SIAM J. SCI. COMPUT.
Vol. 21, No. 1, pp. 88–110

PIVOTED CAUCHY-LIKE PRECONDITIONERS FOR REGULARIZED SOLUTION OF ILL-POSED PROBLEMS*

MISHA E. KILMER† AND DIANNE P. O'LEARY‡

Abstract. Many ill-posed problems are solved using a discretization that results in a least squares problem or a linear system involving a Toeplitz matrix. The exact solution to such problems is often hopelessly contaminated by noise, since the discretized problem is quite ill conditioned, and noise components in the approximate null-space dominate the solution vector. Therefore we seek an approximate solution that does not have large components in these directions. We use a preconditioned conjugate gradient algorithm to compute such a regularized solution. A unitary change of coordinates transforms the Toeplitz matrix to a Cauchy-like matrix, and we choose our preconditioner to be a low rank Cauchy-like matrix determined in the course of Gu's fast modified complete pivoting algorithm. We show that if the kernel of the ill-posed problem is smooth, then this preconditioner has desirable properties: the largest singular values of the preconditioned matrix are clustered around one, the smallest singular values, corresponding to the lower subspace, remain small, and the upper and lower spaces are relatively unmixed. The preconditioned algorithm costs only $O(n \lg n)$ operations per iteration for a problem with n variables. The effectiveness of the preconditioner for filtering noise is demonstrated on three examples.

Key words. regularization, ill-posed problems, Toeplitz, Cauchy-like, preconditioner, conjugate gradient, least squares

AMS subject classifications. 65R20, 45L10, 94A12

PII. S1064827596308974

1. Introduction.

In fields such as seismography, tomography, and signal processing, the process describing the acquisition of data can often be described by an integral equation of the first kind,

$$\int_{\beta_{lo}}^{\beta_{up}} t(\alpha,\beta)\hat{f}(\beta)\,d\beta = \hat{g}(\alpha),$$

where t denotes the kernel, $\hat{f}$ the unknown input function, and $\hat{g}$ the output. Often, it is assumed that values of $\hat{g}$ are known at the points $\alpha_i, i = 1,\ldots,n$. Hence, when a numerical integration rule is used to discretize the integral equation, the equation becomes a system of n linear equations of the form

$$T\hat{f} = \hat{g}.$$

In applications, the kernel is often assumed to be spatially invariant; that is, $t(\alpha,\beta) = t(\alpha-\beta)$. Discretization using a variety of numerical integration rules (e.g., rectangle, trapezoidal, etc.) results in a matrix T having Toeplitz structure when the support

*Received by the editors September 6, 1996; accepted for publication (in revised form) November 17, 1997; published electronically August 16, 1999. This research was supported by National Science Foundation grant CCR 95-03126.

http://www.siam.org/journals/sisc/21-1/30897.html

†Applied Mathematics Program, University of Maryland, College Park, MD 20742. Present address: Department of Mathematics, Tufts University, Medford, MA 02155 (na.mkilmer@na-net.ornl.gov).

‡Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (oleary@cs.umd.edu).

abscissas β_j and the α_i are both uniformly spaced with mesh width h.[1] In the case of n equations and n unknowns we have $T_{ij} = t_{i-j}$ for $1 \le i, j \le n$, and T is therefore constant along diagonals. For simplicity in the operation counts, we shall assume that T is a square $n \times n$ Toeplitz matrix, where n is assumed to be a power of 2. Thus operation counts involving FFTs can be written in terms of $O(n \lg n)$.[2] The case where T is $N \times n, N > n$, can be treated analogously to the square case, as we discuss in section 6.

The discrete inverse problem is to recover $\hat{f}$, given $\hat{g}$ and T. However, the continuous problem is generally ill posed; i.e., small changes in $\hat{g}$ cause arbitrarily large changes in $\hat{f}$. This is reflected in the discrete problem by ill-conditioning in the matrix T. The recovery of $\hat{f}$ then becomes a delicate matter since the recorded data will likely have been contaminated by noise e. In this case, we have measured g rather than $\hat{g}$, where

$$T\hat{f} + e = \hat{g} + e = g. \tag{1}$$

Due to the ill-conditioning of T and the presence of noise, exact solution of the linear system will not lead to a reasonable approximation of $\hat{f}$. Rather, *regularization* is needed in order to compute an approximate solution f. Regularization can be thought of as exchanging the original, ill-posed problem for a more well-posed problem whose solution approximates the true solution. Many regularization methods, both direct and iterative, have been discussed in the literature; see, for example, [12, 17, 9, 5]. In this paper we will primarily be concerned with regularization via conjugate gradient iterations [7, 27, 34], where the regularization parameter is the number of iterations.

Toeplitz matrices have several properties convenient for iterative methods like conjugate gradients: multiplication of a Toeplitz matrix times a vector can be done in $O(n \lg n)$ operations, and circulant preconditioners can be quite efficient [30, 3]. There are some difficulties, though. The inverse of a Toeplitz matrix does not generally have Toeplitz structure, and the fast factorization algorithms for Toeplitz matrices can require as many as $O(n^3)$ flops if pivoting is used to improve stability; see [32, 11, 4], for example.

To overcome these difficulties, we make use of the fact that Toeplitz matrices are related to *Cauchy-like* matrices by fast unitary transformations [19, 8, 10]. Cauchy-like matrices, discussed in detail in section 2, permit fast matrix-vector multiplication. In contrast to Toeplitz matrices, the inverse of a Cauchy-like matrix is Cauchy-like, and complete pivoting can be incorporated in its LDU factorization at a total cost of $O(n^2)$. However, for the special Cauchy-like matrices of interest to us, matrix-vector products with their inverses can be computed in $O(n \lg n)$ operations (see section 4).

The focus of this paper is the development of a Cauchy-like preconditioner that can be used to accelerate convergence of the conjugate gradient (CG) iteration to a

[1] At each meshpoint, the integration weights can be different depending on the integration rule. We assume the weights have been absorbed into $\hat{f}$, so that T is Toeplitz in (1). When t is spatially invariant, discretizing via Galerkin's method will sometimes also yield a matrix with Toeplitz structure (see section 5.2).

[2] When n is a product of small primes, the Winograd FFT algorithms can be used for slightly higher cost. If n is neither a product of small primes nor a power of 2, we suggest that one extend the problem to the desirable dimension, say n^*, by augmenting T with an $n^* - n$ identity matrix and extending the vectors $\hat{f}$ and $\hat{g}$ by some vector with length $n^* - n$. (In this case, T is actually block Toeplitz with two Toeplitz blocks on the diagonal; the only difference in the remainder of the paper is that the variable ℓ defined in section 2 will now have value at most 4, as opposed to 2 for Toeplitz matrices.)

filtered approximate solution of a problem involving a Toeplitz matrix. The regularizing properties of conjugate gradients and our choice of preconditioner are discussed in section 3. Each iteration of our algorithm takes $O(n \lg n)$ operations, and computational issues are discussed in section 4. Section 5 contains numerical results and section 6 presents conclusions and future work.

2. Transformation from Toeplitz to Cauchy-like structure. A Cauchy-like, or generalized Cauchy, matrix C has the form

$$C = \left(\frac{a_i^T b_j}{\omega_i - \theta_j} \right)_{1 \le i,\, j \le n} \quad (a_i, b_j \in \mathcal{C}^{\ell \times 1}; \omega_i, \theta_j \in \mathcal{C}). \tag{2}$$

It can also be defined as the unique solution of the *displacement equation*

$$\Omega C - C\Theta = AB^T, \tag{3}$$

where

$$\Omega = \operatorname{diag}(\omega_1, \ldots, \omega_n), \quad \Theta = \operatorname{diag}(\theta_1, \ldots, \theta_n), \quad A = \begin{pmatrix} a_1^T \\ \vdots \\ a_n^T \end{pmatrix}, \quad B = \begin{pmatrix} b_1^T \\ \vdots \\ b_n^T \end{pmatrix}.$$

The pair (A, B) is the *generator* of C with respect to Ω and Θ, and $\ell \le n$ is called the *displacement rank*. For the matrices and displacement equations of interest here, $\ell = 1$ or 2 [8].

We exploit three important properties of Cauchy-like matrices.

PROPERTY 1. *Row and column permutations of Cauchy-like matrices are Cauchy-like, as are leading principal submatrices.*

This property allows pivoting in fast algorithms for factoring Cauchy-like matrices [19, 8].

PROPERTY 2. *The inverse of a Cauchy-like matrix is Cauchy-like:*

$$C^{-1} = -\left(\frac{x_i^T w_j}{\theta_i - \omega_j} \right)_{1 \le i,\, j \le n} \quad (x_i, w_j \in \mathcal{C}^{\ell \times 1}). \tag{4}$$

Heinig [19] gives an $O(n \lg^2 n)$ algorithm to compute X (with rows x_i^T) and W (with rows w_i^T) given A, B, Θ, and Ω, and explains how, using the FFT, a system involving a Cauchy-like matrix can be solved in $O(n \lg^2 n)$. However, the algorithm is very fragile. It can be unstable for large values of n and, even when used on a well-conditioned matrix, may require pivoting to maintain stability [20, 1]. Alternatively, X and W can be determined from the relations

$$CX = A, \quad W^T C = B^T. \tag{5}$$

The third important property is that Toeplitz matrices also satisfy certain displacement equations [24, 8] which allow them to be transformed via fast Fourier transforms into Cauchy-like matrices [19, 8].

