

CIMGS: AN INCOMPLETE ORTHOGONAL FACTORIZATION PRECONDITIONER*

XIAOGE WANG[†], KYLE A. GALLIVAN[†], AND RANDALL BRAMLEY[†]

Abstract. A new preconditioner for symmetric positive definite systems is proposed, analyzed, and tested. The preconditioner, compressed incomplete modified Gram–Schmidt (CIMGS), is based on an incomplete orthogonal factorization. CIMGS is robust both theoretically and empirically, existing (in exact arithmetic) for any full rank matrix. Numerically it is more robust than an incomplete Cholesky factorization preconditioner (IC) and a complete Cholesky factorization of the normal equations. Theoretical results show that the CIMGS factorization has better backward error properties than complete Cholesky factorization. For symmetric positive definite M-matrices, CIMGS induces a regular splitting and better estimates the complete Cholesky factor as the set of dropped positions gets smaller. CIMGS lies between complete Cholesky factorization and incomplete Cholesky factorization in its approximation properties. These theoretical properties usually hold numerically, even when the matrix is not an M-matrix. When the drop set satisfies a mild and easily verified (or enforced) property, the upper triangular factor CIMGS generates is the same as that generated by incomplete Cholesky factorization. This allows the existence of the IC factorization to be guaranteed, based solely on the target sparsity pattern.

Key words. preconditioner, iterative method, incomplete factorization, incomplete orthogonalization, M-matrix, symmetric positive definite systems, least squares problems

AMS subject classifications. 65F10, 65F20, 65F25, 65F50, 65G05, 65Y20

PII. S1064827594268270

1. Introduction. This work is motivated by the linear least squares problem of finding $x \in \mathbb{R}^n$ which minimizes the value of

$$(1) \quad \|b - Ax\|_2,$$

where $A \in \mathbb{R}^{m \times n}$, $m \geq n$, is a large sparse matrix of full rank and $b \in \mathbb{R}^m$ is an arbitrary vector. Such problems occur frequently in scientific and engineering applications such as linear programming [5], augmented Lagrangian methods for computational fluid dynamics (CFD) [12], and the natural factor method in structural engineering [9, 3].

Minimizing (1) by solving the normal equations $A^T Ax = A^T b$ is a common and often efficient approach because $A^T A$ is symmetric and positive definite. There are many well-developed and reliable methods, both direct and iterative, for solving such systems. In this paper, we present a new preconditioning method for solving the normal equations using the conjugate gradient (CG) iterative method, one which is applicable to more general symmetric positive definite systems. This approach allows the solution of extremely large least squares problems without explicitly forming the normal equations, which requires a potentially large number of floating point operations, and can introduce a loss of information from the original matrix A .

A well-known drawback of this approach is that the condition number of the normal equations is the square of the condition number of A . Orthogonal factorization methods [8] avoid this problem, but they require more floating point operations and if Q is required, as occurs for systems with sequential multiple right-hand sides, they

*Received by the editors May 23, 1994; accepted for publication (in revised form) July 31, 1995. This research was supported by NSF grants CDA-9309746 and CCR-9120105.

<http://www.siam.org/journals/sisc/18-2/26827.html>

[†]Computer Science Department, Indiana University, 215 Lindley Hall, Bloomington, IN 47405-4101 (xwang@cs.indiana.edu, gallivan@csrd.uiuc.edu, bramley@cs.indiana.edu).

potentially can require $\mathcal{O}(mn)$ storage, which is unacceptable for systems with large m . Because the rate of convergence of the CG algorithm is related to the condition number of the matrix to which it is applied, finding an effective preconditioner is crucial. Preconditioning methods that have been proposed and analyzed for the CG algorithm include column scaling, SSOR [4], incomplete Cholesky factorization [11], polynomial preconditioning [1, 2], and incomplete orthogonal factorization [14, 10, 18, 16].

When preconditioning a symmetric positive definite system $Bx = f$, the usual goal is to increase the clustering of the eigenvalues around the value of one. When $B = A^T A$ and the preconditioner is applied to A , a natural target is to make the preconditioned matrix \tilde{A} closer to orthogonal because then $\tilde{A}^T \tilde{A} \approx I$. This suggests using an incomplete orthogonal factorization. Existing incomplete orthogonal factorization preconditioners can be divided into two classes: incomplete Gram–Schmidt methods [14, 10, 16] which give a factorization $A = QR$ with Q not necessarily orthogonal, and incomplete Givens [18], which produce $A \approx QR$ with Q orthogonal. Incomplete Gram–Schmidt-type methods are, in general, robust because they can avoid numerical breakdown when A is full rank. Furthermore, they are effective in accelerating the convergence of CG. The notable drawback is that they are expensive in both floating point operations and storage because like full Gram–Schmidt factorization, they do not take full advantage of sparsity. One way of reducing computations is to use a numerical dropping technique to keep both the Q and R factors sparse, as is done in incomplete LQ (ILQ) [14] and the incomplete Givens method of [18]. The price these methods pay for efficiency is robustness because dropping small entries can lead to zero elements on the diagonal of R . Restart techniques have to be used for these methods to assure robustness. This paper introduces a new preconditioner called compressed incomplete modified Gram–Schmidt (CIMGS); as the name implies, CIMGS is based on an incomplete modified Gram–Schmidt (IMGS) factorization. CIMGS reduces the cost of computing an incomplete orthogonal preconditioner by “compressing” the information carried in A ’s column vectors into dot products of those vectors, which can be used to compute the same factor as the column vectors. In this way, the number of operations is reduced while the preconditioner’s robustness and effectiveness for the CG algorithm is preserved. Furthermore, unlike incomplete Cholesky (IC) factorization, in exact arithmetic the CIMGS factorization completes without breakdown for any full rank matrix A .

The next section describes the new algorithm and analyzes its properties. We show that CIMGS produces the same preconditioner (in exact arithmetic) as IMGS but requires far fewer computations than IMGS does. We also prove that when $A^T A$ is an M-matrix, CIMGS induces a regular splitting. The relationship between CIMGS and incomplete Cholesky factorization is discussed in detail in section 3. Numerical test results showing the effectiveness of *compression* are presented, along with comparisons among CIMGS, IC preconditioned CG, and direct methods. Conclusions and remarks are based on those results.

2. The CIMGS algorithm and its properties. To motivate the CIMGS algorithm, we first describe incomplete modified Gram–Schmidt (IMGS) factorization. Let $P_n = \{(i, j) \mid 1 \leq i < j \leq n\}$ be the set of all index pairs for the strictly upper triangular part of an $n \times n$ matrix, let $P \subseteq P_n$ be a set of index pairs, and assume that the matrix $A \in \mathbb{R}^{m \times n}$ has full column rank. The set P determines which elements of the target incomplete factor R will not be retained in the approximate factorization; that is, P is the set of drop positions. The IMGS factorization algorithm can be

derived easily from the modified Gram–Schmidt factorization of A by setting to zero during the factorization entries of R indexed by P .

Algorithm [Q, R] = IMGS [A, P]

```

begin
  for  $k = 1, 2, \dots, n$ ,
    (1)  $r_{kk} = \|a_k\|_2$ 
    (2)  $q_k = a_k / r_{kk}$ 
    for  $j = k + 1, k + 2, \dots, n$ 
      (3)  $r_{kj} = \begin{cases} 0 & (k, j) \in P \\ q_k^T a_j & (k, j) \notin P \end{cases}$ 
      (4)  $a_j = a_j - q_k r_{kj}$ 
    endfor
  endfor
end

```

Note that if $P = P_n$, then IMGS produces the diagonal matrix $R = \text{diag}(\|a_1\|_2, \dots, \|a_n\|_2)$. At the other extreme, $P = \emptyset$ gives a complete modified Gram–Schmidt factorization. In general, although the matrix Q need not be orthogonal, this factorization will always succeed in producing a nonsingular upper triangular factor R when A has full rank.

THEOREM 1. *If $A \in \mathbb{R}^{m \times n}$, $m \geq n$, has full rank, then IMGS applied with a drop set $P \subseteq P_n$ completes and produces a factorization $A = QR$, where R is an upper triangular matrix with positive diagonal elements and Q is a full rank matrix.*

Proof. Let $a_i^{(j)}$ denote the i th column of A after j steps of IMGS. From the algorithm we can see that $q_i = (a_i - q_1 r_{1i} - \dots - q_{i-1} r_{i-1,i}) / r_{ii}$, for $i = 1, 2, \dots, n$. The algorithm cannot complete if at some step k , $r_{kk} = \|a_k^{(k-1)}\|_2 = 0$. This means that we have $a_k^{(k-1)} = a_k - q_1 r_{1k} - \dots - q_{k-1} r_{k-1,k} = 0$, and so a_k is a linear combination of q_1, q_2, \dots, q_{k-1} . Therefore, the set of vectors $\{q_1, q_2, q_3, \dots, q_{k-1}, a_k\}$ is linearly dependent. However, the vectors $q_1, q_2, q_3, \dots, q_{k-1}$ form a basis of $\text{span}\{a_1, a_2, a_3, \dots, a_{k-1}\}$, and the set of vectors $\{a_1, a_2, a_3, \dots, a_k\}$ is linearly independent because A has full rank. Therefore, $\{q_1, q_2, q_3, \dots, q_{k-1}, a_k\}$ is independent, a contradiction. So if A has full rank, $r_{kk} \neq 0$ for $k = 1, 2, \dots, n$ and the factorization must exist. \square

More detailed studies of IMGS can be found in [16]. In general, IMGS is robust and effective at reducing the number of CG iterations. Its main weakness is the much higher cost of computing the preconditioner compared with other preconditioning methods.

