CIMGS: An Incomplete Orthogonal Factorization Preconditioner

Xiaoge Wang
Department of Computer Science
Indiana University - Bloomington

Kyle Gallivan
Department of Electrical and Computer Engineering
University of Illinois- Urbana

Randall Bramley
Department of Computer Science
Indiana University - Bloomington *

December 16, 1993

Abstract

A new preconditioner (called CIMGS) based on an incomplete orthogonal factorization is derived, analyzed, and tested. Although designed for preconditioning least squares problems, it is also applicable to more general symmetric positive definite matrices. 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, and the conjugate gradient iterative method preconditioned with CIMGS compares favorably with using Cholesky factorization on the normal equations. Theoretical results show that the CIMGS factorization has better backward error properties than complete Cholesky factorization does, and for systems whose normal equations are M–matrices, CIMGS induces a regular splitting, better estimates the complete Cholesky factor R^c as the set of dropped positions gets smaller, and lies between complete Cholesky factorization and incomplete Cholesky factorization in its approximation properties. Those properties usually hold numerically, even when A^TA 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 incomplete Cholesky factorization does. This allows guaranteeing the existence of IC factorization, based solely on the target sparsity pattern.

*Work supported by NSF grants CDA-9309746 and CCR-9120105

1 Introduction

This work is motivated by the linear least squares problem of finding $x \in \Re^n$ which minimizes the value of

$$\| b - Ax \|_2, \tag{1}$$

where $A \in \Re^{m \times n}$, $m \geq n$, is a large sparse matrix of full rank and $b \in \Re^m$ is an arbitrary vector. Such problems occur frequently in scientific and engineering applications such as linear programming [5], augmented Lagrangian methods for CFD [12], and the natural factor method in partial differential equations [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. This 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 the original linear least squares problem. Orthogonal factorization methods [8] avoid this problem, but they require more floating point operations and 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 that it is applied to, 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 1. 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: $A \approx QR$. Existing incomplete orthogonal factorization preconditioners can be divided into two classes: incomplete Gram-Schmidt, such as the methods presented in [14, 10, 16], and incomplete Givens, such as the method in [18]. 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 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 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

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 dotproducts 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 preserving the preconditioner's robustness and effectiveness for the CG algorithm. 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 many fewer computations than IMGS does. We also prove that when A^TA 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 the effectiveness of 'compression' are presented, along with comparisons among CIMGS, IC preconditioned CG, and direct methods. Conclusions and remarks are made based on those results.

2 The CIMGS Algorithm and Its Properties

To motivate the CIMGS algorithm, we first describe incomplete Gram-Schmidt (IMGS) factorization. Let $P_n = \{(i,j) | i \neq j, \ \ 1 \leq i,j \leq n\}$ and assume that the matrix $A \in \Re^{m \times n}$ has full column rank. Let $P \subset P_n$ be a set of index pairs such that $(i,j) \in P$ implies that $1 \leq i < j \leq n$.

The set P determines which elements of the target incomplete factor R will not be retained in the approximate factorization, i.e., P is the set of drop positions. The IMGS factorization algorithm can be easily derived 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 $k = 1, 2, \ldots, n,$
(1) $r_{kk} = \| \ a_k \ \|_2$
(2) $q_k = a_k / r_{kk}$
for $j = k+1, \ k+2, \ \ldots, \ n$
(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

If step (3) is replaced by $r_{kj} = q_k^T a_j$, we get a complete modified Gram–Schmidt factorization. This factorization will always succeed in producing an upper triangular factor R when A has full rank:

Theorem 1 *If $A \in R^{m\times n}$, $m \geq n$, has full rank, then IMGS applied with a drop set $P \subset P_n$ 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 $q_i = (a_i^{(0)} - q_1 r_{1i} - ... - q_{i-1} r_{i-1,i})/r_{ii}$, for $i = 1, 2, ..., n$. The algorithm cannot complete if at some step k, $r_{kk} = \| a_k^{(k-1)} \|_2 = 0$. This means that we have $a_k^{(k-1)} = a_k^{(0)} - q_1 r_{1k} - \ldots - q_{k-1} r_{k-1\,k} = 0$ and so $a_k^{(0)}$ is a linear combination of $q_1, q_2, ..., q_{k-1}$. Therefore, the set of vectors $\left\{q_1, q_2, q_3, \ldots, q_{k-1}, a_k^{(0)}\right\}$ is linearly dependent. However, the vectors $q_1, q_2, q_3, \ldots, q_{k-1}$ form a basis of $span\left\{a_1^{(0)}, a_2^{(0)}, a_3^{(0)}, \ldots, a_{k-1}^{(0)}\right\}$, and the set of vectors $\left\{a_1^{(0)}, a_2^{(0)}, a_3^{(0)}, \ldots, a_k^{(0)}\right\}$ is linearly independent because A has full rank. Therefore, $\left\{q_1, q_2, q_3, \ldots, q_{k-1}, a_k^{(0)}\right\}$ is independent, a contradiction. So if A has full rank, $r_{kk} \neq 0$ for $k = 1, 2, \ldots, n$ and the factorization must exist. □

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 to other preconditioning methods.

The new algorithm CIMGS now described will produce 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 dotproduct 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^TA. 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^TA, the elements of which are dotproducts of the corresponding column vectors. After each step of each of the factorization methods, the relationship is maintained between the reduced matrices. Moreover, it is well–known that 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 maintaining equality of the reduced matrix produced to the normal equations of the reduced matrix produced by IMGS.

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

Algorithm [R]=CIMGS[B,P]
begin
for $k = 1, 2, \ldots, n,$
if $b_{kk} \neq 0$ **then**
(1) $b_{kk} = \sqrt{b_{kk}}$
(2) $r_{kk} = b_{kk}$
for $j = k+1, k+2, \ldots, n$
(3) $b_{kj} = b_{kj}/\sqrt{b_{kk}}$

```
(4)          r_kj = { 0      (k,j) ∈ P
                    { b_kj   (k,j) ∉ P
          endfor
          for  j = k+1, k+2, ..., n
             for  i = k+1, k+2, ..., n
(5)             b_ij = b_ij − b_ki b_kj    (k,j) ∉ P or (k,i) ∉ P
             endfor
          endfor
       else
(6)       quit (incomplete factorization can not complete)
       endif
    endfor
    end
```

Note that the structure of CIMGS is similar to the rank–1 update form of Cholesky factorization, with incompleteness introduced at step (5). Just as with Cholesky factorization, a more efficient version can be obtained by deferring the rank–1 updates until they are needed, but the above formulation is more convenient for the following analysis. 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 be stored in separately.

First we show that CIMGS applied to A^TA produces the same triangular factor as IMGS applied to A.

Theorem 2 *Let $A \in R^{m\times n}$, $m \geq n$, $rank(A) = n$ and $B = A^TA$. If $R = IMGS(A,P)$ and $S^T = CIMGS(B,P)$, then $R = S^T$.*

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 $k = n$. 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 the remaining rows 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^T is given by

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

and columns 2 to n of the remaining rows of B have elements $b_{ij}^{(1)} = b_{ij} - b_{1i}b_{1j}$ 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 $A^{(1)} = \{a_2^{(1)}, a_3^{(1)}, \ldots, a_n^{(1)}\}$. The computation of the rest of S^T 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^T 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 IMGS? 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 says that numerical rank deficiency is less likely to occur for CIMGS than it is for complete Cholesky factorization. That in turn implies that CIMGS may breakdown earlier than IMGS as the condition of A worsens . This will be confirmed by numerical experiments presented in later.

