TRIGONOMETRIC TRANSFORM SPLITTING METHODS FOR REAL SYMMETRIC TOEPLITZ SYSTEMS

ZHONGYUN LIU*, NIANCI WU*, XIAORONG QIN*, AND YULIN ZHANG†

Abstract. In this paper we study efficient iterative methods for real symmetric Toeplitz systems based on the trigonometric transformation splitting (TTS) of the real symmetric Toeplitz matrix A. Theoretical analyses show that if the generating function f of the $n \times n$ Toeplitz matrix A is a real positive even function, then the TTS iterative methods converge to the unique solution of the linear system of equations for sufficient large n. Moreover, we derive an upper bound of the contraction factor of the TTS iteration which is dependent solely on the spectra of the two TTS matrices involved.

Different from the CSCS iterative method in [19] in which all operations counts concern complex operations when the DFTs are employed, even if the Toeplitz matrix A is real and symmetric, our method only involves real arithmetics when the DCTs and DSTs are used. The numerical experiments show that our method works better than CSCS iterative method and much better than the positive definite and skew-symmetric splitting (PSS) iterative method in [3] and the symmetric Gauss-Seidel (SGS) iterative method.

Key words. Sine transform, Cosine transform, Matrix splitting, Iterative methods, Real Toeplitz matrices.

AMS subject classifications. 15A23, 65F10, 65F15.

1. Introduction.

Consider the iterative solution to the following linear system of equations

$$Ax = b \tag{1.1}$$

by the two-step splitting iteration with alternation, where $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ is a symmetric positive definite Toeplitz matrix.

Very often, a Toeplitz matrix A is generated by a function $f(x) \in \mathbf{C}_{2\pi}$, i.e., $[A]_{m,k} = a_{m-k}$, where

$$a_k = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(\theta)e^{-ik\theta}d\theta \ , \ k = 0, \pm 1, \pm 2, \cdots \tag{1.2}$$

the function f is called the generating function of the Toeplitz matrix A.

*School of Mathematics and Statistics, Changsha University of Science and Technology, Changsha 410076, P. R. China (liuzhongyun@263.net).

†Centro de Matemática, Universidade do Minho, 4710-057 Braga, Portugal (zhang@math. uminho.pt).

Toeplitz linear systems arise in a variety of applications in mathematics, scientific computing and engineering, for instance, image restoration storage problems in signal processing, algebraic differential equation, time series and control theory. Those applications have motivated both mathematicians and engineers to develop specific algorithms catering to solving Toeplitz systems for instance, [1, 11, 14] and references therein.

Iterative methods for the linear system of equations (1.1) require efficient splittings of the coefficient matrix A. For example, the Jacobi and the Gauss-Seidel iterations [14, 21] split the matrix A into its diagonal part D, and strictly lower triangular part L and upper triangular part U, respectively, and the generalized conjugate gradient (CG) method and the generalized Lanczos method split the matrix A into its Hermitian and skew-Hermitian parts, see for example [4] and references therein.

Based on the fact that every matrix A naturally has a Hermitian/skew-Hermitian splitting (HSS)

$$A = H + S \quad \text{with} \quad H = \frac{A + A^*}{2} \quad \text{and} \quad S = \frac{A - A^*}{2} \tag{1.3}$$

Bai et al. presented the following HSS iteration in [4] to solve the non-Hermitian positive definite linear system of equations.

The HSS iteration. *Given an initial guess $x^{(0)}$. For $k = 0, 1, 2, \ldots$ until $\{x^{(k)}\}$ converges, compute*

$$\begin{cases} (\alpha I + H)x^{(k+\frac{1}{2})} = (\alpha I - S)x^{(k)} + b \\ (\alpha I + S)x^{(k+1)} = (\alpha I - H)x^{(k+\frac{1}{2})} + b \end{cases}, \tag{1.4}$$

where α is a given positive constant.

