to unstructured grid problems. *SIAM J. Matrix Anal. Applic.*, 13(3):944–961, 1992.

[42] E. de Sturler. IBLU preconditioners based on overlapping subdomains. In D.E. Keyes and J.C. Xu, editors, *Domain decomposition methods for partial differential equations, Proc. 7th Int. Symp. 1993.* AMS, 1994.

[43] J.W. Demmel, M.T. Heath, and H. van der Vorst. Parallel linear algebra. *Acta Numerica*, pages 111–198, 1993.

[44] S. Doi. On parallelism and convergence of incomplete LU factorizations. *App. Numer. Math.*, 7(5):417–436, 1991.

[45] S. Doi and A. Hoshi. Large numbered multicolor MILU preconditioning on SX-3/14. *Int'l J. Computer Math.*, 44:143–152, 1992.

[46] S. Doi and A. Lichnewsky. An ordering theory of incomplete *LU* factorizations on regular grids. Technical report, INRIA, 1990. submitted to SIAM J. Sci. Stat. Comput.

[47] S. Doi and A. Lichnewsky. A graph theory approach for analyzing the effects of ordering on ILU preconditioning. Technical Report 1452, INRIA, Rocquencourt, France, 1991.

[48] J.M. Donato. *Iterative methods for scalar and coupled systems of elliptic equations.* PhD thesis, Dept. of Math., Univ. Calif. at Los Angeles, 1991.

[49] J.M. Donato and T.F. Chan. Fourier analysis of incomplete factorization preconditioners for three-dimensional ansiotropic problems. *SIAM J. Sci. Stat. Comput*, 13:319–338, 1992.

[50] J.J. Dongarra, I.S. Duff, D.C. Sorensen, and Henk A. van der Vorst. *Solving Linear Systems on Vector and Shared Memory Computers.* SIAM, Philadelphia, PA, 1991.

[51] D.F. Dubois, A. Greenbaum, and G.H. Rodrigue. Approximating the inverse of a matrix for use in iterative algorithms on vector processors. *Computing*, 22:257–268, 1979.

[52] I.S. Duff and G.A. Meurant. The effect of ordering on preconditioned conjugate gradients. *BIT*, 29:635–657, 1989.

[53] T. Dupont, R. Kendall, and H. Rachford. An approximate factorization procedure for solving self-adjoint elliptic difference equations. *SIAM J. Numer. Anal.*, 5:559–573, 1968.

[54] V. Eijkhout. Analysis of parallel incomplete point factorizations. *Lin. Alg. Appl.*, 154–156:723–740, 1991.

[55] V. Eijkhout. Beware of unperturbed modified incomplete point factorizations. In Robert Beauwens and Pieter de Groen, editors, *Proceedings of the IMACS International Symposium on Iterative Methods in Linear Algebra, Brussels Belgium*, 1992.

[56] S.C. Eisenstat. Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM J. Sci. Stat. Comput.*, 2:1–4, 1981.

[57] H. Elman and E. Agrón. Ordering techniques for the preconditioned conjugate gradient method on parallel computers. *Comp. Phys. Comm.*, 53:253–269, 1989.

[58] H.C. Elman. A stability analysis of incomplete *LU* factorization. *Math. Comp.*, 47, 1986.

[59] D.J. Evans. The use of pre-conditioning in iterative methods for solving linear equations with symmetric positive definite matrices. *J. Inst. Maths.Applics.*, 4:295–314, 1968.

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

[61] G. Golub, A. Greenbaum, and M. Luskin (eds). *Recent advances in iterative methods.* Springer Verlag, Berlin, 1993.

[62] G.H. Golub and C.F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, 2nd edition, 1989.

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

[64] I. Gustafsson. Modified incomplete cholesky (MIC) methods. In D. Evans, editor, *Preconditioning Methods – Theory and Applications*, pages 265–293. Gordon and Breach, New York, 1983.

[65] W. Hackbusch. *Iterative solution of large sparse linear systems of equations.* Applied Math. Sci. Series, No. 95, Springer Verlag, 1994.

