Electronic Transactions on Numerical Analysis.
Volume 17, pp. 76-92, 2004.
Copyright © 2004, Kent State University.
ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

STRONG RANK REVEALING CHOLESKY FACTORIZATION*

M. GU AND L. MIRANIAN †

Abstract. For any symmetric positive definite $n \times n$ matrix A we introduce a definition of strong rank revealing Cholesky (RRCh) factorization similar to the notion of strong rank revealing QR factorization developed in the joint work of Gu and Eisenstat. There are certain key properties attached to strong RRCh factorization, the importance of which is discussed by Higham in the context of backward stability in his work on Cholesky decomposition of semi-definite matrices. We prove the existence of a pivoting strategy which, if applied in addition to standard Cholesky decomposition, leads to a strong RRCh factorization, and present two algorithms which use pivoting strategies based on the idea of local maximum volumes to compute a strong RRCh decomposition.

Key words. Cholesky decomposition, LU decomposition, QR decomposition, rank revealing, numerical rank, singular values, strong rank revealing QR factorization.

AMS subject classifications. 65F30.

1. Introduction. Cholesky decomposition is a fundamental tool in numerical linear algebra, and the standard algorithm for its computation is considered as one of the most numerically stable among all matrix algorithms. In some applications, it is necessary to compute decomposition with linearly independent columns being separated from linearly dependent ones, i.e., compute the rank revealing decomposition, which is not usually achieved by standard algorithms. One application of rank revealing factorizations is the active-set method discussed by Fletcher [8]. Also, rank revealing factorization can be used to solve least-squares problems using the method proposed by Björck [1, 2].

Usually rank revealing factorizations produce a decomposition with two components: the full-rank portion, and the rank deficient, or redundant, part. In practice, the quality of rank revealing decomposition is governed by the following two distances: how far from singular the full-rank portion is, and how close the exact rank deficient part is to the numerical rank deficient portion, where rank deficiency is estimated with some tolerance. We develop theoretical bounds for full-rank and rank deficient components of strong RRCh decomposition, which are very similar to those obtained for rank revealing QR factorizations, and observe that using algorithms 3 and 4 significantly smaller bounds are obtained in practice.

In particular, consider Cholesky decomposition

$$\Pi A \Pi^T = LDL^T,$$

where $L = \begin{pmatrix} A_k & \\ B_k & I_{n-k} \end{pmatrix}$, $D = \begin{pmatrix} I_k & \\ & C_k \end{pmatrix}$, $A_k \in \mathbb{R}^{k,k}$, $B_k \in \mathbb{R}^{n-k,k}$, $C_k \in \mathbb{R}^{n-k,n-k}$, Π is a permutation matrix, and I_p is $p \times p$ identity matrix. The numerical approximation to the null space of A is

$$N = \begin{pmatrix} -A_k^{-T} B_k^T \\ I_{n-k} \end{pmatrix},$$

which is governed by matrix $W = A_k^{-T} B_k^T$. Hence, we need a pivoting strategy that reveals the linear dependence among columns of a matrix and keeps elements of N bounded by some slow growing polynomials in k and n. Higham in [13] has shown that nearly the tightest possible upper bound for the error $\|A - \hat{L}_k \hat{L}_k^T\|_2$, where $\hat{L}_k$ is the computed $k \times n$ Cholesky

*Received December 27, 2002. Acepted for publication September 25, 2003. Recommended by Paul Van Dooren.

†Department of Mathematics, University of California, Berkeley, CA, USA.

factor, is also governed by W, which implies that stability of the algorithm depends on how small $\|W\|_2$ is.

We introduce a strong rank revealing Cholesky (RRCh) decomposition and propose to perform Cholesky factorization with diagonal pivoting, where on every stage of the factorization we look for the "most linearly independent" column in the "not-yet-factored" portion of the matrix and swap it with the appropriate column in "already-factored" part of the matrix. This pivoting strategy was first introduced by Gu and Eisenstat in [11] and is known as maximum local volumes strategy. Since Cholesky factorization requires significantly less operations than QR, algorithms presented in this paper compute strong rank revealing Cholesky factorization much faster than algorithms that use rank revealing QR factorization.

The rest of the paper is organized as follows. In section 2, we give an overview of the previous results on rank revealing LU and QR factorizations. In section 3, we introduce a definition of strong rank revealing Cholesky decomposition, and, in section 4, we discuss the existence of such factorization. Section 5 contains the first proposed algorithm. Section 6 is devoted to the second algorithm which is based on convex optimization approach. Complete pivoting strategy is discussed in section 7, and numerical experiments are presented in section 8. Concluding remarks are in the final section 9.

2. Previous results on rank revealing LU and QR decompositions.

Assume $A \in \mathbb{R}^{n,m}$ has numerical rank k. Then, according to [19], the factorization

$$\Pi_1 A \Pi_2^T = \begin{pmatrix} L_{11} & \\ L_{21} & I_{n-k} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ & U_{22} \end{pmatrix}, \tag{2.1}$$

where $L_{11}, U_{11} \in \mathbb{R}^{k,k}$, $U_{12} \in \mathbb{R}^{k,m-k}$, $U_{22} \in \mathbb{R}^{m-k,m-k}$, $L_{21} \in \mathbb{R}^{n-k,k}$, $I_{n-k} \in \mathbb{R}^{n-k,m-k}$ and Π_1 and Π_2 are permutation matrices, is a rank revealing LU (RRLU) factorization if

$$\sigma_k(A) \geq \sigma_{\min}(L_{11}U_{11}) \gg \sigma_{\max}(U_{22}) \geq \sigma_{k+1}(A) \approx 0.$$

Given any rank-deficient matrix $A \in \mathbb{R}^{n,m}$, exact arithmetic Gaussian elimination with complete pivoting, unlike partial pivoting, will reveal the rank of the matrix. However, for nearly singular matrices even complete pivoting may not reveal the rank correctly. This is shown in the following example by Peters and Wilkinson [24]:

$$A = \begin{pmatrix} 1 & -1 & -1 & \dots & -1 \\ & 1 & -1 & \dots & -1 \\ & & \ddots & & \vdots \\ & & & & 1 \end{pmatrix}.$$

There are no small pivots, but this matrix has a very small singular value when size of A is sufficiently large.

Several papers, [4, 18, 19], were dedicated to the question of whether there is a pivoting strategy that will force entries with magnitudes comparable to those of small singular values to concentrate in the lower-right corner of U, so that LU decomposition reveals the numerical rank. In [4] the existence of such pivoting is shown for the case of only one small singular value, and for RRLU factorization (2.1) the following bound is obtained:

$$\|U_{22}\|_2 \leq \frac{n\sigma_{n-1}\sigma_n}{\sigma_{n-1} - n\sigma_n},$$

where σ_l denotes lth singular value of A.

Later, in [18] generalized case of more than one small singular value is discussed, and results are summarized in the following theorem:

THEOREM 2.1. *[18]*
Let $A \in \mathbb{R}^{n,n}$ be nonsingular, then for any integer k, $1 \le k < n$ there exist permutations Π_1 and Π_2 such that

$$\Pi_1 A \Pi_2^T = \begin{pmatrix} L_{11} & \\ L_{21} & I_{n-k} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ & U_{22} \end{pmatrix},$$

where L_{11} is unit lower triangular and U_{11} is upper triangular, with $U_{22} = [u_{i,j}]$ bounded by:

$$|u_{i,j}| \le f(n,k)\sigma_{k+1},$$

where

$$f(n,k) = \frac{n!}{k!(n-k)!}\left[1 - \frac{n!}{k!(n-k)!}\frac{\sigma_{k+1}}{\sigma_k}\right]^{-1}$$

for $1 \le i, j \le n-k$ provided that the quantity inside brackets is positive.

