Iterative Methods for Toeplitz-like Matrices

Thomas Huckle
Universität Würzburg, Germany
Visitor of SCCM, Stanford

Abstract. In this paper we will give a survey on iterative methods for solving linear equations with Toeplitz matrices. We introduce a new class of Toeplitz matrices for which clustering of eigenvalues and singular values can be proved. We consider optimal (ω)-circulant preconditioners as a generalization of the circulant preconditioner. For positive definite Toeplitz matrices, especially in the real case, there is a hard competition between the fast, superfast, and iterative solvers. Therefore, it is necessary to get optimal implementations of the iterative solver. We will show different ways to get improved preconditioned conjugate gradient algorithms, and compare the number of flops for the three concurrent methods. Furthermore, we show different ways to deal with nearsingular Toeplitz matrices.
Key Words. Toeplitz matrix, Fourier Transform, preconditioned conjugate gradient method
AMS(MOS) Subject Classifications. 65F10,65N06

0. Introduction

We consider linear equations of the form

$$T_n\, x \;=\; b \tag{1}$$

with a Toeplitz matrix

$$T_n \;=\; (t_{i-j})_{i,j=1}^{n} \;=\; \begin{pmatrix} t_0 & t_{-1} & \cdots & \cdots & t_{1-n} \\ t_1 & t_0 & \ddots & & t_{2-n} \\ \vdots & \ddots & \ddots & t_{-1} & \vdots \\ t_{n-2} & & t_1 & t_0 & t_{-1} \\ t_{n-1} & \cdots & \cdots & t_1 & t_0 \end{pmatrix} .$$

This work was supported by a research grant from the Deutsche Forschungsgemeinschaft

G. Strang [19] was the first to consider circulant preconditioners for solving (1). Circulant matrices are special Toeplitz matrices given by

$$C_n \;=\; \begin{pmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{pmatrix} \;=\; F_n^H \Lambda_n F_n \;,$$

where

$$F_n \;=\; \Big(exp(-2\pi ijk/n) \Big)_{j,k=0}^{n-1}$$

is the Fourier matrix related to the Discrete Fourier Transform, and Λ_n is the diagonal matrix of eigenvalues of C_n [8]. The eigenvalues in Λ_n can be computed by the Fourier Transform applied on the coefficients $(c_0, ..., c_{n-1})$. Furthermore, every product of the form $T_n x$ can be evaluated by defining a circulant matrix C_{2^m}, $2^m \geq 2n-1$, with first row $(t_0, t_{-1}, ..., t_{1-n}, 0, ..., 0, t_{n-1}, ..., t_1)$ such that T_n is the leading $n \times n$ block of C_{2^m}. If an iterative method is used with a circulant matrix as preconditioner then every matrix-vector product can be computed with Fast Fourier-algorithms in $O(n \log(n))$ flops. Here, flops count the number of multiplications and of additions or subtractions. Then, the total number of arithmetic operations is of order $kn \log(n)$ flops with k the number of iterations in the iterative method.

If the given Toeplitz matrix T_n is a leading principal submatrix of an infinite Toeplitz matrix T related to an l_1-sequence $(t_j)_{j=-\infty}^{\infty}$ with

$$\sum_{j=-\infty}^{\infty} |t_j| \leq M < \infty \;, \tag{2}$$

then there can be defined different circulant approximations C_n such that the spectrum of the preconditioned linear system clusters about 1 asymptotically. Therefore k, the number of iterations, will be small in this case. The most important circulant preconditioners are the original preconditioner C_S of Strang, for even n given by the first row [19,5]

$$(\, t_0 \quad t_{-1} \quad \cdots \quad t_{-n/2} \quad t_{n/2-1} \quad \cdots \quad t_1 \,) \;,$$

and the optimal Frobeniusnorm approximation C_F of T. Chan [7,4], that minimizes

$$\|C_n - T_n\|_F \;=\; \|\Lambda_n - B_n\|_F$$

with $C_n = F_n^H \Lambda_n F_n$ and $B_n = F_n T_n F_n^H$. Especially, if T is positive definite and bounded then T_n and C_F are uniformly positive definite and regular, and the spectrum of $C_F - T_n$ clusters about 0 for large n. Therefore, the spectrum of $C_F^{-1} T_n$ clusters about 1.

The complexity of such an iterative method should always be compared with the complexity of the $O(n^2)$ fast Levinson-type solvers and the $O(n \log(n)^2)$ superfast solvers [1,2]. Hence, iterative methods have to be implemented and used very carefully. They will be faster than the fast and/or superfast methods only for a certain range of size n and iteration steps k.

In section 1 we introduce a new class of Toeplitz matrices with clustered spectrum of the preconditioned linear system. This leads to easy generalizations for near-Toeplitz matrices. In section 2 we consider optimal (ω)-circulant preconditioners. In section 3 we discuss different ways to reduce the number of FFT's in the iterative scheme. In section 4 we compare the number of flops for iterative methods and for fast and superfast solvers. In section 5 we give an short overview over different preconditioners for solving nearsingular Toeplitz systems.