Evidently, each iterate of the HSS iteration alternates between the Hermitian part H and the skew-Hermitian part S of the matrix A, analogously to the classical alternating direction implicit iteration (ADI) introduced by Peaceman and Rachford for solving partial differential equations, see [20, 21]. Results associated to the stationary iterative method with alternation can be also found in Benzi and Szyld [10]. Theoretical analysis shows that the HSS iteration (1.4) converges unconditionally to the unique solution of the system of linear equations (1.1), if the coefficient matrix A is a non-Hermitian positive definite matrix. Due to its promising performance and elegant mathematical properties, the HSS scheme immediately attracted considerable attention, resulting in numerous papers devoted to various aspects of this new method, see for instance [2, 3, 5–9, 12, 15, 24] and references therein.

Recently, Gu and Tian in [15] and Chen and Jiang in [12] have respectively tailored the HSS iteration for real positive definite Toeplitz systems by using the reducible properties of symmetric and skew symmetric Toeplitz matrices. That is that each of two systems in (1.4)

is reduced to two subsystems with about half sizes and the computational complexity of their methods is reduced to about half of the HSS iteration at each iteration. However, the computational costs remain $O(n^2)$ flops at each iteration. This seems to be not a very good motivation, because every $n \times n$ Toeplitz matrix A enjoys a circulant and skew-circulant splitting $A = C + S$ (denoted by CSCS) which results in the following CSCS iteration proposed by Ng in [19] for non-Hermitian positive definite Toeplitz systems.

The CSCS iteration. *Given an initial guess $x^{(0)}$. For $k = 0, 1, 2, \ldots$ until $\{x^{(k)}\}$ converges, compute*

$$\begin{cases} (\alpha I + C)x^{(k+\frac{1}{2})} = (\alpha I - S)x^{(k)} + b \\ (\alpha I + S)x^{(k+1)} = (\alpha I - C)x^{(k+\frac{1}{2})} + b \end{cases}, \tag{1.5}$$

where α is a positive constant.

Evidently, the matrices $\alpha I \pm C$ and $\alpha I \pm S$ are circulant matrices and skew-circulant matrices, respectively, which can be diagonalized by the discrete Fourier matrix F and $\hat{F}$, where $\hat{F} = F\Omega$ with Ω a diagonal matrix; see for example [11,14,23]. Thus, the cost of each iteration in (1.5) is $O(n \log n)$ complex flops by using 8 FFTs of n-vectors.

It is shown in [19] that if the real part of the generating function f is positive, then the generated Toeplitz matrix A is positive definite. Furthermore, the splitting matrices C and S are also positive definite for large enough n. In this case the CSCS iteration converges unconditionally to the unique solution of the system of linear equations (1.1). In particular, if the generating function f is real positive even, then the generated Toeplitz matrix A is real symmetric positive definite. Therefore, we may use the CSCS iteration for solving the system of linear equations (1.1) by using FFTs. However, all operation counts above concern complex operations. It is desirable to develop an analogue of (1.5) only involving real arithmetics. This is the main motivation of this paper.

This paper is organized as follows. In next section we first review some basic definitions, notation and preliminaries related to the splittings of the real symmetric Toeplitz matrices based on trigonometric transformations. Then we develop an iterative method based on the splitting above, which is referred to as trigonometric transform splitting and briefly denoted by TTS, for solving (1.1). We finally study the convergence properties and analyze the convergence rate of the TTS iteration. Some computational details are also discussed. Numerical experiments are presented in Section 3 to show the effectiveness of our methods. A brief conclusion is also drawn.

2. The TTS iteration. In this section we first review the trigonometric transform based splitting of a real symmetric Toeplitz matrix, then discuss the convergence of the TTS iteration, and finally give the details of implement of the TTS iteration as well as computational complexity.