However, bounds obtained in [18] may increase very rapidly (faster than exponential, in the worst case) because of its combinatorial nature. In [19] the following improved bounds are obtained:

THEOREM 2.2. *[19]*
Let $A \in \mathbb{R}^{n,n}$ with numerical rank k and $\sigma_1 \ge \ldots \ge \sigma_k \gg \sigma_{k+1} \ge \ldots \ge \sigma_n \ge 0$. There exist permutations Π_1 and Π_2 such that

$$\Pi_1 A \Pi_2^T = \begin{pmatrix} L_{11} & \\ L_{21} & I_{n-k} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ & U_{22} \end{pmatrix},$$

where L_{11} is unit lower triangular and U_{11} is upper triangular. If

$$\frac{\sigma_{k+1}}{\sigma_k} D_{nk} < 1, \quad \textit{where} \;\; D_{nk} = k(n-k) + \min(k, n-k),$$

then

$$\|U_{22}\|_2 \le \frac{\sigma_k \sigma_{k+1} D_{nk}}{\sigma_k - \sigma_{k+1} D_{nk}}$$

and

$$\sigma_{\min}(L_{11}U_{11}) \ge \frac{\sigma_k - \sigma_{k+1} D_{nk}}{D_{nk}}.$$

Pan, in [23], using Schur Complement factorizations and local maximum volumes, deduced the following bounds:

THEOREM 2.3. *[23]*
Let $A \in \mathbb{R}^{n,n}$ with $\sigma_1 \ge \ldots \ge \sigma_k \gg \sigma_{k+1} \ge \ldots \ge \sigma_n \ge 0$. Then there exist permutations Π_1 and Π_2 such that

$$\Pi_1 A \Pi_2^T = \begin{pmatrix} L_{11} & \\ L_{21} & I_{n-k} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ & U_{22} \end{pmatrix},$$

ETNA
Kent State University
etna@mcs.kent.edu

where L_{11} is unit lower triangular and U_{11} is upper triangular,

$$\sigma_{k+1} \leq \|U_{22}\|_2 \leq \big(k(n-k)+1\big)\sigma_{k+1}$$

and

$$\sigma_k \geq \sigma_{\min}(L_{11}U_{11}) \geq \frac{\sigma_k}{k(n-k)+1}.$$

These bounds are very similar to those obtained in rank revealing QR factorizations in [6, 11, 15]. One of the definitions of rank revealing QR factorization presented in [6, 15] is the following: assume $M \in \mathbb{R}^{m,n}$ has numerical rank k, Q is orthogonal, $A_k \in \mathbb{R}^{k,k}$ is upper triangular with nonnegative diagonal entries, $B_k \in \mathbb{R}^{k,n-k}$, $C_k \in \mathbb{R}^{m-k,n-k}$ and Π is a permutation matrix. Then we call factorization

$$M\Pi = QR = Q\begin{pmatrix} A_k & B_k \\ & C_k \end{pmatrix} \tag{2.2}$$

rank revealing QR (RRQR) factorization if

$$\sigma_{\min}(A_k) \geq \frac{\sigma_k(M)}{p(k,n)}, \quad \text{and} \quad \sigma_{\max}(C_k) \leq \sigma_{k+1}(M)\, p(k,n), \tag{2.3}$$

where $p(k,n)$ is a function bounded by low-degree polynomial in k and n. Other, less restrictive definitions are discussed in [6, 22].

RRQR factorization was first introduced by Golub [9], who, with Businger [3], developed the first algorithm for computing the factorization. The algorithm was based on QR with column pivoting, and worked well in practice. However, there are examples (Kahan matrix, [20]) where the factorization it produces fails to satisfy condition (2.3).

Pierce and Lewis in [25] developed an algorithm to compute sparse multi-frontal RRQR factorization. In [21] Meyer and Pierce present advances towards the development of an iterative rank revealing method. Hough and Vavasis in [17] developed an algorithm to solve an ill-conditioned full rank weighted least-squares problem using RRQR factorization as a part of their algorithm. Also, a URV rank revealing decomposition was proposed by Stewart in [26].

In [15] Hong and Pan showed that there exists RRQR factorization with $p(k,n) = \sqrt{k(n-k)+\min(k,n-k)}$ and Chandrasekaran and Ipsen in [6] developed an efficient algorithm that is guaranteed to find an RRQR given k.

In some applications, such as rank deficient least-squares computations and subspace tracking, where elements of $A_k^{-1}B_k$ are expected to be small, RRQR does not lead to a stable algorithm. In these cases strong RRQR, first presented in [11], is being used: factorization (2.2) is called a *strong rank revealing QR (RRQR) factorization* if

1. $\sigma_i(A_k) \geq \dfrac{\sigma_i(M)}{q_1(k,n)}, \quad \sigma_j(C_k) \leq \sigma_{k+j}(M)\, q_1(k,n),$
2. $|(A_k^{-1}B_k)_{i,j}| \leq q_2(k,n)$

for $1 \leq i \leq k$ and $1 \leq j \leq n-k$, where $q_1(k,n)$ and $q_2(k,n)$ are functions bounded by low-degree polynomials in k and n.

Pan and Tang in [22] developed an algorithm that, given $f > 1$ computes strong RRQR with $q_1(k,n) = \sqrt{1+f^2k(n-k)}$ and $q_2(k,n) = f$. Later, in [11], a different, but mathematically equivalent algorithm, was presented by Gu and Eisenstat. The new algorithm was based on the idea of local maximum volumes. The same idea will be used in this paper to develop an efficient algorithms for computing strong rank revealing Cholesky decomposition.

ETNA
Kent State University
etna@mcs.kent.edu

3. Strong rank revealing Cholesky decomposition. In this section, we explore the idea of using significant gaps between singular values to define the numerical rank of a matrix and introduce strong rank revealing Cholesky decomposition.

Given any symmetric positive definite matrix $A \in \mathbb{R}^{n,n}$, we consider a partial Cholesky decomposition with the diagonal pivoting

$$\Pi A \Pi^T = L D L^T, \tag{3.1}$$

where

$$L = \begin{pmatrix} A_k & \\ B_k & I_{n-k} \end{pmatrix}, \quad D = \begin{pmatrix} I_k & \\ & C_k \end{pmatrix},$$

$A_k \in \mathbb{R}^{k,k}$, $B_k \in \mathbb{R}^{n-k,k}$, $C_k \in \mathbb{R}^{n-k,n-k}$, and I_p is $p \times p$ identity matrix. According to the interlacing property of singular values, for any permutation Π, we have

$$\Big(\sigma_i(A_k)\Big)^2 \le \sigma_i(A) \quad \text{and} \quad \sigma_j(C_k) \ge \sigma_{k+j}(A)$$

for $1 \le i \le k$ and $1 \le j \le n-k$. Hence,

$$\Big(\sigma_{\min}(A_k)\Big)^2 \le \sigma_k(A) \quad \text{and} \quad \sigma_{\max}(C_k) \ge \sigma_{k+1}(A).$$

Assume that $\sigma_k(A) \gg \sigma_{k+1}(A) \approx 0$, so that k would be the numerical rank of A. Then we would like to choose permutation matrix Π in such a way that $\sigma_{\min}(A_k)$ is sufficiently large and $\sigma_{\max}(C_k)$ is sufficiently small. In this paper we will call factorization (3.1) *a strong rank revealing Cholesky (RRCh) decomposition* if it satisfies the following conditions:

1. $\Big(\sigma_i(A_k)\Big)^2 \ge \dfrac{\sigma_i(A)}{q_1(k,n)}; \quad \sigma_j(C_k) \le \sigma_{k+j}(A)\, q_1(k,n)$
2. $|(A_k^{-T} B_k^T)_{ij}| \le q_2(k,n)$

for $1 \le i \le k,\ 1 \le j \le n-k$, where $q_2(k,n)$ and $q_1(k,n)$ are bounded by some low degree polynomials in k and n.