The new algorithm CIMGS now described will produce in exact arithmetic the same preconditioner, while greatly reducing the computation cost. The basic idea is to *compress* the information in the column vectors of A into a dot product form without losing the information needed for the computation of the factor R . To understand the meaning of this *compression*, consider the relation between modified Gram–Schmidt factorization of A and complete Cholesky factorization of $A^T A$. In exact arithmetic, they both produce the same factor R . Modified Gram–Schmidt factorization works on A and *sees* only the column vectors. Cholesky factorization works on $A^T A$, the elements of which are dot products of the corresponding column vectors. After each step of each of the factorization methods, the relationship is maintained between the reduced matrices. Moreover, in the sparse case Cholesky factorization can be much more efficient than the modified Gram–Schmidt algorithm. The new algorithm CIMGS is designed to have the efficiency of Cholesky factorization, while producing the same triangular factor as IMGS.

Algorithmically, let $B = A^T A$. When A is a real matrix with full rank, B is symmetric positive definite. Given a drop set $P \subseteq P_n$, CIMGS generates the upper triangular matrix $R \in \mathbb{R}^{n \times n}$ as follows.

Algorithm [R]=CIMGS[B,P]
begin
for $k = 1, 2, \dots, n$,
 if $b_{kk} \neq 0$ **then**
(1) $b_{kk} = \sqrt{b_{kk}}$
(2) $r_{kk} = b_{kk}$
 for $j = k+1, k+2, \dots, n$
(3) $b_{kj} = b_{kj} / \sqrt{b_{kk}}$
(4) $r_{kj} = \begin{cases} 0 & (k, j) \in P \\ b_{kj} & (k, j) \notin P \end{cases}$
 endfor
 for $j = k+1, k+2, \dots, n$
 for $i = k+1, k+2, \dots, n$
(5) $b_{ij} = b_{ij} - b_{ki}b_{kj} \quad (k, j) \notin P \text{ or } (k, i) \notin P$
 endfor
 endfor
 else
(6) *quit (incomplete factorization cannot complete)*
 endif
endfor
end

In practice we take advantage of the symmetry of B by working only on the upper triangular part of B in step 5. Note that the structure of CIMGS is similar to the rank-1 update form of Cholesky factorization, with incompleteness introduced at steps 4 and 5. Just as with Cholesky factorization the CIMGS factorization can be implemented in other ways by deferring the rank-1 updates until they are needed. Note that B is overwritten by intermediate computations, which generate the factor R . The algorithm shows the target factor R being extracted from B at steps 2 and 4, but in practice R need not be stored separately.

First we show that in exact arithmetic CIMGS applied to $A^T A$ produces the same triangular factor as IMGS applied to A .

THEOREM 2. *Let $A \in \mathbb{R}^{m \times n}$, $m \geq n$, $\text{rank}(A) = n$, and let $B = A^T A$. Let R_{IMGS} be the triangular factor produced by IMGS applied to A with a given drop set P , and let R_{CIMGS} be the triangular factor produced by CIMGS applied to B with the same P . Then $R_{\text{IMGS}} = R_{\text{CIMGS}}$.*

Proof. We use induction on n . The $n = 1$ case is trivial. Supposing that the theorem is true for $k = n - 1$, we now prove it for $k = n$. Let $R_{\text{IMGS}} = R = [r_{ij}]$ and $R_{\text{CIMGS}} = S = [s_{ij}]$. After one step of IMGS,

$$r_{11} = \|a_1\|_2 = \sqrt{a_1^T a_1}, \quad r_{1j} = \begin{cases} 0, & (1, j) \in P, \\ \frac{a_1^T a_j}{r_{11}}, & (1, j) \notin P, \end{cases} \quad 2 \leq j \leq n$$

and columns 2 to n of A are updated by

$$a_j^{(1)} = a_j - \frac{r_{1j}}{r_{11}} a_1, \quad 2 \leq j \leq n.$$

After one step of CIMGS the first row of S is given by

$$s_{11} = \sqrt{b_{11}} = \sqrt{a_1^T a_1}, \quad s_{1j} = \begin{cases} 0, & (1, j) \in P, \\ \frac{b_{1j}}{\sqrt{b_{11}}}, & (1, j) \notin P, \end{cases} \quad 2 \leq j \leq n$$

and B is updated by $b_{ij}^{(1)} = b_{ij} - b_{1i}b_{1j}$ if $(1, j) \notin P$ or $(1, i) \notin P$, for $2 \leq i, j \leq n$. Clearly $r_{1j} = s_{1j}$ for all $1 \leq j \leq n$. The computation of the rest of R consists of applying IMGS to the matrix $A^{(1)} = \{a_2^{(1)}, a_3^{(1)}, \dots, a_n^{(1)}\}$. The computation of the rest of S consists of applying CIMGS to the matrix $B^{(1)} = (b_{ij}^{(1)})$, $2 \leq i, j \leq n$. Since

$$a_i^{(1)T} a_j^{(1)} = \begin{cases} a_i^T a_j, & (1, i) \in P \text{ and } (1, j) \in P, \\ a_i^T a_j - r_{1i}r_{1j}, & (1, i) \notin P \text{ or } (1, j) \notin P \end{cases}$$

and

$$b_{ij}^{(1)} = \begin{cases} b_{ij}, & (1, i) \in P \text{ and } (1, j) \in P, \\ b_{ij} - b_{1i}b_{1j}, & (1, i) \notin P \text{ or } (1, j) \notin P, \end{cases}$$

it can easily be seen that $A^{(1)T} A^{(1)} = B^{(1)}$. Using the induction hypothesis, the rest of R computed by IMGS is equal to the rest of S computed by CIMGS. By induction, the theorem is true for any n . \square

Since CIMGS is equivalent to IMGS, Theorem 1 also implies that CIMGS exists when $A^T A$ is positive definite. Since these results assume exact arithmetic, the natural next question to ask is how CIMGS is affected by rounding errors; that is, how does the “compression” technique affect the stability of CIMGS? Earlier analysis has shown that IMGS is less likely than modified Gram–Schmidt to break down due to possible numerical rank deficiency of A [16]. The next theorem bounds the rounding error incurred by CIMGS and shows that CIMGS is less likely to suffer numerical breakdown than complete Cholesky factorization is. The quantity $\lambda_{\min}(G)$ denotes the eigenvalue of minimal modulus for the matrix G .

THEOREM 3. *Let $B \in \mathbb{R}^{n \times n}$ be a symmetric positive definite matrix and μ be the machine precision. Let $P \subseteq P_n$ be a given drop set for the CIMGS algorithm. Let R be the triangular factor produced by CIMGS using the set P , and let U be the matrix of dropped elements, defined as $u_{jk} = b_{jk}$, $(j, k) \in P$, and $u_{jk} = 0$, $(j, k) \notin P$, at step 4 of the CIMGS algorithm. If*

$$(2) \quad C_1(n)\mu\kappa(B) \leq 1,$$

where $\kappa(B)$ is the condition number of B , then the CIMGS factorization completes and there is an error matrix E such that

$$(R + U)^T (R + U) = B + U^T U + E$$

and

$$\|E\|_2 \leq C_2(n)\mu \|B\|_2,$$

where $C_1(n)$ and $C_2(n)$ are constants that depend only on n .