PROPERTY 3. *Every Toeplitz matrix T satisfies an equation of the form*

$$R_1 T - T R_{-1} = AB^T, \tag{6}$$

where $A \in \mathcal{C}^{n\times\ell}$, $B \in \mathcal{C}^{n\times\ell}$, *and*

$$R_\delta = \begin{pmatrix} 0 & 0 & \dots & 0 & \delta \\ 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}.$$

The Toeplitz matrix T *is unitarily related to a Cauchy-like matrix*

$$C = FTS_0^*F^*$$

that satisfies the displacement equation

$$S_1C - CS_{-1} = (FA)(B^TS_0^*F^*), \tag{7}$$

where

$$\begin{aligned} S_1 &= \operatorname{diag}(1, e^{\frac{2\pi i}{n}}, \dots, e^{\frac{2\pi i}{n}(n-1)}), \\ S_{-1} &= \operatorname{diag}(e^{\frac{\pi i}{n}}, \dots, e^{\frac{(2n-1)\pi i}{n}}), \\ S_0 &= \operatorname{diag}(1, e^{\frac{\pi i}{n}}, \dots, e^{\frac{\pi i}{n}(n-1)}), \end{aligned}$$

and F *is the normalized inverse discrete Fourier transform matrix defined by*

$$F = \frac{1}{\sqrt{n}}\left[\exp\left(\frac{2\pi i}{n}(j-1)(k-1)\right)\right]_{1\le j,\,k\le n}.$$

Gohberg, Kailath, and Olshevsky [8] suggest a stable $O(\ell n^2)$ partial pivoting algorithm to factor $C = PLU$. Sweet and Brent [31] show, however, that element growth in this algorithm depends not only on the magnitude of L and U, but on the generator for the Cauchy-like matrix. For our test matrices, partial pivoting alone did not provide the rank-revealing information that we need.

Gu [10] presents an algorithm that can perform a fast $O(\ell n^2)$ variation of LU decomposition with complete pivoting. Recall that in complete pivoting, at every elimination step one chooses the largest element in the current submatrix as the pivot in order to reduce element growth. Gu proposes instead that one find an entry sufficiently large in magnitude by considering the largest 2-norm column of $\bar{A}\bar{B}^T$ corresponding to the part that remains to be factored at each step. This algorithm computes the factorization $C = PLUQ$ [10, Alg. 2] using only the readily determined generators (see section 4), and Gu shows that it is efficient and numerically stable, provided that element growth in the computed factorization is not large. For our purposes it was convenient to set $D = \operatorname{diag}(u_{11}, \dots, u_{nn})$ and $U \leftarrow D^{-1}U$ to obtain the equivalent factorization $C = PLDUQ$.

3. Regularization and preconditioning. If we wanted to solve the linear system $Tf = g$ exactly, we would be finished: using the transformation to Cauchy-like form and the fast factorization algorithms described above, computing this solution would be an easy task. But the solution we seek is an approximate one, having noise filtering properties, so we choose to use an iterative method called CGLS

which, in conjunction with an appropriate preconditioner, produces suitably filtered solutions.

Three assumptions will guide our analysis:

1. The matrix T has been normalized so that its largest singular value is of order 1.
2. The uncontaminated data vector $\hat{g}$ satisfies the discrete Picard condition; i.e., the spectral coefficients of $\hat{g}$ decay in absolute value like the singular values [35, 16].
3. The additive noise is zero-mean white Gaussian. In this case, the components of the error e are independent random variables normally distributed with mean zero and variance ϵ^2.

We need to define the *upper* and *lower* subspaces. Using (1), let $T = \bar{U}\Sigma\bar{V}^T$ be the singular value decomposition of T, and expand the data and the noise in the basis created by the columns of $\bar{V}$:

$$\hat{g} = \sum_{i=1}^{n} \hat{\gamma}_i v_i, \quad e = \sum_{i=1}^{n} \eta_i v_i,$$

with $\hat{\gamma} = \bar{V}^T\hat{g}$ and $\eta = \bar{V}^T e$. Under the white noise assumption, the coefficients η_i are roughly constant in size, while the discrete Picard condition tells us that the $\hat{\gamma}_i$ go to zero at least as fast as the singular values σ_i.

Assumptions 2 and 3 imply the existence of an integer $\bar{m} > 0$ such that for all $i > \bar{m}$, $\hat{\gamma}_i$ are of the same order as η_i and hence these $\hat{\gamma}_i$ are obscured by noise. In addition, assumption 2 ensures that there exists $0 < m \leq \bar{m}$ such that if $i > m$, $|\hat{\gamma}_i| \not\gg |\eta_i|$. We partition the columns of $\bar{V}$ in accordance with these indices as follows. The space spanned by the first m columns of $\bar{V}$ we call the *upper* subspace, while the space spanned by the last $n - \bar{m}$ columns of $\bar{V}$ we define as the *lower* subspace. Hence the upper subspace corresponds to the largest m singular values and the lower subspace corresponds to the smallest $n - \bar{m}$ singular values. Finally, we define the *transition* subspace as the space spanned by the remaining $\bar{m} - m$ columns of V. Since the transition subspace usually corresponds to midrange singular values, the components of the solution lying in this subspace are generally difficult to resolve unless there is a gap in the singular value spectrum.

3.1. Regularization by preconditioned CGs. The standard CG method [22] is an iterative method for solving systems of linear equations for which the matrix is symmetric positive definite. If the matrix is not symmetric positive definite, one can use a variant of standard CG which solves the normal equations in factored form. We refer to the resulting algorithm as CGLS [22]. If the discrete Picard condition holds, then CGLS acts as an iterative regularization method with the iteration index taking the role of the regularization parameter [7, 14, 17]. Convergence is governed by the spread and clustering of the singular values [33]. Therefore, preconditioning is often applied in an effort to cluster the singular values, thus speeding convergence.

In the context of an ill-conditioned matrix T, we require a preconditioner for CGLS which clusters the largest m singular values while leaving the small singular values, and with them the lower subspace, relatively unchanged. In this case, the first few iterations of CGLS will quickly capture the solution lying within the subspace spanned by the first m columns of V. A modest number of subsequent iterations will provide improvement over the transition subspace, without significant contamination from the lower subspace.

3.2. The preconditioner. Given the Toeplitz matrix T, let $\tilde{C} = FTS_0^*F^*$ be its corresponding Cauchy-like matrix. Solving $Tf = g$ is then equivalent to solving

$$\tilde{C}FS_0f = Fg.$$

Note that since F and S_0 are unitary matrices, then

$$\tilde{C} = (F\bar{U})\Sigma\bar{V}^T(S_0^*F^*),$$

where $T = \bar{U}\Sigma\bar{V}^T$ is the singular value decomposition of T. Thus T and $\tilde{C}$ have the same singular values, and there is no mixing of upper and lower subspaces.

A factorization of $\tilde{C}$ using a modified complete pivoting strategy may lead to an interchange of rows (specified by a permutation matrix P) and columns (specified by a permutation matrix Q). Setting $C = P^T\tilde{C}Q^T$, $y = QFS_0f$, and $z = P^TFg$, the problem $Tf = g$ is equivalent to

$$Cy = z. \tag{8}$$

We choose a preconditioner M for the left so that

$$M^{-1}Cy = M^{-1}z$$

and apply CGLS to the corresponding normal equations

$$(M^{-1}C)^*(M^{-1}C)y = (M^{-1}C)^*M^{-1}z. \tag{9}$$

Our choice of preconditioner M is derived from the leading $m \times m$ submatrix of Gu's modified complete pivoting LDU factorization of the matrix C as follows. Let $C = LDU$ and write this equation in block form, where the upper left blocks are $m \times m$:

$$\begin{bmatrix} C_1 & C_2 \\ C_3 & C_4 \end{bmatrix} = \begin{bmatrix} L_1 & 0 \\ L_2 & L_3 \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} U_1 & U_2 \\ 0 & U_3 \end{bmatrix}. \tag{10}$$

Here L_1 and L_3 are lower triangular, U_1 and U_3 are upper triangular, and D_1 and D_2 are diagonal. We choose as our preconditioner the matrix

$$M = \begin{bmatrix} L_1 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} U_1 & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} C_1 & 0 \\ 0 & I \end{bmatrix}.$$

3.3. Properties of the preconditioner. We begin with some theorems about the clustering of the singular values of $M^{-1}C$. It is useful to decompose the matrix $(M^{-1}C)^*(M^{-1}C)$ into the matrix sum

$$\begin{bmatrix} I & C_1^{-1}C_2 \\ (C_1^{-1}C_2)^* & (C_1^{-1}C_2)^*(C_1^{-1}C_2) \end{bmatrix} + \begin{bmatrix} C_3^*C_3 & C_3^*C_4 \\ C_4^*C_3 & C_4^*C_4 \end{bmatrix} \equiv E_1 + E_2 \tag{11}$$

using the block partitioning of the previous section.

Let ϵ_i be the sum of the absolute values of the entries in row i of $C_1^{-1}C_2$, let ϵ_{max} be the largest of these quantities, and let $\hat{s}$ be the largest such row sum for E_2. The case of interest to us is when these quantities are reasonably small.

We denote the kth largest singular value of a matrix Z by $\sigma_k(Z)$ and the kth largest eigenvalue by $\lambda_k(Z)$.

THEOREM 3.1. *The m largest singular values of $M^{-1}C$ lie in the interval*

$$[1, \sqrt{1 + \epsilon_{max} + \hat{s}}].$$

Proof. The upper bound can be obtained by applying Gershgorin's theorem [29, IV.2.1] to bound the eigenvalues of the matrix $(M^{-1}C)^*(M^{-1}C)$, and then taking square roots. The lower bound is somewhat more interesting.

The matrices E_1 and E_2 are Hermitian positive semidefinite, and from the representations

$$E_1 = \begin{bmatrix} I & 0 \\ (C_1^{-1}C_2)^* & 0 \end{bmatrix} \begin{bmatrix} I & (C_1^{-1}C_2) \\ 0 & 0 \end{bmatrix} \qquad \text{and} \qquad E_2 = \begin{bmatrix} C_3^* & 0 \\ C_4^* & 0 \end{bmatrix} \begin{bmatrix} C_3 & C_4 \\ 0 & 0 \end{bmatrix},$$

it is clear that they have rank at most m and $n - m$, respectively.