4. The existence of strong rank revealing Cholesky decomposition. In this section we prove the existence of permutation matrix Π which makes a strong RRCh decomposition possible. It is proven in Theorem 4.2 of this section that permutation matrix obtained using Lemma 4.1 is the one necessary for strong RRCh decomposition with elements of $A_k^{-T} B_k^T$ bounded by slow growing function in k and n.

According to the definition given at the end of the previous section, strong RRCh decomposition requires that *every* singular value of A_k is sufficiently large, *every* singular value of C_k is sufficiently small, and *every* element of $A_k^{-T} B_k^T$ is bounded. As first observed in [11],

$$\det(A) = \det(L)\det(D)\det(L^T) = \det(A_k)^2\ \det(C_k),$$

hence

$$\det(A_k) = \prod_{i=1}^{k} \sigma_i(A_k) = \sqrt{\det(A) / \prod_{j=1}^{n-k} \sigma_j(C_k)}. \tag{4.1}$$

This implies that strong RRCh decomposition also results in a large $\det(A_k)$.

Let us introduce notation.

ETNA
Kent State University
etna@mcs.kent.edu

1. If A is a nonsingular $k \times k$ matrix then $\omega_i(A)$ denotes the 2-norm of the i-th row of A^{-T} and $\omega(A) = (\omega_1(A), \ldots, \omega_k(A))$.
2. For any matrix C, $\gamma_j(C)$ denotes the 2-norm of the j-th column of C.
3. $\Pi_{i,j}$ denotes the permutation matrix obtained by interchanging rows i and j in the identity matrix.
4. In the partial Cholesky factorization

$$X = LDL^T$$

of a matrix $X \in \mathbb{R}^{n,n}$, where L and D are defined in section 3, let $\mathcal{C}_k(X)$ be the pair (L, D).

Now, given k and some real number $f \geq 1$, Algorithm 1 below constructs a strong RRCh decomposition by performing column and row interchanges to maximize $\det(A_k)$.

ALGORITHM 1. *Compute strong RRCh decomposition, given k.*

$(L, D) := \mathcal{C}_k(A)$; $\Pi := I$;
while *there exists i and j such that* $\det(\hat{A}_k)/\det(A_k) > f$,
where $L = \begin{pmatrix} A_k & \\ B_k & I_{n-k} \end{pmatrix}$ *and* $\mathcal{C}_k(\Pi_{i,k+j} A \Pi_{i,k+j}^T) = (\hat{L}, \hat{D})$, ***do***
Find such i and j;
Compute $(L, D) := \mathcal{C}_k(\Pi_{i,k+j} A \Pi_{i,k+j}^T)$ *and* $\Pi := \Pi\, \Pi_{i,k+j}$;
endwhile;

This algorithm interchanges a pair of columns and rows when this increases $\det(A_k)$ by at least a factor of f. Since there is only a finite number of permutations to perform, and none repeat, the algorithm stops eventually.

To prove that Algorithm 1 computes strong RRCh decomposition, we first express $\det(\hat{A}_k)/\det(A_k)$ in terms of $\omega_i(A_k)$, $(C_k)_{jj}$ and $\left(A_k^{-T} B_k^T\right)_{ij}$. Observe that D is symmetric positive definite matrix, hence

$$\sqrt{D} = \begin{pmatrix} I_k & 0 \\ 0 & \sqrt{C_k} \end{pmatrix}$$

is a symmetric positive definite square root of D. Let us write $\bar{L} = L\sqrt{D}$. Then $A = LDL^T = \bar{L}\bar{L}^T$, where $\bar{L}$ is not strictly lower triangular, but instead block lower triangular:

$$\bar{L} = \begin{pmatrix} A_k & 0 \\ B_k & \sqrt{C_k} \end{pmatrix}.$$

Since $\Pi A \Pi^T = \Pi L D L^T \Pi^T = (\Pi\bar{L})(\Pi\bar{L})^T$, the permutation Π swaps rows of $\bar{L}$ and destroys its lower triangular structure. So we use Givens rotations to re-triangularize it, i.e., $\Pi\bar{L} = \tilde{L}Q$, where Q is an orthogonal matrix. Then,

$$\Pi A \Pi^T = \tilde{L} Q Q^T \tilde{L}^T = \tilde{L}\tilde{L}^T,$$

where $\tilde{L}$ is block lower triangular.

Now assume $A = \bar{L}\bar{L}^T$ and $\Pi A \Pi^T = \tilde{L}\tilde{L}^T$, where Π permutes rows i and $k+j$ of $\bar{L}$. The following lemma expresses $\det(\tilde{A}_k)/\det(A_k)$ in terms of $\omega_i(A_k)$, $(C_k)_{jj}$ and $\left(A_k^{-T} B_k^T\right)_{ij}$.

LEMMA 4.1.

$$\frac{\det\left(\tilde{A}_k\right)^2}{\det\left(A_k\right)^2} = \left(A_k^{-T} B_k^T\right)_{ij}^2 + \left(C_k\right)_{jj} \left(\omega_i\left(A_k\right)\right)^2.$$

Proof: To prove this lemma we apply Lemma 3.1, [11] to $\bar{L}^T$ obtaining

$$\frac{\det(\tilde{A}_k)^2}{\det(A_k)^2} = \left(A_k^{-T}B_k^T\right)_{ij}^2 + \left(\gamma_j\left(\sqrt{C_k}\right)\ \omega_i\left(A_k\right)\right)^2.$$

Then we observe that

$$\left(\gamma_j\left(\sqrt{C_k}\right)\right)^2 = \left(e_j\sqrt{C_k}\right)\left(e_j\sqrt{C_k}\right)^T = e_j C_k e_j = (C_k)_{jj},$$

which proves the lemma. □

Let

$$\rho(L,D,k) = \max_{1\le i\le k,\ 1\le j\le n-k}\left(\left(A_k^{-T}B_k^T\right)_{ij}^2 + (C_k)_{jj}\left(\omega_i(A_k)\right)^2\right).$$

Then, according to Lemma 4.1, Algorithm 1 can be rewritten as follows:

ALGORITHM 2. *Compute strong RRCh decomposition, given k.*

$(L,D) := \mathcal{C}_k(A)$; $\Pi := I$;
while $\rho(L,D,k) > f$, ***do***
  Find i and j such that $\left(A_k^{-T}B_k^T\right)_{ij}^2 + (C_k)_{jj}\left(\omega_i(A_k)\right)^2 > f$*;*
  Compute $(L,D) := \mathcal{C}_k(\Pi_{i,k+j}A\Pi_{i,k+j}^T)$ *and* $\Pi := \Pi\ \Pi_{i,k+j}$;
endwhile;

Since Algorithm 1 is equivalent to Algorithm 2, it eventually stops and finds permutation Π for which $\rho(L,D,k) \le f$. This implies that condition (2) of the definition of strong RRCh decomposition in section 3 is satisfied with $q_2(k,n) = f$. Theorem 4.2 discussed below will imply that condition (1) is also satisfied with $q_1(k,n) = \sqrt{1+f^2k(n-k)}$, which means that Algorithms 1 and 2 compute strong RRCh decomposition, given k.