2.1. The TTS. Let's begin with some basic definitions, notation and preliminaries used in the sequel, see [16, 17, 22].

Throughout this paper we denote the first version of the discrete cosine transform (DCT) matrix of order n by $\mathcal{C}_n^I$, the first version of discrete sine transform (DST) matrix of order n by $\mathcal{S}_n^I$, the set of all n-dimensional real vectors by $\mathbb{R}^n$, the set of all $n \times n$ real matrices by $\mathbb{R}^{n\times n}$, the transpose of a vector x by x^T, the transpose of a matrix B by B^T, the conjugate transpose of a matrix B by B^*, the 2-norm by $||\cdot||_2$, the vector $(1, 1, \ldots, 1)^T$ of dimension n by $\mathbf{e}$, the vector $(-1, 1, \ldots, (-1)^n)^T$ by $\mathbf{f}$, the $(q-p)\times n$ restriction matrix by P_{pq} defined by $P_{pq}(x_1, \cdots, x_n)^T = (x_p, \cdots, x_q)^T$ for $p < q$, respectively.

Recall that

$$[\mathcal{S}_n^I]_{m,k} = \sqrt{\frac{2}{n+1}} sin\frac{\pi(m+1)(k+1)}{n+1} \ , \ m, k = 0, 1, \cdots, n-1, \tag{2.1}$$

$$[\mathcal{C}_{n+2}^I]_{m,k} = \sqrt{\frac{2}{n+1}}\varepsilon_k\varepsilon_m cos\frac{\pi mk}{n} \ , \ m, k = 0, 1, \cdots, n+1, \tag{2.2}$$

where

$$\varepsilon_j = \begin{cases} 1, & \text{if} \quad j \neq 0 \quad \text{and} \quad j \neq n+1; \\ \frac{1}{\sqrt{2}}, & \text{if} \quad j = 0 \quad \text{or} \quad j = n+1. \end{cases} \tag{2.3}$$

Let $D_{n+2} = diag(1/\sqrt{2}, 1, \cdots, 1, 1/\sqrt{2})$ be a diagonal matrix of order $n+2$. Let

$$(\hat{\mathcal{C}}_n)_{m,k} = \sqrt{\frac{2}{n+1}} \cos(\frac{\pi mk}{n+1}) \ , \ m, k = 1, 2, \cdots, n.$$

Then the matrix

$$\mathcal{C}_{n+2}^I = \sqrt{\frac{2}{n+1}} D_{n+2}(cos(\frac{\pi mk}{n+1}))_{m,k=0}^{n+1} D_{n+2}, \tag{2.4}$$

which possess the following block structure

$$\mathcal{C}_{n+2}^I = \sqrt{\frac{2}{n+1}} \left[\begin{array}{c|c|c} \frac{1}{2} & \frac{1}{\sqrt{2}}\mathbf{e}^T & \frac{1}{2} \\ \hline \frac{1}{\sqrt{2}}\mathbf{e} & \sqrt{\frac{n+1}{2}}\hat{\mathcal{C}}_n & \frac{1}{\sqrt{2}}\mathbf{f} \\ \hline \frac{1}{2} & \frac{1}{\sqrt{2}}\mathbf{f}^T & \frac{1}{2}(-1)^{n+1} \end{array}\right]. \tag{2.5}$$

Let $\mathbf{a}_{n+2} = [a_0, a_1, \ldots, a_{n+1}]^T \in \mathbb{R}^{n+2}$. *Defining* $\boldsymbol{\lambda} = (\lambda_0, \ldots, \lambda_{n+1})^T$ *by*

$$\boldsymbol{\lambda} = \sqrt{2(n+1)} D_{n+2}\mathcal{C}_{n+2}^I D_{n+2}\mathbf{a}_{n+2}, \tag{2.6}$$

we then have

THEOREM 2.1. [16, 17] *Let* $\boldsymbol{\lambda}$ *be defined as in* (2.6). *Then a real symmetric Toeplitz matrix* $A_n = [a_{|i-j|}]_0^{n-1}$ *admits a splitting*

$$A_n = T_C + T_S, \tag{2.7}$$

where

$$T_C = \frac{1}{2}(\hat{\mathcal{C}}_n \Lambda \hat{\mathcal{C}}_n + R_2) \quad \text{and} \quad T_S = \frac{1}{2}(\mathcal{S}_n^I \Lambda \mathcal{S}_n^I + R_2) \tag{2.8}$$

with $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ *and* $R_2 = \frac{\lambda_0}{n+1}\mathbf{e}\mathbf{e}^T + \frac{\lambda_{n+1}}{n+1}\mathbf{f}\mathbf{f}^T$.