By Corollary IV.4.9 in [29], we know that

$$\lambda_i(E_1) \leq \lambda_i((M^{-1}C)^*(M^{-1}C)). \tag{12}$$

We need to show that $\lambda_k(E_1) \geq 1$. If Y_1 and Y_2 are two $n \times n$ matrices and the rank of Y_2 is n–m, then a theorem of Weyl [23, Thm. 3.3.16] implies $\sigma_n(Y_1 + Y_2) \leq \sigma_m(Y_1)$. Now set

$$Y_1 = \begin{bmatrix} I & C_1^{-1}C_2 \\ 0 & 0 \end{bmatrix}, \qquad Y_2 = \begin{bmatrix} 0 & -C_1^{-1}C_2 \\ 0 & I \end{bmatrix},$$

and notice that the eigenvalues of E_1 are the squares of the singular values of Y_1. But $Y_1 + Y_2$ is the $n \times n$ identity matrix, so by Weyl's result we obtain $\sigma_m(Y_1) \geq 1$. Thus, $\lambda_i(E_1) \geq 1$ for $i = 1, \ldots, m$, and our conclusion follows from (12). ∎

We now study the extent to which preconditioning by M mixes the upper and lower subspaces.

THEOREM 3.2. *Let k be the dimension of the lower subspace, and let*

$$C = [Q_1\, Q_2\, Q_3] \begin{bmatrix} \Sigma_1 & 0 & 0 \\ 0 & \Sigma_2 & 0 \\ 0 & 0 & \Sigma_3 \end{bmatrix} \begin{bmatrix} V_1^* \\ V_2^* \\ V_3^* \end{bmatrix},$$

$$M^{-1}C = \begin{bmatrix} \hat{Q}_1\, \hat{Q}_2\, \hat{Q}_3 \end{bmatrix} \begin{bmatrix} \hat{\Sigma}_1 & 0 & 0 \\ 0 & \hat{\Sigma}_2 & 0 \\ 0 & 0 & \hat{\Sigma}_3 \end{bmatrix} \begin{bmatrix} \hat{V}_1^* \\ \hat{V}_2^* \\ \hat{V}_3^* \end{bmatrix}$$

be singular value decompositions with $V_3, \hat{V}_3 \in \mathcal{C}^{n \times k}$ and $V_1, \hat{V}_1 \in \mathcal{C}^{n \times m}$. Then

$$\|V_1^* \hat{V}_3\|_2 \leq \frac{\hat{\sigma}_{n-k+1}}{\sigma_m} \max\{1, \|C_1\|_2\}. \tag{13}$$

Proof. Using the decompositions we have

$$\begin{aligned} V_1^* \hat{V}_3 &= (V_1^* C^{-1}) M (M^{-1} C \hat{V}_3) \\ &= \Sigma_1^{-1} Q_1^* M \hat{Q}_3 \hat{\Sigma}_3. \end{aligned}$$

Since Q_1 and $\hat{Q}_3$ have orthonormal columns, it follows that

$$\|V_1\hat{V}_3\|_2 \leq \frac{\hat{\sigma}_{n-k+1}}{\sigma_m}\|M\|_2 = \frac{\hat{\sigma}_{n-k+1}}{\sigma_m}\left(\max\{1, \|C_1\|_2\}\right). \qquad \square$$

Next we show that $\hat{\sigma}_j \approx \sigma_j$ for σ_j corresponding to the lower subspace, and thus $\hat{\sigma}_{n-k+1}$ is small. Thus, if C_1 is well conditioned, then we are guaranteed that the upper and lower subspaces remain unmixed.

THEOREM 3.3. *The $(m+i)$th singular value of each of the matrices C and $M^{-1}C$ lies in the interval $[0, \sigma_i(E_2)]$ for $i = 1, \ldots, n-m$.*

Proof. Two theorems due to Weyl for Hermitian matrices Z, Y_1, and Y_2 with $Z = Y_1 + Y_2$ say

$$\lambda_{k+j-1}(Z) \leq \lambda_k(Y_1) + \lambda_j(Y_2) \quad [29, \text{ p. } 210],$$

$$\lambda_n(Y_2) + \lambda_k(Y_1) \leq \lambda_k(Z) \quad [29, \text{ Cor. IV.4.9}].$$

Now from the decomposition in (11), we see $\lambda_n(E_2) = 0$ and $\lambda_{m+1}(E_1) = 0$, and thus

$$0 \leq \lambda_{m+i}((M^{-1}C)^*(M^{-1}C)) \leq \lambda_{m+1}(E_1) + \lambda_i(E_2) = \lambda_i(E_2)$$

for $i = 1, \ldots, (n-m)$.

Also,

$$C^*C = \begin{bmatrix} C_1^* & 0 \\ C_2^* & 0 \end{bmatrix}\begin{bmatrix} C_1 & C_2 \\ 0 & 0 \end{bmatrix} + E_2.$$

We therefore likewise obtain

$$0 \leq \lambda_{m+i}(C^*C) \leq \lambda_i(E_2).$$

The proof is completed by taking square roots. $\square$

These theorems show that the preconditioner will be effective if C_1 is well conditioned and if the row sums of $C_1^{-1}C_2$ and E_2 are small. We now discuss to what extent these conditions hold for integral equation discretizations.

PROPERTY 4. *Let $\tilde{C}$ be a Cauchy-like matrix corresponding to a real Toeplitz matrix T that results from the discretization of a smooth, spatially invariant kernel t, normalized so that the maximum element of T is one. Then for n sufficiently large, there exists $\epsilon \ll 1$ and $m \ll n$ such that all elements of $\tilde{C}$ are less than ϵ in magnitude except for those located in four corner blocks of total dimension $m \times m$.*

To understand why this is true, recall that if $\tilde{A}$ and $\tilde{B}$ are the generators of $\tilde{C}$, where $\tilde{G} = \tilde{A}\tilde{B}^T$, the magnitude of the (k, j)-entry of $\tilde{C}$ is

$$|\tilde{C}_{kj}| = \frac{|\tilde{a}_k^T\tilde{b}_j|}{|\omega_k - \theta_j|}.$$

Thus the largest entries in $\tilde{C}$ appear where the numerator is large or the denominator is small.

The denominator of $\tilde{C}_{kj}$ is $|\omega_k - \theta_j| = |1 - e^{\frac{\pi i}{n}(2(j-k)+1)}|$, which is bounded above by 2. Its smallest entries are attained for $|k-j| \approx 0$ or n, but there are very few small values. In fact, direct computation shows that for $n \geq 100$, at least 95% of the

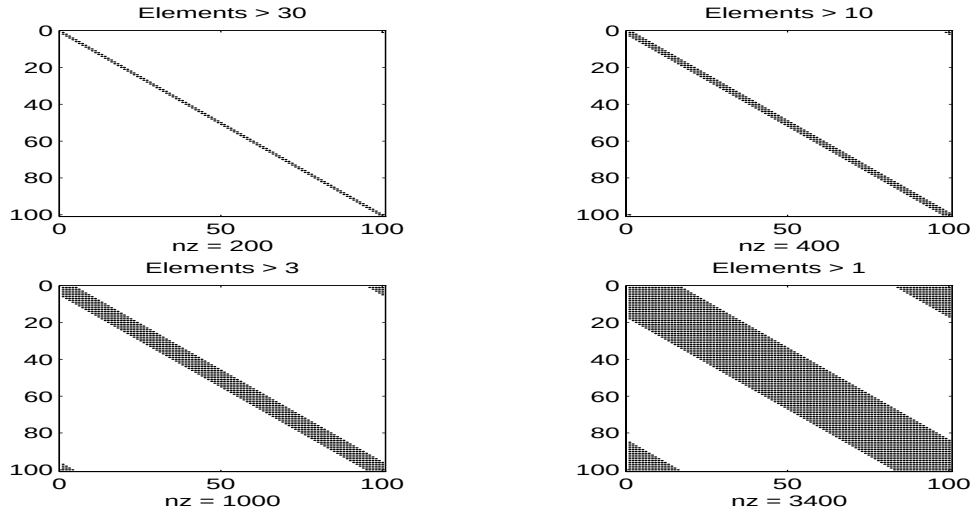

FIG. 1. *Plot revealing* $\left[\frac{1}{|\omega_k-\theta_j|}\right]_{k,\,j=1,\ldots,\,n} > tol$, $n = 100$ *for* $tol = 30, 10, 3,$ *and* 1.

entries in the first row have denominators in the range $[10^{-1}, 2]$, and the other rows have even more in this range. Figure 1 plots values of the matrix

$$\left[\frac{1}{|\omega_k - \theta_j|}\right]_{k,\,j=1,\ldots,\,n}$$

above given tolerance levels for $n = 100$. As expected, there are very few large values, and these occur only near the diagonal and the corners of the matrix.

Now consider the numerators. The formulas for A and B are determined from direct computation in (6). The first column of A is the first unit vector, and the second column is given by

$$[0, t_{n-1}, t_{n-2}, \ldots, t_{p-1}, \ldots, t_1]^T + [t_0, t_{-1}, t_{-2}, \ldots, t_{-(q-1)}, \ldots, t_{-(n-1)}]^T. \tag{14}$$

The first column of B is

$$[t_{-(n-1)}, t_{-(n-2)}, \ldots, t_{-(q-1)}, \ldots, t_{-1}, t_0]^T - [t_1, t_2, \ldots, t_{p-1}, \ldots, t_{n-1}, 0]^T, \tag{15}$$

and the second column is the last unit vector. The generators for $\tilde{C}$ are then $\tilde{A} \equiv FA$ and $\tilde{B} \equiv conj(FS_0)B$, where $conj(\cdot)$ denotes complex conjugation, with F and S_0 as described in Property 3. Therefore, the numerators are

$$|\tilde{a}_k^T \tilde{b}_j| = \left| \frac{1}{\sqrt{n}} conj(\zeta)_j + \frac{1}{\sqrt{n}} e^{\frac{(n-1)}{n}\pi i(1-2j)} \nu_k \right|,$$

where ν_k is the kth entry in the second column of $\tilde{A}$ and ζ_j is the jth entry in the first column of FS_0B. Thus it is the normalized inverse Fourier coefficients of the second column of A and first column of S_0B which determine the magnitude of the numerators, and if t is smooth, these will be large only for small indices j and k.