THEOREM 4.2. *(Theorem 3.2 in [11])*
Suppose we have

$$M = \begin{pmatrix} P_k & Q_k \\ & R_k \end{pmatrix}.$$

Let

$$\rho(M,k) = \max_{i,j}\sqrt{\left(P_k^{-1}Q_k\right)_{ij}^2 + \left(\gamma_j(R_k)\,\omega_i(P_k^T)\right)^2}.$$

If $\rho(M,k) \le f$, where f is some constant and $q_1(k,n) = \sqrt{1+f^2k(n-k)}$, then

$$\sigma_i(P_k) \ge \frac{\sigma_i(M)}{q_1(k,n)},\quad 1\le i\le k$$

and

$$\sigma_j(R_k) \le \sigma_{k+j}(M)q_1(k,n),\quad 1\le j\le n-k.$$

This theorem, applied to $\bar{L}^T$, implies condition (1) of the definition of strong RRCh decomposition, and hence the existence of RRCh decomposition is proved.

Theorem 4.2 and Lemma 4.1 combined together lead to Algorithm 3 which, on every step of the Cholesky decomposition with diagonal pivoting, compares $\hat{\rho}(L,D,k)$ (defined at the beginning of the next section) with f. If $\hat{\rho}(L,D,k) > f$, then Theorem 4.2 and Lemma 4.1 imply that $\det(A_k)$ can be made at least f times bigger by interchanging the appropriate rows in $\bar{L}$. We do the swaps until $\det(A_k)$ is large enough, i.e. $\hat{\rho}(L,D,k) < f$, and then we resume standard Cholesky decomposition with diagonal pivoting.

ETNA
Kent State University
etna@mcs.kent.edu

5. Computing strong RRCh decomposition: Algorithm 3. Given a real number $f \geq 1$ and a tolerance δ, Algorithm 3 computes numerical rank k and produces a strong RRCh decomposition. It is a combination of Algorithms 1 and 2, but uses

$$\hat{\rho}(L, D, k) = \max_{1 \leq i \leq k,\ 1 \leq j \leq n-k} \max\left\{ \left| \left(A_k^{-T} B_k^T\right)_{ij} \right| ;\ \sqrt{\left(C_k\right)_{jj}}\ \left(\omega_i(A_k)\right) \right\}$$

and computes $\omega_i(A_k)$ and $A_k^{-T} B_k^T$ recursively for greater efficiency. Observe that $\hat{\rho}(L, D, k)$ is different from $\rho(A, k)$, as defined in Theorem 4.2, but clearly if $\hat{\rho}(L, D, k) > f$ then $\rho(A, k)$ is also greater than f and hence the result of Theorem 4.2 is applicable to Algorithm 3. In the algorithm below the largest diagonal element of C_k is denoted by $\max(\mathrm{diag}(C_k))$. Update and modification formulas are presented in section 5.1 and 5.2.

ALGORITHM 3.

$k := 0$; $\mathcal{C}_k(A) = (L, D)$; $\Pi := I$;
Initialize $\omega(A_k) \in \mathbb{R}^{1,k}$; $A_k^{-T} B_k^T \in \mathbb{R}^{k,n-k}$;
while $\max(\mathrm{diag}(C_k)) \geq \delta$ ***do***
  $j = \mathrm{argmax}_{1 \leq j \leq n-k}(\max(\mathrm{diag}(C_k)))$;
  $k := k + 1$;
  Compute $\mathcal{C}_k(\Pi_{k,k+j-1} A \Pi_{k,k+j-1}^T)$; $\Pi := \Pi\, \Pi_{k,k+j-1}$;
  Update $\omega(A_k)$; $A_k^{-T} B_k^T$;
  while $\hat{\rho}(L, D, k) \geq f$ ***do***
    Find i and j such that $\left|(A_k^{-T} B_k^T)_{ij}\right| \geq f$ *or* $\sqrt{(C_k)_{jj}}\ \omega_i(A_k) \geq f$
    Compute $\mathcal{C}_k(\Pi_{i,k+j} A \Pi_{i,k+j}^T)$, $\Pi := \Pi\, \Pi_{i,k+j}$;
    Modify $\omega(A_k)$; $A_k^{-T} B_k^T$;
  endwhile
endwhile

5.1. Updating formulas. Throughout sections 5.1 and 5.2 lower-case letters denote row vectors, upper-case letters denote matrices, and Greek letters denote real numbers.

On the $(k-1)$th step we have

$$L = \begin{pmatrix} A_{k-1} & \\ B_{k-1} & I_{n-k+1} \end{pmatrix} \quad \text{and} \quad D = \begin{pmatrix} I_{k-1} & \\ & C_{k-1} \end{pmatrix}.$$

Let

$$C_{k-1} = \begin{pmatrix} \alpha^2 & w \\ w^T & K_{11} \end{pmatrix}; \quad B_{k-1} = \begin{pmatrix} b \\ B \end{pmatrix}; \quad A_{k-1}^{-T} B_{k-1}^T = (u^T, U).$$

By Cholesky factorization, we have that

$$A_k = \begin{pmatrix} A_{k-1} & \\ b & \alpha \end{pmatrix},\ B_k = \left(B; w^T/\alpha\right),\ A_k^{-1} = \begin{pmatrix} A_{k-1}^{-1} & \\ -u/\alpha & 1/\alpha \end{pmatrix}.$$

Then,

$$\omega_i^2(A_k) = \omega_i^2(A_{k-1}) + u_i^2/\alpha^2 \quad \text{for} \quad 1 \leq i \leq k-1;\ \ \omega_k^2(A_k) = 1/\alpha^2$$

and

$$A_k^{-T} B_k^T = \begin{pmatrix} U - u^T w/\alpha^2 \\ w/\alpha^2 \end{pmatrix}.$$

ETNA
Kent State University
etna@mcs.kent.edu

5.2. Modifications formulas. We want to interchange the ith and $(k+j)$th diagonal elements of $\Pi A \Pi^T$. This corresponds to interchanging the ith and $(k+j)$th rows of L, which will be done by reducing this general interchange to the interchange between the kth and $(k+1)$st rows.

5.2.1. Interchange within the last $n-k$ diagonal elements of A. If $j > 1$, then interchanging the $(k+j)$th and $(k+1)$st columns of L^T corresponds to interchanging the $(k+1)$st and $(k+j)$th diagonal elements of L and D. This interchange leaves L lower triangular and D block-diagonal and will only swap the first and jth columns of B_k^T and $A_k^{-T} B_k^T$.

5.2.2. Interchange within the first k diagonal elements of A. If $i < k$, then we interchange the kth and ith rows of L. This leaves D unchanged, and L is not lower triangular any more. But we can re-triangularize it using Givens rotations. Denote $\Pi_{k,i} L = \hat{L} Q$, where $\hat{L}$ is lower triangular. Then

$$\Pi_{k,i} L D L^T \Pi_{k,i}^T = \hat{L} Q D Q^T \hat{L}^T = \hat{L} D \hat{L}^T,$$

since Q affects only the upper left $k \times k$ corner of D, which is identity. Now $\hat{A}_k = \Pi_{k,i} A_k Q^T$, hence $\hat{A}_k^{-1} = Q A_k^{-1} \Pi_{k,i}^T$, and post-multiplication by an orthogonal matrix does not affect the 2-norms of rows of A_k^{-T}. This implies that $\omega(A_k)$ just had its kth and ith elements permuted. Also, $A_k^{-T} B_k^T$ has its kth and ith rows permuted.

The main cost of this reduction comes from computing re-triangularization of L after swapping its ith and kth rows, which takes about $3k(2n-k)$ flops.