Theorem 3 *Let B be a symmetric positive definite matrix and μ be the machine precision. Let P be a given drop set of zero positions 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. If*

$$\kappa(B) \geq C_1(n)\mu, \tag{2}$$

where $\kappa(B)$ is the condition number of B, then 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

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

be conformally partitioned. Denote

$$L = R + U = \begin{pmatrix} l_{11} & l_{12}^T \\ 0 & L_{22} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12}^T + u_{12}^T \\ 0 & R_{22} + U_{22} \end{pmatrix}.$$

Consider the computation of the first row of L. It is the same as the computation of first row of Cholesky factor of B, except during the rank-1 update of the trailing submatrix. CIMGS only updates those elements in position (i,j) such that at least one of $(1,i)$ or $(1,j)$ is not in P, while Cholesky factorization updates elements in positions (i,j) where both $(1,i)$ and $(1,j)$ are not in P.

Let $B^{(2)}_{CIMGS}$ and $B^{(2)}_{CHOL}$ be the reduced matrices after one step of CIMGS and Cholesky factorization, respectively. After one step of CIMGS, we have

$$\begin{pmatrix} l_{11} & 0 \\ l_{12} & I \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & B^{(2)}_{CIMGS} \end{pmatrix} \begin{pmatrix} l_{11} & l_{12}^T \\ 0 & I \end{pmatrix} = B + \begin{pmatrix} 0 & 0 \\ 0 & u_{12}u_{12}^T \end{pmatrix} + E^{(1)}_{CIMGS}.$$

where $E^{(1)}_{CIMGS}$ denotes the error caused by finite precision computations during the first step, and $B^{(2)}_{CIMGS} = B^{(2)}_{CHOL} + u_{12}u_{12}^T$. 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, $E^{(1)}_{CIMGS}$ and $E^{(1)}_{CHOL}$ are equal in those positions for which updating is performed. For the other positions, the elements of $E^{(1)}_{CIMGS}$ are equal to 0. Using Wilkinson's estimates [17], we get

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

Since $B^{(2)}_{CIMGS}$ equals the reduced matrix after applying one step of Cholesky factorization to $B + u_{12}u_{12}^T + E^{(1)}_{CIMGS}$, again using Wilkinson's result gives

$$\begin{aligned} \| B^{(2)}_{CIMGS} \|_2 &\leq \| B_{22} \|_2 + \| u_{12}u_{12}^T \|_2 + c_1\mu \| B \|_2 \\ \leq \| B_{22} \|_2 + \| l_{12}l_{12}^T \|_2 + c_1\mu \| B \|_2 &\leq c_4 \| B \|_2 \end{aligned}$$

where c_4 is a constant depending on n.

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

$$B^{(2)}_{CIMGS} = L^{(2)^T}L^{(2)} - U^{(2)^T}U^{(2)} - E^{(2)}_{CIMGS}$$

and

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

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

$$L^TL - U^TU - E_2 - E_1 = B,$$

where $E_1 = E^{(1)}_{CIMGS}$, and E_2 is $E^{(2)}_{CIMGS}$ augmented by a null first row and column. 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.

If the minimal eigenvalue of $B + U^T U$ is greater than the error, the factorization process will not break down. That is, if

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

completing the proof. □

Since $U^T U$ is symmetric positive semidefinite, the smallest eigenvalue of $B + U^T U$ is greater than or equal to the smallest eigenvalue of B. So numerical breakdown is less likely to happen for CIMGS factorization than it is for Cholesky factorization.

Since the goal of incomplete orthogonalization preconditioning is to approximate an orthogonal factorization, it is important to estimate the closeness of the CIMGS factor to the factor obtained using complete Gram–Schmidt factorization. When $A^T A$ is an M-matrix, we have the following result:

Theorem 4 *Let $A \in R^{m \times n}$ have full rank. If $A^T A$ is an M-matrix and $Q \in R^{m \times n}$, $R \in R^{n \times n}$ are the matrices that are produced by applying IMGS with a given $P \subset P_n$, 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 0, and $\rho(E) < 1$, where $\rho(E)$ is the spectral radius of E.

The proof can be found in [16].

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 of $A^T A$ 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 1 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 1 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 $A^T A = L L^T - \hat{E}$ is a regular splitting, where L is the 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 necessarily 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:

Theorem 5 *Let $B \in R^{n\times n}$ be a symmetric positive definite M-matrix, and let $P_1 \subseteq P_2$ be zero position sets. If R_1 and R_2 are the CIMGS factors produced by using P_1 and P_2 respectively, then $R_1 \leq R_2$.*

The notation $R_1 \leq R_2$ is used to indicate componentwise inequality. The next Lemma is needed for the proof of this Theorem.

Lemma 1 *Let $A \in \Re^{n\times n}$ and $B \in \Re^{n\times n}$ 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 non-zero 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

$$R = \begin{pmatrix} r_{11} & R_{12}^T \\ 0 & R_{22} \end{pmatrix}, \quad T = \begin{pmatrix} t_{11} & T_{12}^T \\ 0 & T_{22} \end{pmatrix},$$

$$A = \begin{pmatrix} a_{11} & A_{12}^T \\ A_{12} & A_{22} \end{pmatrix}, \quad \textit{and} \quad B = \begin{pmatrix} b_{11} & B_{12}^T \\ B_{12} & B_{22} \end{pmatrix}$$

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 $b_{1i} \leq 0$, we also have

$$r_{1i} = \begin{cases} a_{1i}/r_{11} \leq b_{1i}/r_{11} \leq b_{1i}/t_{11} = t_{1i} & (1,i) \notin P \\ 0 = t_{1i} & (1,i) \in P. \end{cases}$$

Let f and g be the vectors of dropped elements from the first step of CIMGS applied to A and B, respectively. Then $f \leq g \leq 0$ and $R_{12} \leq T_{12} \leq 0$. R_{22} and T_{22} are the CIMGS factors of $A^{(2)} = A_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T$ and $B^{(2)} = B_{22} - T_{12}g^T - gT_{12}^T - T_{12}T_{12}^T$, and therefore

$$\begin{aligned} A^{(2)} &= A_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T \\ &\leq B_{22} - R_{12}f^T - fR_{12}^T - R_{12}R_{12}^T \\ &\leq B_{22} - T_{12}f^T - fT_{12}^T - T_{12}T_{12}^T \\ &\leq B_{22} - T_{12}g^T - gT_{12}^T - T_{12}T_{12}^T \\ &\leq B^{(2)}. \end{aligned}$$

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

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

$$B = \begin{pmatrix} b_{11} & b^T \\ b & B_{22} \end{pmatrix},$$

$$R_1 = \begin{pmatrix} r_{11}^{(1)} & R_{12}^{(1)} \\ & R_{22}^{(1)} \end{pmatrix}, \text{ and } R_2 = \begin{pmatrix} r_{11}^{(2)} & R_{12}^{(2)} \\ & R_{22}^{(2)} \end{pmatrix}$$

be partitioned so that B_{22}, $R_{22}^{(1)}$, and $R_{22}^{(2)}$ are of order $n-1$. After one step of CIMGS with P_1 and P_2, respectively, $r_{11}^{(1)} = \sqrt{b_{11}} = r_{11}^{(2)}$ and since $B_{1i} \leq 0$,

$$r_{1i}^{(1)} = \begin{cases} b_{1i}/r_{11}^{(1)} = b_{1i}/r_{11}^{(2)} = r_{1i}^{(2)} & (1,i) \notin P_1 \text{ and } (1,i) \notin P_2 \\ b_{1i}/r_{11}^{(1)} \leq 0 = r_{1i}^{(2)} & (1,i) \notin P_1 \text{ and } (1,i) \in P_2 \quad i = 2, \ldots, n \\ 0 = r_{1i}^{(2)} & (1,i) \in P_1 \text{ and } (1,i) \in P_2 \end{cases} \tag{3}$$

This implies $R_{12}^{(1)} \leq R_{12}^{(2)}$.