Therefore,

$$\frac{|\tilde{a}_k^T \tilde{b}_j|}{|\omega_k - \theta_j|} \leq \frac{1}{\sqrt{n}}(|\nu_k| + |\zeta_j|) \ll 1$$

away from the corners. Thus $\tilde{C}$ can be permuted to contain the large elements in the upper left block, and any pivoting strategy that produces such a permutation will give a suitable preconditioner for our scheme.

We have observed that if Gu's algorithm is applied to a matrix with this structure, then C_1 will contain the four corner blocks. The interested reader is referred to [10] for details on the complete pivoting strategy, but the key fact is that Gu makes his pivoting decisions based on the size of elements in the generator $\bar{A}\bar{B}^T$ corresponding to the block that remains to be factored. The resulting Cauchy-like preconditioner C_1 for the matrix C then has the property that the first m singular values of the preconditioned matrix are clustered, and that the invariant subspace corresponding to small singular values of C is not much perturbed. Thus we expect that the initial iterations of CGLS will produce a solution that is a good approximation to the noise-free solution.

4. Algorithmic issues. Our algorithm is as follows.

ALGORITHM 1 (Solving $Tf = g$).

1. Compute the generators for the matrix $\tilde{C} = FTS_0^*F^*$ using (14) and (15).
2. Determine an index m to define the size of the partial factorization of $\tilde{C}$ and factor $\tilde{C} = PLDUQ$.
3. Set $C = P^T\tilde{C}Q^T$ and $z = P^TFg$.
4. Determine the $m \times m$ leading principal submatrix, C_1, of C and let $M = \begin{bmatrix} C_1 & 0 \\ 0 & I \end{bmatrix}$. (See (10).)
5. Compute an approximate solution $\tilde{y}$ to $M^{-1}Cy = M^{-1}z$ using a few steps of CGLS.
6. The approximate solution in the original coordinate system is $f = S_0^*F^*Q^T\tilde{y}$.

When to stop the CGLS iteration in order to get the best approximate solution is a well-studied but open question (for instance, see [18] and the references therein). We do not solve this problem, but we consider the other algorithmic issues in the following subsections.

4.1. Determining the size of C_1. The choice of the parameter m determines the number of clustered singular values in the preconditioned system. It influences the amount of work per iteration and, perhaps more importantly, the mixing of upper and lower subspaces. We use a simple heuristic in our numerical experiments. We compute the Fourier transform of the data vector $\hat{g}$ and determine the index m for which the Fourier coefficients start to level off. This is presumed to be the noise level, and the factorization is truncated here.

4.2. Computing the preconditioner. Since $\tilde{C}$ satisfies the displacement equation (3), with $\Omega = S_1$ and $\Theta = S_{-1}$, it follows that C_1 satisfies

$$\Omega_1 C_1 - C_1 \Theta_1 = A_1 B_1^T,$$

where Ω_1 and Θ_1 are the leading principal submatrices of $P^T \Omega P$ and $Q \Theta Q^T$, respectively, and A_1 and B_1 contain the first m rows of $P^T \tilde{A}$ and $Q^T \tilde{B}$, respectively.

Thus the matrix C_1^{-1} has entries

$$C_1^{-1} = -\left(\frac{x_i^T w_j}{\tilde{\theta}_i - \tilde{\omega}_j} \right)_{1 \le i,\, j \le n}, \tag{16}$$

where $\tilde{\theta}_i$ and $\tilde{\omega}_j$ are the elements of Θ and Ω that appear in Θ_1 and Ω_1, respectively, and, from (5), the vectors x_i^T and w_j^T are rows of X_1 and W_1 defined as

$$C_1 X_1 = A_1, \quad W_1^T C_1 = B_1^T. \tag{17}$$

Computing X_1 and W_1 costs $O(m^2)$ operations, given the factorization of C_1 and the matrices A_1 and B_1. The factorization of C_1 is obtained during the modified complete pivoted partial factorization of $\tilde{C}$ in $O(mn - m^2/2 + m/2)$ operations. Since $O(n \lg n)$ operations are required to compute A_1 and B_1 by means of FFTs, the total cost to initialize our preconditioner is $O(mn + n \lg n)$ operations.

4.3. Applying the preconditioner. Let r be a vector of length m and assume that no pivoting was done when $\tilde{C}$ was factored. Heinig [19] states that $C_1^{-1} r$ may be written as

$$C_1^{-1} r = \sum_{j=1}^{\ell} -(X_1)_j \cdot (C_0 (W_1)_j \cdot r),$$

where $(X_1)_j$ is the jth column of X_1, $(W_1)_j$ is the jth column of W_1, and C_0 is the Cauchy matrix $C_0 = (\frac{1}{\theta_i - \omega_j})_{1 \le i,\, j \le m}$. The notation $\cdot$ denotes the componentwise product of two vectors.

Fast multiplication by the matrix C_0 requires finding the coefficients of a polynomial whose roots are the elements of Θ_1 and Ω_1 [6], and this process can be unstable. To avoid this difficulty, realizing that the elements of S_{-1} and S_1 are roots of unity, we extend C_0 to a matrix of size $n \times n$ satisfying the displacement equation (2) with $\Omega = S_{-1}$ and $\Theta = S_1$. Now multiplication of a vector by the extended matrix C_0 can be reduced to multiplication by diagonal and antidiagonal matrices and FFTs. This observation allows us to develop a mathematically equivalent algorithm for computing $s = C_1^{-1} r$ which costs only $O(n \lg n)$ operations.

ALGORITHM 2 (Forming $s = C_1^{-1} r$).
Set $s = 0$.
For $j = 1, \ldots, \ell$, do
1. Compute $\hat{r} = W_j \cdot r$.
2. Extend $\hat{r}$ by zeros so that $\hat{r}$ is of length n.
3. Set $\hat{r} \leftarrow C_0 \hat{r}$ (see below).
4. Truncate $\hat{r}$ to length m.
5. Set $s = s + X_j \cdot \hat{r}$.

The product $C_1^{-*}r$ can be computed similarly.

If pivoting was done during factorization, the vector $\hat{r}$ should be multiplied by P after step 2 and by Q before step 4.

This formulation allows $C_1^{-1}r$ to be computed in $O(n \lg n)$ operations in a stable manner, using an observation of Finck, Heinig, and Rost [6] that any Cauchy-like matrix can be factored as

$$C_0 = \mathrm{diag}(h(\theta_1), \ldots, h(\theta_n))^{-1} V(\theta) H V(\omega)^T, \tag{18}$$

where $V(\omega)$ and $V(\theta)$ are the Vandermonde matrices whose second columns contain the diagonal elements of Ω and Θ, respectively. The matrix H is a Hankel matrix, i.e., one in which elements on the antidiagonals are constant. The first row is equal to the coefficients of the polynomial $h(u) = \prod_{i=1}^{n}(u - \omega_i)$ except for the leading one. Since, from Property 3, Ω and Θ contain roots of unity, products of the matrix C_0 with a vector are very simple to compute:

- $h(u) = u^n - 1$, so H has a single nonzero diagonal extending from southwest to northeast.
- $V(\omega)^T$ is $\sqrt{n}F$, where F is the normalized, discrete, inverse Fourier transform matrix defined in Property 3.
- $V(\theta)$ is the matrix product $\sqrt{n}FS_0$, where the diagonal matrix S_0 is defined in Property 3.

Thus products $C_0\hat{r}$ can be computed stably in $O(n \lg n)$ operations. Since $\ell = 2$ at most, the preconditioner can be applied to a vector in $O(n \lg n)$ operations, given X_1 and W_1 that have been computed according to (17). This is the same order as the number of operations to apply C to a vector, since $C = PFTS_0^*F^*Q$ and the product of a Toeplitz matrix with a vector can be computed in $O(n \lg n)$ operations by embedding the matrix in a circulant matrix [2]. Thus, each iteration of CGLS costs $O(n \lg n)$ operations.

5. Numerical results. In this section we summarize results of our algorithm on three test problems using MATLAB and IEEE floating point double precision arithmetic. Our measure of success in filtering noise is the *relative error*, the 2-norm of the difference between the computed estimate f and the vector $\hat{f}$ corresponding to zero noise, divided by the 2-norm of $\hat{f}$. Since one does not usually know the exact solution, other measures, such as the norm of the residual, must be monitored in order to determine when to stop iterating. Choosing the optimal regularization parameter is a difficult task and the interested reader is referred to [15, 18, 17] for heuristics to determine the regularization parameter. In each experiment, we apply the CGLS iteration with Cauchy-like preconditioner of size m. The value $m = 0$ corresponds to no preconditioning.

We compare our method to the preconditioning scheme of Hanke, Nagy, and Plemmons [14]. In the one-dimensional case, their preconditioner is determined by forming the T. Chan circulant matrix approximant to T, computing the eigenvalues via one-dimensional FFTs, and then replacing all the eigenvalues below a certain tolerance with ones. When cutoff = 20, for example, the largest 20 eigenvalues remain unchanged and the last $n - 20$ are replaced by 1s. It requires $O(n \lg n)$ operations to initialize their preconditioner by means of FFTs.