1. Frobeniusnorm and Clustering

Instead of the l_1-condition let us introduce the family of Toeplitz matrices that satisfy

$$\sum_{j=-\infty}^{\infty} |j t_j^2| \;\leq\; M \;< \infty \,. \tag{3}$$

In many examples the coefficients of T_n satisfiy (3) and the l_1-condition, but in general (3) and (2) define different classes. If we consider the preconditoner C_S we get for the eigenvalues of $T_n - C_S$ the inequality [15]

$$\sum_{m=1}^{n} |\lambda_m(T_n - C_S)|^2 \;\leq\; \|T_n - C_S\|_F^2 \;\leq\; \sum_{j=-\infty}^{\infty} |j(t_j - t_{n-j})^2| \;\leq\; 2 \sum_{j=-\infty}^{\infty} |j t_j^2| \;\leq\; 2M \,.$$

Hence, the spectrum of $C_S - T_n$ is clustered about 0, and for uniformly bounded and nonsingular C_S the spectrum of the preconditioned system $C_S^{-1}T_n$ will cluster about 1 asymptotically if (3) is satisfied. To estimate the singular values we consider

$$\|C_S^H C_S - T_n^H T_n\|_F \;=\; \|(C_S^H - T_n^H)C_S + T_n^H(C_S - T_n)\|_F \;\leq\; (\|C_S\|_2 + \|T_n\|_2)\|C_S - T_n\|_F \;.$$

Thus, the sigular values of the preconditioned system are also clustered about 1 if T_n and C_S are uniformly bounded and nonsingular. The same results hold for the optimal circulant matrix C_F.

As a first advantage of this approach, the clustering of the eigenvalues can be shown also for nonsymmetric Toeplitz matrices. Furthermore, using (3) and the Frobeniusnorm it is easy to generalize the clustering property to all products and sums of Toeplitz matrices. For example, for $A = T_1T_2T_3$ with Toeplitz matrices T_j, $j = 1, 2, 3$, with (3), and related circulant preconditioners C_j, $j = 1, 2, 3$, the circulant preconditioner $C = C_1C_2C_3$ gives immidiately that $\|T_1T_2T_3 - C\|_F$ is bounded, and therefore the eigenvalues cluster about 0. Hence, the whole theory developed for Toeplitz matrices can be generalized to products and sums of Toeplitz matrices as long as (3) is satisfied. An important example are matrices of low displacement rank which can be defined as matrices of the form $A = L_1U_1 + ... + L_kU_k$ with upper and lower Toeplitz matrices L and U [13].

2. Optimal (ω)-circulant Preconditioners

Connected with circulant matrices is the class of skewcirculant matrices

$$S_n \;=\; \begin{pmatrix} s_0 & s_{n-1} & \cdots & s_2 & s_1 \\ -s_1 & s_0 & s_{n-1} & \cdots & s_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & s_{n-1} \\ -s_{n-1} & -s_{n-2} & \cdots & -s_1 & s_0 \end{pmatrix} ,$$

that can be reduced to the circulant case by $\bar{\Omega}S_n\Omega \;=\; C_n$ with $\Omega = diag(1, exp(\pi i/n), ..., ..., exp(\pi i(n-1)/n))$. Skewcirculant matrices are also Toeplitz matrices, and every circulant preconditioner yields a related skewcirculant preconditioner [15]. In some cases C_F will

lead to a faster convergence, in other cases S_F, where S_F minimizes $\|S_F - T_n\|_F$. Hence, we are looking for criteria to choose C_F or S_F as preconditioner.

Let us formulate the whole problem in a more general setting. Define the unitary diagonal matrix $D = diag(1, \omega, ..., \omega^{n-1})$ for a given number $\omega = exp(i\phi)$. Then, for a Toeplitz matrix T_n also $DT_n\bar{D}$ is Toeplitz. Now, we can consider the class of (ω)-circulant matrices [8] defined by $C(\omega) = DC_n\bar{D}$ for circulant matrices C_n. Thus, with ϕ we have an additional degree of freedom. Note, that for $\phi = 0$ we get the circulant matrices and for $\phi = \pi/n$ we get the skewcirculant matrices as special cases. The problem of computing the optimal (ω)-circulant preconditioner can now be written in the form

$$\min \|D(\omega)C_n\bar{D}(\omega) - T_n\|_F = \min \|C_n - \bar{D}T_nD\|_F = \min \|C_n - T_n(\omega)\|_F \ .$$