Proof. Our proof of the theorem is again by induction on n . Let

$$B = \begin{pmatrix} b_{11} & B_{12}^T \\ B_{12} & B_{22} \end{pmatrix}, R = \begin{pmatrix} r_{11} & R_{12}^T \\ 0 & R_{22} \end{pmatrix}, U = \begin{pmatrix} 0 & U_{12}^T \\ 0 & U_{22} \end{pmatrix}$$

be partitioned so that B_{22} , R_{22} , and U_{22} are of order $n - 1$. Denote

$$S = R + U = \begin{pmatrix} s_{11} & S_{12}^T \\ 0 & S_{22} \end{pmatrix} = \begin{pmatrix} r_{11} & R_{12}^T + U_{12}^T \\ 0 & R_{22} + U_{22} \end{pmatrix}.$$

Let $B_{CIMGS}^{(1)}$ and $B_{CHOL}^{(1)}$ be the trailing $n - 1$ by $n - 1$ principal submatrix of B after one step of CIMGS and Cholesky factorization, respectively. After one step of CIMGS we have

$$\begin{pmatrix} s_{11} & 0 \\ S_{12} & I \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & B_{CIMGS}^{(1)} \end{pmatrix} \begin{pmatrix} s_{11} & S_{12}^T \\ 0 & I \end{pmatrix} = B + \begin{pmatrix} 0 & 0 \\ 0 & U_{12}U_{12}^T \end{pmatrix} + E_{CIMGS}^{(1)},$$

where $E_{CIMGS}^{(1)}$ denotes the error caused by finite precision computations during the first step. Similarly, define $E_{CHOL}^{(1)}$ as the error matrix from one step of full Cholesky factorization. Note that in positions (i, j) with $(1, i) \notin P$ or $(1, j) \notin P$, the updating is exactly the same as that in one step of Cholesky factorization, while at other positions no updating occurs. Therefore, $E_{CIMGS}^{(1)}$ and $E_{CHOL}^{(1)}$ are equal in those positions for which updating is performed. For the other positions, the elements of $E_{CIMGS}^{(1)}$ are equal to zero. Using Wilkinson's estimates [17], we get

$$\| E_{CIMGS}^{(1)} \|_2 \leq \| E_{CHOL}^{(1)} \|_2 \leq c_1 \mu \| B \|_2.$$

$B_{CIMGS}^{(1)}$ equals the reduced matrix after applying one step of Cholesky factorization to $B + U_{12}U_{12}^T$ without rounding error on the positions (i, j) such that $(1, i) \in P$ and $(1, j) \in P$. Let G denote the trailing principal submatrix of order $n - 1$ of $E_{CIMGS}^{(1)}$. Again using Wilkinson's result gives

$$\begin{aligned} \| B_{CIMGS}^{(1)} \|_2 &\leq \| B_{22} + U_{12}U_{12}^T + G \|_2 \\ &\leq \| B_{22} \|_2 + \| U_{12}U_{12}^T \|_2 + c_1 \mu \| B \|_2 \\ &\leq \| B_{22} \|_2 + \| S_{12}S_{12}^T \|_2 + c_1 \mu \| B \|_2 \leq c_4 \| B \|_2, \end{aligned}$$

where c_4 is a constant depending on n .

From the computation of one step of CIMGS we can see that $B_{CIMGS}^{(1)}$ is also positive definite when condition (2) holds. Using the induction hypothesis,

$$B_{CIMGS}^{(1)} = S_{22}^T S_{22} - U_{22}^T U_{22} - E_{CIMGS}^{(2)}$$

and

$$\| E_{CIMGS}^{(2)} \|_2 \leq C_2(n - 1)\mu \| B_{CIMGS}^{(1)} \|_2 \leq c_5 \mu \| B \|_2,$$

where c_5 is a constant depending on n . So we get

$$S^T S - U^T U - E_2 - E_1 = B,$$

where $E_1 = E_{CIMGS}^{(1)}$, and E_2 is $E_{CIMGS}^{(2)}$ augmented by a first row and column of zeros. It follows that

$$\| E \|_2 = \| E_1 + E_2 \|_2 \leq \| E_1 \|_2 + \| E_2 \|_2 \leq c_1 \mu \| B \|_2 + c_5 \mu \| B \|_2.$$

Letting $C_2(n) = c_1 + c_5$ establishes the error bounds.

Because $\lambda_{\min}(B) \leq \lambda_{\min}(B + U^T U)$, condition (2) implies

$$\frac{1}{\|(B + U^T U)^{-1}\|_2} \geq C_1(n)\mu \|B\|_2,$$

and so the factorization process will not break down. The proof is complete. \square

From the proof, the rounding errors in CIMGS have the same bound as those in complete Cholesky factorization. Since $\lambda_{\min}(B + U^T U)$ is typically larger than $\lambda_{\min}(B)$, numerical breakdown is less likely to happen in CIMGS factorization than it is in Cholesky factorization.

Since the goal of incomplete orthogonalization preconditioning is to approximate an orthogonal factorization, it is important to estimate the closeness of the IMGS factor to the factor obtained using complete Gram–Schmidt factorization. When $A^T A$ is an M-matrix, we have the following result, the proof of which can be found in [16].

THEOREM 4. *Let $A \in \mathbb{R}^{m \times n}$ have full rank. If $A^T A$ is an M-matrix and $Q \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{n \times n}$ are the matrices that are produced by applying IMGS with a given $P \subseteq P_n$, then*

$$Q^T Q = R^{-T} A^T A R^{-1} = I - E$$

is a regular splitting with $E \geq 0$, all of the diagonal elements of E equal to zero, and $\rho(E) < 1$, where $\rho(E)$ is the spectral radius of E .

If $A^T A$ is an M-matrix, Theorem 4 bounds the distance between Q and an orthogonal matrix since it implies that $\rho(Q^T Q) \leq 2$. Unfortunately, it does not guarantee an improvement in the condition number of $Q^T Q$ compared to $A^T A$ in general, since $\lambda_{\min}(A^T A)$ is not necessarily a lower bound on $\lambda_{\min}(Q^T Q)$. In practice, however, we have found that one step of IMGS tends to behave like one step of MGS in that one of the eigenvalues is brought closer to one and the remaining ones tend to stay in an interval whose lower and upper bounds do not significantly worsen. For MGS one of the eigenvalues is made exactly one, and the remaining ones are in an interval bounded by the minimum and maximum eigenvalues of the normal equations of the original matrix.

A similar result in [11] shows that $A^T A = LL^T - \hat{E}$ is a regular splitting, where L is the lower triangular IC factor of $A^T A$, $\hat{E} \geq 0$, and $\rho(\hat{E}) < 1$. It is straightforward to transform this result into one similar to Theorem 4. Specifically, if L is used to precondition the least squares problem, then it can be shown that $\tilde{Q}^T \tilde{Q} = L^{-1} A^T A L^{-T} = I - \tilde{E}$ is a regular splitting, satisfying conditions on \tilde{E} identical to those on E of Theorem 4. However, as is shown in the next section, CIMGS and IC do not, in general, produce the same triangular factor for a given drop set P .

Certainly the choice of P will affect the quality of the preconditioner. Intuitively, the more elements retained in the factor, the better CIMGS should approximate complete Gram–Schmidt. For general matrices A , we have not been able to rigorously establish this heuristic, but when $A^T A$ is an M-matrix, CIMGS has the following monotonicity property, where the notation \leq is used to indicate componentwise inequality.

THEOREM 5. *Let $B \in \mathbb{R}^{n \times n}$ be a symmetric positive definite M-matrix, and let $P_R \subseteq P_S \subseteq P_n$ be drop sets. If R and S are the CIMGS factors produced by using P_R and P_S , respectively, then $R \leq S$.*

The next Lemma is needed for the proof of this Theorem.

LEMMA 2.1. *Let $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times n}$ be symmetric positive definite M -matrices where $A \leq B$. If R and T are the upper triangular matrices from the CIMGS factorization of A and B , respectively, with the same nonzero position set P , then $R \leq T$.*

Proof. We prove the proposition by induction on n . Clearly the result holds for $n = 1$; assume that it holds for matrices of order $n - 1$. Let

$$R = \begin{pmatrix} r_{11} & R_{12}^T \\ 0 & R_{22} \end{pmatrix}, \quad T = \begin{pmatrix} t_{11} & T_{12}^T \\ 0 & T_{22} \end{pmatrix}, \quad A = \begin{pmatrix} a_{11} & A_{12}^T \\ A_{12} & A_{22} \end{pmatrix}, \quad \text{and} \\ B = \begin{pmatrix} b_{11} & B_{12}^T \\ B_{12} & B_{22} \end{pmatrix}$$

be the corresponding partitioned matrices for a matrix of order n , so that R_{22} , T_{22} , A_{22} , and B_{22} are of order $n - 1$. After one step of CIMGS on A and B , we have $0 < r_{11} = \sqrt{a_{11}} \leq \sqrt{b_{11}} = t_{11}$ and, since $A_{12} \leq B_{12} \leq 0$ by hypothesis, $R_{12} \leq T_{12} \leq 0$. Let $f = (f_2, f_3, \dots, f_n)^T$ and $g = (g_2, g_3, \dots, g_n)^T$ be the vectors of elements dropped by the first step of CIMGS applied to A and B , respectively. Then $f \leq g \leq 0$ and the reduced matrices R_{22} and T_{22} are the CIMGS factors of $\tilde{A} = A_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T$ and $\tilde{B} = B_{22} - T_{12}g^T - gT_{12}^T - T_{12}T_{12}^T$, and, therefore,

$$\begin{aligned} \tilde{A} &= A_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T \\ &\leq B_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T \\ &\leq B_{22} - T_{12}f^T - fT_{12}^T - T_{12}T_{12}^T \\ &\leq B_{22} - T_{12}g^T - gT_{12}^T - T_{12}T_{12}^T \\ &\leq \tilde{B}. \end{aligned}$$

By the induction hypothesis $R_{22} \leq T_{22}$, and so $R \leq T$ and the lemma is true for matrices of any order $n \geq 1$. \square