Let $f = (f_2, f_3, \ldots, f_n)$ and $g = (g_2, g_3, \ldots, g_n)$ be the vectors of elements dropped by CIMGS with P_1 and P_2, respectively. Then

$$f_i = \begin{cases} b_{1i}/r_{11}^{(1)} = b_{1i}/r_{11}^{(2)} = g_i & (1,i) \in P_1 \text{ and } (1,i) \in P_2 \\ 0 \geq b_{1i}/r_{11}^{(2)} = g_i & (1,i) \notin P_1 \text{ and } (1,i) \in P_2 \quad i = 2, \ldots n. \\ 0 = g_i & (1,i) \notin P_1 \text{ and } (1,i) \notin P_2 \end{cases} \tag{4}$$

and so $0 \geq f \geq g$. Also note that $R_{12}^{(1)} + f = R_{12}^{(2)} + g$.

Now let $B_{22}^{(1)}$ and $B_{22}^{(2)}$ be the reduced matrices of order $n-1$ produced by one step of CIMGS with P_1 and P_2, respectively. It follows that

$$\begin{aligned} B_{22}^{(1)} &= B_{22} - (R_{12}^{(1)} + f)^T (R_{12}^{(1)} + f) + f^T f \\ &\leq B_{22} - (R_{12}^{(1)} + f)^T (R_{12}^{(1)} + f) + g^T g \\ &\leq B_{22} - (R_{12}^{(2)} + g)^T (R_{12}^{(2)} + g) + g^T g \\ &= B_{22}^{(2)}. \end{aligned}$$

Let $R_{22}^{(1)}$ be the CIMGS factor for $B_{22}^{(1)}$ using P_1, $R_{22}^{(2)}$ be the CIMGS factor for $B_{22}^{(2)}$ using P_2, and T be the CIMGS factor for $B_{22}^{(2)}$ using P_1. By Lemma 1, $R_{22}^{(1)} \leq T$. By the induction hypothesis $T \leq R_{22}^{(2)}$. Together we have $R_1 \leq R_2$ and therefore the theorem is true for matrices of order n. □

Let R^c be the complete Cholesky factor of B. R^c is equal to the CIMGS factor produced by using the pattern $P^c = \emptyset$. Clearly, we have $P^c \subset P_1 \subset P_2$ and, by Theorem 5, $R^c \leq R_1 \leq R_2$. If $E_1 = R_1 - R^c$ and $E_2 = R_2 - R^c$, then we have $E_1 \leq E_2$. In this sense we can say that R_1 better approximates R^c. Informally, if fewer elements are dropped, i.e., if a smaller drop set P is used, the resulting IMGS factor better approximates the Cholesky factor componentwise.

3 Relations between CIMGS and IC

It is reasonable to ask how CIMGS compares to IC when they are both applied to $A^T A$, because the two algorithms have similar structure. In general, CIMGS generates a different

factor from IC. CIMGS is more stable than IC in the sense that CIMGS will not break down when A is full rank, while in practice IC break down frequently. Furthermore, there is a close relationship between CIMGS and IC when certain conditions are imposed on the sparsity pattern of the target factor.

Theorem 6 *Suppose $A \in R^{m\times n}$, $B = A^TA$, and the set $P \subset P_n$ has the property that for any $1 \le i,j,k \le n$, with $i < j$ and $i < k$, the condition $(i,j) \in P$ and $(i,k) \notin P$ 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 set P respectively, then $R = U$.*

Two observations will make the following proof easier to understand.

Observation 1: Suppose $A \in R^{n\times n}$ and $B \in R^{n\times n}$ are symmetric positive definite matrices for which IC completes using a drop set $P \subset P_n$. 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: Let $A \in R^{m\times n}$ and $P \subset P_n$. Let $B = A^TA$, and suppose $R^{(1)}$ and $B^{(1)}$ are the matrices that arise from one step of CIMGS on B using position set P, so that $B = {R^{(1)}}^T B^{(1)} R^{(1)}$. Suppose $\tilde{B}^{(1)}$ and $U^{(1)}$ are the matrices that arise from one step of IC on B with the same P so that $B = {U^{(1)}}^T \tilde{B}^{(1)} U^{(1)}$. Then $U^{(1)} = R^{(1)}$ and $b_{ij}^{(1)} = \tilde{b}_{ij}^{(1)}$ for those $1 < i,j \le n$, such that either both $(1,i)$ and $(1,j)$ are in P, or both $(1,i)$ and $(1,j)$ are not in P.

Observation 2 can be seen from the algorithms for CIMGS and IC, which differ only in the updating step. CIMGS updates the elements in position (i,j) where either $(1,i)$ or $(1,j)$ are kept. In IC, only the elements in positions (i,j) where both $(1,i)$ and $(1,j)$ are kept. This causes the two algorithms to differ at positions (i,j) where one and only one of $(1,i)$ and $(1,j)$ are kept at the first step. All the other elements are the same after one step of CIMGS and IC. With these observations, we prove theorem 6.

Proof: By 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. For $k = n$, one step of CIMGS gives $B = {R^{(1)}}^T B^{(1)} R^{(1)}$ with $B^{(1)} = (b_{ij}^{(1)})$ and

$$R^{(1)} = \begin{pmatrix} r_{11} & r_{12} \ldots r_{1n} \\ 0 & I \end{pmatrix}.$$

The remaining steps of CIMGS can be viewed as applying CIMGS to the submatrix of $B^{(1)}$ from rows and columns 2 to n, which is an $(n-1)\times(n-1)$ matrix. Let R_2 denote the resulting $(n-1)\times(n-1)$ triangular matrix. The overall CIMGS factor R then has the form $\begin{pmatrix} 1 & 0 \\ 0 & R_2 \end{pmatrix} R^{(1)}$

On the other hand, one step of IC gives $B = {U^{(1)}}^T \tilde{B}^{(1)} U^{(1)}$, where $\tilde{B}^{(1)} = (\tilde{b}_{ij})$ and

$$U^{(1)} = \begin{pmatrix} u_{11} & u_{12} \ldots u_{1n} \\ 0 & I \end{pmatrix}.$$

The remaining steps of IC consist of applying IC to the submatrix of $\tilde{B}^{(1)}$ from rows and columns 2 to n. Denote the resulting $(n-1) \times (n-1)$ triangular matrix by U_2. The resulting IC factor U is $\begin{pmatrix} 1 & 0 \\ 0 & U_2 \end{pmatrix} U^{(1)}$.

By observation 2, $R^{(1)} = U^{(1)}$ and $b_{ij}^{(1)} = \tilde{b}_{ij}^{(1)}$ for those positions where either both $(1,i)$ and $(1,j)$ are in P or both are not in P. We now show that $R_2 = U_2$. Compare the elements of $b_{ij}^{(1)}$ and $\tilde{b}_{ij}^{(1)}$ where $(i,j) \notin P$. Note that only these elements affect the IC factor. If either both $(1,i)$ and $(1,j)$ are in P or both $(1,i)$ and $(1,j)$ are not in P, we have $b_{ij}^{(1)} = \tilde{b}_{ij}^{(1)}$ from observation 2. In other words, $b_{ij}^{(1)} \neq \tilde{b}_{ij}^{(1)}$ is possible only for positions (i,j) such that either $(1,i) \in P$ and $(1,j) \notin P$ or $(1,i) \notin P$ and $(1,j) \in P$. By hypothesis, in these cases $(i,j) \in P$. Therefore, $b_{ij}^{(1)} \neq \tilde{b}_{ij}^{(1)}$ can only be true if $(i,j) \in P$.