For fixed ω we have to replace t_j in T_n by $t_j\omega^j$ to get $T_n(\omega)$, and the resulting minimization problem has an unique circulant solution C_n with first row

$$\frac{1}{n} \ (\ nt_0 \quad (n-1)t_{-1}\omega^{-1} + t_{n-1}\omega^{n-1} \quad \cdots \quad \cdots \quad t_{1-n}\omega^{1-n} + (n-1)t_1\omega^1 \ ) \ .$$

In a second step we can compute ω that minimizes $\|C_n - \bar{D}T_nD\|_F$, where D and C_n are functions of ω only. The matrix $C_n - \bar{D}T_nD$ has the first row

$$\frac{1}{n} \ (\ 0 \quad t_{-1}\omega^{-1} - t_{n-1}\omega^{n-1} \quad \cdots \quad \cdots \quad t_{1-n}\omega^{1-n} - t_1\omega \ ) \ .$$

Hence, we have to minimize

$$\sum_{j=1}^{n-1}(n-j)\Big(|t_{-j}\omega^{-j} - t_{n-j}\omega^{n-j}|^2 + |t_j\omega^j - t_{j-n}\omega^{j-n}|^2\Big) \ =$$

$$= \sum_{j=1-n, j\neq 0}^{n-1} |t_j|^2 - 2 \, Real\Big[\Big(\sum_{j=1}^{n-1} \bar{t}_{-j}t_{n-j}\Big)\omega^n\Big] \ .$$

To minimize this last expression we have to set ω such that $\omega = exp(i\phi/n)$ with

$$\phi \ = \ arg\Big(\sum_{j=1}^{n-1} \bar{t}_j t_{j-n}\Big) \ . \tag{4}$$

If T_n is a real matrix the preconditioner is supposed to be also real. Thus, we consider only circulant or skewcirculant matrices. In view of (4), we have to prefer the skewcirculant preconditioner if $\sum_{j=1}^{n-1} t_j t_{j-n}$ is negative, otherwise the circulant approximation.

For the real case some special comments are necessary. First, we always want to use FFT of real data, because this can be done faster. The FFT of a real vector leads to a complex conjugate even vector,

$$x \in \mathbf{R}^n \to F_n x = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \bar{y}_3 \\ \bar{y}_2 \end{pmatrix} = y \text{ complex conjugate even .}$$

The Fast Fourier Transform for such vectors can be evaluated with the same complexity as in the real case (or can be reduced to the real case) [22]. We can compute $F_n y$ by using one real FFT in the form

$$a = F_n(real(y) + imag(y)) \ , \ F_n y = real(a) - imag(a) \in \mathbf{R}^n \ . \tag{5}$$

For skewcirculant matrices we have to deal with products of the form $F\bar{\Omega}x$ with real x. First note, that for real x this Fourier Transform gives a quarter wave even vector [22]

$$F_n \bar{\Omega} x \ = \ \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ \bar{y}_2 \\ \bar{y}_1 \end{pmatrix} \ .$$

We want to compute y by using real FFT. This can be done by evaluating

$$z = F_n \, real(\bar{\Omega} x) \ \text{ and } \ y_1 = \sum_{j=1}^{n} exp(\pi i (j-1)/n) x_j \ . \tag{6}$$

From (6), we get the wanted solution y by the recursion $y_{j+1} = 2z_{j+1} - y_j$ for $j = 2, 3, ..., (n+1)/2$. On the other side, if we have given a quarter wave even vector y, then $z = \bar{\Omega} F_n y$ is real and can be computed by

$$a = \bar{\Omega} F_n(real(y) + imag(y)) \ , \ \bar{\Omega} F_n y = real(a) - imag(a) \ . \tag{7}$$

3. Iterative Solution with Minimal Number of FFT's

Any circulant preconditioner C_n for T_n is nothing else than a diagonal preconditioner for $B_n = F_n T_n F_n^H$. We only have to replace the original equations $T_n x = b$ with preconditioner $C_n = F_n^H \Lambda_n F_n$ by $B_n(F_n x) = (F_n b)$ with preconditioner Λ_n. Now, the matrix B_n can be characterized in different ways. First, $T_n = C + S$ can be written as the sum of a circulant and a skewcirculant matrix C and S. Hence, we get

$$B_n = F_n T_n F_n^H = F_n(C+S)F_n^H = \Lambda_C + (F_n \Omega F_n^H)\Lambda_S(F_n \bar{\Omega} F_n^H) = \Lambda_C + C_u \Lambda_S C_u^H \quad (8a)$$

with the unitary circulant matrix $C_u = F_n \Omega F_n^H$, and Λ_C, Λ_S the eigenvalue matrices of C and S. A similar representation can be found in [20]. The first column of C_u is given by

$$\frac{2}{n} \quad \frac{1}{1 - exp(\pi i(1-2j)/n)} \quad \text{for } j = 0, 1, ..., n-1 \ .$$