Proof of Theorem 5. We use induction on n . For $n = 1$ the result holds trivially. Assume that it is true for matrices of order $n - 1$. Let

$$B = \begin{pmatrix} b_{11} & B_{12}^T \\ B_{12} & B_{22} \end{pmatrix}, \quad R = \begin{pmatrix} r_{11} & R_{12}^T \\ & R_{22} \end{pmatrix}, \quad \text{and} \quad S = \begin{pmatrix} s_{11} & S_{12}^T \\ & S_{22} \end{pmatrix}$$

be partitioned so that B_{22} , R_{22} , and S_{22} are of order $n - 1$. After one step of CIMGS with P_R and P_S , respectively, $r_{11} = \sqrt{b_{11}} = s_{11}$ and since $B_{12} \leq 0$,

$$(3) \quad r_{1i} = \begin{cases} \frac{b_{1i}}{r_{11}} = \frac{b_{1i}}{s_{11}} = s_{1i}, & (1, i) \notin P_R \text{ and } (1, i) \notin P_S, \\ \frac{b_{1i}}{r_{11}} \leq 0 = s_{1i}, & (1, i) \notin P_R \text{ and } (1, i) \in P_S, \\ 0 = s_{1i}, & (1, i) \in P_R \text{ and } (1, i) \in P_S. \end{cases} \quad i = 2, \dots, n,$$

This implies $R_{12} \leq S_{12}$.

Let f and g be the vectors of elements dropped by the first step of CIMGS with P_R and P_S , respectively. Then $R_{12} + f = S_{12} + g \leq 0$, and since $R_{12} \leq S_{12}$, we have $0 \geq f \geq g$.

Now let B_R and B_S be the reduced matrices of order $n - 1$ produced by one step of CIMGS with P_R and P_S , respectively. It follows that

$$\begin{aligned} B_R &= B_{22} - (R_{12} + f)(R_{12} + f)^T + ff^T \\ &\leq B_{22} - (R_{12} + f)(R_{12} + f)^T + gg^T \\ &= B_{22} - (S_{12} + g)(S_{12} + g)^T + gg^T = B_S. \end{aligned}$$

Let T be the CIMGS factor for B_S using P_R , and note that R_{22} is the CIMGS factor of B_R using P_R and S_{22} is the CIMGS factor of B_S using P_S . By Lemma 2.1, $R_{22} \leq T$. By the induction hypothesis $T \leq S_{22}$, so $R_{22} \leq S_{22}$. Therefore, $R \leq S$ and the theorem is true for matrices of order n . \square

Let R_{CHOL} be the complete Cholesky factor of B . R_{CHOL} is equal to the CIMGS factor produced by using the pattern $P_0 = \emptyset$. Since $P_0 \subset P_R \subseteq P_S$, by Theorem 5 $R_{CHOL} \leq R \leq S$. If $E_R = R - R_{CHOL}$ and $E_S = S - R_{CHOL}$, then $E_R \leq E_S$. In this sense we can say that R better approximates R_{CHOL} . Informally, if fewer elements are dropped, i.e., if a smaller drop set P is used, the resulting CIMGS factor better approximates the complete Cholesky factor componentwise.

3. Relations between CIMGS and IC. It needs to be emphasized that CIMGS is not equivalent to IC although the structures of both are similar to Cholesky factorization. This can be seen from the following: first, CIMGS guarantees the existence when the matrix $B = A^T A$ is positive definite while IC does not have this property. Second, from the description of CIMGS in section 2 it can also be seen that CIMGS is not equal to IC. If step 5 in the CIMGS algorithm is replaced by

$$b_{ij} = b_{ij} - b_{ki}b_{kj} \quad (k, j) \notin P \text{ and } (k, i) \notin P,$$

then we get the IC algorithm. Note the change from *or* to *and* in the statement. This is the source of the extra operations and intermediate storage that CIMGS needs compared with IC. CIMGS performs computations on some of those positions (i, j) where one and only one of (k, j) and (k, i) is not in P for any k , while IC does not.

On the other hand, when certain conditions are imposed on the sparsity pattern of the target factor, the next theorem shows that CIMGS generates the same upper triangular factor as IC does, and both implicitly pass through the same intermediate factors to achieve this. Theorem 1 shows that IMGS will not fail for a full rank matrix, and Theorem 2 shows that CIMGS is equivalent to IMGS. Putting these statements together provides a way to guarantee existence of the IC factorization, based solely on the target sparsity pattern.

THEOREM 6. *Suppose $B \in \mathbb{R}^{n \times n}$ is symmetric and positive definite, and the set $P \subseteq P_n$ has the property that for any $1 \leq i < j < k \leq n$, the condition that one and only one of (i, j) and (i, k) is in P implies that $(j, k) \in P$. If R and U are the upper triangular matrices that arise from the CIMGS and IC on B using the drop set P , respectively, then $R = U$.*

Two observations will make the following proof easier to understand.

Observation 1: Suppose $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times n}$ are symmetric positive definite matrices for which IC completes using a drop set $P \subseteq P_n$. Let the matrices U and T be the upper triangular factors that IC gives for A and B , respectively. If $a_{ij} = b_{ij}$ for all $(i, j) \notin P$, then $T = U$.

Observation 2: After one step of CIMGS and IC applied to a symmetric positive definite matrix A using the same drop set P , the resulting first rows of the triangular factors are the same. In addition, the reduced matrices from which the rest of the factors are computed are also identical except for positions (i, j) where one and only one of $(1, i)$ and $(1, j)$ are in P . CIMGS updates the values of the elements at those positions, while IC does not change their values.

With these observations we prove Theorem 6.

Proof of Theorem 6. We use induction on n . When $n = 2$, it is trivial to show that the assertion is true. Assume that for $k < n$ the assertion is true. As indicated in Observation 2, the first steps of CIMGS and IC on B generate the same first row

of their respective triangular factors. The remaining steps of the algorithms consists of applying them to the updated trailing principal submatrices of order $n - 1$. Let $B_{CIMGS}^{(1)}$ and $B_{IC}^{(1)}$ be those trailing submatrices, for CIMGS and IC, respectively. As indicated in Observation 2, $B_{CIMGS}^{(1)}$ and $B_{IC}^{(1)}$ are equal except for positions (i, j) where exactly one of $(1, j)$ and $(1, i)$ is in P . The property of P in the hypotheses implies that all such positions (i, j) are in P . Let T be the triangular factor obtained from applying IC to $B_{CIMGS}^{(1)}$. By Observation 1 T is equal to the factor obtained by applying IC to $B_{IC}^{(1)}$. By the induction hypothesis, the IC factorization of $B_{CIMGS}^{(1)}$ is the same as the CIMGS factorization of $B_{CIMGS}^{(1)}$. Therefore, CIMGS applied to $B_{CIMGS}^{(1)}$ gives the same factor as IC applied to $B_{IC}^{(1)}$. Together with the equality of the first rows of the two factors, this gives $R = U$, and so by induction the theorem holds for all n . \square

Theorem 6 establishes a connection between IC and CIMGS. From this connection we can derive the following result regarding IC applied to an arbitrary symmetric positive definite matrix. This result is important because it allows us to guarantee the existence of the IC factor based only on the target nonzero pattern. Other modifications of IC have been proposed that allow the factorization to avoid breakdown, but they generally consist of ad hoc modifications of the elements as the factorization proceeds. This result allows, for example, a priori assurance that IC can be applied to matrices from a finite element mesh, based solely on the geometry of the mesh.

THEOREM 7. *Let $B \in \mathbb{R}^{n \times n}$ be a symmetric positive definite matrix. If $P \subseteq P_n$ has the property that for any $1 \leq i < j < k \leq n$, the condition that one and only one of (i, j) and (i, k) is in P implies $(j, k) \in P$, then the IC factorization algorithm completes successfully.*

Proof. There exists a full rank matrix $A \in \mathbb{R}^{n \times n}$, such that $B = A^T A$. According to Theorems 1 and 2, CIMGS applied to B completes successfully. By Theorem 6, the conditions on the set P assure that IC applied to B generates the same upper triangular matrix as CIMGS. So IC completes successfully and generates the same upper triangular matrix as CIMGS. \square

This result allows us, under certain restrictions, to view IC as a member of the class of incomplete Gram–Schmidt factorizations. On the other hand, the property of the set P described in the above theorem can be viewed as a condition that can guarantee the existence of IC when the matrix is symmetric positive definite. Furthermore, it is easy to modify the target sparsity pattern in order to satisfy the hypothesis of Theorem 7; see [16] for more details.

In general, CIMGS gives a different factor R from the one given by IC. If both methods generate the factor successfully, which is better in accelerating CG convergence? Again, the assumption that $A^T A$ is an M-matrix allows us to prove the relationship: the complete Cholesky factor is closer to the CIMGS factor than it is to the IC factor. To establish this result, two lemmas are stated here. Their proofs can be found in [16].

