JOURNAL OF COMPLEXITY 8, 1–21 (1992)

Parallel Solution of Toeplitzlike Linear Systems*

VICTOR PAN

Computer Science Department, State University of New York at Albany, Albany, New York 12222, and Mathematics and Computer Science Department, Lehman College, CUNY, Bronx, New York 10468

Received June 28, 1990

The known algorithms invert an $n \times n$ Toeplitz matrix in sequential arithmetic time $O(n \log^2 n)$, but their parallel time is not better than linear in n. We present a weakly numerically stable algorithm that numerically inverts an $n \times n$ well-conditioned Toeplitz matrix A in sequential time $O(n \log^4 n \log \log n)$ or in parallel time of $O(\log^3 n)$ using $O(n \log n \log \log n)$ processors. The algorithm keeps all its advantages being applied to the inversion of Toeplitzlike matrices, having smaller displacement ranks (which makes it applicable to pseudo-inversion of Toeplitzlike matrices of full rank).

1. INTRODUCTION

Computations with Toeplitz, Toeplitzlike, and other well-structured dense matrices have numerous applications in the sciences and engineering (see Bunch (1985) and Kailath (1987) for surveys and further references), and they have received much attention from researchers. In particular, $O(n \log^2 n)$ arithmetic operations suffice in order to solve a linear system with an $n \times n$ Toeplitz matrix $A = [a_{ij}]$ (where $a_{ij} = a_{i-j}$, $i, j = 0, 1, \ldots, n-1$) by means of the various algorithms of Brent *et al.* (1980), Bitmead and Anderson (1980), Musicus (1981), deHoog (1987), Ammar and Gragg (1988), and Chun and Kailath (1991). Moreover, the algorithms compute the first and the last columns of the inverse matrix A^{-1}, and then any column can be immediately computed by means of at most

* Supported by NSF Grants CCR-8805782 and CCR-9020690 and PSC-CUNY Award Nos. 661340, 668541, and 669290.

0885-064X/92 $3.00

nine fast Fourier transforms (hereafter referred to as FFTs). All these algorithms only allow parallel acceleration by factors ranging from $\log n$ to $\log^2 n$, but even such an acceleration is considered to be of a great practical value (see Kailath, 1987, p. 74).

Some other known algorithms solve Toeplitz linear systems in time polylogarithmic in n but only at the expense of using very many processors, so that the total work of such algorithms (represented by the product of their time and processor bounds) exceeds n^2 (compare Csanky, 1976; Borodin *et al.*, 1982). Thus we arrived at the question: Can a Toeplitz linear system of n equations be solved in polylogarithmic time using, say, n processors? The positive answer to this question was well known for some specific subclasses of Toeplitz systems, such as triangular Toeplitz and circulant linear systems, and more recently, has been extended to well-conditioned symmetric Toeplitz systems, but remained a challenge for general Toeplitz linear systems.

This paper presents a weakly numerically stable (according to the definition of Bunch (1987)) parallel algorithm for the numerical solution of well-conditioned Toeplitz linear systems (based on a novel approach) that supports the desired parallel polylogarithmic time bound and preserves a near optimum (within a polylog factor) total work bound.

Our algorithm relies on Newton's iteration and inherits its self-correcting property (although in principle, this approach may be similarly combined with any other iterative scheme for matrix inversion, rather than with Newton's). Given a well-conditioned $n \times n$ Toeplitz matrix A such that $\log \kappa = O(\log n)$, $\kappa = \|A\|_2 \|A^{-1}\|_2$, we numerically compute the first and the last columns of the inverse A^{-1} of A (with error norm less than $\|A^{-1}\|_1 / 2^N$, $N = n^c$ for any fixed constant c), and we use $O(\log^3 n)$ parallel arithmetic steps and $O(n \log n \log\log n)$ processors. We also extend our algorithm and the latter complexity estimates to the solution of any well-conditioned Toeplitzlike linear system, that is, of the system whose coefficient matrix has *displacement rank* bounded by a constant. To be certain, we assume the customary PRAM model of parallel computations (see, for instance, Karp and Ramachandran, 1990; Borodin *et al.* 1982), where in each step each processor performs at most one arithmetic operation, but our results hold over any parallel computer model that supports (within a constant factor) the bounds of Table I on parallel time and on the number of processors for the elementary computations listed there. (Our entire algorithm is immediately reduced to these computations; note the simplicity of data movement between the processors in the case of these computations, which means low cost of processor communication and synchronization.)

Our techniques may be of independent interest; they include Newton's iteration for matrix inversion performed in a novel way that preserves the structure of the input matrix; a homotopic method of computing a good

TABLE I
PARALLEL COMPLEXITY OF BASIC COMPUTATIONS

Parallel computation for	Parallel time (t_{par})	Processors (p)	Source
1. FFT at n points (n is a power of 2)	$3(1 + \log_2 n)$	$2n$	(Pease, 1968)
2. Summation of n numbers	$2[\log_2 n]$	$[n/\log_2 n]$	(Quinn, 1987)
3. Inner product	$2[\log_2 n] + 1$	$[n/\log_2 n]$	Reduce to summation
4. $n \times n$ matrix times a vector	$2[\log_2 n] + 1$	$n[n/\log_2 n]$	n inner products
5. Product of $n \times n$ matrices	$2[\log_2 n] + 1$	$n^2[n/\log_2 n]$	n^2 inner products
6. $n \times n$ Toeplitz matrix by a vector (n is a power of 2)	$1 + 9(2 + \log_2 n)$	$4n$	Reduce to 3 FFTs at $2n$ points, (Aho *et al.*, 1976)

initial approximation for Newton's iteration; and a technique of approximation to a matrix by a Toeplitzlike matrix.

We use the notation (t, p) for the pairs of the upper bounds on parallel time and the number of processors; for instance, for the parallel cost of FFT, such a pair is given by $(3(1 + \log_2 n), 2n)$. The expressions in this form for the parallel computational cost of our inversion algorithm are quite complicated, and we usually simplify them in two ways: either by deleting the smaller order terms and writing $\sim(t, p)$ [say, $\sim(3 \log_2 n, 2n)$ for FFT] or by using asymptotic estimates and writing $O_A(t, p)$ to denote that we have the upper bounds $O(st)$ on the parallel arithmetic time (that is, on the number of parallel arithmetic steps) and, simultaneously, $O(p/s)$ on the number of processors for any s, $1 \leq s \leq p$. This includes Brent's (1974) *slowdown principle* of parallel computing, according to which it suffices to slow down the computations by $O(s)$ times in order to perform them with by s times fewer processors.

We organize the paper as follows: In the next two sections, we recall some customary definitions and simple facts of the theory of matrix computations (in Section 2) and of displacement representation of a matrix (Section 3). In Section 4, we recall Newton's iteration for matrix inversion and consider it in the case of the inversion of Toeplitzlike matrices, assuming that we have a good initial approximation. In Section 5, we show how to compute such a good initial approximation for any well-conditioned matrix. In Section 6, we modify Newton's iteration in order to preserve a Toeplitzlike structure of the auxiliary matrices. Finally, in Section 7, we estimate the computational cost of the resulting Newton's modified algorithm for the solution of Toeplitzlike linear systems.