5.2.3. Interchange between kth and $(k+1)$st diagonal elements of A. Suppose we have

$$\Pi_{k,k+1} A \Pi_{k,k+1}^T = \Pi_{k,k+1} L D L^T \Pi_{k,k+1}^T.$$

Here, $\Pi_{k,k+1} L$ is not lower triangular anymore, so we re-triangularize it $\Pi_{k,k+1} L = \hat{L}^T Q$, where $\hat{L}$ is lower triangular. Notice that in matrix $\Pi_{k,k+1} L$ only the element $(k, k+1)$ needs to be reduced to zero, hence Q is a single Givens rotation matrix that affects only the kth and $(k+1)$st rows of a matrix if multiplied from the right, and the kth and $(k+1)$st column if multiplied from the left. Let's assume that we performed the $(k+1)$st step of Cholesky decomposition zeroing out elements to the right of element $L(k+1, k+1)$. We have

$$\Pi_{k,k+1} L D L^T \Pi_{k,k+1}^T = \hat{L} Q^T D Q \hat{L}^T = \hat{L} D \hat{L}^T,$$

where we used the fact that $Q^T D Q = D$, since the upper-left $(k+1) \times (k+1)$ corner of D is just the identity. Write

$$L = \begin{pmatrix} A_{k-1} & & & \\ b_1 & \beta & & \\ b_2 & \beta\mu & \beta\nu & \\ B & c_1^T & c_2^T & I_{n-k-1} \end{pmatrix} \quad \text{and} \quad \hat{L} = \begin{pmatrix} A_{k-1} & & & \\ b_2 & \hat{\beta} & & \\ b_1 & \beta\mu/\rho & \beta\nu/\rho & \\ B & \hat{c}_1^T & \hat{c}_2^T & I_{n-k-1} \end{pmatrix},$$

where $\rho = \sqrt{\mu^2 + \nu^2}$, $\hat{\beta} = \beta\rho$, $\hat{c}_1 = (\mu c_1 + \nu c_2)/\rho$, and $\hat{c}_2 = (\nu c_1 - \mu c_2)/\rho$. From the expression for L we can see that

$$A_k^{-1} = \begin{pmatrix} A_{k-1}^{-1} & 0 \\ -u/\beta & 1/\beta \end{pmatrix},$$

ETNA
Kent State University
etna@mcs.kent.edu

where $u = b_1 A_{k-1}^{-1}$ is computed using back substitution. Also,

$$A_k^{-T} B_k^T \equiv \begin{pmatrix} u_1^T & U \\ \mu & u_2 \end{pmatrix} = \begin{pmatrix} A_{k-1}^{-T} & -u^T/\beta \\ 0 & 1/\beta \end{pmatrix} \begin{pmatrix} b_2^T & B^T \\ \beta\mu & c_1 \end{pmatrix},$$

so that

$$b_2 A_{k-1}^{-1} = u_1 + \mu u, \;\; A_{k-1}^{-T} B^T = U + u^T c_1/\beta, \;\; u_2 = c_1/\beta.$$

It follows that

$$\hat{A}_k^{-1} = \begin{pmatrix} A_{k-1}^{-1} & 0 \\ -b_2 A_{k-1}^{-1}/\hat{\beta} & 1/\hat{\beta} \end{pmatrix} = \begin{pmatrix} A_{k-1}^{-1} & 0 \\ -(u_1+\mu u)/\hat{\beta} & 1/\hat{\beta} \end{pmatrix}, \text{ and}$$

$$\hat{A}_k^{-T} \hat{B}_k^T = \begin{pmatrix} (1-\beta\mu^2/(\hat{\beta}\rho))u^T - (\beta\mu/(\hat{\beta}\rho))u_1^T & A_{k-1}^{-T}B^T - (u_1^T + \mu u^T)\hat{c}_1/\hat{\beta} \\ \beta\mu/(\hat{\beta}\rho) & \hat{c}_1/\hat{\beta} \end{pmatrix}.$$ Simplifying,

$$1 - \beta\mu^2/(\hat{\beta}\rho) = 1 - \mu^2/\rho^2 = \nu^2/\rho^2, \;\; \beta\mu/(\hat{\beta}\rho) = \mu/\rho^2.$$

We also have

$$\begin{aligned} A_{k-1}^{-T}B^T - (u_1^T + \mu u^T)\hat{c}_1/\hat{\beta} &= U + u^T c_1/\beta - \mu u^T \hat{c}_1/\hat{\beta} - u_1^T \hat{c}_1/\hat{\beta} \\ &= U + u^T(\rho c_1 - \mu\hat{c}_1)/\hat{\beta} - u_1^T\hat{c}_1/\hat{\beta} \\ &= U + \nu u^T \hat{c}_2/\hat{\beta} - u_1^T \hat{c}_1/\hat{\beta}. \end{aligned}$$

Substituting these relations into the matrix, we get

$$\hat{A}_k^{-1}\hat{B}_k = \begin{pmatrix} (\nu^2 u^T - \mu u_1^T)/\rho^2 & U + \nu u^T\hat{c}_2/\hat{\beta} - u_1^T\hat{c}_1/\hat{\beta} \\ \mu/\rho^2 & \hat{c}_1/\hat{\beta} \end{pmatrix}.$$

Then $\omega_k(\hat{A}_k) = 1/\hat{\beta}$, and

$$\omega_i(\hat{A}_k)^2 = \omega_i(A_k)^2 + (u_1+\mu u)_i^2/\hat{\beta}^2 - u_i^2/\beta^2, \;\; \text{for} \;\; 1 \le i \le k-1.$$

The cost of the $(k+1)$st step of Cholesky decomposition is about $2(n-k)^2$ flops, the cost of computing u is about k^2 flops, and the cost of computing $A_k^{-T}B_k^T$ is about $4k(n-k)$ flops, hence the "grand total" cost of modifications is about $4n^2 + 2nk - 2k^2$ flops.

Reasoning very similar to the analysis performed in [11] section 4.4 shows that the total number of interchanges (within the inner loop) up to the k-th step of Algorithm 1 is bounded by $k \log_f \sqrt{n}$, which guarantees that the Algorithm 3 will halt.

6. Computing strong RRCh decomposition using max norm and 2-norm estimators: Algorithm 5. In this section we discuss how convex optimization approach can be used as an alternative to carrying on modifications and updates in sections 5.2 and 5.1 and present Algorithm 5.

The convex optimization method was first developed by William Hager in [12]. It is based on the idea of finding a maximum of an appropriate function over a convex set (the maximum will be attained at a vertex of that set) by jumping from vertex to vertex according to a certain rule. The vertices can be visited only once, and since we have finitely many vertices, the algorithm will halt in a finite number of iterations.

ETNA
Kent State University
etna@mcs.kent.edu

We wish to develop estimators for

$$\|A_k^{-T}B_k^T\|_{\max} = \max_{i,j} |A_k^{-T}B_k^T|_{i,j} \text{ and } \omega_{\max}(A_k) = \max_j(\omega_j(A_k))$$

using the method described below, instead of carrying on modifications and updates described in section 5.

Assume H is an invertible square matrix, and M is any $n \times m$ matrix, (here H will be matrix A_k and M will be $A_k^{-T}B_k^T$). Denote

$$\|H\|_{max\ row} := \omega_{\max}(H) = \max_i(\omega_i(H))$$

and

$$\|M\|_{\max} := \max_{i,j} |(M_{ij})| \,.$$

We are going to use the following lemma which is discussed in [14]:

LEMMA 6.1.

$$\|M\|_{\max} = \max_{x\neq 0} \frac{\|Mx\|_\infty}{\|x\|_1} \quad \textit{and} \quad \|H\|_{max\ row} = \max_{x\neq 0} \frac{\|Hx\|_2}{\|x\|_1}.$$

Using this lemma and the algorithm discussed in [12] we obtain an estimator for $\|A_k^{-T}B_k^T\|_{\max}$:

ALGORITHM 4.