B_n can be also written using a Hermitian circulant matrix C_H in the form [16]

$$B_n \ = \ \Lambda_F \ + \ diag(a-d) * C_H \ + \ C_H * diag(d-a) \quad (8b)$$

with

$$a = \sqrt{n} F_n \begin{pmatrix} 0 \\ t_1 \\ \vdots \\ t_{n-1} \end{pmatrix} \quad , \quad d = \sqrt{n} F_n^H \begin{pmatrix} 0 \\ t_{-1} \\ \vdots \\ t_{1-n} \end{pmatrix} \ ,$$

and

$$C_H e_0 = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} 0 \\ \frac{1}{1-exp(-2\pi i/n)} \\ \vdots \\ \frac{1}{1-exp(-2\pi i(1-n)/n)} \end{pmatrix} \ .$$

The eigenvalues of C_H are given by

$$\lambda_0 = \frac{1-n}{2} \ \text{ and } \ \lambda_j = \frac{2j-n-1}{2n} \quad \text{for } j = 1, ..., n-1 \ .$$

In (8) the matrix-vector product with B_n takes the same number of flops as the matrix-vector product with T_n itself (4 FFT's of size n against 2 FFT's of size $2n$). But if we solve the linear system in B_n the preconditioner is a diagonal matrix and needs no more

FFT-calls. Using B_n, the costs of the iterative solver are mainly 4 FFT-calls of size n per iteration. In the symmetric or Hermitian case the computing of the representation (8b) takes one FFT-call less than (8a). Note, that Λ_F are the eigenvalues of the preconditioner, and that $d = conj(a)$ in the Hermitian or symmetric case.

Solving the system in B_n in this form is optimal if n is a power of 2. For general n the Fast Fourier Transform may be very slow, and therefore we prefer a formulation where we can extend all the occuring matrices to a power of 2. We can reduce the use of FFT's of size n if we apply the iterative method to the original system (1). In this case, we have to compute the eigenvalues of the circulant preconditioner and the entries of C_F^{-1} with 2 FFT-calls of size n. Then we can replace C_F^{-1} and T_n by circulant matrices of the size 2^m. In each iteration we use only fast FFT's of size 2^m.

Thus, for each n one should decide in advance whether the Fourier Transform of size n or of size 2^m is faster. In the first case, the formulation (8) will be better, in the second case one should use (1) with the extensions of C_F^{-1} and T_n to a power of 2. For n with very slow Fourier Transform, one should better solve (1) with fast Levinson-type methods.

In the real case, the optimal implementations of the DFT take $2.5\, n \log(n)$ flops [2,22]. In this case, the new linear system in B_n gets complex, but all appearing complex vectors $F_n b$, $diag(\Lambda_C)$, $diag(\Lambda_S)$, $diag(\Omega)$, $diag(\Lambda_F)$, a, and d will be complex conjugate even or square wave even. From (5-7) we now how to reduce the Fourier Transform of such vectors to the real case. Hence, the costs per iteration step in the real case are given by $10\, n \log(n)$ flops for solving (1) with B_n, n a power of 2, or $10\, m 2^m$ flops with the extended circulant preconditioner. In the complex case, the optimal implementations of the DFT take $5\, n \log(n)$ real flops. The costs per iteration are $20\, n \log(n)$ or $20\, m 2^m$ flops [1,22]. The costs in the beginning and end are 5 FFT's of size n for (8) in the general case, 4 FFT's of size n for (8b) in the symmetric or Hermitian case, or 2 FFT's of size n for (1).

For unsymmetric Krylov methods there is a further way to reduce the number of FFT-calls [12]. For example, the Lanczos iteration, can be written with the recursions

$$v_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1} \ , \tag{9}$$

$$w_{j+1} = A^T w_j - \alpha_j w_j - \beta_j w_{j-1} \ . \tag{10}$$

Now, Toeplitz matrices are persymmetric. With the counteridentity J, $J(j,k) = \delta_{j+k-1,n}$, they satisfy

$$T_n^T J = J T_n \ .$$

If the Lanczos iteration starts with vectors v_0 and $w_0 = Jv_0$, then the second recursion generates $w_j = Jv_j$. Hence, we can recover w_j from v_j by a multiplication with J. Obviously, this is true not only for the special matrix J but for any other general P. Hence, if a matrix satisfies $A^T P = PA$, then for w_{j+1} instead of $A^T w_j$ we can evaluate Pv_{j+1}, which may be much cheaper.

Here, we are interested in preconditioned and transformed Toeplitz systems. First, let us consider the preconditioned system. Then, A is not Toeplitz, but e.g. $A = C_n T_n$. We have to set $P = JT_n$ which gives $A^T P = T_n^T C_n^T J T_n = J T_n C_n T_n = PA$. Instead of products with T_n^T and C_n^T we only need $JT_n v_j$ in the second recursion, and $T_n v_j$ has to be computed also in (9) for v_{j+1}. Hence, we need no further FFT-calls for (10) and the number of flops reduces by a factor 2.