2. Some Definitions and Auxiliary Facts

We start by recalling some customary definitions and simple facts of the theory of matrix computations (Golub and van Loan, 1989; Parlett, 1980; Wilkinson, 1965). $\mathbf{R}_{p,q}$ denotes the linear space of $p \times q$ real matrices, which are vectors if $q = 1$ or $p = 1$. $O_{m,n}$ denotes the $m \times n$ null matrix, I_n the $n \times n$ identity matrix; we write I and O unless the matrix size is not clear from the context. $\text{diag}(a_1, \ldots, a_k)$ denotes the $k \times k$ diagonal matrix with the entries $a_1, \ldots, a_k$ on the diagonal. $X = [X_1, \ldots, X_k]$ denotes the $1 \times k$ block vector whose entries are vectors or matrices. $\text{trace } W = \sum_i w_{ii}$ denotes the trace of a matrix $W = [w_{ij}]$, W^T its transpose, $\det W$ its determinant, and $\|W\|_h$ its h-norm,

$$\|W\|_h = \max \|W\mathbf{v}\|_h / \|\mathbf{v}\|_h, \tag{2.1}$$

where the maximization is over all the nonzero vectors $\mathbf{v}$, $h = 1, 2, \infty$. For an $n \times n$ matrix $W = [w_{ij}]$, we have

$$\|W\|_1 = \|W^T\|_\infty = \max_j \sum_i |w_{ij}|, \tag{2.2}$$

$$\|W\|_1 / \sqrt{n} \le \|W\|_2 = \|W^T\|_2 \le \|W\|_1 \sqrt{n}, \tag{2.3}$$

$$\max_{i,j} |w_{ij}| \le \|W\|_2 \le \max_{i,j} |w_{ij}| \, n. \tag{2.4}$$

If a matrix W is *symmetric and positive definite* (we abbreviate as s.p.d.) with eigenvalues $\lambda_1, \ldots, \lambda_n$ such that $0 < \lambda_n \le \cdots \le \lambda_2 \le \lambda_1$, then

$$\|W\|_2 = \lambda_1, \qquad \|W^{-1}\|_2 = 1/\lambda_n, \tag{2.5}$$

and consequently,

$$\|W + aI\|_2 = \lambda_1 + a, \qquad \|(W + aI)^{-1}\|_2 = 1/(\lambda_n + a), \tag{2.6}$$

for any positive scalar a. A matrix $W = [w_{ij}]$ is called *diagonally dominant* if $2|w_{ii}| > \sum_j |w_{ij}|$, for all i, or $2|w_{jj}| > \sum_i |w_{ij}|$, for all j:

$$\kappa_p(M) = \|M\|_p \|M^{-1}\|_p \quad \text{for } p = 1, 2, \infty \text{ and for } a \textit{ nonsingular matrix } M. \tag{2.7}$$

The singular value decomposition (SVD) of an $n \times n$ matrix W of rank $r \le n$ can be defined as follows:

$$W = U \sum V^T,\ U \in \mathbf{R}_{n,r},\ V \in \mathbf{R}_{n,r},\ U^T U = V^T V = I_r, \quad (2.8)$$

$$\sum = \operatorname{diag}(\sigma_1, \ldots, \sigma_r), \qquad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0. \quad (2.9)$$

The matrix W uniquely determines the matrix Σ of the relations (2.8)–(2.9) and, if all the singular values σ_k are distinct, then the matrices U and V too. Let us denote

$$W = GH^T, \qquad G = U \sum, \qquad H = V. \quad (2.10)$$

For an integer d such that $0 \leq d \leq r$, let U_d, V_d, G_d, and H_d denote the four matrices formed by the first d columns of the four matrices U, V, G, and H, respectively, so that

$$W_d = G_d H_d^T,\ G_d = U_d \sum_d,\ H_d = V_d,\ \sum_d = \operatorname{diag}(\sigma_1, \ldots, \sigma_d). \quad (2.11)$$

Then W_d has rank d, and (see Golub and van Loan, 1989, p. 73)

$$\|W - W_d\|_2 \leq \|W - Y\|_2 \qquad \text{for all matrices } Y \text{ of rank at most } d. \quad (2.12)$$

Our results stated for real matrices also hold for complex ones, with complex Hermitian matrices playing the role of real symmetric ones.

Hereafter, log and ln stand for the binary and exponentially based logarithms, respectively, unless the base of log is specified.

3. Displacement Representations of a Matrix

In this section, we recall the definitions and some fundamental properties of the displacement representation of matrices. Such a representation characterizes Toeplitz structure of matrices. Informally, the matrix is more Toeplitzlike if its displacement rank is smaller.

Definition 3.1. $L(\mathbf{v})$ denotes a lower triangular Toeplitz matrix with the first column $\mathbf{v}$. J is the reversion matrix, filled with zeros except for its main antidiagonal entries filled with ones. Z denotes the matrix, zero everywhere except for its first subdiagonal, filled with ones. Thus $J\mathbf{v} = [v_k, \ldots, v_0]^T$, $Z\mathbf{v} = [0, v_0, \ldots, v_{k-1}]^T$, for $\mathbf{v} = [v_0, \ldots, v_k]^T$.

Definition 3.2 (Kailath *et al.*, 1979). A pair of $n \times d$ matrices G and H is called a *ZZ^T-displacement generator* (of length d), for an $n \times n$ matrix S, if

$$F_+(S) = S - ZSZ^T = GH^T. \tag{3.1}$$

Such a pair is called a Z^TZ-displacement generator (of length d) for an $n \times n$ matrix S if

$$F_-(S) = S - Z^TSZ = JG\,H^TJ. \tag{3.2}$$

$d_+(S)$ [respectively, $d_-(S)$] denotes the ZZ^T- (respectively, Z^TZ-) *displacement rank* of S, defined as the minimum integer d in the decomposition (3.1) [respectively, (3.2)]. We drop the prefices (ZZ^T-) and/or (Z^TZ-) where distinguishing between them is not important or is clear from the context.

Remark 3.1. For any Toeplitz matrix A, the decompositions (3.1) and (3.2), with $d = 2$ and with $S = A$, are immediately defined, so that $d_+(A) \leq 2$, $d_-(A) \leq 2$.

PROPOSITION 3.1 (Chun *et al.*, 1987; Kailath *et al.*, 1979). *A pair* (G, H) *of* $n \times d$ *matrices* $G = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_d]$ *and* $H = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_d]$ *is a* ZZ^T- *(respectively, an* Z^TZ-*) displacement generator of length* d *for an* $n \times n$ *matrix* S *if and only if*

$$S = L_+(G, H) = \sum_{i=1}^{d} L(\mathbf{g}_i)L^T(\mathbf{h}_i) \tag{3.3}$$

[respectively, if and only if

$$S = L_-(G, H) = \sum_{i=1}^{d} L^T(\mathbf{g}_i)L(\mathbf{h}_i)]. \tag{3.4}$$

COROLLARY 3.1 (see Bitmead and Anderson, 1980, Lemma 5). *For any pair of vectors* $\mathbf{g}$ *and* $\mathbf{h}$ *of the same dimension,* $L(\mathbf{g})L^T(\mathbf{h}) = L^T(\mathbf{u}) + L(\mathbf{v}) - L^T(ZJ\mathbf{g})L(ZJ\mathbf{h})$, $L^T(\mathbf{g})L(\mathbf{h}) = L^T(\mathbf{s}) + L(\mathbf{t}) - L(ZJ\mathbf{g})L^T(ZJ\mathbf{h})$ *where* J *and* Z *are the matrices of Definition* 3.1, $\mathbf{u}^TJ$ *is the last row of* $L(\mathbf{g})L^T(\mathbf{h})$, $J\mathbf{v}$ *is the last column of* $L(\mathbf{g})L^T(\mathbf{h})$, $\mathbf{s}^T$ *is the first row of* $L^T(\mathbf{g})L(\mathbf{h})$, *and* $\mathbf{t}$ *is the first column of* $L^T(\mathbf{g})L(\mathbf{h})$.