Pick x at random, such that $\|x\|_1 = 1$
loop
  Compute $A_k^T y = B_k^T x$, $z^T = \xi^T A_k^{-T} B_k^T$,
  where $\xi = sign((y)_\imath)e_\imath$, $\imath = \text{argmax}_j |y_j|$ and $e_\imath$ is $\imath$-th unit vector
  if $\|z\|_\infty < z^T x$
    stop
  else
    $\imath = \text{argmax}_j |z|$;
    $x := e_\imath$;
  end
endloop

Estimator for computing $\|A_k^{-1}\|_{max\ row}$ is the same as the previous one with the exception of different computation of z, which becomes:

$$A_k y = x,\ A_k^T z = \xi, \text{ where } \xi = 2A_k^{-T}x.$$

We wish to construct an algorithm similar to Algorithm 3, but using the above estimations instead of carrying on modifications and updates described in section 5. Algorithm 5, presented in this section, performs regular Cholesky decomposition with diagonal pivoting until $\max(\text{diag}(C_k)) \le \delta$ for some small δ. Every, for instance, $n/10$th step we use convex optimization approach to estimate the largest entry of $\omega(A_k)$ and max norm of $A_k^{-T}B_k^T$ and find i and j that correspond to these values. While

$$\max_{i,j}(|A_k^{-T}B_k^T|_{i,j}) > f, \text{ or } \sqrt{\max_j((C_k)_{jj})}\ \max_i(\omega_i(A_k)) > f$$

we do the interchanges. When there is no need for swaps we estimate $\omega_{\max}(A_k)$ to see if $1/\omega_{\max}(A_k) = 1/\omega_i(A_k)$ is smaller then δ. A simple calculation shows that if we permute i-th and k-th rows of A_k and re-triangularize it back, obtaining new matrix $\hat{A}_k = \Pi A_k Q$, then

ETNA
Kent State University
etna@mcs.kent.edu

$\hat{A}_k(k,k) = 1/\omega_k(\hat{A}_k) = 1/\omega_i(A_k)$. Hence if $\hat{A}_k(k,k) = 1/\omega_i(A_k) \le \delta$ then we should go one step back to $k-1$ and consider $(k-1)$-st stage of the Cholesky decomposition, since we do not want to have elements less than δ on the diagonal of A_k.

The main advantage of convex optimization approach over modifications and updates is efficiency. The overall cost of estimations combined with the potential swaps is usually $O(n^2)$, while modifications combined with swaps cost $O(n^3)$.

ALGORITHM 5.

$k := 0;\ \ \mathcal{C}_k(A) = (L, D);\ \ \Pi := I;$
while $\max(\mathrm{diag}(C_k)) \ge \delta$ ***do***
  $j = \mathrm{argmax}_{1 \le j \le n-k}(\max(\mathrm{diag}(C_k)));$
  $k := k + 1$;
  Compute $\mathcal{C}_k(\Pi_{k,k+j-1} A \Pi^T_{k,k+j-1});\ \Pi := \Pi\, \Pi_{k,k+j-1};$
do
  while $\hat{\rho}(L, D, k) \ge f$ ***do***
    Find i *and* j *such that* $\left|(A_k^{-T} B_k^T)_{ij}\right| \ge f$ *or* $\sqrt{(C_k)_{jj}}\, \omega_i(A_k) \ge f$
    Compute $\mathcal{C}_k(\Pi_{i,k+j} A \Pi^T_{i,k+j}),\ \Pi := \Pi\, \Pi_{i,k+j};$
  endwhile
  if $1/\omega_i(A_k) \ge \delta$ ***break endif***
  $k := k - 1;$
enddo
endwhile

In the following table let N_1 and N_2 be the average number of iterations it took to estimate $\max_i(\omega_i(A_k))$ and $\|A_k^{-T} B_k^T\|_{\max}$.The average of the ratios of the actual values over the estimated ones is denoted by Q_1 and Q_2.

Matrix	Order	Estimation of $\|A_k^{-T} B_k^T\|_{\max}$		Estimation of $\max_i(\omega)$	
		N_1	Q_1	N_2	Q_2
Kahan	96	2	3.4635	2	1.3955
	192	2	3.4521	2	1.0788
	384	2	3.4462	2	1.1053
Extended Kahan*	96	2	1.0000	2	1.7684
	192	2	1.0000	2	2.2863
	384	2	1.0000	2	1.0000
Generalized Kahan	96	2	2.3714	2	1.1778
	192	2	1.0000	2	1.9075
	384	2	1.0000	2	1.9992

On average, it takes about two “jumps” from vertex to vertex of the appropriate convex S to obtain the desired result. Hence, in algorithm 5 we just solve two systems two times, which takes about $8k^2$ flops.

There may be only finitely many iterations in the “**do-enddo**” loop, so the algorithm will eventually halt.

7. Effects of pivoting on Cholesky factorization.

7.1. Complete Pivoting. In this section we give an example, discovered in [5, section 2], of a symmetric positive semi-definite matrix for which Cholesky decomposition with complete pivoting does not result in strong RRCh decomposition because condition (2) of the definition in section 3 is violated.

ETNA
Kent State University
etna@mcs.kent.edu

Consider the following matrix, discovered by Higham in [13]:

$$R(\theta) = \mathrm{diag}(1, s, \dots, s^{r-1}) \begin{pmatrix} 1 & -c & -c & \dots & -c & -c & \dots & -c \\ & 1 & -c & \dots & -c & -c & \dots & -c \\ & & 1 & & \vdots & \vdots & & \vdots \\ & & & \ddots & \vdots & \vdots & & \vdots \\ & & & & 1 & -c & \dots & -c \end{pmatrix} \in \mathbb{R}^{r \times n},$$

where $c := \cos(\theta),\ s := \sin(\theta)$ for some θ. Let us scale $R(\theta)$, i.e., put $U(\theta) = \mathrm{diag}(r, r-1, \dots, 1)R(\theta)$ and consider matrix $C(\theta) = U(\theta)^T U(\theta)$. When we perform Cholesky decomposition with complete pivoting on $C(\theta)$, we will observe that, because of the way this matrix is designed, no pivoting will be necessary. Suppose

$$C(\theta) = \begin{pmatrix} C_{11}(\theta) & C_{12}(\theta) \\ C_{21}(\theta) & C_{22}(\theta) \end{pmatrix} = \begin{pmatrix} A_r & \\ B_r & I_{n-r} \end{pmatrix} \begin{pmatrix} I_r & \\ & C_r \end{pmatrix} \begin{pmatrix} A_r^T & B_r^T \\ & I_{n-r} \end{pmatrix}.$$

Then

$$C_{11}(\theta)^{-1} C_{12}(\theta) = A_r^{-T} B_r^T.$$

It is proven in [5, Lemma 2.3] that

$$\|C_{11}(\theta)^{-1} C_{12}(\theta)\|_2 = \|A_r^{-T} B_r^T\|_2 \to \sqrt{\frac{1}{3}(n-r)(4^r - 1)} \ \text{ as } \ \theta \to 0. \tag{7.1}$$

Simple calculation shows that

$$\|A_r^{-T} B_r^T\|_{\max} = c(1+c)^{r-1}.$$

To make the limit proved above more practical, we have to bound θ away from zero to avoid $\mathrm{diag}(1, s, \dots, s^{r-1})$, and hence $C(\theta)$, being too singular. We want s^{r-1} to be greater than some fixed tolerance ϵ, which usually is on the order of the machine precision. Simple manipulation with Taylor expansions shows that quantity $\|A_r^{-T} B_r^T\|_{\max}$ grows like

$$O\left(e^{\sqrt{-2r\log(\epsilon)}}\right) \tag{7.2}$$

for large r instead of a factor of 2^r as in (7.1).