Our method is similar to their method in that we also rely on a rank-revealing factorization to determine an appropriate cutoff which is used to form the preconditioner. However, our preconditioner does not require preliminary approximation of the Toeplitz matrix by a circulant matrix, and therefore we expect our method

to outperform theirs in cases when the Toeplitz matrix is not well approximated by its T. Chan circulant approximant; for example, in cases when the $|t_j|$ do not decay sufficiently quickly as j increases (in particular, see Example 2).

We note that the value m depends on the noise level $\|e\|_2/\|\hat{g}\|_2$ and the rate of decay of the singular values (related to the smoothness of the kernel). For smooth kernels t and large values of n, we therefore expect $m \ll n$. In this case, the cost of initializing our preconditioner, $O(mn + n \lg n)$ operations, will not be much more than initializing the preconditioner in [14] (see Example 2).

5.1. Signal processing example (Example 1). As mentioned in the introduction, Toeplitz matrices often arise in signal processing (one-dimensional image reconstruction problems). As an example, we consider the 256×256 Toeplitz matrix T whose entries are defined by

$$t_{i,\,j} = \begin{cases} \frac{4}{51}\phi(\alpha_i - \beta_j) & \text{if } |i-j| \le 15, \\ 0 & \text{otherwise,} \end{cases}$$

where

$$\alpha_i = \beta_i = \frac{4i}{51}, \qquad i = 1, 2, \ldots, 256,$$

and

$$\phi(\gamma) = \frac{1}{2\sqrt{\pi}\delta}\exp\left(-\frac{\gamma^2}{4\delta^2}\right), \qquad \delta = 0.3.$$

This matrix is the one used in Example 4 of [2]. The authors note that such matrices may occur in image restoration contexts as "prototype problems" and are used to model certain degradations in the recorded image.

The condition number of T is 6.16×10^7. We wish to solve the equation $Tf = g$, where g denotes the noisy data vector for which $\|e\|_2/\|\hat{g}\|_2$, the noise level, is about 10^{-3}. The uncorrupted data, $\hat{g}$, and exact numerical solution,[3] $\hat{f}$, are displayed in Figure 2. The Fourier coefficients of g are shown in Figure 3. Using these coefficients, an appropriate cutoff value m was determined as explained in section 4.1.

The solid line in Figure 4 shows the convergence of CGLS on the unpreconditioned Toeplitz system. The minimal value of the relative error, 2.18×10^{-1}, was achieved at 117 iterations. Convergence of CGLS on the preconditioned system involving the Cauchy-like matrix is also shown in Figure 4 for two different values of m. Table 5.1 gives an idea of the sensitivity of the algorithm to the choice of m. By increasing m from 41 to 51, for example, we slightly increase the minimum relative error in favor of the fewer number of iterations required to reach a regularized solution. From Table 5.1 we observe that the number of iterations for the preconditioned system is substantially less than for the unpreconditioned when m is chosen appropriately.

For a cutoff of 51, the method of [14] gives a solution with relative error of 2.19×10^{-1} in 25 iterations. In contrast, for $m = 51$, our method took only 15 iterations to reach almost the same relative error value. For any cutoff value it took the method of [14] more iterations to reach a reasonable regularized solution in general than it took our method. To further compare the two methods, we computed the average decrease in relative error per iteration by dividing the minimum relative error

[3] We first determined $\hat{f}$ using MATLAB's square function $\hat{f} = square(2\pi v * .3)$ with $v = [0{:}.1{:}25.5]$, then computed $\hat{g} = T\hat{f}$.

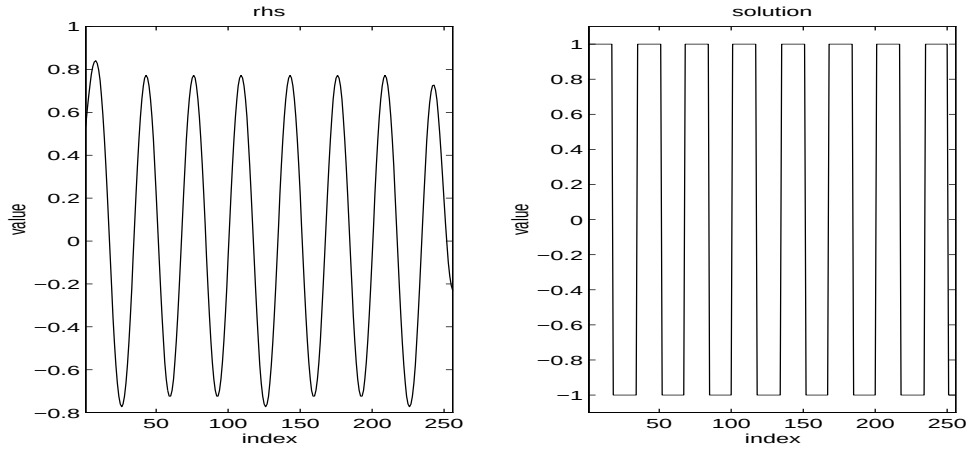

FIG. 2. *Uncontaminated data vector (left) and exact solution vector (right) for Example* 1.

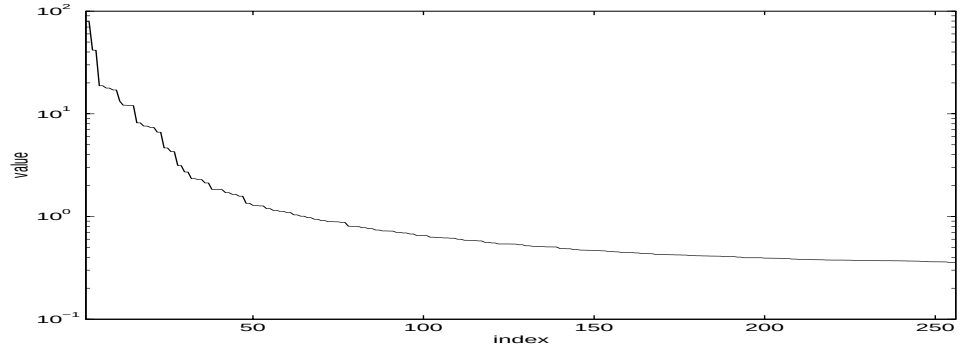

FIG. 3. *Fourier coefficients of the noisy data for Example* 1.

by the number of iterations required to achieve it. This tells us how much we gain, on average, for the price of one iteration for a given preconditioner. For m and the cutoff set to 51, these numbers are .0147 and .0088 for our method and the method of [14], respectively. For m and the cutoff set to 31, these numbers are .0044 and .0034. So in both cases, our method is slightly better.

The singular values of T and of the preconditioned matrix $M^{-1}C$ for $m = 51$ are shown in Figure 5. As predicted by the theory in section 3.3, the first 51 singular values of $M^{-1}C$ are clustered very tightly around one and the smallest singular values have been left virtually untouched. To test the robustness of the theorems, we computed the quantities appearing in the theorems. In Theorem 3.1, for example, $\epsilon_{max} = \|C_1^{-1}C_2\|_\infty = 2.91 \times 10^1$ and $\hat{s} = \|E_2\|_\infty = 3.79 \times 10^{-2}$, giving an upper bound of 5.49 on the largest singular value of the preconditioned matrix; in fact $\sigma_1(M^{-1}C) = 2.79$, but we found $\sigma_2(M^{-1}C), \ldots, \sigma_{51}(M^{-1}C) \in [1, 1.07]$. In Theorem 3.2, $\|C_1\|_2 = 9.94$ and the quantity $\frac{1}{\sigma_{51}}(\max\{1, \|C_1\|\})$ was equal to 1.57×10^2. (This quantity was approximately equal to the condition number of C_1, which was 2.74×10^2.) Since, as Figure 5 indicates, the preconditioned matrix has singular values more than four orders of magnitude smaller than 10^2, Theorem 3.2 does imply that the upper and lower subspaces remain relatively unmixed.

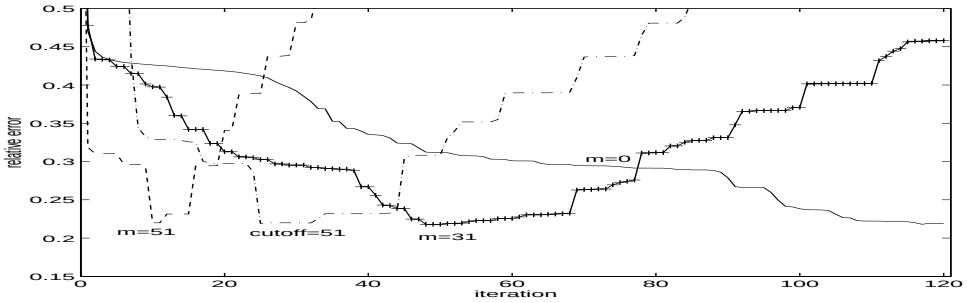

FIG. 4. *Relative error in computed solution for our preconditioner when* $m = 0$, $m = 31$, *and* $m = 51$ *and relative error in computed solution for the method in* [14] *with cutoff* $= 51$ *eigenvalues*; *Example* 1.

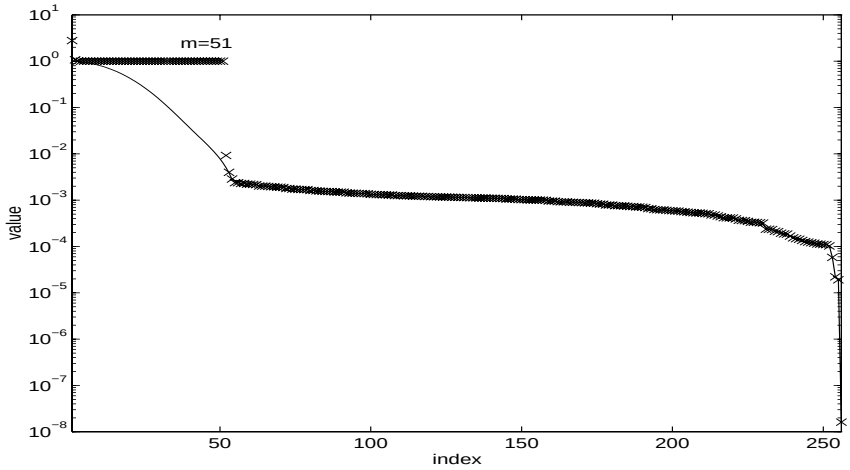

FIG. 5. *Singular values of* C *(solid line) and* $M^{-1}C$ *(*$\times$*'s) for Example* 1, $m = 51$.