Corollary 3.1 immediately defines a Z^TZ- (respectively, a ZZ^T-) displacement generator of length at most $d + 2$ for every matrix A given with its ZZ^T- (respectively, Z^TZ-) displacement generator of length d, so that

$$d_-(A) - 2 \leq d_+(A) \leq d_-(A) + 2. \tag{3.5}$$

If $n/2$ substantially exceeds the value d of the decompositions (3.3) and/or (3.4), then it is economical to represent the matrix A with the $2dn$ entries of its displacement generator.

DEFINITION 3.3. Hereafter, we freely identify computing a matrix with computing its displacement generator.

Given two matrices A and B with their displacement generators of lengths d_1 and d_2, respectively, we may, of course, combine the generators together into a displacement generator of length $d_1 + d_2$ for the matrix $A + B$. Furthermore, we have the following fundamental result of Chun *et al.* (1987) (which we state specifying the computational cost estimates omitted in Chun *et al.* (1987)):

PROPOSITION 3.2. *Given ZZ^T- (or Z^TZ)$-$displacement generators of length d_1 for an $n \times n$ matrix A and of length d_2 for an $n \times n$ matrix B, then ZZ^T- (or Z^TZ)$-$ displacement generators of length at most $d_1 + d_2 + 1$ for the matrices A^TA and AB can be computed for the parallel cost $(3 \log(4nd_1d_2), 4nd_1d_2)$.*

Proof. Indeed, it is easily verified that

$$F_+(AB) = F_+(A)B + AF_+(B) - F_+(A)F_+(B) - (ZA\mathbf{e}_1)(\mathbf{e}_n^TBZ)$$

where $\mathbf{e}_i$ denotes the ith unit coordinate vector, and (3.1) defines $F_+(S)$ for $S = A$, $S = B$, and $S = AB$. We seek the matrix $F_+(AB)$ in the format (3.1); we also use this format for the representation of the matrices $F_+(A)$ and $F_+(B)$ and of their products with each other and with the matrices A and B, and we use the format (3.3) for the matrices A and B. Apart from the multiplication of the two matrices $F_+(A)$ and $F_+(B)$, the computation is essentially reduced to multiplications of d_1 vectors by B and of A by d_2 vectors, which are in turn reduced to a pair of d_1d_2 concurrent multiplications of $n \times n$ triangular Toeplitz matrices by vectors and to the summation of the resulting vectors. Application of the auxiliary estimates of Table I yields Proposition 3.2. ■

To conclude, we recall the following fact:

PROPOSITION 3.3 (Kailath *et al.*, 1979). *Let A be a nonsingular matrix. Then $d_+(A) = d_-(A^{-1})$, $d_-(A) = d_+(A^{-1})$.*

Remark 3.2. The approach of this section can be extended to measure non-Toeplitz structure of matrices as well, in which case the operators $A - ZAZ^T$ and $B - Z^TBZ$ in the representations (3.1) and (3.2) should be replaced by other matrix operators such as $FA - AF$ or $A - FAF^T$ for appropriate matrices F (see Kailath *et al.*, 1978; Gohberg *et al.*, 1986; Chun and Kailath, 1991; Pan, 1990).

4. Newton's Iteration for the Inversion of Toeplitzlike Matrices

We base our parallel solution of a Toeplitzlike linear system on Newton's iteration for matrix inversion,

$$X_{k+1} = X_k(2I - WX_k) = (2I - X_kW)X_k, \qquad k = 0, 1, \ldots, \quad (4.1)$$

which quadratically converges to the solution $X = W^{-1}$ of the matrix equation $f(X) = X^{-1} - W = O$.

Proposition 4.1. *Let matrices $X_1, X_2, \ldots$ be defined by Newton's iteration* (4.1), *given matrices W and X_0. Let $K = 2^k$ for $k = 1, 2, \ldots$. Then*

$$I - WX_k = (I - WX_{k-1})^2, \qquad \|WX_k - I\| \le \|WX_{k-1} - I\|^2 \le \|WX_0 - I\|^K$$

for any fixed matrix norm and for $k = 1, 2, \ldots$. Furthermore, if $q(0) = \|WX_0 - I\| < 1$, then the matrix W is nonsingular, and

$$\|W^{-1}\| \le \|X_0\|/(1 - q(0)), \qquad \|X_k - W^{-1}\| \le q(0)^K\|W^{-1}\| \quad \text{for } K = 2^k,\ k = 0, 1, \ldots.$$

In the next section, we reduce the inversion of any matrix A to the inversion of matrices W for which $q(0)$ is noticeably less than 1, so that Proposition 4.1 implies rapid convergence to W^{-1} in terms of the overall number of steps (4.1) involved. The overall computational cost of this process, however, also depends on the computational cost of each step (4.1). We assume hereafter that the matrices W and X_0 are given with their ZZ^T– displacement generators of lengths d and $d(0)$, respectively, and moreover, that d is bounded by a constant as $n \to \infty$,

$$d(0) \le d + 2 = O(1) \qquad \text{as } n \to \infty \quad (4.2)$$

(compare Remark 5.3 below).

Now we apply Proposition 3.2 and deduce that the matrices X_k of the equation (4.1) can be recursively computed with their ZZ^T–displacement generators of lengths

$$d(k) \le 2^k d(0) + (2^k - 1)(d + 3) \qquad \text{for } k = 1, 2, \ldots,$$

and thus, due to (4.2),

$$d(k) \le 2^k(2d + 5) - d - 3 = O(2^k). \quad (4.3)$$

Step k of (4.1) is essentially reduced to multiplication of X_k by W and of $2I - X_kW$ by X_k for the computational cost

$$(t(k), p(k)) \le (3 \log(16dn^2d(k)^2(d(k) + d + 2)), 4nd(k)(d(k) + d + 2))$$
$$\sim (3 \log(16dn^2d(k)^3), 4nd(k)^2) = O_A(k + \log n, n4^k). \quad (4.4)$$

(This follows from Proposition 3.2.)

Now, we may estimate the cost of approximating W^{-1} by using Newton's iteration.