[66] Wolfgang Hackbusch and Gabriel Wittum (eds.), editors. *Incomplete Decompositions (ILU) - Algorithms, Theory, and Applications.* Vieweg, Braunschweig, 1993.

[67] V.P. Il'in. Incomplete factorization methods. *Sov. J. Numer. Anal. Math. Modelling*, 3:179–198, 1988.

[68] A. Jameson and E. Turkel. Implicit schemes and LU decomposition. *Math. Comp.*, 37:385–397, 1981.

[69] D.S. Kershaw. The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations. *J. Comp. Phys.*, 26:43–65, 1978.

[70] R. Kettler. Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods. In W. Hackbusch and U. Trottenberg, editors, *Multigrid methods: Proc., Koln-Porz, Nov. 1981*, pages 502–534. Lecture notes in Math. 1228, Springer Verlag, Berlin, 1982.

[71] J.C.C. Kuo and T.F. Chan. Two-color fourier analysis of iterative algorithms for elliptic problems with red/black ordering. *SIAM J. Sci. Stat. Comp.*, 11:767–793, 1990.

[72] M.M. Magolu. Modified block-approximate factorization strategies. *Numer. Math.*, 61:91–110, 1992.

[73] M.M. Magolu. Analytical bounds for block approximate factorization methods. *Lin. Alg. Applic.*, 179:33–57, 1993.

[74] T.A. Manteuffel. An incomplete factorization technique for positive definite linear systems. *Math. Comp.*, 34:473–497, 1980.

[75] 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*, 31:148–162, 1977.

[76] J.A. Meijerink and H.A. van der Vorst. Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems. *J. Comp.Phys.*, 44(1):134–155, 1981.

[77] G. Meurant. The block preconditioned conjugate gradient method on vector computers. *BIT*, 24:623–633, 1984.

[78] G. Meurant. Domain decomposition methods for partial differential equations on parallel computers. *Int. J. Supercomputing Appls.*, 2:5–12, 1988.

[79] Y. Notay. Solving positive (semi)definite linear systems by preconditioned iterative methods. In O. Axelsson and L.Yu. Kolotilina, editors, *Preconditioned Conjugate Gradient Methods*, pages 105–125, Nijmegen, 1989. Lecture Notes in Mathematics, vol. 1457.

[80] Y. Notay. DRIC: a dynamic version of the RIC method. *Numer. Lin. Alg. Applic.*, 1994. to appear.

[81] T.A. Oliphant. An implicit numerical method for solving two-dimensional time-dependent diffusion problems. *Quart. Appl. Math.*, 19:221–229, 1961.

[82] T.A. Oliphant. An extrapolation process for solving linear systems. *Quart. Appl. Math.*, 20:257–267, 1962.

[83] J.M. Ortega. *Introduction to vector and parallel solution of linear systems.* Plenum Press, 1988.

[84] O. Østerby and Z. Zlatev. *Direct methods for sparse matrices.* Number 157 in Lecture Notes in Computer Science. Springer Verlag, Berlin, Heidelberg, New York, 1983.

[85] C. Pommerell. *Solution of large unsymmetric systems of linear equations*, volume 17 of *Series in Micro-electronics.* Hartung-Gorre, Konstanz, 1992. Ph.D. thesis, ETH, Zurich.

[86] G. Radicati di Brozolo and M. Vitaletti. Sparse matrix-vector product and storage representations on the IBM 3090 with Vector Facility. Technical Report 513-4098, IBM-ECSEC, Rome, July 1986.

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

[88] J. Schlichting and H.A.van der Vorst. Solving 3D block bidiagonal linear systems on vector computers. *Journal of Comp. and Appl. Math*, 27:323–330, 1989.

[89] H. Stone. Iterative solution of implicit approximations of multidimensional partial differetntial equations. *SIAM J. Numer. Anal.*, 5:530–558, 1968.

[90] C.H. Tong. The preconditioned conjugate gradient method on the Connection Machine. In H. Simon, editor, *Scientific application of the Connection Machine*, pages 188–213. World Scientific, 1989.

[91] C.H. Tong. *Parallel preconditioned conjugate gradient methods for elliptic partial differential equations.* PhD thesis, Dept. of Comp. Sci., Univ. of Calif. at Los Angeles, 1990.