TABLE 5.1
Minimum relative errors achieved for various values of m; *Example* 1.

	Our method		Method of [14]	
m (cutoff)	Minimum rel. error	Achieved at iter.	Minimum rel. error.	Achieved at iter.
0	2.18×10^{-1}	117		
31	2.18×10^{-1}	50	2.18×10^{-1}	65
41	2.19×10^{-1}	27	2.19×10^{-1}	48
51	2.20×10^{-1}	15	2.19×10^{-1}	25
61	2.32×10^{-1}	8	2.32×10^{-1}	30

5.2. Phillips test problem (Example 2). Next we consider the discretized version of the well-known first-kind Fredholm integral equation studied by Phillips [28]. The kernel of the integral equation is given by $t(\alpha, \beta) = \phi(\alpha - \beta)$ where ϕ is defined by

$$\phi(\gamma) = \begin{cases} 1 + \cos(\frac{\gamma\pi}{3}), & |\gamma| < 3, \\ 0 & |\gamma| \geq 3, \end{cases}$$

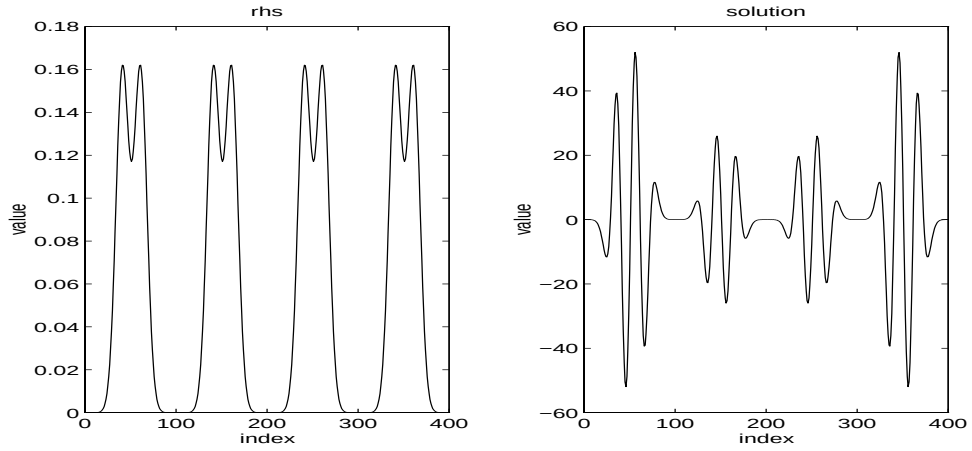

FIG. 6. *Uncontaminated data vector (left) and exact solution vector (right) for Example* 2.

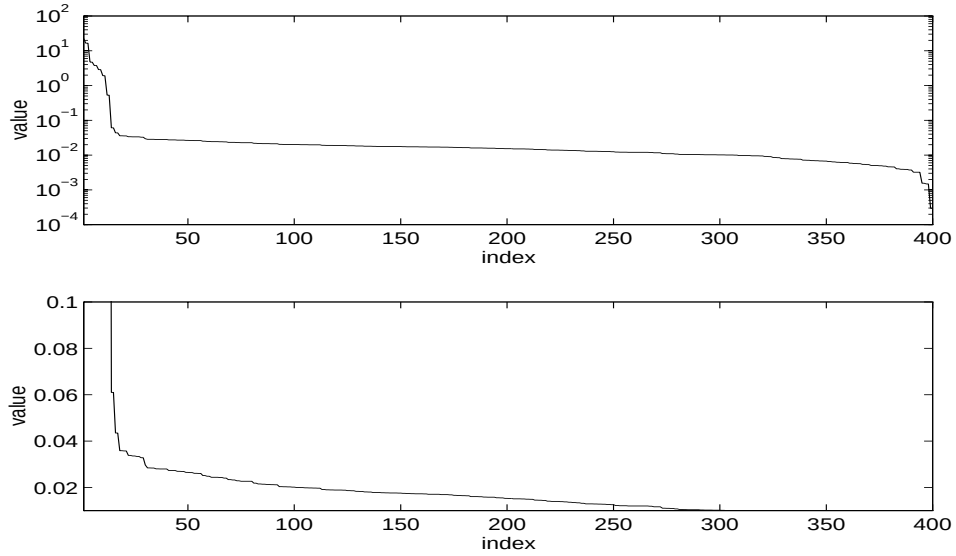

FIG. 7. *Fourier coefficients of the noisy data for Example* 2, *two different scales.*

and the limits of integration are -6 and 6. We used Hansen's MATLAB Regularization Toolbox, described in [17], to generate the corresponding 400×400 symmetric Toeplitz matrix whose condition number was approximately 6.8×10^8. T was then scaled by 6. In this code, the integral equation is discretized by the Galerkin method with orthonormal box functions. The uncorrupted data vector[4] is shown in Figure 6. The noise level was 1×10^{-2} for this problem.

It was difficult to determine the appropriate cutoff value m, as Figure 7 indicates, but Table 5.2 and Figure 8 show that the savings in the number of iterations to convergence can be substantial. In addition, for several values of m, the minimum

[4]We set $\hat{g} = [s, s, s, s]$ with $s = (\phi(v, -1))^2 + (\phi(v, 1))^2$, where $v = [-5{:}0.1{:}4.9]$ and $\phi(v, \lambda) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-(v-\lambda)^2}{2}\right)$. Then $\hat{f}$ was taken to be the exact numerical solution of the problem.

TABLE 5.2
Minimum relative errors achieved for various values of m, Example 2.

	Our method		Method of [14]	
m (cutoff)	Minimum rel. error	Achieved at iter.	Minimum rel. error.	Achieved at iter.
0	5.71×10^{-2}	301		
33	5.64×10^{-2}	66	5.83×10^{-2}	485
36	5.63×10^{-2}	54	5.72×10^{-2}	474
39	5.56×10^{-2}	53	5.85×10^{-2}	458
42	5.82×10^{-2}	46	5.83×10^{-2}	490
45	5.81×10^{-2}	32	6.22×10^{-2}	425
48	4.79×10^{-2}	21	5.48×10^{-2}	433
51	4.68×10^{-2}	23	2.70×10^{-2}	295
54	4.90×10^{-2}	17	3.01×10^{-2}	260
57	5.02×10^{-2}	10	3.80×10^{-2}	220
60	3.57×10^{-2}	10	3.93×10^{-2}	236
63	4.21×10^{-2}	3	3.67×10^{-2}	136
66	5.06×10^{-2}	11	5.04×10^{-2}	200

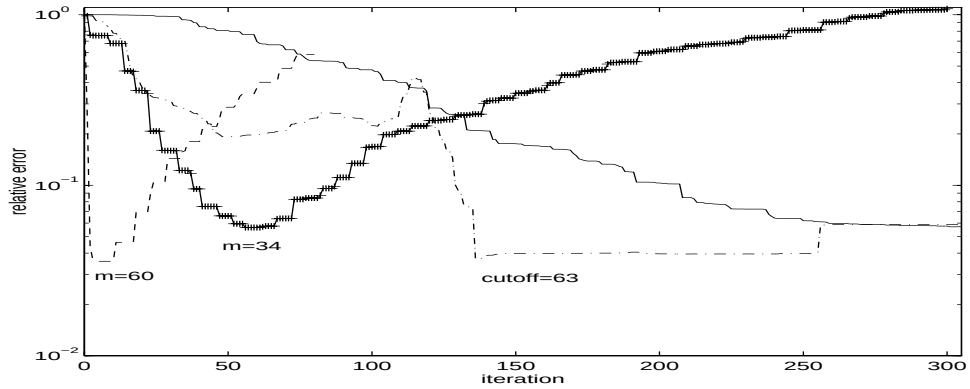

FIG. 8. *Relative error in computed solution for our preconditioner when $m = 0$, $m = 34$, and $m = 60$ and relative error in computed solution for the method in* [14] *with cutoff $= 63$ eigenvalues; Example 2.*

relative error is somewhat lower than the minimum obtained for the unpreconditioned problem. For example, after 302 iterations, CGLS on the unpreconditioned problem achieved a minimum relative error of 5.71×10^{-2}. For $m = 60$, however, a minimum relative error of 3.57×10^{-2} was reached in only 10 iterations. Again, we note that the method of [14] achieves similar, and sometimes lower, relative error values, but the number of iterations required to achieve these values, and thus the total cost, is very high in comparison.