Let $\tilde{U}_2$ be the IC factor of the submatrix of $B^{(1)}$ consisting of rows and columns 2 to n. By observation 1, $U_2 = \tilde{U}_2$. Also, according to the induction hypothesis, $\tilde{U}_2 = R_2$. These two conditions taken together imply $U_2 = R_2$ completing the $k = n$ case. By induction, the assertion is true for all n. □

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 $B \in R^{n \times n}$ be a symmetric positive definite matrix. If $P \subset P_n$ has the property that $(i,j) \in P$ and $(i,k) \notin P$ implies $(j,k) \in P$ for all $1 \leq i \leq n$, then the IC factorization algorithm completes successfully.*

Proof: Let $B \in R^{n \times n}$ be a symmetric positive definite matrix. There exists a matrix $A \in R^{m \times n}$, $m \geq n$ with full rank such that $B = A^T A$. According to Theorem 1, IMGS applied to A completes successfully, yielding an upper triangular matrix R with positive diagonal elements such that $A = QR$. By Theorem 6, the conditions on the set P assure that IC applied to $B = A^T A$ generates the same upper triangular matrix as IMGS. So IC completes successfully and generates an upper triangular matrix L^T which is equal to R. □

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^TA is an M–matrix allows us to prove the relationship: the CIMGS factor is closer than the IC factor to the complete Cholesky factor. To establish this result, two lemmas are stated here without proof.

Lemma 2 . *Let $A, B \in R^{n\times n}$ be symmetric positive definite M-matrices with $A \geq B$. Then $R \geq T$, where $R^TR = A$ and $T^TT = B$ are Cholesky factorizations.*

Lemma 3 *Let $A, B \in R^{n\times n}$ be symmetric positive definite M-matrices, with $A \geq B$. Let P be a non-zero position matrix for incomplete Cholesky factorization. Then $R \geq T$, where R and T are the incomplete Cholesky factors of A and B, respectively, using the same P.*

Theorem 8 *Assume that $A \in R^{m\times n}$ is such that $B = A^TA$ is a symmetric positive definite M-matrix. Let R_{CHOL} and R_{IC} be the upper triangular matrices from Cholesky factorization of B and incomplete Cholesky factorization of B with the pattern P, respectively. Let R_{IMGS} be the upper triangular matrix from IMGS on A with the same pattern P. The following relation is satisfied:*

$$R_{CHOL} \leq R_{IMGS} \leq R_{IC},$$

Furthermore, $E_1 \leq E_2$, where

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

Proof: Since IMGS on A generates the same upper triangular matrix R as CIMGS on A^TA, in the proof we use the CIMGS algorithm. The proof is carried out in two parts.

(1) We prove $R_{CHOL} \leq R_{IMGS}$ by induction on the size of the problem. The inequality is trivial for $n = 1$. Assume that it holds for $n - 1$. Partition $B = \begin{pmatrix} b_{11} & b_{12} \\ b_{12}^T & B_{22} \end{pmatrix}$ and $P = \begin{pmatrix} p_{11} & p_{12} \\ p_{12}^T & P_{22} \end{pmatrix}$, and let $R_{CHOL} = \begin{pmatrix} r_{11} & r_{12} \\ 0 & R_{22} \end{pmatrix}$, $R_{IMGS} = \begin{pmatrix} t_{11} & t_{12} \\ 0 & T_{22} \end{pmatrix}$ be the factors from Cholesky factorization and CIMGS factorization of A^TA, respectively. Applying one step of Cholesky factorization and CIMGS factorization to A^TA, we get

$$r_{11} = \sqrt{b_{11}}, \quad r_{12} = \frac{b_{12}}{r_{11}}$$

and

$$t_{11} = \sqrt{b_{11}}, \quad t_{12} = \left(\frac{b_{12}}{t_{11}}\right)_{|p_{12}}.$$

It is easy to see that $r_{11} = t_{11}$ and $r_{12} \leq t_{12}$. Let g_{12} denote the dropped elements from the first step of CIMGS, so that $r_{12} = t_{12} + g_{12}$ and $t_{12}^T g_{12} = 0$.

R_{22} is the Cholesky factor of $B^{(1)} = B_{22} - r_{12}^T r_{12}$, and T_{22} is the CIMGS factor of $\tilde{B}^{(1)} = B_{22} - t_{12}^T g_{12} - g_{12}^T t_{12} - t_{12}^T t_{12}$. Again both $B^{(1)}$ and $\tilde{B}^{(1)}$ are M-matrices. It is not difficult to see that $B^{(1)} \leq \tilde{B}^{(1)}$, because

$$\begin{aligned} B^{(1)} &= B_{22} - r_{12}^T r_{12} \\ &= B_{22} - t_{12}^T t_{12} - t_{12}^T g_{12} - g_{12}^T t_{12} - g_{12}^T g_{12} \\ &= \tilde{B}^{(1)} - g_{12}^T g_{12} \end{aligned}$$

and $g_{12}^T g_{12} \geq 0$. Let $L^T L = \tilde{B}^{(1)}$ be the Cholesky factorization of $\tilde{B}^{(1)}$. By Lemma 2, $R_{22} \leq L$, and by the induction assumption, we have $L \leq T_{22}$. So $R_{22} \leq T_{22}$, which implies $R_{CHOL} \leq R_{IMGS}$. By induction, this is true for any n.

(2) Next we prove $R_{IMGS} \leq R_{IC}$ 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 $R_{IC} = \begin{pmatrix} l_{11} & l_{12} \\ 0 & L_{22} \end{pmatrix}$ be the incomplete Cholesky factor of $A^T A$. Applying one step of incomplete Cholesky factorization to $A^T A$, we have $l_{11} = \sqrt{b_{11}}$, and for components not in P, $l_{12} = \left(\frac{b_{12}}{l_{11}}\right)$. So $l_{11} = t_{11}$ and $l_{12} = t_{12}$. Now L_{22} is the incomplete Cholesky factor of $\hat{B}^{(1)} = B_{22} - l_{12}^T l_{12} \geq \tilde{B}^{(1)}$, and $\hat{B}^{(1)}$ is also an M–matrix. Let $\hat{L}$ be the incomplete Cholesky factor of $\hat{B}^{(1)}$. By the induction hypothesis, we have $T_{22} \leq \hat{L}$. From Lemma 3, $\hat{L} \leq L_{22}$, so $T_{22} \leq L_{22}$. Together, these imply $T \leq L$. By induction, the inequality holds for matrices of any order $n \geq 1$. □

Now we know that when $A^T A$ is an M-matrix, CIMGS will be a better approximation of the full Cholesky factorization than IC with the same sparsity pattern. Experiments on general matrices also show that 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 test

4.1 Test environment

The experiments in this section use test problems that 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 non-zeros (nnz(A)), the number of the non-zeros in the normal equations (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 $B \in \Re^{m \times n}$, denoted $dense(B)$, is the percentage ratio of actual non-zero elements to the maximum possible, i.e., $100(nnz(B)/mn)$.

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 non-zero elements – $421 \leq nnz(A) \leq 78298$. The density of the matrices

ranges from less than 1% to slightly more than 7%. As expected, both the density and non-zero element count increase significantly for the normal equations associated with the test matrices – density reaches the neighborhood of 40% for some matrices, and one problem has almost 400,000 non-zero elements. Most are reasonably conditioned but there are a few which 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 computational fluid dynamics 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 the set of problems, we generate a right-hand side vector consistent with a solution vector whose components are all equal to 1. In this case, we can easily check the accuracy of the method. We measure the following quantities for the accurancy of the method: $\| x - x^* \|_2$, $\| Ax - b \|_2$, $\frac{\|x-x^*\|_2}{\|x*\|_2}$ and $\frac{\|Ax-b\|_2}{\|b\|_2}$.