THEOREM 4.1. *Fix a matrix norm, positive ε and $q(0)$, both less than 1, natural d and $d(0) \le d + 2$, and the ZZ^T−displacement generators of length d for an $n \times n$ matrix W and of length $d(0)$ for an $n \times n$ matrix X_0 such that*

$$\|W X_0 - I\| \le q(0);$$

also let

$$k_\varepsilon = \lceil \log | \log \varepsilon / \log q(0) | \rceil. \quad (4.5)$$

Then it suffices to perform k_ε steps (4.1) of Newton's iteration, in order to compute a ZZ^T−displacement generator of length $d(k_\varepsilon) \le 2^{k_\varepsilon}(2d + 5) - d - 3$ for the matrix X_{k_ε} such that

$$\|X_{k_\varepsilon} - W^{-1}\| \le \varepsilon \|W^{-1}\|.$$

These steps can be performed for the overall computational cost $(t_\varepsilon, p_\varepsilon)$ where

$$t_\varepsilon = 6k_\varepsilon(3k_\varepsilon + \log(16d(2d + 5)^3n^2)), \quad (4.6)$$

$$p_\varepsilon = 4^{k_\varepsilon+1} n(2d + 5)^2/t_\varepsilon = 4n(2d + 5)^2(\log \varepsilon / \log q(0))^2/t_\varepsilon. \quad (4.7)$$

Proof. Combining (4.3), (4.4), and Proposition 4.1 gives us the estimates of Theorem 4.1, except that we need to refine the processor bound by applying Brent's slowdown principle of parallel computing. Indeed, applying (4.4) for $k = 1, 2, \ldots, k_\varepsilon$, we obtain the cost bounds $(t_\varepsilon^*, p_\varepsilon^*)$, $t_\varepsilon^* = t_\varepsilon/2$, $p_\varepsilon^* = p(k_\varepsilon)$ where t_ε is defined by (4.6), k_ε by (4.5), and $p(k)$ by (4.4). Slowing down by the factor of $t_\varepsilon/2^k$ the $(k_\varepsilon - k)$th step (4.1), for $k = 1, 2, \ldots, k_\varepsilon$, we decrease by the same factor the processor bound of this step and thus arrive at (4.6) and (4.7). ∎

Hereafter, let b denote $\log(1/\varepsilon)$, $\varepsilon = 2^{-b}$,

$$h = 1/\log(1/q(0)), \qquad h < \tfrac{1}{5} \text{ for } q(0) < \tfrac{1}{32}. \tag{4.8}$$

Then $k_\varepsilon \le \lceil \log(0.2b) \rceil$,

$$\begin{aligned} t_\varepsilon &\le 6\lceil \log(0.2b) \rceil (3\lceil \log(0.2b) \rceil + \log(16d(2d+5)^3 n^2) \\ &= O(\log b \log(bn)), \end{aligned} \tag{4.9}$$

$$p_\varepsilon \le (0.8b)^2 n(2d+5)^2/t_\varepsilon = O(b^2 n/(\log b \log(bn))). \tag{4.10}$$

In the next sections, we reduce approximating A^{-1} (for a given Toeplitzlike matrix A) to a sequence of Newton's iterations (4.1) applied to some auxiliary matrices W (with b appropriately bounded from above) so as to ensure (4.8) for each such application of (4.1).

Remark 4.1. If W is a Toeplitz matrix, then $d = d(0) \le 2$, $d(k) \le 7 * 2^k - 5$, $k = 0, 1, \ldots$, and similarly, the bounds (7.1) of Section 6 can be improved to $d_+(\tilde{X}_k) \le 2$, $d(k+1) \le 9$ in such a case.

5. Numerical Inversion of a Symmetric Positive Definite Matrix

In this section, we start with a positive tolerance value ε and with a general symmetric positive definite $n \times n$ matrix A and compute a desired approximation $\tilde{A}^{-1}$ to A^{-1} such that

$$\|\tilde{A}^{-1} - A^{-1}\|_2 \le \varepsilon \|A^{-1}\|_2. \tag{5.1}$$

Symmetrization extends this result to any nonsingular matrix A, since $A^{-1} = (A^T A)^{-1} A$ (see also Remark 5.2 below).

We compute this approximation $\tilde{A}^{-1}$ by applying our method of the homotopic variation of the matrix diagonal, that is, we recursively apply Newton's iteration to invert the matrices $A_j = A + p^j aI$ for $j = 0, 1, \ldots$ where a and p are positive scalars, a is large (so that it is easy to invert the matrix A_0), p is small enough (so that A_m^{-1} is a good initial approximation to A^{-1} already for a moderately large m) but is not too small (so that A_{j-1}^{-1} is a good initial approximation to A_j^{-1} for each j).

Let us specify this algorithm and estimate its computational cost. We apply the definitions of Section 2, as well as the following definitions:

Definition 5.1. A is an s.p.d. matrix; p, κ_2, δ, and q are positive scalars; k and m are natural numbers such that $q = 1 - p(1+p)/2$, $\kappa_2 =$

$\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 \geq 1;$

$$m \geq \frac{\ln(n\kappa_2/(1-p)^2)}{\ln(1/p)}, \quad \frac{1}{\delta} \geq \frac{2(p+\kappa_2)}{p(1-p)}, \quad k = \lceil \log(\log \delta/\log q) \rceil. \tag{5.2}$$

For example, setting $q = q(0) < \frac{1}{32}$ implies the relations

$$p < 0.97902, \qquad m \geq 47.14 \ln(2269.7\, n\kappa_2), \tag{5.3}$$

$$k \geq \log(0.2 \log(97.3(0.979 + \kappa_2))). \tag{5.4}$$

Algorithm 5.1.

Input. An s.p.d. matrix $A = [a_{ij}]$ and a positive ε.

Output. A matrix $\tilde{A}^{-1}$ satisfying the bound (5.1).

The Initialization Stage. Choose scalars m, p, and k according to Definition 5.1 (also compare Remark 5.1 below); denote $A_{-1} = aI$; compute the scalar a and the matrix $\tilde{A}_{-1}^{-1}$ such that

$$a = n \max_{i,j} |a_{ij}|/(1-p), \qquad \tilde{A}_{-1}^{-1} = A_{-1}^{-1} = I/a. \tag{5.5}$$

Stage j, $j = 0, 1, \ldots, m$. Let $A_j = A + a_jI$, $a_j = ap^j$, and apply k iteration steps (4.1) with $W = A_j$ and with $X_0 = \tilde{A}_{j-1}^{-1}$ in order to compute the matrix $X_k = \tilde{A}_j^{-1}$; in this case (see Propositions 4.1 above and 5.1c below),

$$\|\tilde{A}_j^{-1} - A_j^{-1}\|_2 \leq \delta \|A_j^{-1}\|_2 \tag{5.6}$$

if $\delta = q^K$ and $K = 2^k$.

Stage $m + 1$. Apply sufficiently many steps of the iteration (4.1) where $W = A$, $X_0 = \tilde{A}_m^{-1}$ in order to compute and to output an approximation $\tilde{A}^{-1}$ to A^{-1} satisfying the bound (5.1).

The relations of Definition 5.1 imply the following properties:

PROPOSITION 5.1. $\|A_jA_{j-1}^{-1} - I\|_2 \leq 1 - p$ *for all* j.
(b) $\|A\, A_m^{-1} - I\|_2 \leq 1 - p$.
(c) $\|A_j\tilde{A}_j^{-1} - I\|_2 \leq q$.
(d) $\|A\, \tilde{A}_m^{-1} - I\|_2 \leq q$.