Figure 9 illustrates that, as in Example 1, the first m singular values of the preconditioned matrix are clustered around one and the singular values corresponding to the lower subspace remain almost unchanged. In particular, when $m = 60$ the

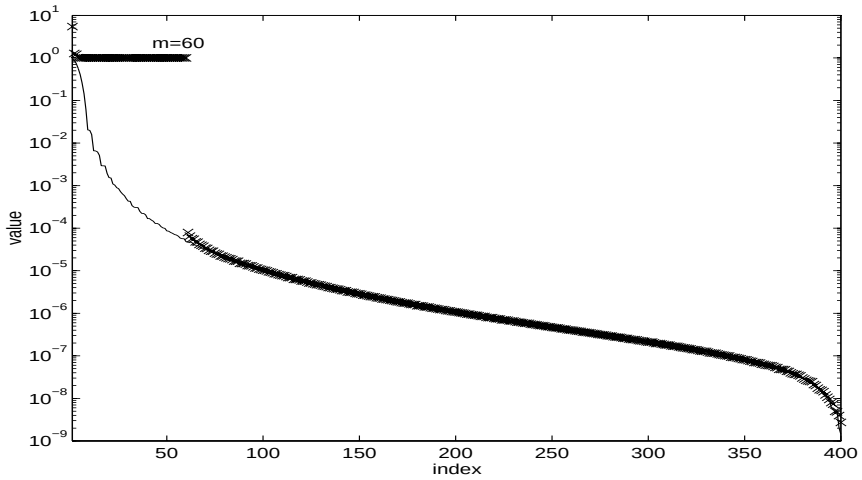

FIG. 9. *Singular values of C (solid line) and $M^{-1}C$ ($\times$'s) for Example* 2, $m = 60$.

quantities that play a role in the upper bound of Theorem 3.1 are $\epsilon_{\max} = \|C_1^{-1}C_2\|_\infty = 6.73 \times 10^1$ and $\hat{s} = \|E_2\|_\infty = 6.69 \times 10^{-3}$; in this case, the theorem predicts that the largest singular value of the preconditioned matrix will be less than 8.3. In fact, the largest singular value of $M^{-1}C$ was 5.5, with the second largest singular value 1.27. In Theorem 3.2, the quantity which premultiplies $\hat{\sigma}_{n-k+1}$ is 2.07×10^4, which is about the same order of magnitude as the condition number of C_1, 7.15×10^4. However, for $k = 175$, the upper bound in (13) is already on the order of 10^{-3}, and so we observe that the upper and lower subspaces remain relatively unmixed as the theorem predicts. So even when C_1 is moderately ill conditioned, very small singular values ensure that the upper and lower subspaces do not mix.

5.3. Nonsymmetric example (Example 3). Finally, since both previous examples involve symmetric Toeplitz matrices, for our third example we chose to work with a 100×100 matrix T. We first define the vectors

$$t_j^{(1)} = \begin{cases} \frac{4}{51}\phi(.15, \alpha_1 - \beta_j), & 1 \le j \le 10, \\ 0 & \text{otherwise}, \end{cases}$$

$$t_j^{(2)} = \begin{cases} \frac{4}{51}\phi(.18, \alpha_1 - \beta_j), & 1 \le j \le 11, \\ 0 & \text{otherwise}, \end{cases}$$

where α_i, β_j are as in Example 1 and

$$\phi(\delta, \gamma) = \frac{10}{3\sqrt{\pi}} \exp\left(\frac{-\gamma^2}{4\delta^2}\right),$$

and use the MATLAB command *toeplitz*$(t^{(1)}, t^{(2)})$ to generate the matrix T.

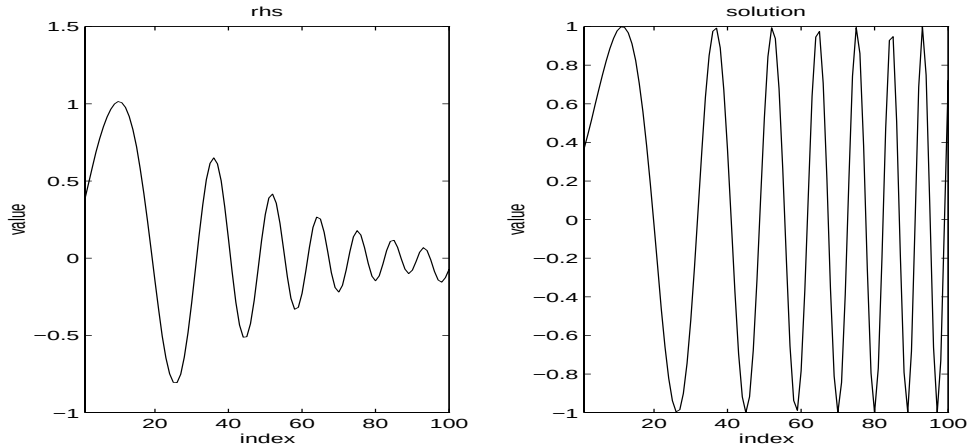

FIG. 10. *Uncontaminated data vector (left) and exact solution vector (right) for Example* 3.

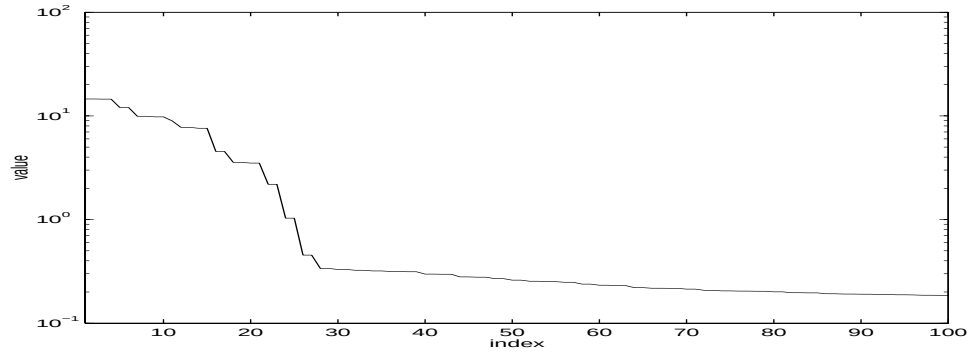

FIG. 11. *Fourier coefficients of the noisy data for Example* 3.

The condition number of T is approximately 4.67×10^7. We first defined the exact solution[5] shown in Figure 10. The uncorrupted data were obtained by calculating $\hat{g} = T\hat{f}$ and are also shown in Figure 10. White noise was added to $\hat{g}$ to obtain the noisy data whose Fourier coefficients are shown in Figure 11, where the noise level was determined to be 1×10^{-3}.

As Figure 12 and Table 5.3 indicate, the minimum relative error obtained with no preconditioning was 2.12×10^{-2} in 50 iterations. For values of m close to 33, however, the preconditioned system converges in fewer than 10 iterations to the same minimum relative error. The method of [14] compares about the same as in the other examples; that is, a smaller relative error could be achieved, but at the expense of over twice the number of iterations that our method requires to achieve its minimum relative error. As in the first example, we can compare the average gain per iteration for particular values of m/cutoff. The method of [14] achieves its smallest minimum relative error value of 1.55×10^{-2} when the cutoff is 25; it takes 25 iterations, giving an average gain of .00062 per iteration. In contrast, our method achieves its smallest minimum relative error of 2.12×10^{-2} after 9 iterations with $m = 33$, yielding an average gain

[5] $\hat{f} = \sin([1{:}0.1{:}10.9]^2 \cdot \frac{3\pi}{25})$.

TABLE 5.3
Minimum relative errors achieved for various values of m; Example 3.

	Our method		Method of [14]	
m (cutoff)	Minimum rel. error	Achieved at iter.	Minimum rel. error.	Achieved at iter.
0	2.12×10^{-2}	50		
21	2.13×10^{-2}	25	1.83×10^{-2}	34
25	2.12×10^{-2}	19	1.55×10^{-2}	25
29	2.19×10^{-2}	12	1.98×10^{-2}	24
33	2.12×10^{-2}	9	2.62×10^{-2}	23
37	2.15×10^{-2}	8	3.23×10^{-2}	20
41	2.50×10^{-2}	7	3.70×10^{-2}	19

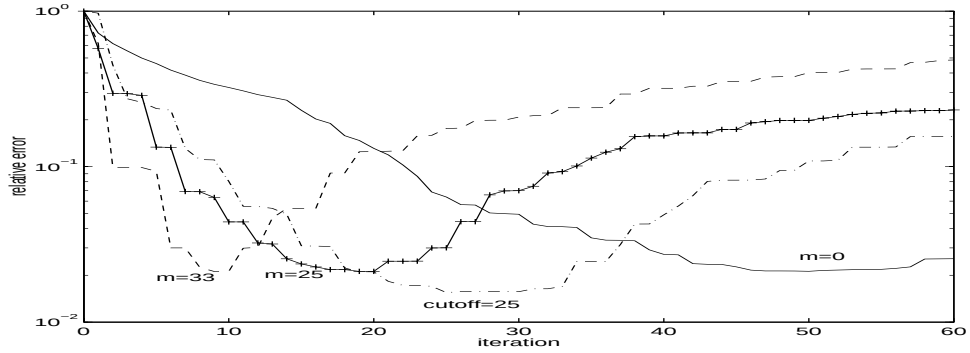

FIG. 12. *Relative error in computed solution for our preconditioner when* $m = 0$, $m = 25$, *and* $m = 33$ *and relative error in computed solution for the method in* [14] *with cutoff* $= 25$ *eigenvalues*; *Example* 3.

of .00236 per iteration.

We also observe from Figure 13 that when our preconditioner is defined for $m = 33$, the largest 33 singular values cluster around 1, and the small singular values are left almost unchanged. For $m = 33$ we found $\epsilon_{max} = \|C_1^{-1}C_2\|_\infty = 1.35 \times 10^1$, and $\hat{s} = \|E_2\|_\infty = 5.10 \times 10^{-2}$; thus Theorem 3.1 predicts the largest singular value of $M^{-1}C$ is bounded above by about 3.8. In fact, the largest singular value is 2.4, while the next largest singular value is only 1.09. In Theorem 3.2 we found $\|C_1\|_2 = 1.13$ so that the quantity multiplying $\hat{\sigma}_{n-k+1}$ on the right-hand side of (13) was 5.71×10^1, just over half the condition number of C_1. Thus, for a suitable value of k in (13), the mixing between the upper and lower subspaces is small.