The CIMGS and incomplete Cholesky factorizations used for comparison in the experiments were implemented in standard Fortran. The packages SPARSPAK-A [6] used for Cholesky factorization and SPARSPAK-B [7] are also in Fortran but benefit from the more careful consideration typical of a numerical software package. All tests were run on a single processor of an Alliant FX/2800. In addition to the algebraic quantities mentioned above, floating point operation counts and the cpu time were also collected in the experiments and are used to assess the efficiency of the various approaches.

A conjugate gradient iterative method, 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. Of course, even though 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 6 of the 30 test problems produced the desired accuracy within n iterations. After $2n$ iterations, a total of 9 matrices satisfy the requirement. Finally, after $2m$ only 2 more are added for a total of 11 out of 30 matrices. If the accuracy condition is altered to constrain the residual 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 5 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.

name	m	n	nnz(A)	nnz(A^TA)	dense(A)	dense(A^TA)
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
conev8	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
fffff800	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

Table 1: Characteristics of test matrices

4.2 Implementation of CIMGS and its performance

As with Cholesky factorization, by reorganizing the computations we can get different versions of CIMGS. The version given earlier can be viewed as a rank-1 update approach. The experiments use a delayed update version. On the i^{th} step, we form the i^{th} row of A^TA, store it in B, then perform all the CIMGS modifications for that row before going on to the next row. During this computation we may need to store elements in a row that are in the drop set, but are needed for the modifications required for later rows. This organization facilitates the use of a very simple dynamic data structure to hold the row as it evolves from A^TA, the original B, to the final form of B. The fill–in elements are computed using a simple pattern union computation for each sparse row triad performed.

The implemented version of CIMGS is:

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

(1) $$B(i, i:n) = \begin{cases} B(i, i:n) - r_{ki} B(k, i:n) & (k,j) \notin P \\ B(i, i:n) - b_{ki} R(k, i:n) & (k,j) \in P \end{cases}$$

endfor
if $b_{ii} > 0$ **then**
(2) $b_{ii} = \sqrt{b_{ii}}$
(3) $B(i, i:n) = B(i, i:n)/b_{ii}$
for $j = i, i+1, \ldots, n$

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

endfor
else
(5) *quit (incomplete factorization can not complete)*
endif
endfor
end

In the algorithm description, there is no specification for the selection of P except for the exclusion of diagonal elements. Ways of selecting P can be divided into two classes: dynamic and static. By dynamic we mean that P is unknown until the completion of the factorization. For example, using a drop tolerance to select retained elements is a dynamic method. Assume that A is normalized so that the diagonal elements of A^TA are equal 1 and all the off diagonal elements are less than or equal to 1. Note that the computation of CIMGS will not make the elements greater than 1. So we can safely choose the 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. For example, P can be selected so that the target factor will have the same sparsity pattern as the normal equations, the way incomplete Cholesky factorization usually does.

By experiment we found that the 'compression' technique is effective in reducing the computational cost of computing the preconditioner. CIMGS produces the preconditioner much more efficiently than IMGS for a given pattern selection strategy. Table 2 shows the number of test matrices in various ranges of speedup, where speedup is defined as the ratio of operation counts of CIMGS over IMGS for dynamic pattern selection. Speedups defined as the ratio of execution times follow similar profiles. The data clearly shows substantial improvement in the production of the preconditioner – a factor of over 2000 in one case. In addition, the quality of the preconditioners produced by CIMGS are the same as for IMGS, with the exception of one problem for which CIMGS fails to produce the preconditioner R, and two problems for which CIMGS is unexpectedly superior. The fact that there is one problem that CIMGS can not produce the factor R while IMGS does confirms the analysis presented earlier showing that CIMGS is more likely than IMGS to breakdown due to the numerical deficiency.

speedup	1-30	30-60	60-100	100-200	200-500	500-1000	>1500
# matrices	6	5	2	9	4	1	2

Table 2: Number of matrices in selected ranges of the ratio of IMGS operation to CIMGS operation count for the preconditioner phase with dynamic pattern selection

Figure 1 demonstrates the reduction in total operations count (for performing the factorization and applying the iterative method) by plotting the ratio of operation counts for IMGS over those of CIMGS for the 30 test cases. In all cases this ratio is no smaller than one.

As with IMGS and ICGS the performance of CIMGS with dynamic dropping is sensitive to the choice of ϵ. For example, when $\epsilon = 0.02$ we find that for BELLMEDT most of the time CIMGS takes to solve the problem is in the factorization phase. We can then increase the size of ϵ to reduce the cost of factorization and to improve the overall performance. Figure 2 shows the effects of varying ϵ for this problem. On the other hand, for problem SAYLR4, the preconditioning quality of R is not good enough and the iterative phase takes an excessive amount of time. Reducing ϵ allows the iterative method to solve the problem, and improves performance as shown in Figure 3. The optimal value of ϵ is unknown in general. This is a potential difficulty of CIMGS, which it shares 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. In any case, this flexibility of pattern selection may allow us to find a better preconditioner after a more careful selection process.

4.3 Comparison of CIMGS with IC

Previously we showed that when A^TA is an M–matrix, R_{CIMGS} better approximates R_{CHOL} than R_{IC} does, so we expect that R_{CIMGS} will perform better than R_{IC} as a preconditioner.