[92] A. van der Ploeg. *Preconditioning for sparse matrices with applications.* PhD thesis, Rijksuniversiteit Groningen, The Netherlands, 1994.

[93] H. van der Vorst. Iterative solution methods for certain sparse linear systems with a nonsymmetric matrix arising from pde-problems. *J. Comp. Phys.*, 44:1–19, 1981.

[94] H. van der Vorst. The convergence behaviour of preconditioned CG and CGS in the presence of rounding errors. In O. Axelsson and L.Y. Kolotilina, editors, *Preconditioned conjugate gradient methods.* vol. 1457, Lecture notes in Math., Springer Verlag, 1990.

[95] H. A. van der Vorst. Large tridiagonal and block tridiagonal linear systems on vector and parallel computers. *Parallel Computing*, 5:45–54, 1987.

[96] H. A. van der Vorst. ICCG and related methods for 3D problems on vector computers. *Computer Physics Communications*, 53:223–235, 1989.

[97] H.A. van der Vorst. A vectorizable variant of some ICCG methods. *SIAM J. Sci. Stat. Comput.*, 3:350–356, 1982.

[98] H.A. van der Vorst. High performance preconditioning. *SIAM J. Sci. Stat. Comput.*, 10:1174–1185, 1989.

[99] Henk A. van der Vorst and Gerard G. L. Sleijpen. The effect of incomplete decomposition preconditioning on the convergence of conjugate gradients. In Wolfgang Hackbusch and Gabriel Wittum, editors, *Incomplete Decompositions (ILU) - Algorithms, Theory, and Applications, Proceedings of the Eightth GAMM-Seminar, Kiel, January 24-26, 1992, Notes on Numerical Fluid Dynamics, Volume 41, Vieweg, Braunschweig*, pages 179–187, 1993.

[100] R.S. Varga. Factorization and normalized iterative methods. In R.E. Langer, editor, *Boundary problems in differential equations*, pages 121–142. Univ. of Wisconsin Press, Madison, 1960.

[101] R.S. Varga. *Matrix Iterative Analysis.* Prentice-Hall Inc., Englewood Cliffs, NJ, 1962.

[102] R.S. Varga, E.B. Saff, and V. Mehrman. Incomplete factorization of matrices and connection with h-matrices. *SIAM J. Numer. Anal.*, 17:787–793, 1980.

[103] T. Washio and K. Hayami. Parallel block preconditioning based on SSOR and MILU. *Numer. Lin. Alg. with Applic.*, 1994. to appear.

[104] P. Wesseling. A robust and efficient multigrid method. In W. Hackbusch and U. Trottenberg, editors, *Multigrid methods: Proc., Koln-Porz, Nov. 1981*, pages 614–630. Lecture notes in Math. 1228, Springer Verlag, Berlin, 1982.

[105] P. Wesseling. Theoretical and practical aspects of a multigrid method. *SIAM J. Sci. Stat. Comp.*, 3:387–407, 1982.

[106] G. Wittum. On the robustness of ILU smoothing. *SIAM J. Sci. Stat. Comp.*, 10:699–717, 1989.

[107] G. Wittum. An ILU-based smoothing correction scheme. In W. Hackbusch, editor, *Parallel algorithms for PDEs, Proc.,Kiel, Jan. 1990*, pages 228–240. Vieweg, Braunschweig, 1991. Notes on NUmerical Fluid Mechanics, volume 31.

[108] Z. Wosnicki. *Two-sweep iterative methods for solving large linear systems and their application to the numerical solution of multi-group multi-dimensional neutron diffusion equation.* PhD thesis, Inst. of Nuclear Research, Swierk, 1973.

[109] D.P. Young, R.G. Melvin, F.T. Johnson, J.E. Bussoletti, L.B. Wigton, and S.S. Samanth. Application of sparse matrix solvers as effective preconditioners. *SIAM J. Sci. Stat. Comp.*, 10(6):1186–1199, 1989.

[110] Z. Zlatev. *Computational methods for general sparse matrices.* Kluwer Acad. Pub., Dordrecht, Boston, London, 1991.