Note that by A_n we have only given a_j for $j = 0, 1, ..., n-1$. Hence, we have still freedom in choosing a_n and a_{n+1}. Usually, if we have no further information, we set $a_n = a_{n+1} = 0$ or we can get a_n, a_{n+1} according to (1.2). Especially, it is also possible to choose a_n and a_{n+1} such that $\lambda_0 = \lambda_{n+1} = 0$, and (2.7) reduces to $A_n = \frac{1}{2}(\mathcal{S}_n^I \Lambda \mathcal{S}_n^I + \hat{\mathcal{C}}_n \Lambda \hat{\mathcal{C}}_n)$ for this special diagonal matrix Λ.

We remark here that various definitions of Λ and/or choices of trigonometric transformations may result in different TTS'. See, for example, [16] for more details.

Based on the splitting (2.7), we can now develop the following TTS iteration.

The TTS iteration. *Given an initial guess* $x^{(0)}$. *For* $k = 0, 1, 2, \ldots$ *until* $\{x^{(k)}\}$ *converges, compute*

$$\begin{cases} (\alpha I + T_C)x^{(k+\frac{1}{2})} = (\alpha I - T_S)x^{(k)} + b \\ (\alpha I + T_S)x^{(k+1)} = (\alpha I - T_C)x^{(k+\frac{1}{2})} + b \end{cases}, \tag{2.9}$$

where α is a positive constant.

Note that we can reverse roles of the matrices T_C and T_S in the above TTS iterative method so that we may first solve the system of linear equations with coefficient matrix $\alpha I + T_S$ and then solve the system of linear equations with coefficient matrix $\alpha I + T_C$. Theoretical analysis shows that the TTS iteration (2.9) converges to the unique solution of the system of linear equations $Ax = b$, if T_C and T_S are positive definite.

The two-half steps at each TTS iteration require exact solutions with the $n \times n$ matrices $\alpha I + T_C$ and $\alpha I + T_S$. Since both $\alpha I + T_C$ and $\alpha I + T_S$ are real symmetric positive definite matrices with Toeplitz-plus-Hankel structure. The linear systems with the coefficient matrices $\alpha I + T_C$ and $\alpha I + T_S$ may be solved efficiently by PCG. Thus, the resulting method is similar to the IHSS iteration in [6], called the inexact TTS iteration which will be studied in future. Here we focus on the fast and exact solvers for the inner linear subsystems in the TTS iteration, which will be discussed in next subsection. We remark here that the TTS

iterative method can be equivalently rewritten as the following matrix-vector form,

$$x^{(k+1)} = \mathscr{H}(\alpha)x^{(k)} + \mathscr{G}(\alpha)b,\ k = 0, 1, \cdots, \tag{2.10}$$

where

$$\mathscr{H}(\alpha) = (\alpha I + T_S)^{-1}(\alpha I - T_C)(\alpha I + T_C)^{-1}(\alpha I - T_S), \tag{2.11}$$

is the iteration matrix of the TTS iteration (2.9) and

$$\mathscr{G}(\alpha) = 2\alpha(\alpha I + T_S)^{-1}(\alpha I + T_C)^{-1}.$$

In fact, (2.10) may also result from the splitting

$$A = \mathscr{M}(\alpha) - \mathscr{N}(\alpha)$$

of the coefficient matrix A, with

$$\begin{cases} \mathscr{M}(\alpha) = \frac{1}{2\alpha}(\alpha I + T_C)(\alpha I + T_S) \\ \mathscr{N}(\alpha) = \frac{1}{2\alpha}(\alpha I - T_C)(\alpha I - T_S) \end{cases}.$$