If we consider the system where the circulant preconditioner is written as a diagonal matrix then $A = \Lambda B_n$. We can define

$$P = F_n^H J T_n F_n^H = (F_n^H J F_n^H) B_n = \Delta B_n = \Lambda^{-1}\Delta(\Lambda B_n)$$

with $\Delta = F_n^H J F_n^H = \bar{\Omega}^2$. Hence, also in this case (10) needs no further FFT-calls, and the iterative method takes only 4 FFT's of order n. That reduces for example in the complex case the flops per iteration by a factor 3 from $60\, n \log(n)$ to $20\, n \log(n)$.

These ideas can be combined with the ideas of section 2 for (ω)-circulant preconditioner. In recent papers the Fourier Transform is replaced by the Sine or Cosine Transform to construct preconditioner for Toeplitz systems [3,9]. With

$$S_n = \Big(sin(\pi jk/(n+1))\Big)_{j,k=1}^{n} \ \text{ and } \ C_n = \Big(cos(\pi jk/(n+1))\Big)_{j,k=1}^{n} \tag{11}$$

one considers the class of matrices $A_S = S_n \Lambda S_n$ which can be written as a special class of Toeplitz-plus-Hankel matrices. A_C can be defined in the same way with the cosine function. It is easy to check that the results of this section can be applied to this new preconditioners.

For example, it is possible to write a Toeplitz matrix in the form $T_n = A_S + A_C$, this is equivalent to the circulant/skewcirculant representation of T_n. We can again replace the original system with preconditioner A_S by the matrix $B_n = S_n T_n S_n$ with a diagonal preconditioner. It is straightforward to formulate the transpose-free method for this new linear system, too.

4. Comparison with Fast and Superfast Toeplitz Solver

If we want to solve a linear system (1) then we have three different methods that can be very fast.

- The classical fast $O(n^2)$-methods are very easy to implement and for small dimension always optimal. In the indefinite case there may occur breakdowns in the recursion if leading principal submatrices are singular, but meanwhile there exist look-ahead versions that overcome this difficulties in nearly all cases (see e.g. [11]). Furthermore, fast algorithms do not depend on the factorization of n.
- The superfast $O(n \log(n)^2)$-methods are restricted to the symmetric or Hermitian positive definite case [1,2]. As the Fast Fourier Transform the superfast method of Ammar/Gragg [1,2] depends heavily on the prime factor decomposition of n. For certain n the method can be very slow, optimal behaviour will occur if n is a power of 2. But in this case for example the Ammar/Gragg code is faster than Levinson already for $n \geq 256$. In a recent paper [17] the author introduced a superfast method that is independent of the factorization of n, but slower than the Ammar/Gragg method for n a power of 2. For superfast solvers it is very important to find optimal formulations of the algorithm to reduce the use of FFT's and to use optimal implementations of the Fast Fourier Transform, especially in the real case.
- Iterative method may have difficulties if in the preconditioner some of the eigenvalues are near 0. But in this case, one can replace this small numbers by 1 without changing the convergence too much (Consider herefore that we want to minimize $I - \Lambda_n^{-1} B_n$, and if a diagonal element λ_j is very small than by defining $\lambda_j = 1$ we change the

Frobeniusnorm only slightly). In the computation of the circulant preconditioner there have to be used at least two FFT-calls of exact dimension n. This may be slow for some n. All other FFT-calls can be forced to a power of 2. To improve the overall costs of iterative methods we have to include the results of the proceeding sections. In an implementation of the method it is also necessary to use optimal implementations of the Fast Fourier Transform for the real or complex case.

Now, let us compare the number of flops for fast, superfast and iterative methods in the symmetric or Hermitian positive definite case and for n a power of 2. We consider three different problems. First, we want to compute only one special right hand side, which allows us to build a Cholesky-decomposition or a Gohberg-Semencul formula for T_n^{-1}. Second, we consider the problem of solving (1) with a given general vector b. Third, we consider the problem of solving (1) for a large number of different right hand sides.

In the real case, the Levinson algorithm takes $2n^2$ flops, and with its output we can form a Cholesky-factorization for T_n^{-1} [2]. Therefore, every other linear system (1) can be solved in $2n^2$ additional flops. The superfast algorithm needs $8n\log(n)^2)$ flops in the real case [2]. With the computed special solution we can form a Gohberg-Semencul formula for $T_n^{-1} = L_1L_1^T - L_2L_2^T$ with lower or upper triangular Toeplitz matrices. Hence, every further right hand side needs $2 * 2.5 * 2n\log(n)$ flops for computing circulant extensions for L_1 and L_2 and $8 * 2.5 * 2n\log(n)$ for the matrix-vector products in $T_n^{-1}x$.