Proof. $A_jA_{j-1}^{-1} - I = ap^{j-1}(1-p)A_{j-1}^{-1}$ and $\|A_{j-1}^{-1}\|_2 \leq 1/(ap^{j-1})$ (since A_{j-1} is s.p.d.), and Proposition 5.1a follows. Proposition 5.1b is implied by the relations $A\, A_m^{-1} - I = -ap^mA_m^{-1}$, $(1-p)a \leq n\|A\|_2$, $\|A_m^{-1}\|_2 \leq \|A^{-1}\|_2$,

and by the first inequality of (5.2). To prove Proposition 5.1c, observe that $\|A_j\tilde{A}_{j-1}^{-1} - I\|_2 = \|A_jA_{j-1}^{-1} - I\|_2 + \|A_j(\tilde{A}_{j-1}^{-1} - A_{j-1}^{-1})\|_2$, combine this bound with Proposition 5.1a and with the relations $\|\tilde{A}_{j-1}^{-1} - A_{j-1}^{-1}\|_2 \le \|A_{j-1}^{-1}\|_2\|A_{j-1}\tilde{A}_{j-1}^{-1} - I\|_2$, $\|A_j\|_2 = \|A\|_2 + pa_{j-1}$, and $\|A_{j-1}^{-1}\|_2 \le \min\{1/a_{j-1}, \|A^{-1}\|_2\}$, and deduce that $\|A_j\tilde{A}_{j-1} - I\|_2 \le (1 - p) + \delta(p + \kappa_2) \le q$, by using the relations (5.6) and then (5.2). Actually, we deduce Proposition 5.1c and the bound (5.6) together, by induction on j. Then we similarly deduce Proposition 5.1d. ■

COROLLARY 5.1. *A matrix $\tilde{A}^{-1}$ satisfying* (5.1) *for $\varepsilon \ge \delta = p(1 - p)/(2(p + \kappa_2))$ can be computed by using $m + 1 \sim \log(n\kappa_2)/\log(1/p)$ stages of Algorithm 5.1, each stage involving $k \sim \log(\log \kappa_2/\log(1/(1 - p)))$ Newton's steps. $\lceil \log(\log \varepsilon/\log \delta) \rceil$ additional Newton's steps at Stage $m + 1$ suffice if $\varepsilon < \delta$.*

Remark 5.1. It is, of course, desirable to choose smaller natural values k and m satisfying (5.2), and thus we need a sharper upper bound on κ_2. We may recall that

$$\kappa_2 \le \kappa_2^+ = n\|X_k\|_1\|W\|_1/(1 - q(k)) = O(\kappa_2),$$
$$q(k) = \sqrt{n}\|I - WX_k\|_1 \ge \|I - WX_k\|_2, \quad (5.7)$$

for $W = A_j$ and for all j and k, and set $1/\delta = 2(p + \kappa_2^+)/p(1 - p)$. κ_2^+ is generally a coarse upper bound on κ_2, and we may try a heuristic improvement of the latter formula just by testing the inequalities (c) and (d) of Proposition 5.1 for smaller values k and m.

Remark 5.2. Algorithm 5.1 can be applied to the inversion of any $n \times n$ nonsingular symmetric matrix $A = [a_{ij}]$ if we simply replace the value a in the equations (5.5) by $a = n\sqrt{-1}\max_{i,j}|a_{ij}|/q$ and to the inversion of any diagonally dominant matrix A if we replace the equations (5.5) by setting $a = (1 - q)/q$, $\tilde{A}_{-1}^{-1} = I/(1 + a)$ and if we replace the 2-norms by the p-norms of matrices throughout for $p = 1$ or $p = \infty$. In both cases, Proposition 5.1 and all the results of this section can be easily extended. Furthermore, Algorithm 5.1 can be extended by means of symmetrization in order to invert any nonsingular matrix A (not necessarily s.p.d.), for we may invert the s.p.d. matrix A^TA and then compute $A^{-1} = (A^TA)^{-1}A^T$. The resulting asymptotic upper bounds on the complexity of parallel inversion of general well-conditioned matrices A repeat the estimates of Pan and Reif (1985), but we next improve these estimates in the case where A is a Toeplitz or Toeplitzlike matrix.

Remark 5.3. $d_+(A_j^{-1}) = d_-(A_j) \le d_+(A_j) + 2 \le d_+(A) + 3$ [see (3.5)], but a more careful application of Corollary 3.1 implies that $d_+(A_j^{-1}) \le d_+(A)$ for all j, and similarly, $d_-(A_j^{-1}) \le d_-(A)$ for all j, which justifies

(4.2), since we choose A_j^{-1} as the initial approximations of Newton's iterations of Algorithm 5.1.

6. Modified Newton's Iteration for the Numerical Inversion of Toeplitzlike Matrices

When we apply Algorithm 5.1 to the inversion of a Toeplitz or Toeplitzlike matrix A, the displacement rank of the computed approximations $\tilde{A}_j^{-1}$ to A_j^{-1} and, therefore, the computational cost of step k of Newton's iteration (4.1), grow as k grows. This means a high computational cost of Newton's iteration (4.1) unless it converges already for smaller k, say, for k of the order of log log n. Generally, such a fast convergence of Newton's iteration (4.1) cannot be guaranteed. To solve the problem, we periodically replace Newton's iterates X_k by their approximations having small displacement ranks. We essentially exploit the observation that for larger k, the matrix X_k can be closely approximated by a matrix having a small displacement rank, namely, by A_j^{-1}.

To specify these steps, we again apply Newton's iteration (4.1) to a Toeplitzlike matrix W (in the context of Algorithm 5.1, $W = A_j$ at stage $j \leq m$ and $W = A$ at stage $m + 1$). Now, however, we modify this iteration: having performed its step $k = k(0)$ [for $k(0)$ to be specified below by (7.6) and (7.7)], we then shift to the modified iteration,

$$X_{k+1} = \tilde{X}_k(2I - W\tilde{X}_k), \qquad k = k(0), k(0) + 1, \ldots, \tag{6.1}$$

where the matrix $\tilde{X}_k$ is an approximation to the matrix X_k obtained with its small length displacement generator by means of the following algorithm applied to the matrix $B = X_k$:

Algorithm 6.1.

Input. An $n \times n$ matrix B with its ZZ^T (or its Z^TZ)–displacement generator (G, H) of length r and a natural $d \leq r$.

Output. A ZZ^T (or a Z^TZ)–displacement generator (G_d, H_d) of length d for the $n \times n$ matrix B_d, where $B_d - ZB_dZ^T = G_dH_d^T$ (or $B_d - Z^TB_dZ = G_dH_d^T$, respectively), that minimizes the norm $\|GH^T - G_dH_d^T\|_2$.

Computations. Compute SVD (2.8), (2.9) of the matrix $W = GH^T$ and define the matrices G_d and H_d by (2.10) and (2.11).

Clearly, the output displacement generator satisfies the desired minimization property, due to (2.12), and in the Appendix, we also prove (by using Proposition 3.1) the following bound (being of independent interest too):

PROPOSITION 6.1. *Let B and B_d be defined in Algorithm 6.1. Let C be any $n \times n$ matrix having ZZ^T (or Z^TZ) – displacement rank at most d, and let $\rho = \|B - C\|_2$. Then $\|B_d - C\|_2 \leq (1 + 2n(r - d))\,\rho$.*

Remark 6.1. $B_d = B$ if $\rho = 0$, so that we have an algorithm that computes a ZZ^T – displacement generator of length $d_+(A)$ and a Z^TZ – displacement generator of length $d_-(A)$ for a matrix A given with its longer displacement generators.