LEMMA 3.1. *Let $A, B \in \mathbb{R}^{n \times n}$ be symmetric positive definite M-matrices with $A \geq B$. Then $R \geq T$, where R and T are the Cholesky factors of A and B , respectively.*

LEMMA 3.2. *Let $A, B \in \mathbb{R}^{n \times n}$ be symmetric positive definite M-matrices, with $A \geq B$. Let $P \subseteq P_n$. Then $R \geq T$, where R and T are the respective incomplete Cholesky factors of A and B using the same P .*

THEOREM 8. *Let $B \in \mathbb{R}^{n \times n}$ be a symmetric positive definite M-matrix. Let R_{CHOL} , R_{IC} , and R_{CIMGS} be the upper triangular matrices from complete Cholesky,*

incomplete Cholesky, and CIMGS factorization of B , respectively, where the same drop set P is used for IC and CIMGS. The following relation is satisfied:

$$R_{CHOL} \leq R_{CIMGS} \leq R_{IC}.$$

Furthermore, $E_1 \leq E_2$, where

$$\begin{aligned} E_1 &= R_{CIMGS} - R_{CHOL}, \\ E_2 &= R_{IC} - R_{CHOL}. \end{aligned}$$

Proof. The proof is carried out in two parts.

(1) We prove $R_{CHOL} \leq R_{CIMGS}$ by induction on the size of the problem. The inequality is trivial for $n = 1$. Assume that it holds for $n - 1$. Partition $B = \begin{pmatrix} b_{11} & B_{12}^T \\ B_{12} & B_{22} \end{pmatrix}$, $R_{CHOL} = \begin{pmatrix} r_{11} & R_{12}^T \\ & R_{22} \end{pmatrix}$, and $R_{CIMGS} = T = \begin{pmatrix} t_{11} & T_{12}^T \\ & T_{22} \end{pmatrix}$ such that B_{22} , R_{22} , and T_{22} are of order $n - 1$. Let $U_{CIMGS} = \begin{pmatrix} 0 & U_{12}^T \\ & U_{22} \end{pmatrix}$ be the matrix of dropped entries from CIMGS, as defined in Theorem 3. Then $T_{12}^T U_{12} = 0$. Applying one step of Cholesky factorization and CIMGS factorization to B , we get

$$r_{11} = \sqrt{b_{11}}, \quad R_{12}^T = r_{11}^{-1} B_{12}^T$$

and

$$t_{11} = \sqrt{b_{11}}, \quad T_{12}^T + U_{12}^T = r_{11}^{-1} B_{12}^T.$$

Because B is an M-matrix, $R_{12} \leq T_{12} \leq 0$.

R_{22} is the Cholesky factor of $B_{CHOL}^{(1)} = B_{22} - R_{12}R_{12}^T$, and T_{22} is the CIMGS factor of $B_{CIMGS}^{(1)} = B_{22} - T_{12}T_{12}^T - U_{12}U_{12}^T - T_{12}U_{12}^T$. Again both $B_{CHOL}^{(1)}$ and $B_{CIMGS}^{(1)}$ are M-matrices. Furthermore,

$$\begin{aligned} B_{CHOL}^{(1)} &= B_{22} - R_{12}R_{12}^T \\ &= B_{22} - T_{12}T_{12}^T - T_{12}U_{12}^T - U_{12}T_{12}^T - U_{12}U_{12}^T \\ &= B_{CIMGS}^{(1)} - U_{12}U_{12}^T \end{aligned}$$

and $U_{12}U_{12}^T \geq 0$ imply that $B_{CHOL}^{(1)} \leq B_{CIMGS}^{(1)}$. Let S be the Cholesky factor of $B_{CIMGS}^{(1)}$. By Lemma 3.1 $R_{22} \leq S$, and by the induction hypothesis $S \leq T_{22}$. So $R_{22} \leq T_{22}$, which together with the inequality $R_{12} \leq T_{12}$ implies $R_{CHOL} \leq R_{CIMGS}$. By induction, this is true for any n .

(2) Next we prove $R_{CIMGS} \leq R_{IC}$ using a similar induction proof. It holds trivially for $n = 1$. Assume that it is true for $n - 1$. Then for matrices of order n , let $R_{IC} = C = \begin{pmatrix} c_{11} & C_{12}^T \\ 0 & C_{22} \end{pmatrix}$ be the incomplete Cholesky factor of B . Applying one step of incomplete Cholesky factorization to B , we have $c_{11} = \sqrt{b_{11}}$, and $C_{12} = T_{12}$ by Observation 2. Now C_{22} is the incomplete Cholesky factor of $B_{IC}^{(1)} = B_{22} - C_{12}C_{12}^T$, and $B_{IC}^{(1)}$ is also an M-matrix. Then $B_{CIMGS}^{(1)} \leq B_{IC}^{(1)}$ because

$$\begin{aligned} B_{CIMGS}^{(1)} &= B_{22} - T_{12}T_{12}^T - T_{12}U_{12}^T - U_{12}T_{12}^T \\ &= B_{22} - C_{12}C_{12}^T - T_{12}U_{12}^T - U_{12}T_{12}^T \\ &= B_{IC}^{(1)} - T_{12}U_{12}^T - U_{12}T_{12}^T \end{aligned}$$

and $T_{12}U_{12}^T \geq 0$. Let G be the incomplete Cholesky factor of $B_{CIMGS}^{(1)}$. By the induction hypothesis, $T_{22} \leq G$. From Lemma 3.2, $G \leq C_{22}$, so $T_{22} \leq C_{22}$. Together, these imply $T \leq C$. By induction, the inequality holds for matrices of any order $n \geq 1$. \square

Now we know that when B is a symmetric positive definite M-matrix, CIMGS will be a better elementwise approximation of the full Cholesky factorization than IC with the same sparsity pattern. Experiments on general matrices also show that for the same sparsity pattern, CG preconditioned by CIMGS takes fewer iterations than that preconditioned by IC, when both successfully produce a preconditioner. These results will be shown in section 4.

4. Numerical testing. Testing has been performed to assess the importance of the family of methods obtained by different implementations of CIMGS. We first compare CIMGS with IMGS to demonstrate that the compression strategy helps in reducing the computational cost of IMGS, and to assess how it affects the quality of the preconditioner. Next, we compare CIMGS with IC to evaluate the robustness and efficiency of this new family of methods. Results from the direct methods of QR factorization and Cholesky factorization of the normal equations are also presented to give the overall picture of where preconditioned iterative methods fit among methods of solving linear least squares problems.

4.1. Test environment. The test problems include systems from applications problems and from the RUA and RRA sets of the Harwell–Boeing collection. Characteristics of the matrices, including the number of rows (m), the number of columns (n), the number of nonzeros ($\text{nnz}(A)$), the number of the nonzeros in the normal equations ($\text{nnz}(A^T A)$), the density of the matrix, and the density of the normal equations, are given in Table 1. The density of a matrix $B \in \mathbb{R}^{m \times n}$, denoted $\text{dense}(B)$, is the percentage ratio of actual nonzero elements to the maximum possible, i.e., $100(\text{nnz}(B)/mn)$.

The collection includes 30 matrices, of which 10 are square. The sizes of the matrices vary considerably both in terms of dimensions— $115 \leq m \leq 16640$ and $82 \leq n \leq 3564$ —and number of nonzero elements— $421 \leq \text{nnz}(A) \leq 78298$. The density of the matrices ranges from less than 1% to slightly more than 7%. As expected, the density increases significantly for the normal equations associated with the test matrices, reaching the neighborhood of 40% for some matrices. Most are reasonably conditioned, but a few, such as CONEV8, are ill conditioned. The matrices are grouped according to the application source. The first set, AMOCO1 to WELL1033, is from the RRA portion of the Harwell–Boeing collection. AMOCO1 is a seismic tomography problem, while BELLADIT and BELMEDT are based on information retrieval problems. The group from CONEV8 to STRAT8 is from CFD problems where for some algorithms they are used to compute orthogonal projections. The final group of rectangular matrices, BNL1 to WOODW, is from a collection of linear programming problems available on NETLIB. We have also included a group of square matrices from the RUA set of the Harwell–Boeing collection, FS_760_1 to STEAM2.

For each matrix we generate a right-hand side vector consistent with a solution vector whose components are all equal to one. This allows checking the accuracy of the methods using both the residual vector norm $\|Ax - b\|_2$ and the error vector norm $\|x - x^*\|_2$. More comprehensive testing which included inconsistent problems [16] has lead to the same conclusions as in this paper.