The iterative method in an optimal implementation takes $10\,n\log(n)$ flops in every iteration step. The computations at the beginning and at the end of the algorithm for getting the preconditioner and transforming the linear equation, take $4 * 2.5\,n\log(n)$ flops for Λ_F, a, F_nb, and $x = F_n^H y$ (8b). For each size n and number k, we can now compare the costs for the different methods to decide what algorithm should be used. Here, k denotes the number of iterations necessary for solving the linear system iteratively.

(1) Solving (1) for special right hand side: For $k \leq 0.2n/\log(n) - 1$ the iterative method is faster then Levinson, and for $k \leq 0.8\log(n) - 1$ the iterative method is faster than the superfast algorithm. The superfast algorithm is better than the fast algorithm for $n \geq 256$ [2]. Hence, for every n we can derive upper bounds k_1^* and k_2^* for k; if k is larger than this bounds then the iterative method will be too slow compared with the

fast (k_1^*) or superfast (k_2^*) method. The boldface numbers show whether the fast or the superfast method gives the sharper bound for k.

n	32	64	128	256	512	1024	2048	4096	8192	16392	32784	65568
k_1^*	**0**	**1**	**2**	**5**	10	19	36	67	125	233	435	818
k_2^*	3	3	4	**5**	**6**	**7**	**7**	**8**	**9**	**10**	**11**	**11**

Table 1. Allowed iterations compared with fast/superfast algorithm for special b

(2) Solving (1) for a general given right hand side. The fast algorithm gives the bound $k \leq k_1^* = 0.4 * n/\log(n) - 1$ and the superfast method gives $k \leq k_2^* = (0.8\log(n) + 5) - 1$.

n	32	64	128	256	512	1024	2048	4096	8192	16392	32784	65568
k_1^*	**1**	**3**	**6**	11	21	39	73	135	251	467	872	1637
k_2^*	8	8	9	**10**	**11**	**12**	**12**	**13**	**14**	**15**	**16**	**16**

Table 2. Allowed iterations compared with fast/superfast algorithm for general b

(3) For many right hand sides we count the costs only for solving (1) with each b. The Cholesky decomposition gives $k \leq k_1^* = 0.2n/\log(n) - 0.5$ and Gohberg-Semencul $k \leq k_2^* = 4 - 0.5$.

n	32	64	128	256	512	1024	2048	4096	8192	16392	32784	65568
k_1^*	**0**	**1**	**3**	5	10	19	36	67	125	233	436	818
k_2^*	3.5	3.5	3.5	**3.5**	**3.5**	**3.5**	**3.5**	**3.5**	**3.5**	**3.5**	**3.5**	**3.5**

Table 3. Allowed iterations compared with fast/superfast algorithm for many b

For complex Toeplitz system we count a complex addition as two flops and a complex multplication as 6 flops. Then, Levinson takes $(2+6)n^2$ flops, the same as the Cholesky-decompostion-vector product. The superfast algorithm takes $(4*2+2*6)n\log(n)^2$ flops [1], and the evaluation of Gohberg-Semencul $(2+8)*5*2\,n\log(n)$. The iterative method takes $2*5*2n\log(n)$ flops per iteration and $4*5\,n\log(n)$ at the beginning and end.

(1) Solving (1) for special b: For $k \leq k_1^* = 0.4n/\log(n) - 1$ the iterative method is faster then Levinson, and for $k \leq k_2^* = \log(n) - 1$ the iterative method is faster than the superfast algorithm. The superfast algorithm is better than the fast algorithm for

$n \geq 128$.

(2) Solving (1) for a general given right hand side: The fast algorithm gives the bound $k \leq k_1^* = 0.8 * n/\log(n) - 1$ and the superfast method gives $k \leq k_2^* = (\log(n) + 5) - 1$.

(3) For many right hand sides the Cholesky-decomposition gives $k \leq k_1^* = 0.4n/\log(n) - 0.5$ and Gohberg-Semencul $k \leq k_2^* = 4 - 0.5$.

Hence, we see that iterative methods are competetive in the positive definite case only for large n and small k. For many right hand sides the use of the Gohberg-Semencul formula leads to a mininmal number of flops. In the indefinite or nearsingular case iterative methods may give a higher accuracy.

In the unsymmetric case we have to compare only the classical $O(n^2)$ methods with the iterative solver. Here, the number of flops depends on the used iterative method, for example a transpose-free method in the real case will take $10\, n \log(n)$ flops per iteration and a fast Levinson-type method will take $6n^2$ flops.

5. Iterative Methods for Near-singular Toeplitz Systems

In the case of nearsingular Toeplitz matrices circulant preconditioners will again lead to a good approximation to T_n, but now C_F can get also nearsingular and thus the clustering of $C_F - T_n$ does not lead to a clustering of $C_F^{-1} T_n$. There are different circulant and non-circulant preconditioners suggested for this case.