To decrease the computational cost of performing Algorithm 6.1, we modify it by first computing a (recursive) QR-factorization of the $n \times r$ matrices G and H (see Golub and van Loan, 1989, pp. 211–213). Then it remains to compute the product $R(G)R^T(H)$ of the triangular factors $R(G)$ of G and $R(H)$ of H, and the SVD of this product defines the SVD of $W = GH^T$.

In the applications in this paper, $r = O(\log n)$ [see (7.1), (7.7), and (7.10) below], so that we may ignore the smaller cost of computing the SVD of $R(G)\,R^T(H)$ (this cost can be estimated by combining the results of Pan (1987) and Bini and Pan (1991)) and deduce the bound

$$(\tilde{t}, \tilde{p}) \sim (6r \log n,\ nr/\log n) \tag{6.2}$$

on the computational cost of performing Algorithm 6.1 after the above modification (the parallel time bound of (6.2) can be decreased for larger r by using the results of Pan (1987)).

7. COMPUTATIONAL COST OF THE MODIFIED NEWTON'S ITERATION

In this section we estimate the computational cost of approximating to A^{-1} by means of the iteration (4.1), (6.1) and assuming that (4.2) holds for the matrix $W = A$, so that A is a well-conditioned matrix,

$$\log \kappa_2 \leq \log \kappa_2^+ = O(\log n) \tag{7.1}$$

(compare Remark 5.1). We start by estimating the complexity of approximating W^{-1} given two well-conditioned matrices W and X_0 satisfying (4.2) and (4.8). Applying the modified iteration (6.1), Algorithm 6.1, and Proposition 3.2, we bound the displacement ranks of $\tilde{X}_k$ and X_{k+1} as follows:

$$d_+(\tilde{X}_k) \leq d + 2, \qquad d(k+1) = d_+(X_{k+1}) \leq 3d + 7, \qquad k = k(0), k(0) + 1, \ldots \tag{7.2}$$

(also compare Remark 4.1).

The bounds (4.3) and (7.1) suggest that we decrease the cost of our computations if we shift to the iteration (6.1) as soon as we can, by choosing smaller $k(0)$. The transition from $Y = X_k$ to $Y = \tilde{X}_k$, however, generally increases the residual norm $\|WY - I\|_2$, and this may hurt the rapid convergence to W^{-1}, unless k is large enough, such that the residual norms

$$q(k) = \|WX_k - I\|_2, \qquad \tilde{q}(k) = \|W\tilde{X}_k - I\|_2 \tag{7.3}$$

are small enough. Below, we specify $k(0)$ such that for all $k \geq k(0)$ we may safely apply the step (6.1), without endangering the fast convergence to W^{-1}.

Proposition 4.1 implies that $q(k+1) \leq (\tilde{q}(k))^2$, and it suffices to ensure that $\tilde{q}(k)$ be sufficiently small, so that

$$q(k+1) \leq \tilde{q}(k)^2 \leq q(k)^{1+\alpha(k)}, \qquad 1 > \alpha(k) \geq \alpha, \tag{7.4}$$

for $q(k)$ and $\tilde{q}(k)$ defined by (7.2) and for a fixed positive constant α. Then $q(k)$ monotone decreases to 0 with the convergence order of at least $1 + \alpha$ as $k \to \infty$; furthermore, $\alpha(k)$ increases to 1 as $k \to \infty$, so that the rate of the convergence of $q(k)$ to 0 ultimately becomes quadratic.

Now, we shall find a smaller k for which (7.4) holds, and shall let it be denoted $k(0)$. To find such a value k, we just need to estimate the values $q(k)$, and we do this by fixing some positive $q(0) < \frac{1}{32}$ and by recursively applying Proposition 4.1. (As an alternate heuristic approach to estimating $q(k)$, we may estimate the sum of the singular values of the matrices $R_s - ZR_sZ^T$, where $R_s = WX_s - I$, $s = k, k+1$, for the selected candidate values k.)

Next, we estimate such a desired $k = k(0)$, which also furnishes us with an upper bound on the overall computational cost of approximating W^{-1} with a fixed error bound. The value $k(0)$ and the latter cost estimate are defined in terms of an upper bound on the value $\log \kappa$, where κ is the spectral condition number of the matrix W, that is, $\kappa = \kappa_2(W)$, and we assume that $\log \kappa = O(\log n)$ [see (7.1)].

We observe that $\|X_k - W^{-1}\|_2 \leq \|W^{-1}\|_2 q(k)$, and Proposition 6.1 [applied to $B = X_k$, $C = W^{-1}$, $B_d = \tilde{X}_k$ with $r = d(k)$ and with d replaced by $d + 2 \leq d_+(W^{-1})$] implies that $\|\tilde{X}_k - W^{-1}\|_2 \leq \|W^{-1}\|_2 q(k)(1 + 2n(d(k) - d - 2))$. We combine this bound with the relations $\tilde{q}(k) = \|W\tilde{X}_k - I\|_2 \leq \|W\|_2 \|\tilde{X}_k - W^{-1}\|_2$, substitute $\kappa = \|W\|_2 \|W^{-1}\|_2$, and obtain that $\tilde{q}(k) \leq \kappa q(k)(1 + 2n(d(k) - d - 2))$. It follows from this bound and from the equations (6.1) that

$$q(k+1) \leq (\tilde{q}(k))^2 \leq (\kappa q(k)(1 + 2n(d(k) - d - 2)))^2.$$

Therefore, the bound (7.4) holds if

$$((1 + 2n(d(k) - d - 2))\kappa)^2 \le (1/q(k))^{1-\alpha}. \tag{7.5}$$

Thus, we just need to satisfy the inequality (7.5) for $k = k(0)$ in order to satisfy the bound (7.4) for all k (provided that α is fixed between 0 and 1).

The inequality (4.3) implies that $d(k) - d - 2 = (2^k - 1)(2d + 5)$ for $k = 0, 1, \ldots, k(0)$, and Proposition 4.1 implies that

$$q(k(0)) \le (q(0))^{K(0)} \qquad \text{for } K(0) = 2^{k(0)}.$$

Substitute these estimates for $q(k)$ and $d(k)$ into the inequality (7.5) for $k = k(0)$, take logarithms on both sides, and obtain that it suffices if $k(0)$ satisfies the following inequality:

$$(1 - \alpha)2^{k(0)} \ge 2h \log((1 + 2n2^{k(0)+1}(2d + 5))\kappa) \tag{7.6}$$

for h of (4.8) and for a fixed constant α between 0 and 1 (to be specific, we let $\alpha = \frac{1}{5}$). We choose a value $k(0)$ satisfying the latter inequality but otherwise as small as possible, so that

$$k(0) \sim \log\log(n\kappa),\ d(k(0)) \sim 0.5(2d + 5)\log(nd\kappa), \tag{7.7}$$

and we may bound $k(0)$ and $d(k(0))$ by applying our upper estimates for $\log\log(n\kappa)$ and $\log(nd\kappa)$ (obtained, for instance, by applying the relations (5.7) with κ_2 replaced by κ). In particular, (7.7) implies that $d(k(0)) \sim 4.5\log(n\kappa)$ for $d = 2$, and that under (4.2) and (7.1), $k(0) = O(\log\log n)$, $d(k(0)) = O(\log n)$.