The CIMGS and incomplete Cholesky factorizations used for comparison in the experiments were implemented in standard Fortran. The packages SPARSPAK-A [6]

TABLE 1
Characteristics of test matrices.

name	m	n	$\text{nnz}(A)$	$\text{nnz}(A^T A)$	$\text{dense}(A)$	$\text{dense}(A^T A)$
amoco1	1436	330	35210	27686	7.43	25.42
belladit	374	82	1343	2395	4.38	35.60
bellmedt	5831	1033	52012	372255	0.86	34.89
illc1850	1850	713	10608	5633	0.80	1.11
well1850	1850	713	10608	5633	0.80	1.11
well1033	1033	321	5765	2469	1.74	2.40
cone8	3362	484	15852	5135	0.97	2.19
dunes8	5414	771	25430	6998	0.61	1.18
strat8	16640	2205	78298	21757	0.21	0.45
bnl1	1576	632	9152	28005	0.92	7.01
ffff800	854	525	6235	10625	1.39	3.85
gen	1500	780	3276	5816	0.28	0.96
nzfri	3521	624	15903	8406	0.72	2.16
pilot4	1000	411	5145	6899	1.25	4.08
scsd6	1350	148	5666	2248	2.84	10.26
seba	1028	516	4874	52432	0.92	19.69
shell	1775	537	4900	2748	0.51	0.95
ship12s	2869	1042	8284	6388	0.28	0.59
standata	1075	360	3038	1833	0.79	1.41
woodw	8405	1099	37478	21525	0.41	1.78
fs_760_1	760	760	5976	13957	1.03	2.42
fs_760_3	760	760	5976	13957	1.03	2.42
gre_115	115	115	421	692	3.18	5.22
hwatt_2	1856	1856	11550	27445	0.34	0.80
mc_fe	765	765	24382	73254	4.17	12.52
orsreg_1	2205	2205	14133	24203	0.29	0.50
pde_9511	961	961	4681	6420	0.51	0.70
pores_2	1224	1224	9613	12723	0.64	0.85
saylr4	3564	3564	22316	38793	0.18	0.31
steam2	600	600	13760	20237	3.82	5.62

used for Cholesky factorization and SPARSPAK-B [7] used for QR factorization are also in Fortran but benefit from the more careful consideration typical of a numerical software package. Floating point operation counts (obtained by instrumenting the Fortran code) are used to compare the efficiency of the various approaches. Although this does not measure potentially important performance features such as pipelining and data locality, it does provide a machine-independent measure.

Conjugate gradients on the normal equations (CGLS) [13] is used in the experiments as the basic iterative method to solve the least squares problems and, for square matrices, the nonsymmetric linear systems. Although CGLS does not form the normal equations explicitly, its convergence depends on their spectrum. We would, therefore, expect CGLS to have difficulty converging for problems that are not well conditioned. Applying CGLS to the test suite confirms this expectation. Since the test problems are consistent we can use $\|x - x^*\|_2 / \|x^*\|_2 \leq 10^{-6}$ as the condition that determines acceptable accuracy. Unfortunately, this proved to be a very difficult condition for CGLS to fulfill. Only six of the 30 test problems produced the desired accuracy within n iterations. After $2n$ iterations, a total of nine matrices satisfy the requirement. Finally, after $2m$ only two more are added for a total of 11 out of 30 matrices. If the stopping condition is altered to use the residual norm by requiring $\|b - Ax\|_2 \leq 10^{-6}$ or $\|b - Ax\|_2 / \|b\|_2 \leq 10^{-6}$, the situation improves somewhat. After n iterations, 10 matrices have satisfied the requirement. An additional five matrices have acceptable

errors after $2n$ steps and after $2m$ iterations a total of 16 matrices out of 30 are solved satisfactorily. Therefore, some form of preconditioning is required for CGLS to be a viable solution technique for these test problems.

4.2. Implementation of CIMGS and its performance. As with Cholesky factorization, reorganizing the computations gives different versions of CIMGS. The version given earlier can be viewed as a rank-1 update approach, while our experiments use a delayed update version. On the i th step, we form the i th row of $A^T A$, store it in B , then perform all the CIMGS modifications for that row before going on to the next row. Locations of fill-in elements are computed using a simple pattern union computation for each sparse row triad performed. This factorization may store elements in a row that are in the drop set but are needed for the modifications required for later rows. This in turn means that CIMGS may require more intermediate storage than IC with the same pattern, although the final factors used in the iterations will require the same amount of storage. In the worst case, the intermediate storage needed by CIMGS is bounded by the space needed by the complete Cholesky factorization with the same matrix ordering.

The implemented version of CIMGS is as follows.

Algorithm [R]=CIMGS[B,P]
begin
for $i = 1, 2, \dots, n$,
 for $k = 1, 2, \dots, i-1$,
 for $j = i, i+1, \dots, n$,

$$b_{ij} = \begin{cases} b_{ij} - b_{ki}b_{kj} & (k, i) \notin P \\ b_{ij} - b_{ki}r_{kj} & (k, i) \in P \end{cases}$$

 endfor
 endfor
 if $b_{ii} > 0$ **then**

$$b_{ii} = \sqrt{b_{ii}}$$

 for $j = i, i+1, \dots, n$

$$b_{ij} = b_{ij}/b_{ii}$$

$$r_{ij} = \begin{cases} 0 & (k, j) \in P \\ b_{ij} & (k, j) \notin P \end{cases}$$

 endfor
 else
 quit (incomplete factorization cannot complete)
 endif
endfor
end

The drop set P can be determined dynamically or statically. By *dynamic* we mean that P is selected as the factorization proceeds. For example, using a drop tolerance to select retained elements is a dynamic method. Suppose that A is normalized so that the diagonal elements of $A^T A$ are equal to one and all the off diagonal elements have magnitudes no greater than one. Note that the CIMGS computations will not make the magnitudes greater than one. So we can safely choose a tolerance ϵ between 0 and 1. When the magnitude of a computed element is smaller than ϵ , this element is dropped, or we say this position is selected into P . By *static* we mean that P is determined before the numerical computation starts.

Note that in the implementation of IC, only computations that involve elements in positions that are retained in the incomplete factor are performed. This is because positions that are in P will not affect the values of the final factor R . The situation

TABLE 2
Comparison of CIMGS and IMGS.

	Static Pattern				Dynamic Pattern			
	Iterations		Operations		Iterations		Operations	
	CIMGS	IMGS	CIMGS	IMGS	CIMGS	IMGS	CIMGS	IMGS
amoco1	141	140	8.77	145.55	174	173	6.78	194.41
belladit	14	14	0.17	2.24	8	8	0.17	2.66
bellmedt	16	16	324.30	5475.47	18	18	84.01	4251.12
illc1850	296	305	0.47	4.71	87	87	0.86	120.38
well1850	180	180	0.47	4.71	37	37	1.14	149.75
well1033	88	87	0.12	1.77	38	38	0.24	16.87
cone8	1	1	0.38	30.61	1	1	0.27	264.47
dunes8	20	20	1.89	45.66	13	13	1.38	682.05
strat8	89	89	1.61	317.46	99	99	0.74	23565.04
bnl1	162	162	9.54	86.22	103	103	5.66	129.18
ffff800		NC	Fail	13.83		NC	Fail	102.40
gen	41	41	0.23	4.38	32	32	0.07	6.61
nzfri	57	57	7.52	98.27	46	610	10.03	1294.32
pilot4	61	61	0.97	10.59	22	22	0.53	31.42
scsd6	21	21	0.08	6.23	11	11	0.06	14.18
seba	70	70	24.92	142.32	73	73	8.94	148.50
shell	23	23	0.87	4.65	16	16	1.37	177.87
ship12s	35	35	0.08	1.96	26	26	0.07	11.19
standata	151	NC	0.29	3.55	76	75	0.65	66.33
woodw	34	34	6.61	207.19	27	27	2.27	972.28
fs_760_1	4	4	8.40	21.28	7	7	0.06	5.54
fs_760_3	NC	NC	8.40	21.28	NC	NC	4.51	174.80
gre_115	34	34	0.07	0.10	13	13	0.13	0.92
hwatt_2	1	1	5.86	70.89	1	1	4.50	2060.69
mc_fe	235	234	12.60	118.32	709	717	3.14	148.59
orserg_1	312	312	30.94	68.50	507	507	0.35	84.12
pde_9511	26	26	0.69	9.58	17	17	1.06	295.14
pores_2	NC	NC	5.46	28.69	NC	NC	0.64	71.35
saylr4	NC	NC	27.26	182.24	2277	2279	1.92	2976.69
steam2	7	7	7.02	24.64	28	28	0.10	4.80

is more complicated for CIMGS, where some of the elements in P carry intermediate information that will eventually affect the value of R . On the other hand, not all of the elements in P will affect the final value of R and computing them involves a *compute-then-drop* operation, which wastes time and space. A detailed discussion of an algorithm that identifies these unnecessary computations by symbolic analysis can be found in [16]. The data shown here include such unnecessary computations, but it should be kept in mind that performance may be improved further given an efficient symbolic analysis algorithm that identifies and eliminates unnecessary computations.