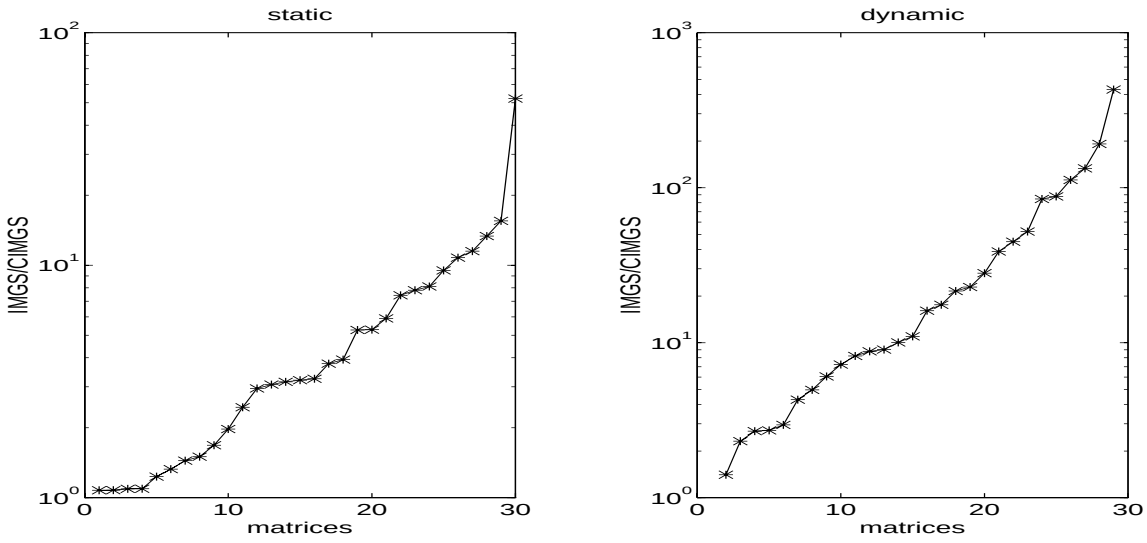


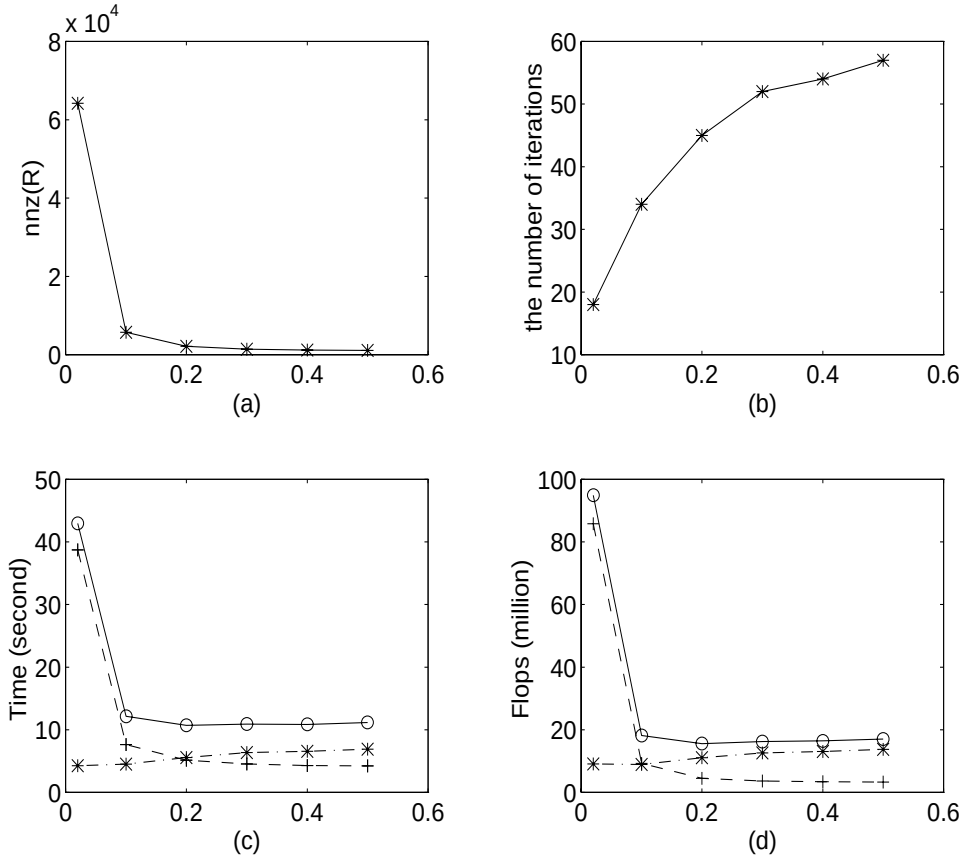
Figure 1: Ratio of total operation counts: IMGS to CIMGS (static and dynamic)

What if the normal equations are not an M-matrix? Since we do not yet have analytic results to answer this question, we next seek to experimentally determine which is better by numerical testing.

In general, IC is much more likely to fail than CIMGS. This is the primary advantage of incomplete orthogonal factorization preconditioning methods over IC preconditioning. It is known that IC fails so often that it is not practical to use it without modification for general problem solving. For problems for which IC exists without modification, we now compare the performance of IC with CIMGS.

When the pattern is chosen to be that of the normal equations, the number of iterations of CGLS preconditioned by CIMGS and IC respectively are shown in Table 3. In the table, the space shown is the number of floating point numbers for the factorization. Operations are counted in millions of floating point operations in the factorization phase. The operation count is the same in each iteration for both IC and CIMGS preconditioner because they produce an R with the same structure. So the iteration number can be used to evaluate the number of operations for the iterative phase.

This table shows that in general, with the drop set P from the complement of the set of positions of non-zero elements of normal equations. the factors R generated by these methods perform similarly as preconditioners. IC is likely to take one or two more iterations than the others. Although the difference is small, it seems to agree with the theoretical analysis for the case when A^TA is an M-matrix. This does not provide a firm basis to conjecture that the theoretical results can be extended to more general problems, because the testing is limited. We do not yet know if this comparison will change as the non-zero pattern is varied or how it changes as, e.g., the angles between the columns of A changes. We also do not know how large the difference between the IC and CIMGS preconditioners can become. Here we only



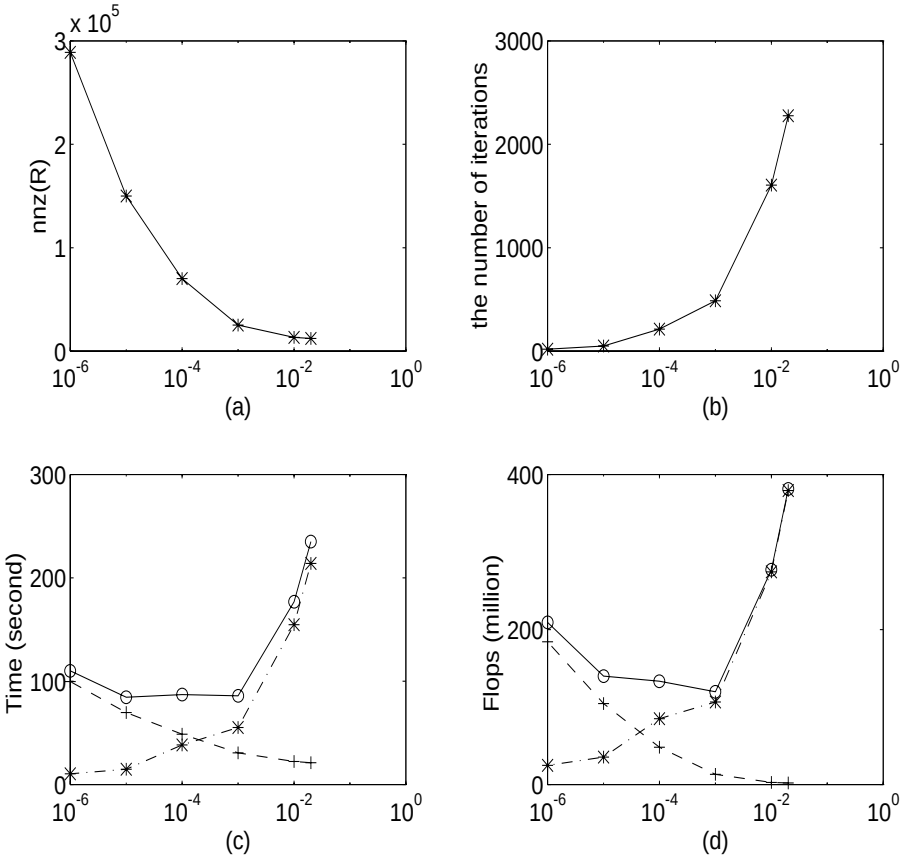
Solid line: Total time or flops.
Dash line: CIMGS factorization time or flops.
Dot line: CGLS iteration time or flops.

Figure 2: Effect of ϵ on the cost of CIMGS factorization BELLMEDT

see a small difference; more testing is needed to answer these question.

As mentioned in the previous section, there are methods that can modify the pattern so that IC will exist. For problems for which IC needs some pattern modification to avoid numerical breakdown, the preconditioner R produced by IC using a modified pattern is the same as the R produced by CIMGS using the same pattern. The performance of the IC preconditioner will be the same (or very close) as that of the CIMGS preconditioner.

From the table we can also see that the differences in the cost of computing these factorizations are quite significant and in general IC is much cheaper than CIMGS. So in those cases where both CIMGS and IC successfully complete use the same static pattern, the cost of the factorization and therefore the cost of solving the system heavily favors IC. CIMGS does have a significant advantage over IC and other preconditioners – CIMGS is the only robust preconditioner that does not have to rely on elaborate restart mechanisms to achieve



Solid line: Total time or flops.
Dash line: CIMGS factorization time or flops.
Dot line: CGLS iteration time or flops.