We note that $\mathscr{H}(\alpha) = \mathscr{M}^{-1}(\alpha)\mathscr{N}(\alpha)$, $\mathscr{G}(\alpha) = \mathscr{M}^{-1}(\alpha)$ and $\mathscr{G}(\alpha)A = I - \mathscr{H}(\alpha)$. If $\rho(\mathscr{H}(\alpha)) \ll 1$, then the spectrum of the preconditioned matrix $\mathscr{G}(\alpha)A$ would be clustered around 1. In this case, the preconditioner $\mathscr{G}(\alpha)$ could be served as a good preconditioner, referred to as TTS precondtioner, for solving the linear system (1.1) by the preconditioned conjugate gradient method (PCG).

2.2. The relevant algorithms of the TTS iteration. In this subsection we will show how to fast compute the products of the matrices $(\alpha I + T_C)^{-1}$, $(\alpha I + T_S)^{-1}$, $\alpha I - T_C$ and $\alpha I - T_S$ with a vector $\mathbf{v}$ via DCT I and DST I.

We first show the calculation of $(\alpha I - T_C)\mathbf{v}$. Because $\hat{\mathcal{C}}_n\mathbf{v}$ can be computed by

$$\begin{bmatrix} * \\ \hat{\mathcal{C}}_n\mathbf{v} \\ * \end{bmatrix} = \mathcal{C}^I_{n+2}\begin{bmatrix} 0 \\ \mathbf{v} \\ 0 \end{bmatrix}.$$

Thus, $(\alpha I - T_C)\mathbf{v}$ can be easily obtained from

$$\begin{bmatrix} * \\ (\alpha I - T_C)\mathbf{v} \\ * \end{bmatrix} = \frac{1}{2}\mathcal{C}^I_{n+2}(2\alpha I_{n+2} - \Lambda_{n+2})\mathcal{C}^I_{n+2}\begin{bmatrix} 0 \\ \mathbf{v} \\ 0 \end{bmatrix},$$

which can be written as the following algorithm.

Now, we show how to fast compute $(\alpha I + T_C)^{-1}\mathbf{v}$. To do this, we need to embed $\alpha I + T_C$ into an $(n+2) \times (n+2)$ matrix M of the following form

Algorithm 1 Calculate $(\alpha I - T_C)\mathbf{v}$.

1: to compute $\mathbf{u} = \mathcal{C}^I_{n+2}(0, \mathbf{v}^T, 0)^T \in \mathbb{R}^{n+2}$ by DCT I.
2: to compute $\mathbf{w} = (2\alpha I_{n+2} - \Lambda_{n+2})\mathbf{u}$.
3: to compute $\mathbf{z} = \mathcal{C}^I_{n+2}\mathbf{w}$ by DCT I.
4: to compute $\mathbf{y} = \frac{1}{2}P_{2,n+1}\mathbf{z} \in \mathbb{R}^n$.

$$M = \frac{1}{2}\mathcal{C}^I_{n+2}(2\alpha I_{n+2} + \Lambda_{n+2})\mathcal{C}^I_{n+2}. \tag{2.12}$$

A simple but tedious straightforward calculation shows that

$$M = \begin{bmatrix} \gamma_1 & \mathbf{g}_1^T & \gamma_2 \\ \mathbf{g}_1 & \alpha I + T_C & \mathbf{g}_2 \\ \gamma_2 & \mathbf{g}_2^T & \gamma_3 \end{bmatrix}, \tag{2.13}$$

with

$\gamma_1 = \alpha + \frac{1}{n+1}(\frac{\lambda_0}{4} + \frac{1}{2}\mathbf{e}^T\Lambda\mathbf{e} + \frac{\lambda_{n+1}}{4})$, $\gamma_2 = \frac{1}{n+1}(\frac{\lambda_0}{4} + \frac{1}{2}\mathbf{e}^T\Lambda\mathbf{f} + \frac{(-1)^{n+1}\lambda_{n+1}}{4})$,