The practical growth rate is much slower than the theoretical, but it is still super-polynomial, implying that Cholesky decomposition with complete pivoting will not be strong RRCh decomposition because condition (2) in the definition is violated.

Matrix $A_r^{-T} B_r^T$ also plays a key role in backwards error analysis for Cholesky decomposition of semi-definite matrices, as discussed by Higham in [13]. As he shows quantity $\|W\|_2 = \|A_r^{-T} B_r^T\|_2$ contributes greatly to the accuracy of Cholesky decomposition. If we perform Cholesky factorization without any of the pivoting strategies described in Algorithms 3 and 4 we get the following bounds obtained by Higham for the error of the Cholesky decomposition:

$$\|A - \hat{L}_r \hat{L}_r^T\|_2 \le 2\big(2r(r+1) + \theta\big)\big(\|W\|_2 + 1\big)^2 u \|A\|_2 + O(u^2),$$

where u is the machine precision, θ is some small constant, $\hat{L}_r$ is the computed r-th Cholesky factor, and r is the computed rank of A. As discussed in [5, section 4], the above bound is about the best result that could have been expected and reflects the inherent sensitivity of $A - L_r L_r^T$ (with L_r being the precise r-th Cholesky factor) to small perturbations of A.

7.2. Generalization of $R(\theta)$. In this section we construct generalizations to the Kahan matrix. We construct a matrix, such that Cholesky decomposition with diagonal pivoting fails to find a strong RRCh decomposition for the same reason as in the previous section: condition (2) of the definition of strong RRCh decomposition is violated.

Consider the matrices

$$U_n = \left(u_1, \ldots, u_n\right)^T; \;\; V_n = \left(v_1, \ldots, v_n\right)^T,$$

where u_i and $v_i \in \mathbb{R}^{p\times 1}$ and $p \ll n$. We introduce an upper triangular matrix $M_n \in \mathbb{R}^{n\times n}$:

$$M_n(i,j) = \begin{cases} 1, \text{ if i = j} \\ 0, \text{ if i < j} \\ -u_i^T v_j, \text{ if i > j} \end{cases} \quad . \tag{7.3}$$

Let's call matrix $P_n = \operatorname{diag}(1, s, \ldots, s^{n-1})M_n$ a Generalized Kahan matrix, and consider matrix $A = P_n^T P_n$ with numerical rank k. If we choose column scaling appropriately then there will be no diagonal swaps during Cholesky decomposition with diagonal pivoting. Then for matrix A we have $A_{11}^{-1}A_{12} = -M_k^{-1}U_k V_{n-k}^T$, where M_k is the upper left $k\times k$ corner of M_n, U_k is the top k rows of U_n, and V_{n-k} is the lower $(n-k)$ rows of V_n. Lemma 7.1 proved below gives a clue as to how to design matrices M_n by choosing u_i and v_i appropriately and to have $A_{11}^{-1}A_{12}$ grow faster than any polynomial.

Let's compute $X_k^T = \left(x_1, \ldots, x_k\right)^T = M_k^{-1}U_k$ explicitly.

LEMMA 7.1.

$$x_j^T = u_j^T \prod_{i=j+1}^{k} \{1 + v_i u_i^T\}$$

Proof by induction:
Base case: if $j = k$, then $x_k^T = u_k^T$ – true.
Inductive step: $j \to j-1$
We have that $M_k X^T = U_k$, hence

$$x_{j-1}^T - u_{j-1}^T v_j x_j^T - u_{j-1}^T v_{j+1} x_{j+1}^T - u_{j-1}^T v_{j+2} x_{j+2}^T - \ldots - u_{j-1}^T v_k x_k^T = u_{j-1}^T.$$

Combining terms and using the inductive assumption, we get

$$x_{j-1}^T = u_{j-1}^T \left[\sum_{i=j}^{k} v_i x_i^T + 1\right] = u_{j-1}^T \left[\sum_{i=j}^{k} v_i u_i^T \prod_{p=i+1}^{k}\{1 + v_p u_p^T\} + 1\right].$$

After simplifying notation with $t_m := v_m u_m^T$, we obtain:

$$\begin{aligned}
x_{j-1}^T &= u_{j-1}^T \left[\sum_{i=j}^{k} t_i \prod_{p=i+1}^{k}\{1 + t_p\} + 1\right] \\
&= u_{j-1}^T \left[\sum_{i=j}^{k} t_i \left(1 + \sum_{p=i+1}^{k} t_p + \sum_{k>p_1>p_2>i+1} t_{p_1}t_{p_2} + \ldots\right) + 1\right] \\
&= u_{j-1}^T \left(\sum_{p=j}^{k} t_p + \sum_{k>p_1>p_2>j} t_{p_1}t_{p_2} + \sum_{k>p_1>p_2>p_3>j} t_{p_1}t_{p_2}t_{p_3}\ldots + 1\right) \\
&= u_{j-1}^T \prod_{i=j}^{k}\{1 + t_i\} = u_{j-1}^T \prod_{i=j}^{k}\{1 + v_i u_i^T\}. \;\square
\end{aligned}$$

ETNA
Kent State University
etna@mcs.kent.edu

If we put $p = 1$, $u_i = 1$ and $v_j = c$ for all i and j, then we obtain exactly the Kahan matrix. In general, matrix P_n can have more than one small singular value, whereas the Kahan matrix has only one.

8. Numerical experiments implemented in Matlab. We used the following sets of test matrices M:

1. **Random**: is an $n \times n$ matrix with random entries, chosen from a uniform distribution on the interval $(0, 1)$.
2. **Scaled Random**: a random matrix, whose ith row is scaled by factor η^i, where $\eta = 20\epsilon$ and the machine precision is $\epsilon = 1.1 \times 10^{-16}$.
3. **GKS**: an upper triangular matrix whose jth diagonal element is $1/\sqrt{j}$ and whose (i, j) element is $-1/\sqrt{j}$, for $j > i$.
4. **Kahan**: we choose $c = 0.285$
5. **Extended Kahan**: the matrix $M = S_{3l}R_{3l}$, where

$$S_{3l} = \mathrm{diag}(1, \xi, ..., \xi^{3l-1})$$

and

$$R_{3l} = \begin{pmatrix} I_l & -\phi H_l & 0 \\ & I_l & \phi H_l \\ & & \mu I_l \end{pmatrix}$$

where we choose l is a power of 2; $\xi > 0$, $\phi > 1/\sqrt[4]{4l-1}$, $\phi = 0.285$, $\mu = 20\epsilon/\sqrt{n}$ and $\xi^2 + \phi^2 = 1$; $0 < \mu \ll 1$ and $H_l \in \mathbb{R}^{l \times l}$ is a symmetric Hadamard matrix.
6. **Extended Kahan***: we choose $f = \phi^2 l$ and $\delta = 4l^2\sigma_{2l+1}(A)$
7. **Generalized Kahan**: described in section 7.2, U_n and V_n consist of $n/48$ blocks; put $k = n/48$, $c = 0.285$ and $f = \sqrt{n}$.