Figure 3: Effect of ϵ on the cost of the iterative phase of CIMGS for SAYLR4

robustness when the factorization fails to exist. This enables CIMGS to use its adjustability to improve the overall solution process efficiency by using a smaller and more problem dependent non-zero set to generate its preconditioner, possibly offseting the added cost of its factorization compared to IC and similar methods.

This can be seen from the data in Figure 4. For the 12 problems where IC exists and allowed convergence, the pattern selection was adjusted for CIMGS[1]. The CIMGS version is either comparable to, or appreciably better than, the IC version, indicating that with careful adjustment the CIMGS preconditioner can be competitive with or superior to the standard and usually efficient IC preconditioner.

Note that in the implementation of IC, we only keep the positions that the elements will

[1]A description of the adjustments is given in the discussion of the comparison to the normal equations.

Name	Iterations		Space		Operations	
	cimgs	IC	cimgs	ic	cimgs	ic
belladit	14	15	3401	2395	0.17	0.08
bellmedt	16	17	533070	372255	324.30	142.32
dunes8	20	21	119523	6998	1.89	0.06
conev8	1	1	20591	5135	0.38	0.05
strat8	89	91	9558	21757	1.61	0.18
fs_760_1	4	4	232165	13957	8.70	0.02
gre_115	34	35	5749	692	0.07	
steam2	7	7	113548	20237	7.32	0.53
nzfri	57	57	189907	8406	7.52	0.06
scsd6	21	21	11026	2248	0.32	0.03
shell	23	22	95739	2748	0.87	0.01
woodw	34	35	156426	21525	6.61	0.29

Table 3: Comparison of CIMGS and IC.

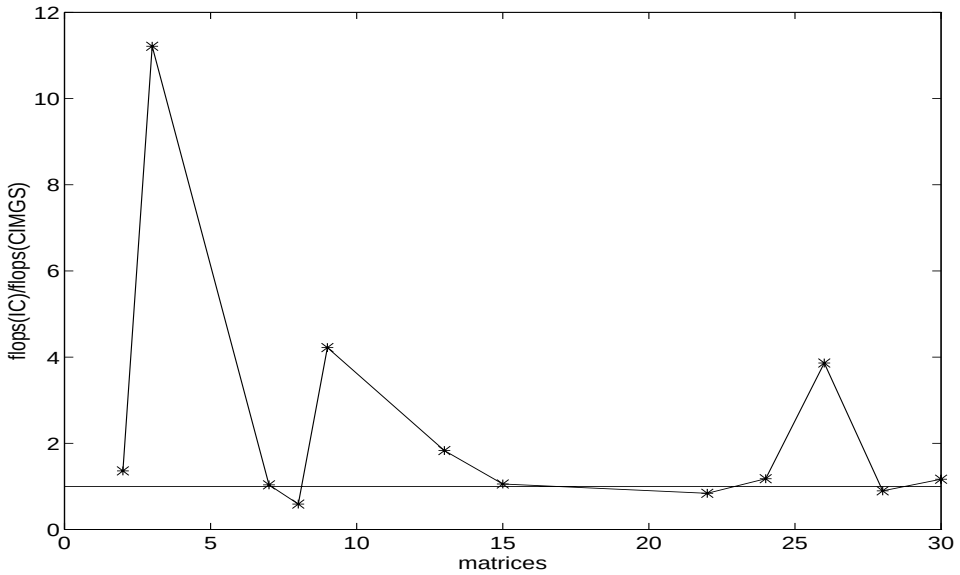


Figure 4: Ratio of total operations for CGLS/IC to total operations for CGLS/CIMGS with adjusted pattern selection

be saved and only do the computations that involve the elements that in those positions. Obviously we need not compute for positions that are in P, because we know those positions will not effect the values of the final factor R. The situation is more complicated for CIMGS, where we not only need to compute the elements not in the drop set P, but also need to compute some of the elements in P because they carry intermediate information that will eventually affect the value of R. On the other hand, it is not necessary to compute all the elements in P because some of them will not affect the value of R. Computing them will involve 'compute–then–drop', which is a waste of time and space. A detailed discussion of an algorithm that identifies these unnecessary computations by symbolic analysis can be found in [16]. The data we show here includes 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.

4.4 Comparison of CIMGS with direct methods

Is the CIMGS preconditioner effective enough to compete with direct methods? We also conducted experiments comparing CIMGS preconditioned CG method with two direct methods: the QR factorization of SPARSPAK-B, and SPARSPAK-A applied to the normal equations.

From the experiments we found that SPARSPAK-B is able to solve all the problems except MC_FE and STEAM2, for which $\| Ax - b \|_2 \leq 10^{-6}$ 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, however. Comparing with other methods tested we find that SPARSPAK-B is the most robust method. There are some problems for which CIMGS breaks down during the factorization when using certain drop sets P, and so a pattern adjustment may be needed. Comparing the number of floating point operations and the time used by the methods, SPARSPAK-B need more floating point operations than the other methods. CGLS preconditioned by CIMGS is more efficient by this measure.

The normal equations method implemented using SPARSPAK-A is much more efficient than the QR in SPARSPAK-B but is less robust. Table 4 shows the number of problems for which SPARSPAK-A fails to generate the Cholesky factor using various reorderings, and the number of problems for which SPARSPAK-A with the corresponding orderings provides the most efficient solution.

Method	Reverse Cuthill-McKee	Refined Quotient Tree	Minimum Degree
# Failure	6	5	6
# Best	9	1	15

Table 4: Performance of SPARSPAK-A

Compared with CGLS preconditioned by CIMGS, the normal equations method is less robust in the sense that it fails on more problems than the CGLS/CIMGS combination.

Furthermore, there is almost no space for improvement in the direct method. For CGLS preconditioned by CIMGS, there is great flexibility in pattern and ordering selection so that if CIMGS with a pattern fails to compute a factor R, we can change to another pattern.

So with a careful selection of the sparsity pattern, CGLS preconditioned by CIMGS can be as efficient as the normal equations method. For example, consider the following two sets of performance tests. The first set, in Figure 6, shows the ratio of flops when using SPARSPAK-A with reverse Cuthill–McKee (RCM) ordering to the flops when using CGLS with CIMGS preconditioning. The second set, in Figure 5, shows the ratio of times. The results were from having 50% of the problems use either the static pattern of the normal equations, or a dynamic pattern with $\epsilon = 0.02$, whichever is better, and the remaining 50% of the problems use dynamic pattern selection with no more than 3 refinements of ϵ.