We also explicitly estimate the number k_ε^* of steps (6.1) sufficient in order to ensure that

$$\|\tilde{X}_k - W^{-1}\|_2 \le \varepsilon\|W^{-1}\|_2 \qquad \text{for } k = k(0) + k_\varepsilon^* \tag{7.8}$$

and for a fixed positive ε (tolerance to the output errors).

Let $\gamma = \varepsilon/(1 + 2n(2d + 5))$, apply the inequality (7.4) with $\alpha = \frac{1}{5}$, and deduce that a total of less than $k_\varepsilon^* = \max\{0, \lceil \log_{1.2}(\log(1/\gamma)/\log(1/q(k(0))))\rceil\}$ steps (6.1) suffice in order to ensure that

$$\|X_k - W^{-1}\|_2 \le \gamma\|W^{-1}\|_2 \qquad \text{for } k = k(0) + k_\varepsilon^*.$$

Now again, apply Proposition 6.1 to $B = X_k$, $C = W^{-1}$, $B_d = \tilde{X}_k$ with $r = 3d + 7$ and with d replaced by $d + 2$ and deduce the bound (7.8).

To estimate $1/q(k(0))$, apply the inequality (7.5) for $k = k(0)$ and $\alpha = \frac{1}{5}$, that is,

$$1/q(k(0)) \geq ((1 + 2n(d(k(0)) - d - 2)\kappa)^{2.5}; \tag{7.9}$$

this completes estimating

$$\begin{aligned} k_\varepsilon^* &= \max\left\{0, \left[\log_{1.2}\left(\frac{\log(1/\gamma)}{\log(1/q(k(0)))}\right)\right]\right\} \\ &\sim \max\left\{0, \log_{1.2}\left(\frac{\log(n/\varepsilon)}{\log(nd\kappa)}\right)\right\}, \end{aligned} \tag{7.10}$$

so that $k_\varepsilon^* = O(\log(\log(n/\varepsilon)/\log n))$ under (4.2) and (7.1).

Summarizing the above relations and applying (6.2) and Proposition 3.2, we deduce the following bounds:

PROPOSITION 7.1. *Given a well-conditioned matrix W and an initial approximation to W^{-1} satisfying* (4.2) *and* (4.8), *modified Newton's iteration* (4.1), (6.1) *approximates W^{-1} within the error bound* (7.8) *for the parallel cost of its four stages given by*

$$(t_0, p_0) \leq (k(0)t(k(0)), p(k(0)) = O_A(\log n \log\log n, n \log^2 n)$$

[see (4.4), (7.6), (7.7)],

$$(t_\varepsilon, p_\varepsilon) \leq (3k_\varepsilon^* \log(32dn^2(d + 2)^3, 8n(d + 2)^2) = O_A(k_\varepsilon^* \log n, n)$$

[see (4.4), (7.2), (7.10)], so that $(t_\varepsilon, p_\varepsilon) = O_A(\log^2 n, n)$ if $\log\log(1/\varepsilon) = O_A(\log\log n)$,

$$(\tilde{t}_0, \tilde{p}_0) \sim (6d(k(0)) \log n, nd(k(0))/\log n) = O_A(\log^2 n, n)$$

[see (6.2), (7.7)],

$$(\tilde{t}_\varepsilon, \tilde{p}_\varepsilon) \sim (6k_\varepsilon^*(3d + 7) \log n, (3d + 7)n/\log n)$$

[see (6.2), (7.2)], so that $(t_\varepsilon, p_\varepsilon) = O_A(\log^2 n, n/\log n)$ *if* $\log\log(1/\varepsilon) = O(\log n)$. *Here (t_0, p_0) and $(\tilde{t}_0, \tilde{p}_0)$ denote the parallel cost of the first $k(0)$ steps* (4.1) *and of the transition from $X_{k(0)}$ to $\tilde{X}_{k(0)}$, respectively, whereas $(t_\varepsilon, p_\varepsilon)$ denotes the parallel cost of the further steps* (6.1) *excluding the cost $(\tilde{t}_\varepsilon, \tilde{p}_\varepsilon)$ of the transition from X_k to $\tilde{X}_k$ in these steps.*

COROLLARY 7.1. *The overall cost bounds of approximating W^{-1} under the assumptions of Proposition 7.1 are given by $O_A(\log^2 n, n \log n \log\log n)$ provided that* $\log\log(1/\varepsilon) = O(\log n)$.

Now, we combine the latter estimates with the results of Section 5.

Given an s.p.d. matrix A with its displacement generator of length d, we apply Algorithm 5.1, but replace the iteration (4.1) by the modified Newton iteration (4.1), (6.1) (with ε replaced by a value δ satisfying the bound (5.2) at each stage j of Algorithm 5.1 for $j = 0, 1, \ldots, m$ and with ε denoting the tolerance to the output error at stage $m + 1$ of Algorithm 5.1), so that the value $q(0)$ is identified with the value q of Section 5, and the value h of the equation (4.8) is proportional to the value $1/p$ of Section 5. Equations (5.2) and (7.1) imply that $O(\log n)$ applications of the modified Newton iteration will suffice.

COROLLARY 7.2. *Let a well-conditioned $n \times n$ matrix A be given with its displacement generator of length d. Let $\varepsilon = 2^{-b}$ be a positive constant, $\log b = O(\log n)$ as $n \to \infty$. Then an approximation to A^{-1} by a matrix $\tilde{A}^{-1}$ satisfying the inequality* (5.1) *can be computed for the cost $O_A(\log^3 n, n \log n \log\log n)$.*

Remark 7.1. The reader may specify the constants hidden in the "O_A" notation of Corollary 7.1, by using the estimates of Proposition 7.1; moreover, we believe that these estimates and constants can be further improved, in particular, by means of improving Algorithm 6.1 [compare also the tentative heuristic refinement indicated in Remark 5.1 and in the comments following (7.4)].

Remark 7.2. Our approach can be applied by computing the pseudo-inverse (the Moore–Penrose generalized inverse) $A^+ = (A^TA)^{-1}A^T$ of a full rank Toeplitz or Toeplitzlike matrix A either by means of the inversion of the matrix A^TA or, more directly, by using Newton's iteration (4.1), (6.1), which converges to A^+ if it converges at all. This also gives the least-squares solution of linear systems $A\mathbf{x} = \mathbf{b}$ for Toeplitz or Toeplitzlike matrices A. For a Toeplitz matrix A having full column rank, a displacement generator of length 4 for A^+ is shown in Kailath and Chun (1989).