For each matrix $A = M^T M$ we chose $n = 96, 192, 384$, set $f = 10\sqrt{n}$ and $\delta = 3 \times 10^{-13} \times \|A\|_2$. In Algorithm 5 we set $p = 4$ for $n = 96$, $p = 10$ for $n = 192$, $p = 20$ for $n = 384$. The results are summarized in the table below. Theoretical upper bounds for

$$\max_{i,j}\left(\frac{\sigma_i(L)}{\sigma_i(A_k)}, \frac{\sqrt{\sigma_j(C_k)}}{\sigma_{k+j}(L)}\right) \quad \text{and} \quad \max_{i,j}|(A_k^{-1}B_k)|_{i,j}$$

are $q_1(k, n) = \sqrt{1 + f^2 k(n-k)}$ and $q_2(k, n) = \begin{cases} f & \text{if k < n} \\ 0 & \text{if k = n} \end{cases}$. We observe from our experiments that theoretical bounds are much larger than these obtained in practice.

Denote $Q_1 := \max_{i,j}\left(\frac{\sigma_i(L)}{\sigma_i(A_k)}, \frac{\sqrt{\sigma_j(C_k)}}{\sigma_{k+j}(L)}\right)$ and $Q_2 := \max_{i,j}|(A_k^{-1}B_k)|_{i,j}$, $N :=$ number of iterations in the inner **while** loops; Est denotes results of Algorithm 3, and Mod denotes results of Algorithm 5.

Matrix	Order	Rank	N		Q_1			Q_2		
			Est	Mod	Est	q_1	Mod	Est	q_2	Mod
Random	96	96	0	0	1	1	1	0	0	0
	192	192	0	0	1	1	1	0	0	0
	384	384	0	0	1	1	1	0	0	0
Scaled random	96	43	0	0	3.10	4.7×10^3	3.10	1.54	98	1.54
	192	82	0	0	3.63	1.3×10^4	3.63	1.18	139	1.18
	384	158	0	0	6.24	3.7×10^4	6.24	1.29	196	1.29
GKS	96	95	0	0	1.12	9.5×10^2	1.12	0.71	98	0.71
	192	191	0	0	1.09	1.9×10^3	1.09	0.71	139	0.71
	384	383	0	0	1.07	3.8×10^3	1.07	0.71	196	0.71
Kahan	96	95	1	1	2.54	9.5×10^2	2.54	0.98	98	0.98
	192	191	1	1	1.26	1.9×10^3	1.26	0.98	139	0.98
	384	298	1	1	8.15	3.1×10^4	8.15	0.98	196	0.98
Extended Kahan	96	64	0	0	5.27	4.4×10^3	5.27	2.60	98	2.60
	192	128	0	0	10.0	1.3×10^4	10.0	5.20	139	5.20
	384	256	0	0	16.9	3.5×10^4	16.9	10.4	196	10.4
Extended Kahan*	96	64	8	32	2.97	1.2×10^2	1.49	2.60	2.60	0.38
	192	128	11	64	6.04	4.7×10^2	1.09	5.20	5.20	0.19
	384	256	3	128	12.1	1.9×10^3	1.5	10.4	10.4	0.96
Generalized Kahan	96	94	1	1	4.04	1.3×10^2	4.04	2.35	9.8	2.35
	192	134	2	2	21.7	1.2×10^3	19.9	6.21	13.9	6.59
	384	131	2	2	13.4	3.6×10^3	14.5	4.21	19.6	4.12

9. Conclusion. We have introduced a definition of strong rank revealing Cholesky decomposition, similar to the notion of strong rank revealing QR factorization. We proved the existence of such decomposition for any symmetric positive definite $n \times n$ matrix and presented two efficient algorithms for computing it. Numerical experiments show that if k is the numerical rank of A, then bounds which govern the gap between k-th and $(k+1)$-st singular values of matrix A and the norm of the approximate null space of A obtained in practice using our algorithms are several orders of magnitude smaller than theoretical ones. Algorithms presented in this paper produce strong rank revealing Cholesky factorization at lesser cost than analogous algorithms which use strong rank revealing QR factorization.

REFERENCES

[1] A. Björck, *Numerical methods for least squares problems*, SIAM, Philadelphia, PA, USA, 1996.
[2] A. Björck, *A direct method for the solution of sparse linear least squares problems*, in Large scale matrix problems, A. Björck, R. J. Plemmons, H. Schneider, eds., North-Holland, 1981.
[3] P.A. Businger, G.H. Golub, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269-276.
[4] T.F. Chan, *On the existence and computation of LU factorizations with small pivots*, Math. Comp., 42 (1984), pp. 535-547.
[5] T.C. Chan, *An efficient modular algorithm for coupled nonlinear systems,* Research Report YALEU/DCS/RR-328, Sept. 1984.
[6] S. Chandrasekaran, I. Ipsen, *On rank revealing QR factorizations,* SIAM J. Matrix Anal. Appl., 15 (1994), pp. 592-622.
[7] J.W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.
[8] R. Fletcher, *Expected conditioning,* IMA J. Numer. Anal., 5 (1985), pp. 247-273.
[9] G.H. Golub, *Numerical methods for solving linear least squares problems*, Numer. Math., 7 (1965), pp. 206-216.
[10] G.H. Golub, C.F. Van Loan, *Matrix Computations*, second ed., Johns Hopkins University Press, Baltimore, MD, USA, 1989.
[11] M. Gu, S.C. Eisenstat, *An efficient algorithm for computing a strong rank revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848-869.
[12] W. W. Hager, *Condition estimates*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 311-317.

[13] N.J. Higham, *Analysis of the Cholesky decomposition of a semi-definite matrix*, in M.G. Cox and S.J. Hammarling, eds., Reliable Numerical Computation, Oxford University Press, 1990, pp. 161-185.
[14] N.J. Higham, *Accuracy and stability of numerical algorithms*, SIAM, January 1996.
[15] Y.P. Hong, C.T. Pan, *Rank revealing QR factorizations and the singular value decompositions*, Math. Comp., 58 (1992), pp. 213-232.
[16] R.A. Horn, C.R. Johnson, *Topics in Matrix Analysis,* Cambridge University Press, Cambridge, 1991.
[17] P. Hough, S. Vavasis, *Complete orthogonal decomposition for weighted least squares,* SIAM J. Matrix Anal. Appl., 18 (1997), pp. 369-392.
[18] T.-S. Hwang, W.-W. Lin, E.K. Yang, *Rank revealing LU factorizations*, Linear Algebra Appl., 175 (1992), pp. 115-141.
[19] T.-S. Hwang, W.-W. Lin, D. Pierce, *Improved bound for rank revealing LU factorizations*, Linear Algebra Appl., 261 (1997), pp. 173-186.
[20] W. Kahan, *Numerical linear algebra*, Canad. Math. Bull., 9 (1966), pp. 757-801.
[21] C.D. Meyer, D. Pierce, *Steps towards an iterative rank revealing method,* Boeing Information and Support Services, ISSTECH-95-013, November 30, 1995.
[22] C.-T. Pan, P.T. Tang, *Bounds on singular values revealed by QR factorization*, BIT, 39(4) (1999), pp. 740-756.
[23] C.-T. Pan, *On the existence and computation of rank revealing LU factorizations,* Linear Algebra Appl., 316 (2000), pp. 199-222.
[24] G. Peters, J.H. Wilkinson, *The least-squares problem and pseudo-inverses*, Comput. J., 13 (1970), pp. 309-316.
[25] D. Pierce, J.G. Lewis, *Sparse multi-frontal rank revealing QR factorization,* SIAM J. Matrix Anal. Appl., 18 (1997), pp. 159-180.
[26] G.W. Stewart, *Updating a rank revealing ULV decomposition,* SIAM J. Matrix Anal. Appl., 14 (1993), pp. 494-499.