Table 2 shows two sets of comparisons. One set uses a drop set P so that the target factor will have the same sparsity pattern as the normal equations, as is usually done with incomplete Cholesky factorization. The other set of examples uses a dynamic drop set P with $\epsilon = 0.02$, so that elements with magnitude less than 2% of the maximal magnitude are dropped. For each set of examples, we list the number of iterations taken in the iterative phase of solutions using both CIMGS and IMGS preconditioners. This quantity can be used to compare the quality of the preconditioners because both use the same drop set P , and so the costs per iteration of applying them are the same. We also list the number of operations in millions used to compute the preconditioners, showing how much compression reduces the cost. Failure of the factorization to exist

is indicated by the word **Fail**, and failure of the iterative method to converge within the maximum allowed number of iterations is indicated by **NC**. From the table we observe the following facts:

- In no case did IMGS take fewer operations than CIMGS. The ratio of operation counts of IMGS over CIMGS varies from 1.4 to 197 for static patterns and from 7 to 31845 for dynamic patterns.
- Compression can increase the chance of numerical breakdown due to finite precision computations, in the same way that Cholesky factorization on the normal equations is more likely to fail than QR factorization on the original matrix. For problem FFFFF800 CIMGS factorization using both static and dynamic patterns fails, while IMGS with the same pattern succeeds. However, for this problem IMGS fails in the iterative phase.
- Compression does not degrade the quality of preconditioners. With a few exceptions, the number of iterations does not change. For those problems where CIMGS takes more iterations, usually only one additional iteration is needed. For problems where IMGS takes more iterations large differences can occur, such as the problem STANDATA with a static pattern and NZFRI with a dynamic pattern. Although this testing is too limited to claim that IMGS is likely to require more iterations than CIMGS, it does show that the compression technique does not significantly degrade the quality of the preconditioner.
- The reduction in cost in using CIMGS is more significant when using a dynamic pattern. Using a dynamic pattern involves *compute-then-drop* because it is not determined if an element is in drop set until after it is computed. This computation in CIMGS involves only a multiplication and subtraction, while in IMGS it is a dot product of two vectors of length n .

These experiments show that the compression technique used in CIMGS is effective in preserving the robustness and improving the efficiency of IMGS. Other experiments not shown here using a wider variety of pattern selection methods indicate that with the same pattern, CIMGS always uses fewer operations than IMGS does in computing the preconditioners. Furthermore, as in the experiments above the number of iterations remains unchanged with a only a few exceptions.

4.3. Comparison of static CIMGS with other methods. Table 3 shows the number of floating point operations in millions required by various methods. Failure of the factorization to exist is indicated by the word **Fail**, and failure of the iterative method to converge within the maximum allowed number of iterations is indicated by **NC**. The method QR is QR factorization on the original matrix, method NE is full Cholesky factorization of the normal equations, IC is incomplete Cholesky factorization combined with CGLS, and static and dynamic CIMGS are the versions given earlier. For the direct methods a minimum degree reordering was used; this reordering gave the best performance for the test set.

In comparing static CIMGS with IC, the same drop set and iterative method was used for both. From the table we note the following facts:

- IC factorization fails to produce the preconditioner in 18 problems without reordering and 16 problems with minimum degree ordering out of a total of 30 problems. The failure rate is so high that without modification IC is not useful in practice. By contrast, CIMGS factorization fails on one problem without reordering and three problems with minimum degree ordering. Since CIMGS factorization must complete in exact arithmetic, these failures must be caused

TABLE 3
Total operations of methods.

	SPARSPAK		IC		CIMGS		CIMGS		CIMGS
	QR	NE			Static		Hybrid		Dyn.
name	md	md	no	md	no	md	no	md	no
amoco1	70.96	3.37	Fail	Fail	46.27	21.29	46.35	22.65	47.01
belladit	2.76	0.16	0.34	0.31	0.42	0.35	0.34	0.31	0.31
bellmedt	$\geq 10^4$	345.76	174.25	174.25	354.51	335.63	174.25	174.25	94.87
illc1850	4.96	0.15	Fail	16.59	20.63	15.89	20.67	16.59	7.20
well1850	4.98	0.15	Fail	13.14	12.76	11.07	12.79	13.14	3.93
well1033	0.88	0.04	Fail	2.04	3.17	1.98	3.18	2.04	1.66
cone8	25.24	Fail	0.26	0.26	0.59	0.44	0.26	0.26	0.47
dunes8	27.34	0.75	3.47	3.31	5.14	3.35	3.47	3.31	3.57
strat8	180.83	4.87	43.98	152.40	44.45	133.77	43.98	152.40	44.15
bnl1	187.98	11.46	Fail	Fail	35.47	37.27	36.44	37.76	17.06
ffff800	6.68	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail
gen	2.11	0.28	Fail	Fail	2.12	1.72	2.18	1.78	1.29
nzfri	41.93	0.92	6.65	Fail	14.12	5.64	6.65	5.69	16.25
pilot4	13.35	1.01	Fail	Fail	4.34	2.36	4.35	2.37	1.50
scsd6	2.70	0.07	0.85	0.85	0.90	0.87	0.85	0.85	0.52
seba	41.78	Fail	Fail	Fail	42.13	Fail	42.38	Fail	15.35
shell	5.42	Fail	0.79	0.90	1.68	0.88	0.79	0.90	2.03
ship12s	2.54	0.13	Fail	Fail	2.73	2.73	2.78	2.78	1.78
standata	2.96	Fail	Fail	Fail	4.11	Fail	4.12	Fail	3.00
woodw	356.44	3.47	10.18	11.28	16.23	12.73	10.18	11.28	8.71
fs_760_1	6.41	2.26	0.66	0.66	8.84	1.45	0.66	0.66	0.36
fs_760_3	69.54	Fail	Fail	Fail	NC	NC	NC	NC	NC
gre_115	0.15	0.05	0.19	0.18	0.26	0.18	0.19	0.18	0.24
hwatt_2	123.57	37.29	Fail	Fail	6.19	5.22	6.90	5.24	4.79
mc.fe	28.23	10.59	Fail	Fail	106.97	18.40	107.06	18.85	110.48
orserg_1	219.83	63.95	Fail	Fail	84.44	719.97	84.60	719.99	49.56
pde_9511	5.34	1.65	Fail	Fail	2.08	4.91	2.10	4.92	2.22
pores_2	12.02	3.72	Fail	Fail	NC	NC	NC	NC	NC
saylr4	360.51	108.13	Fail	Fail	NC	NC	NC	NC	381.41
steam2	22.11	7.40	1.89	1.89	8.38	4.29	1.89	1.89	2.25

by finite precision arithmetic. Notice that complete Cholesky factorization (NE) also fails on these problems.

- CIMGS failed to allow convergence on three problems. However, IC failed in the factorization phase on those three problems. In no case has IC succeeded while CIMGS failed, showing that CIMGS is more robust than IC.
- When both CIMGS and IC succeed, IC is generally more efficient. For the 12 out of 30 problems without reordering where both succeeded, IC takes fewer operations than CIMGS. With minimum degree reordering, the difference between CIMGS and IC is less significant. Minimum degree reordering reduces the fill-in of Cholesky factorization as well as the intermediate storage and computations of CIMGS. Furthermore, minimum degree reordering does not change the number of operations needed in IC factorization, but the quality of the preconditioner is lessened, increasing the number of iterations needed. For the 14 minimum degree reordered problems where both IC and CIMGS succeed, IC takes fewer operations eight times. These results suggest that finding a proper reordering for CIMGS is important for efficiency.

In terms of numbers of iterations, CIMGS performance is similar to that of IC. As shown by Theorem 8, when $A^T A$ is an M-matrix, CIMGS produces a better approx-

imation to the complete Cholesky factor than IC does. For the test problems where both preconditioners exist (and for which $A^T A$ need not be an M-matrix), IC generally requires one or two more iterations than CIMGS does. Only for SHELL without reordering did CIMGS require more iterations than IC (22 iterations versus 21). With minimum degree reordering, there were 14 problems where both CIMGS and IC succeeded. On 13 of those, CIMGS required fewer iterations. On the minimum degree reordered problem WOODW, CIMGS required one more iteration than IC did.

The primary reason CIMGS efficiency improves so dramatically with minimum degree reordering is that the amount of intermediate storage (and operations on those intermediate quantities) is reduced greatly. The ratio of storage required by CIMGS to that required by IC ranges from 1.42 to 34.8 for problems without reordering. That range becomes 1.17 to 3.6 for minimum degree reordered matrices. This indicates that finding a good ordering is important for CIMGS. Note that in any case, the intermediate space CIMGS needs cannot be greater than that of full Cholesky factorization.

The comparison of CIMGS and IC suggests using a hybrid method: perform IC factorization and if it fails, switch to CIMGS. Such an approach will allow retaining the efficiency of IC when it exists and having the additional robustness of CIMGS otherwise. The operation counts for this hybrid method are also shown in Table 3. Note that since minimum degree reordering lessens the difference in operation counts between IC and CIMGS, the hybrid method with minimum ordering does not show significant advantage over CIMGS.