First, if we have no additional information on T_n available, then one can apply Strangs preconditioner in a modified way [21,14], or the Sine-Transform-based approximation [9,10,18].

Now, let us assume that we have further information on the singularity of T_n. If T_n is the leading principal submatrix of the infinite singular Toeplitz matrix T, then each singularity of T is related to a zero of the corresponding function $f(z)$ on the unit disk D. Now, we can try to construct a band Toeplitz matrix T_B that reflects this singularity and gives a good approximation on T_n [18,6]. A band Toeplitz matrix corresponds to a polynomial in z, and to remove the singularity in $T_B^{-1} T_n$ we choose the polynomial such

that it has the same zeros as $f(z)$. Moreover, we get a good approximation on T_n at the same time by considering an optimization problem of the type [6]

$$\min_{p(z)} \max_{z \in D} \frac{|f(z) - p(z)|}{|f(z)|} \ .$$

Here, we will show an easier way find an optimal band preconditioner with given singularity. We consider a similar problem

$$\min \|T_B - T_n\|_F \ \ , \text{ with } p(z_s) = 0 \text{ whenever } f(z_s) = 0 \ . \tag{12}$$

Let us assume that $f(z)$ has only one zero $z_0 = exp(i\phi_0)$ on the unit disk. Then, the minimization problem (12) reads as

$$\min \|T_B - T_n\|_F \ \text{ subject to } \ \sum_{j=l}^{-r} b_j exp(ij\phi_0) = 0 \ .$$

We can solve the constraint condition for b_0, replace b_0, and get

$$\min\Big(n|t_0 + \sum_{j=l, j\neq 0}^{-r} b_j z_0^j|^2 + \sum_{j\neq 0, j=l}^{-r} (n - |j|)|b_j - t_j|^2\Big) \ .$$

This leads to a very simple linear equation for the unknowns $b = (b_{-r}, ..., b_{-1}, b_1, ..., b_l)^T$ in terms of $z = (z_0^r, ..., z_0, z_0^{-1}, ..., z_0^{-l})^T$, $t = (t_{-r}, ..., t_{-1}, t_1, ..., t_r)^T$, and $e = (1, ..., 1)^T$ of the form

$$\Big((n-1)diag(z) + n(ez^H)\Big) \ = \ (n-1)(t. * z) - nt_0 e \ .$$

The underlying matrix is diagonal plus a rank-1 perturbation and can be solved very efficiently for example with the Sherman/Morisson/Woodbury-formula. For every further zero in f we get a further rank-1 term in this linear system.

After a band preconditioner (12) or [6] is found one can define a circulant or Sine Transform-based preconditioner for T_B and use this as preconditioner for T_n [18,12].

A method that in addition can always be applied to improve the convergence of an iterative method is the use of double-preconditioning for Toeplitz matrices introduced in [13]. Here, we want to apply left and right preconditioning at the same time. If we have computed a band or circulant approximation M_l on T_n then the preconditioned equation

$M_l^{-1}T_n$ is a matrix of displacement rank 3. Hence, the theory for circulant preconditioning for matrices with low displacement rank [13] can be applied to find a further (ω)-circulant approximation M_r on $M_l^{-1}T_n$. For band matrix M_l this may be a circulant matrix and for circulant M_l this should be a skewcirculant matrix (otherwise we get the identity matrix for M_r). This second approximation can be used as a right preconditioner. So, we arrive at the linear system

$$\left(M_l^{-1}T_nM_r^{-1}\right)\left(M_rx\right) \;=\; M_lb\;.$$

In the circulant/skewcirculant case we can transform this system again in the form that both preconditioner appear as diagonal matrices like (8) by using the circulant/ skewcirculant representation $T_n = C + S$ and

$$\Lambda_l^{-1}F_nT_n\Omega F_n^H\Lambda_r^{-1} = \Lambda_l^{-1}(\Lambda_C F_n^H\Omega F_n + F_n\bar{\Omega}F_n^H\Lambda_S)\Lambda_r^{-1}\;.$$

Thus, for nearsingular equations double-preconditioning with band or circulant Toeplitz matrices leads to a large reduction in the number of iterations (for numerical examples see [13]).

6. Conclusions

We have shown a method to extend the theory of circulant preconditioning to a wider class of Toeplitz and near-Toeplitz matrices. Furthermore, we have introduced the optimal (ω)-circulant Frobeniusnorm approximation as preconditioner. If an iterative scheme is used for solving (1) then one should use different methods for real and complex data, also dependent on the prime factorization of n. A comparison with the fast and superfast Toeplitz solvers for positive definite matrices shows that the number of iterations in the preconditioned conjugate gradient algorithm has to be small. Otherwise, fast and superfast methods are faster than the iterative method. For nearsingular Toeplitz matrices double-preconditioning with band and circulant matrices removes the singularity, and often leads to a fast convergence. A program package that includes all aspects of iterative Toeplitz solvers is in preparation.