APPENDIX: PROOF OF PROPOSITION 6.1

For brevity, we only consider the Z^TZ-case. To estimate the norm $\|B_d - C\|_2$, let us [similarly to the equations (3.4) for $S = B$ and for d replaced by r] denote that

$$B = L_-(G, H) = \sum_{i=1}^{r} L^T(\mathbf{g}_i)L(\mathbf{h}_i), \tag{A.1}$$

$$B_d = L_-(G_d, H_d) = \sum_{i=1}^{d} L^T(\mathbf{g}_i)L(\mathbf{h}_i), \tag{A.2}$$

where $\mathbf{g}_i$ and $\mathbf{h}_i$ denote columns i of the matrices G and H, respectively, for $i = 1, \ldots, r$, which are also columns i of the matrices G_d and H_d, respectively, for $i = 1, \ldots, d$ [compare Eqs. (2.10) and (2.11)]. Equations (2.1) and (2.8)–(2.9) imply that

$$\|\mathbf{g}_i\|_2 = \|\mathbf{h}_i\|_2 = \sigma_i \qquad \text{for } i = 1, \ldots, r. \tag{A.3}$$

It follows from Eqs. (A.1) and (A.2) that

$$\|B - B_d\|_2 = \| \sum_{i=d+1}^{r} L^{\mathrm{T}}(\mathbf{g}_i) L(\mathbf{h}_i) \|_2 \le \sum_{i=d+1}^{r} \|L^{\mathrm{T}}(\mathbf{g}_i) L(\mathbf{h}_i)\|_2 .$$

Apply Eqs. (A.3) and deduce that

$$\|B - B_d\|_2 \le n \sum_{i=d+1}^{r} \sigma_i^2 . \tag{A.4}$$

Apply Eq. (3.2) for $S = B$ and $S = C$ to define the matrices $F_-(B)$ and $F_-(C)$ and deduce that

$$\|F_-(B) - F_-(C)\|_2 \le 2\rho. \tag{A.5}$$

Let $\sigma_i^2(B)$ and $\sigma_i^2(C)$ denote the ith singular values of the matrices $F_-(B)$ and $F_-(C)$, respectively, so that $\sigma_1(B) \ge \sigma_2(B) \ge \cdots \ge \sigma_d(B) \ge 0$, $\sigma_1(C) \ge \cdots \ge \sigma_d(C) \ge 0$, and

$$\sigma_i(C) = 0 \qquad \text{if } i > d. \tag{A.6}$$

Apply the well-known estimate (see Golub and van Loan, 1989, p. 428) and deduce that

$$|\sigma_i^2(B) - \sigma_i^2(C)| \le \|F_-(B) - F_-(C)\|_2 \qquad \text{for all } i.$$

Substitute the relations (A.5) and (A.6) and obtain that $\sigma_i^2(B) \le 2\rho$ if $i > d$, so that $\sum_{i=d+1}^{r} \sigma_i^2(B) \le 2(r - d)\rho$. Substitute this bound into the inequality (A.4) and deduce that

$$\|C - B_d\|_2 \le \|C - B\|_2 + \|B - B_d\|_2 \le (1 + 2n(r - d))\rho.$$

Acknowledgments

I thank Thomas Kailath for his suggestion of using the SVDs of the generators, the referee for several helpful comments, and also Sally Goodall and Joan Bentley for typing this paper.

References

AHO, A. V., HOPCROFT, J. E., AND ULLMAN, J. D. (1976), "The Design and Analysis of Computer Algorithms," Addison-Wesley, Reading, MA.

AMMAR, G. S., AND GRAGG, W. G. (1988), Superfast solution of real positive definite Toeplitz systems, *SIAM J. Matrix Anal. Appl.* **9**(1), 61–76.

BINI, D., AND PAN, V. (1991), Parallel complexity of tridiagonal symmetric eigenvalue problem, *in* "Proc. 2nd Ann. ACM-SIAM Symp. on Discr. Algor.," pp. 384–393.

BITMEAD, R. R., AND ANDERSON, B. D. O. (1980), Asymptotically fast solution of Toeplitz and related systems of linear equations, *Linear Algebra Appl.* **34**, 103–116.

BORODIN, A., VON ZUR GATHEN, J., AND HOPCROFT, J. (1982), Fast parallel matrix and GCD computation, *Inform. and Control* **52**(3), 241–256.

BRENT, R. P. (1974), The parallel evaluation of general arithmetic expressions, *J. ACM*, **21**(2), 201–206.

BRENT, R. P., GUSTAVSON, F. G., AND YUN, D. Y. Y. (1980), Fast solution of Toeplitz systems of equations and computation of Padé approximations, *J. Algorithms* **1**, 259–295.

BUNCH, J. R. (1985), Stability of methods for solving Toeplitz systems of equations, *SIAM J. Sci. Statist. Comput.* **6**(2), 349–364.

BUNCH, J. R. (1987), The weak and strong stability of algorithms in numerical linear algebra, *Linear Algebra Appl.* **88/89**, 49–66.

CHUN, J., AND KAILATH, T. (1991), Divide-and-conquer solution of least-squares problems for matrices with displacement structure, *SIAM J. Matrix Anal. Appl.* **12**, 128–145.

CHUN, J., KAILATH, T., AND LEV-ARI, H. (1987), Fast parallel algorithm for QR-factorization of structured matrices, *SIAM J. Sci. Statist. Comput.* **8**(6), 899–913.

CSANKY, L. (1976), Fast parallel matrix inversion algorithms, *SIAM J. Comput.* **5**(4), 618–623.

DEHOOG, F. R. (1987), On the solution of Toeplitz systems, *Linear Algebra Appl.* **88/89**, 123–138.

GOHBERG, I., KAILATH, T., AND KOLTRACHT, I. (1986), Efficient solution of linear systems of equations with recursive structure, *Linear Algebra Appl.* **80**, 81–113.

GOLUB, G. H., AND VAN LOAN, C. F. (1989), "Matrix Computations," Johns Hopkins Univ. Press, Baltimore, MD.

KAILATH, T. (1987), Signal processing applications of some moment problems, *in* "Proc. AMS Symp. in Applied Math.," Vol. 37, pp. 71–100.

KAILATH, T., AND CHUN, J. (1989), Generalized Gohberg–Semencul formulas for matrix inversion, *Oper. Theory Adv. Appl.* **40**, 231–246.

KAILATH, T., KUNG, S.-Y., AND MORF, M. (1979), Displacement ranks of matrices and linear equations, *J. Math. Anal. Appl.* **68**, 395–407.

KAILATH, T., VIERA, A., AND MORF, M. (1978), Inverses of Toeplitz operators, innovations, and orthogonal polynomials, *SIAM Rev.* **20**(1), 106–119.

KARP, R., AND RAMACHANDRAN, V. (1990), A survey of parallel algorithms for shared memory machines, *in* "Handbook of Theoretical Computer Science," pp. 869–941, North-Holland, Amsterdam.

MUSICUS, B. R. (1981), "Levinson and Fast Choleski Algorithms for Toeplitz and Almost Toeplitz Matrices," Internal Report, Lab. of Electronics, M.I.T.

PAN, V. (1987), Complexity of parallel matrix computations, *Theoret. Comput. Sci.* **54**, 65–85.

PAN, V. (1990), On computations with dense structured matrices, *Math. Comp.* **55**(191), 179–190.

PAN, V., AND REIF, J. (1985), Efficient parallel solution of linear systems, *in* "Proc. 17th Ann. ACM Symp. on Theory of Computing," pp. 143–152.

PARLETT, B. N. (1980), "The Symmetric Eigenvalue Problem," Prentice-Hall, Englewood Cliffs, NJ.

PEASE, M. C. III (1968), An adaption of the fast Fourier transform for parallel processing, *Journal of ACM*, **15**(2), 252–264.

QUINN, M. J. (1987), "Designing Efficient Parallel Algorithms for Parallel Computers," McGraw-Hill, NY.

WILKINSON, J. H. (1965), "The Algebraic Eigenvalue Problem," Clarendon Press, Oxford.