The results show that CIMGS is more robust than IC, at the cost of more operations and intermediate storage. The natural question to ask is if the price is too high to make the method practically useful. Examining direct methods on these problems can help answer that. From Table 3 QR is the most robust method, able to solve all the problems. For problem MC_FE and STEAM2, $\|Ax - b\|_2 \leq 10^{-6}$ could not be reached because $\|b\|_2$ is too large, 3.955×10^{13} for MC_FE and 5.266×10^{10} for STEAM2. The relative residuals for the solution of these two problems are small. NE has six failures. Compared to one and three failures of CIMGS in the factorization phase with and without ordering, respectively, CIMGS seems more robust. This is consistent with the result of Theorem 3, which suggested that static CIMGS would be more robust than full Cholesky factorization. In terms of efficiency, QR needs more floating point operations than the iterative methods on most of the test problems and NE needs fewer operations than iterative methods on the problems where it succeeds. In term of storage, the storage needed by CIMGS is bounded by the storage needed by normal equations method and depends also on the drop set P .

Table 3 also shows the result of CIMGS using dynamic pattern with $\epsilon = 0.02$ as a fixed drop tolerance. When compared to static CIMGS without reordering, only one additional success occurs: SAYLR4. However, the efficiency is improved for 22 of the test problems, sometimes by more than a factor of 24. Note that the test problems come from different sources and a single drop tolerance as was used here may not be suitable for all problems. Determining the drop tolerance is still a matter of experimentation. CIMGS shares this potential difficulty with other drop tolerance methods such as ILUT preconditioning [15]. In general, the more ill conditioned problems are, the smaller ϵ may need to be to get a preconditioner of adequate quality. Our tests show that CIMGS has a great flexibility in pattern selection without losing robustness. This flexibility of pattern selection may allow us to find a better preconditioner after a more careful selection process, possibly based on application-dependent features.

4.4. Summary of testing results. The tests show that CIMGS greatly reduces the number of operations needed when compared with IMGS. In terms of robustness of the factorization phase, it failed half as often as complete Cholesky factorization of the normal equations reordered using a minimum degree algorithm. When considering both natural and minimum degree reordered matrices static CIMGS factorization failed only four times, compared to 34 factorization failures by IC. If total robustness is considered (iteration plus factorization phases), static CIMGS failed less than a third as often as IC did.

In terms of the number of total (iterative and factorization) operations required, for the problems where the IC factorization existed, it was usually more efficient than CIMGS, especially when there is no reordering to reduce the intermediate operations for CIMGS. Using a hybrid approach of first trying IC and, if the factorization fails, using CIMGS allows combining the efficiency of IC with the robustness of CIMGS. However, even a simple minimum degree reordering makes the costs of CIMGS comparable with those of IC, without sacrificing robustness. In addition, using a dynamic pattern selection further reduces both the numbers of operations and the amount of intermediate storage that CIMGS requires, again without sacrificing robustness.

5. Conclusion. This paper introduces a new preconditioning algorithm, CIMGS. A detailed study of the theoretical and numerical properties of CIMGS shows that it is robust both theoretically and empirically, existing (in exact arithmetic) for any full rank matrix. Numerically it is more robust than an incomplete Cholesky factorization, and CGLS preconditioned with dynamic CIMGS compares favorably with using Cholesky factorization on the normal equations. This finding suggests that with CIMGS preconditioning, CGLS can be a viable method for practical use.

Additional theory shows that CIMGS is equivalent to IMGS, the factorization has better backward error properties than complete Cholesky factorization, and for systems whose normal equations are M-matrices, CIMGS induces a regular splitting, better estimates the complete Cholesky factor R^c as the drop set P gets smaller, and lies between complete Cholesky factorization and incomplete Cholesky factorization in its approximation properties. Typically, those properties seem to hold numerically, even when $A^T A$ is not an M-matrix. When the drop set satisfies a mild and easily verified (or enforced) property, the upper triangular factor CIMGS generates is the same as the one generated by incomplete Cholesky factorization. This allows the existence of an IC factorization to be guaranteed, based solely on the target sparsity pattern.

There are several issues left for further research. First, we need to have a more efficient algorithm to identify unnecessary computations used in the current implementation for static sparsity patterns which should bring down the computation cost further. Second, new reordering algorithms need to be found to reduce the intermediate data storage and computations. Existing reorderings generally target minimizing fill-in during complete factorization, or minimizing bandwidth of the matrix. Possibly by adapting them we can develop a new heuristic more suitable for improving the performance of CIMGS, by minimizing the intermediate computations.

Selecting an optimal target sparsity pattern, that is, the drop set P , could be crucial to the success of CIMGS. We need to have a fast way of selecting the pattern, but even if this is not practical, for problems where the same pattern can be used over and over, it may still be worthwhile to search for a near optimal pattern.

Parallel processing is an important issue, which has not been discussed here. Although CIMGS has a structure similar to Cholesky factorization, which is not as

rich in parallelism as Gram–Schmidt-type factorization, we can still exploit parallelism in the algorithm by utilizing a block bordered diagonal matrix structure. Because of the great flexibility of sparsity pattern selection allowed in CIMGS, it is feasible to combine sparsity pattern selection strategies with matrix ordering techniques to get better performance using parallel processing.

The preconditioning method proposed here is applied to CG-type iterative methods. How will they perform when combined with other types of iterative methods, for example, row projection methods, GMRES, or Lanczos-based methods? Of particular interest is adapting the preconditioner to the particular iterative method. We are presently investigating the relationship of near-orthogonality of the coefficient matrix of a system of linear equations to the convergence behavior of a collection of iterative methods and the implications of the use of CIMGS as a preconditioner.

Another potential research area is to extend the relationship between CIMGS and incomplete Cholesky to unsymmetric matrices. Currently we are developing a CIMGS-like algorithm for ILU factorization which can be more robust and efficient than standard ILU factorization methods.

REFERENCES

- [1] S. ASHBY, *Polynomial Preconditioning for Conjugate Gradient Methods*, Ph.D. thesis, University of Illinois, Urbana-Champaign, Urbana, IL, 1987; Tech. report 1355, Department of Computer Science, University of Illinois, Urbana-Champaign, Urbana, IL.
- [2] S. ASHBY, *Minimax polynomial preconditioning for Hermitian linear systems*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 766–789.
- [3] M. W. BERRY AND R. J. PLEMMONS, *Algorithms and experiments for structural mechanics on high-performance architectures*, Comput. Methods Appl. Mech. Engrg., 64 (1987), pp. 487–507.
- [4] A. BJÖRCK, *SSOR preconditioning methods for sparse least squares problems*, in Proc. of the Computer Science and Statistics: 12th Annual Symposium on the Interface, J. F. Gentleman, ed., University of Waterloo, Waterloo, Ontario, Canada, May 1979, pp. 21–25.
- [5] I. CHIO, C. L. MONMA, AND D. SHANNO, *Further development of a primal-dual interior point method*, ORSA J. Comput., 2 (1990), pp. 304–311.
- [6] E. CHU, A. GEORGE, J. LIU, AND E. NG, *SPARSPAK: Waterloo Sparse Matrix Package User's Guide for SPARSPAK-A*, Tech. report CS-84-36, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1984.
- [7] A. GEORGE AND E. NG, *SPARSPAK: Waterloo Sparse Matrix Package, User's Guide for SPARSPAK-B*, Tech. report CS-84-37, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1984.
- [8] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 2nd ed., The Johns Hopkins University Press, Baltimore, MD, 1989.
- [9] M. T. HEATH, R. J. PLEMMONS, AND R. C. WARD, *Sparse orthogonal schemes for structural optimization using the force method*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 514–532.
- [10] A. JENNINGS AND M. A. AJIZ, *Incomplete methods for solving $A^T Ax = b$* , SIAM J. Sci. Statist. Comput., 5 (1984), pp. 978–987.
- [11] J. A. MEIJERINK AND H. A. VAN DER VORST, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), pp. 148–162.
- [12] M. FORTIN AND R. GLOWINSKI, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, North-Holland, Amsterdam, 1983.
- [13] C. PAIGE AND M. SAUNDERS, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
- [14] Y. SAAD, *Preconditioning techniques for nonsymmetric and indefinite linear systems*, J. Comput. Appl. Math., 24 (1988), pp. 89–105.
- [15] Y. SAAD, *SPARSKIT: A Basic Tool Kit for Sparse Matrix Computations*, Tech. report, Center for Supercomputing Research and Development, University of Illinois, Urbana-Champaign, Urbana, IL, 1990.

- [16] X. WANG, *Incomplete Factorization Preconditioning for Linear Least Squares Problems*, Ph.D. thesis, University of Illinois, Urbana-Champaign, Urbana, IL, 1993; Tech. report UIUCDCS-R-93-1834, Department of Computer Science, University of Illinois, Urbana-Champaign, Urbana, IL.
- [17] J. H. WILKINSON, *A priori error analysis of algebraic processes*, in Proc. International Congress of Mathematicians, Moscow: Izdat. Mir, 1968, pp. 119–129.
- [18] Z. ZLATEV AND H. B. NIELSON, *Solving large and sparse linear least squares problems by conjugate gradient algorithm*, Comput. Math. Appl., 15 (1988), pp. 185–202.