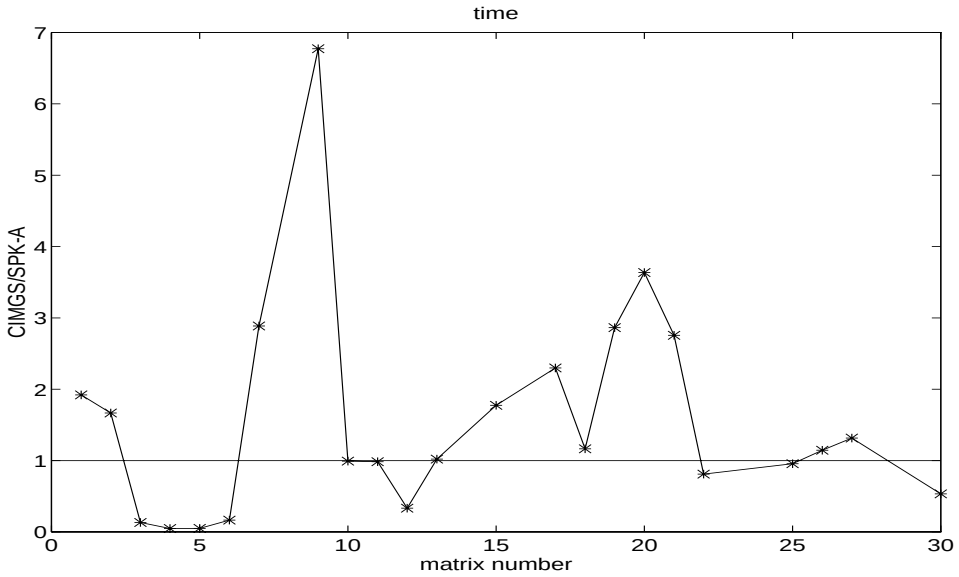


Figure 5: Comparison of times for SPARSPAK-A and CGLS preconditioned by CIMGS

From Figure 6 we see that for some problems, such as BELLMEDT, ILLC1850, WELL1850 and WELL1033, the iterative method works much more efficiently than the normal equations method. For other problems such as STRAT8 and PROSE_2, the normal equations method works better.

The results in Figure 5 differ from those comparing the number of operations. In terms of time, there are more problems for which the normal equations method is better than CGLS preconditioned by CIMGS. This inconsistency is due in part to the difference in optimization levels afforded the two codes. The implementation of CIMGS is not optimized compared to SPARSPAK, and some implementation details of CIMGS can be bettered. For example, our symbolic analysis and data structure allocation can be improved by properly adapting the fast symbolic Cholesky factorization result. This will is a goal of future research.

In summary, compared with the QR method, a robust direct method of solving linear least squares problems, we conclude that for most test problems, using CGLS with CIMGS

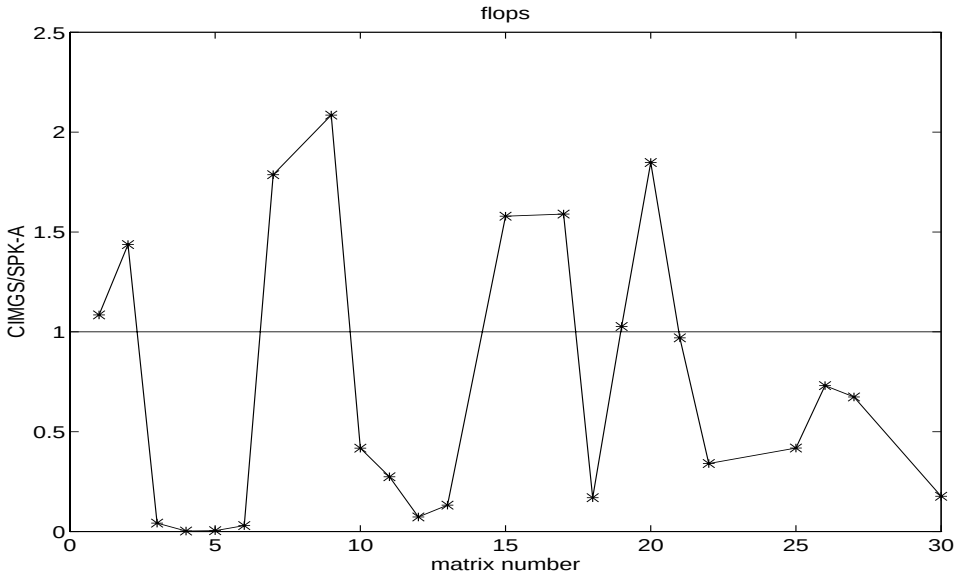


Figure 6: Comparison of operations for SPARSPAK-A and CGLS preconditioned by CIMGS

is much more efficient. On the other hand, the normal equations method, on the problems succeeds in solving, is more efficient. However, if we are free to use the adjustability of the CIMGS preconditioner, we find that CGLS with CIMGS preconditioning can be as efficient as the normal equations method.

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 preconditioner, and CG preconditioned with CIMGS compares favorably with using Cholesky factorization on the normal equations. This suggests that with CIMGS preconditioning, CG can be a viable method for practical use for least squares problems.

Additional theory shows that CIMGS is equivalent to IMGS, the factorization has better backward error properties than complete Cholesky factorization does, 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. Those properties usually seem to numerically hold, even when A^TA 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 incomplete Cholesky factorization does. This allows guaranteeing the existence of IC factorization, 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. This should bring down the computation cost further, and we are currently working on this issue.

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, i.e., 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 of the parallel processing.

The preconditioning method proposed here is applied to CG type iterative methods. How will they perform when combined with other type 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 for the use of CIMGS as a preconditioner.

Another potential research area is to extend the relation of CIMGS with Cholesky and incomplete Cholesky to unsymmetric matrices. It will be interesting and useful if there is a algorithm which has a similar relation with LU factorization and incomplete LU factorization, and avoids the problem of numerical breakdown.

References

[1] S. Ashby, *Polynomial Preconditioning for Conjugate Gradient Methods*, PhD thesis, University of Illinois Urbana-Champaign, 1987. Also available as Tech. Rep. 1355, Department of Computer Science, University of Illinois – Urbana.

[2] ——, *Minimax polynomial preconditioning for Hermitian linear systems*, SIAM J. Mat. Anal. Appl., 12 (1991), pp. 766–789.

[3] M. W. Berry and R. J. Plemmons, *Algorithms and experiments for structural mechanics on high-performance architectrues*, Computer Methods in Applied Mechanics and Engineering, 64 (1987), pp. 487–507.

[4] A. Björck, *SSOR preconditioning methods for sparse least squares problems*, in Proceedings of the Computer Science and Statistics: 12-th 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 Journal on Computing, 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. Rep. CS-84-36, Department of Computer Science, University of Waterloo, 1984.

[7] A. George and E. Ng, *SPARSPAK: Waterloo sparse matrix package, user's guide for SPARSPAK-B*, Tech. Rep. CS-84-37, Department of Computer Science, University of Waterloo, 1984.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins, 2nd ed., 1989.

[9] M. T. Heath, R. J. Plemmons, and R. C. Ward, *Sparse orthogonal schemes for structural optimization using the force method*, SIAM Journal of Scientific and Statistical Computing, 5 (1984), pp. 514–532.

[10] A. Jennings and M. A. Ajiz, *Incomplete methods for solving $A^TAx = b$*, SIAM Journal of Scientific and Statistical Computing, 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*, Mathematics of Computation, 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, 1983.

[13] C. Paige and M. Saunders, *Solution of sparse indefinite systems of equations and least squares problems*, Tech. Rep. STAN-CS-73-399, Stanford University, 1973.

[14] Y. Saad, *Preconditioning techniques for nonsymmetric and indefinite linear systems*, Journal of Computational and Applied Mathematics, 24 (1988), pp. 89–105.

[15] Y. Saad, *SPARSKIT: a basic tool kit for sparse matrix computations*, tech. rep., Center for Supercomputing Research and Development, University of Illinois, Urbana, Illinois, 1990.

[16] X. WANG, *Incomplete factorization preconditioning for linear least squares problems*, PhD thesis, University of Illinois Urbana-Champaign, 1993. Also available as Technical Report # UIUCDCS-R-93-1834, Department of Computer Science, University of Illinois at Urbana – Champaign.

[17] J. H. WILKINSON, *A priori error analysis of algebraic processes*, in Proc. International Congress of Mathematicians, 1968, pp. 119–129.

[18] Z. ZLATEV AND H.B.NIELSON, *Solving large and sparse linear least squares problems by conjugate gradient algorithm*, Computers and Mathematics with Applications, 15 (1988), pp. 185–202.