Acknowledgement.

This paper is written while the author was visiting Stanford University. The hospitality of Prof. Gene Golub is gratefully acknowledged. The author is also indebted to Roland Freund for many useful hints and discussions.

References

[1] Ammar,G.S.,Gragg,W.B.: Implementation and Use of the Generalized Schur Algorithm, *Computational and Combinatorial methods in Systems Theory*, Byrnes,C.I., and Lindquist,A., eds., North-Holland, Amsterdam, pp. 265-280, 1986.

[2] Ammar,G.S.,Gragg,W.B.: Superfast solution of real positive definite Toeplitz systems, *SIAM J. Matrix Anal. Appl.*, Vol. 9, pp.61-76, 1988.

[3] Boman,E., Koltracht,I.: Fast Transform Based Preconditioners for Toeplitz Equations, *Preprint*, August 1993.

[4] Chan,R.: The spectrum of a family of circulant preconditioned Toeplitz systems, *SIAM J. Numer. Anal. 26(2)*, 503-506,1989.

[5] Chan,R.,Strang,G.: Toeplitz equations by conjugate gradients with circulant preconditioner, *SIAM J. Sci. Stat. Comput. 10*, 104-119, 1989.

[6] Chan,R., Tang,P.: Fast Band-Toeplitz Preconditioners for Hermitian Toeplitz Systems, *SIAM J.Sci.Comp. 15(1)*, pp. 164-171, 1994

[7] Chan,T.: An optimal circulant preconditioner for Toeplitz systems, *SIAM J. Sci. Stat. Comput. 9(4)*, 766-771, 1988.

[8] Davis,P.J.: Circulant Matrices, *John Wiley, New York*, 1979.

[9] Di Benedetto,F.: Analysis of Preconditioning Techniques for Ill-conditioned Toeplitz Matrices, *Proc. ERCIM Workshop on Numerical Linear Algebra, Pisa, CNR*, pp. 5-10, 1992.

[10] Di Benedetto,F., Fiorentino,G., and Serra,S.: C.G. Preconditioning for Toeplitz Matrices, *Computers Math. Appl.* 25, pp. 35-45, 1993.

[11] Freund,R.W.: A look-ahead Bareiss algorithm for general Toeplitz matrices, *AT&T*

Numerical Analysis Manuscript, Bell Laboratories, Murray Hill,NJ, 1993.

[12] Freund,R.W.: Transpose-free quasi-minimal residual methods for non-Hermitian linear systems, *Recent Advances in Iterative Methods (G. Golub, A. Greenbaum, and M. Luskin, eds.), The IMA Volumes in Mathematics and its Applications*, Vol. 60, pp. 69-94, Springer-Verlag, New York, 1994.

[13] Freund,R.W., Huckle,T.: Iterative Solution of Linear Systems with Low Displacement Rank by Conjugate Gradient-type Algorithms, in preparation. Householder Meeting, Lake Arrowhead, 1993.

[14] Hanke,M., Nagy,J.: Toeplitz Approximate Inverse Preconditioner for Banded Toeplitz matrices, *SMU Math Report 93-7*, Southern Methodist Univ. 1993, submitted to Num. Alg..

[15] Huckle,T.K.: Circulant/Skewcirculant Matrices as Preconditioners for Hermitian Toeplitz Systems, *Proceedings of the IMACS Conference on Iterative Methods in Linear Algebra, Brusseles, April 1991.*

[16] Huckle,T.: Some Aspects of Circulant Preconditioners, *SIAM J. Sci. Stat. Comput.* 14, pp. 531-541, 1993.

[17] Huckle,T.: Superfast Solution of Linear Equations with Low Displacement Rank, *Manuscript SCCM-93-15. Computer Science Department, Stanford University, Stanford, CA*, December 1993.

[18] Serra,S.: Preconditiong Techniques for Ill-conditioned Block Toeplitz Systems with Nonnegative Generating Functions, *Tech. Rep. 15/93, Univ. Pisa*, June 1993.

[19] Strang,G.: A proposal for Toeplitz matrix computations, *Studies in Applied Mathematics 74*, 171-176,1986.

[20] Tismenetsky,M.: A Decomposition of Toeplitz Matrices and Optimal Circulant Preconditioning, in *Linear Algebra & Appl. 154-156*, 105-121, 1991.

[21] Tyrtyshnikov,E.: Circulant Preconditioners with Unbounded Inverses: Why Non-Optimal Preconditioners May Possess a Better Quality Than Optimales Ones, *Colorado Conference on Iterative Methods*, Breckenridge, CO, April 5-9, 1994.

[22] Van Loan,C.: Computational Frameworks for the Fast Fourier Transform, *SIAM Publications*, Phil., 1992.