6. Conclusions. We have developed an efficient algorithm for computing regularized solutions to ill-posed problems with Toeplitz structure. This algorithm makes use of a unitary transformation to a Cauchy-like system and iterates using the CGLS algorithm preconditioned by a rank-m partial factorization with pivoting. By exploiting properties of the transformation, we showed that each iteration of CGLS costs only $O(n \lg n)$ operations for a system of n variables, the same as the cost per iteration of the method in [14].

Our theory predicts that for banded Toeplitz matrices we can expect the precon-

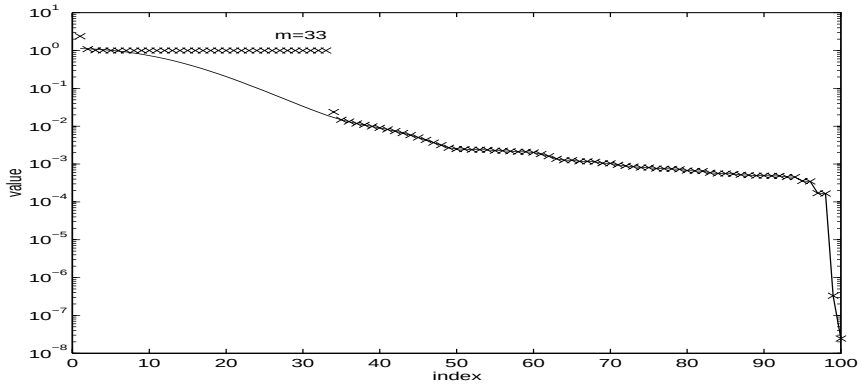

FIG. 13. *Singular values of C (solid line) and $M^{-1}C$ ($\times$'s) for Example* 3, $m = 33$.

ditioner determined in the course of Gu's fast modified complete pivoting algorithm to cluster the largest singular values of the preconditioned matrix around 1, keep the smallest singular values small, and not mix the upper and lower subspaces. Thus CGLS produces a good approximate solution within a small number of iterations. Our results illustrate the effectiveness of our preconditioner for an optimal value of m and for values in a neighborhood of the optimal one. The results further illustrate the advantage of our method over the preconditioned scheme of [14] in terms of reaching a reasonable regularized solution in many fewer iterations. Hence, our algorithm is both efficient and practical.

Determining the optimal value of m can be difficult, and it appears better to underestimate the value rather than to overestimate it. Advances in computing truly rank-revealing factorizations of Cauchy-like matrices will yield corresponding advances in our algorithm.

We note a left preconditioner can be defined similarly in the case when T has dimension $N \times n$, where $N > n$. In this case we have $\tilde{C} = F_N T S_0^* F_n^*$, where the subscript on F denotes the dimension of the normalized Fourier transform matrix and S_0 is $n \times n$. We determine the square principal submatrix C_1 as before and augment it by an $(N-m) \times (N-m)$ identity matrix so that M is now $N \times N$. The proofs of Theorems 3.1 and 3.3 remain the same. Theorem 3.2 can be adapted to the $N \times n$ case by adding $N-n$ rows of zeros to the Σ and $\hat{\Sigma}$ matrices, appending the $N \times (N-n)$ matrices Q_4 and $\hat{Q}_4$ in the definition of the singular value decompositions, and replacing C^{-1} in the proof with $C^\dagger$.

Similar ideas are valid for preconditioners of the form

$$\begin{bmatrix} C_1 & & \\ & C_2 & \\ & & I \end{bmatrix},$$

where C_1 and C_2 are both Cauchy-like. In practice, C_2 can be determined by computing a partial factorization of the trailing submatrix of C, remaining after C_1 is removed. This method saves time in the precomputation of M but more iterations may be required for convergence.

There are other unitary and real orthogonal transforms relating Toeplitz and Cauchy-like matrices (see [21, 25, 8], for instance). The particular transform exploited here allows us to apply the preconditioner in a fast and stable way as discussed in

section 4.3. However, Hermitian structure is not preserved under this transformation, and therefore a method such as CGLS which solves the normal equations must be used. As an alternative, when T is Hermitian, one might apply a minimal residual variant of CG called MR-II [13] with an appropriate symmetric preconditioner to the original Toeplitz system.

Finally, we note that these ideas have been extended in [26] to the case of computing regularized solutions to two-dimensional problems in which T is block Toeplitz with Toeplitz blocks.

Acknowledgment. We would like to thank Dr. Gohberg for introducing us to Cauchy-like matrices and giving us a preprint of [8], thus providing the inspiration for this work.

REFERENCES

[1] A. Bojanczyk, *personal communication*, 1996.
[2] R. Chan, J. Nagy, and R. Plemmons, *Circulant preconditioned Toeplitz least squares iterations*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 80–97.
[3] T. Chan, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 766–771.
[4] T. F. Chan and P. C. Hansen, *A lookahead Levinson algorithm for general Toeplitz systems*, IEEE Proc. Signal Processing, 40 (1992), pp. 1079–1090.
[5] R. D. Fierro, G. H. Golub, P. C. Hansen, and D. P. O'Leary, *Regularization by truncated total least squares*, in Proceedings of the Fifth SIAM Conference on Applied Linear Algebra, J. G. Lewis, ed., SIAM, Philadelphia, 1994, pp. 102–105.
[6] T. Finck, G. Heinig, and K. Rost, *An inversion formula and fast algorithms for Cauchy-Vandermonde matrices*, Linear Algebra Appl., 183 (1993), p. 179.
[7] H. E. Fleming, *Equivalence of regularization and truncated iteration in the solution of ill-posed problems*, Linear Algebra Appl., 130 (1990), pp. 133–150.
[8] I. Gohberg, T. Kailath, and V. Olshevsky, *Fast Gaussian elimination with partial pivoting of matrices with displacement structure*, Math. Comp., 64 (1995), pp. 1557–1576.
[9] W. Groetsch, *Theory of Tikhonov Regularization for Fredholm Equations of the First Kind*, Pitman, Boston, MA, 1984.
[10] M. Gu, *Stable and efficient algorithms for structured systems of linear equations*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 279–306.
[11] M. H. Gutknecht and M. Hochbruck, *Look-ahead Levinson and Schur Algorithms for Non-Hermitian Toeplitz Systems*, Numer. Math., 70 (1995), pp.181–227.
[12] M. Hanke, *Regularization with differential operators: An iterative approach*, Numer. Funct. Anal. Optim., 13 (1992), pp. 523–540.
[13] M. Hanke and J. Nagy, *Restoration of atmospherically blurred images by symmetric indefinite conjugate gradient techniques*, Inverse Problems, 12 (1996), pp. 157–173.
[14] M. Hanke, J. Nagy, and R. Plemmons, *Preconditioned iterative regularization for ill-posed problems*, Numerical Linear Algebra, L. Reichel, A. Ruttan, and R. S. Varga, eds., de Gruyter, Berlin, 1993, pp. 141–163.
[15] M. Hanke and T. Raus, *A general heuristic for choosing the regularization parameter in ill-posed problems*, SIAM J. Sci. Comput, 17 (1996), pp. 956–972.
[16] P. C. Hansen, *The discrete Picard condition for discrete ill-posed problems*, BIT, 30 (1990), pp. 658–672.
[17] P. C. Hansen, *Rank Deficient and Discrete Ill-Posed Problems*, Ph.D. thesis, Technical University of Denmark, Lyngby, Denmark, 1995.
[18] P. C. Hansen and D. P. O'Leary, *The use of the L-curve in the regularization of discrete ill-posed problems*, SIAM J. Sci. Comput., 14 (1993), pp. 1487–1503.
[19] G. Heinig, *Inversion of generalized Cauchy matrices and other classes of structured matrices*, in Linear Algebra in Signal Processing, IMA Vol. Math. Appl. 69, Springer-Verlag, New York, 1994, pp. 95–114.
[20] G. Heinig, *personal communication.* 1996.
[21] G. Heinig and A. Bojanczyk, *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices.* I. *Transformations*, Linear Algebra Appl., 254 (1997), pp. 193–226.

[22] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Natl. Bur. Standards, 49 (1952), pp. 409–436.
[23] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1991.
[24] T. Kailath, S. Kung, and M. Morf, *Displacement ranks of matrices and linear equations*, J. Math. Anal. Appl., 78 (1979), pp. 395–407.
[25] T. Kailath and V. Olshevsky, *Displacement structure approach to discrete-trigonometric-transform based preconditioners of the G. Strang type and of T. Chan type*, Calcolo, 33 (1996), pp. 191–208.
[26] M. Kilmer, *Cauchy-Like Preconditioners for Two-Dimensional Ill-Posed Problems*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 777–799.
[27] C. Lanczos, *Solution of systems of linear equations by minimized iterations*, J. Res. Natl. Bur. Standards, 49 (1952), pp. 33–53.
[28] D. L. Phillips, *A technique for the numerical solution of certain integral equations of the first kind*, J. ACM, 9 (1962), pp. 84–97.
[29] G. W. Stewart and J. G. Sun, *Matrix Perturbation Theory*, Academic Press, New York, 1990.
[30] G. Strang, *A proposal for Toeplitz matrix calculations*, Stud. Appl. Math., 74 (1986), pp. 171–176.
[31] D. Sweet and R. Brent, *Error analysis of a fast partial pivoting method for structured matrices*, Advanced Signal Processing Algorithms, Proc. SPIE, 2363 (1995), pp. 266–280.
[32] D. Sweet, *The use of pivoting to improve the numerical performance of Toeplitz matrix algorithms*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 468–493.
[33] A. van der Sluis and H. van der Vorst, *The rate of convergence of conjugate gradients*, Numer. Math, 48 (1986), pp. 543–560.
[34] A. van der Sluis and H. van der Vorst, *Sirt- and CG-type methods for the iterative solution of sparse linear least-squares problems*, Linear Algebra Appl., 130 (1990), pp. 257–302.
[35] J. M. Varah, *Pitfalls in the numerical solution of linear ill-posed problems*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 164–176.