$\gamma_3 = \alpha + \frac{1}{n+1}(\frac{\lambda_0}{4} + \frac{1}{2}\mathbf{f}^T\Lambda\mathbf{f} + \frac{\lambda_{n+1}}{4})$, $\mathbf{g}_1 = \frac{1}{n+1}(\frac{\lambda_0}{2\sqrt{2}}\mathbf{e} + \frac{\sqrt{n+1}}{2}\hat{\mathcal{C}}_n\Lambda\mathbf{e} + \frac{\lambda_{n+1}}{2\sqrt{2}}\mathbf{f})$,

$\mathbf{g}_2 = \frac{1}{n+1}(\frac{\lambda_0}{2\sqrt{2}}\mathbf{e} + \frac{\sqrt{n+1}}{2}\hat{\mathcal{C}}_n\Lambda\mathbf{f} + \frac{(-1)^{n+1}\lambda_{n+1}}{2\sqrt{2}}\mathbf{f})$.

Clearly, the inverse of M in (2.12) can be easily solved, once α and Λ_{n+2} are determined. We hope to obtain $(\alpha I + T_C)^{-1}$ from M^{-1}. For this purpose, let's partition M in (2.12) into the following form

$$M = \left[\begin{array}{ccc|c} \gamma_1 & \mathbf{g}_1^T & & \gamma_2 \\ \mathbf{g}_1 & \alpha I + T_C & & \mathbf{g}_2 \\ \hline \gamma_2 & \mathbf{g}_2^T & & \gamma_3 \end{array}\right] \equiv \begin{bmatrix} M_{11} & \mathbf{t} \\ \mathbf{t}^T & \gamma_3 \end{bmatrix},$$

where $\mathbf{t} = \begin{bmatrix} \gamma_2 \\ \mathbf{g}_2 \end{bmatrix} \in \mathbb{R}^{n+1}$, $M_{11} = \begin{bmatrix} \gamma_1 & \mathbf{g}_1^T \\ \mathbf{g}_1 & \alpha I + T_C \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$.

If M_{11} is nonsingular, then M is of the following 2×2 block decomposition

$$M = \begin{bmatrix} I_{n+1} & 0 \\ \mathbf{t}^T M_{11}^{-1} & 1 \end{bmatrix} \begin{bmatrix} M_{11} & \mathbf{t} \\ 0 & \sigma \end{bmatrix},$$

where $\sigma = \gamma_3 - \mathbf{t}^T M_{11}^{-1}\mathbf{t} \in \mathbb{R}$, which is referred to as the *Schur complement* of the matrix M corresponding to the sub-matrix M_{11}.

Denoting

$$M^{-1} = \begin{bmatrix} M_{11} & \mathbf{t} \\ 0 & \sigma \end{bmatrix}^{-1} \begin{bmatrix} I_{n+1} & 0 \\ \mathbf{t}^T M_{11}^{-1} & 1 \end{bmatrix}^{-1} \equiv \begin{bmatrix} N_{11} & \mathbf{s} \\ \mathbf{s}^T & \beta \end{bmatrix}$$

gives

$$\begin{cases} N_{11} = M_{11}^{-1} + M_{11}^{-1}\mathbf{t}(\gamma_3 - \mathbf{t}^T M_{11}^{-1}\mathbf{t})^{-1}\mathbf{t}^T M_{11}^{-1} \\ \mathbf{s} = -M_{11}^{-1}\mathbf{t}(\gamma_3 - \mathbf{t}^T M_{11}^{-1}\mathbf{t})^{-1} \\ \beta = (\gamma_3 - \mathbf{t}^T M_{11}^{-1}\mathbf{t})^{-1} \end{cases},$$

which results in

$$M_{11}^{-1} = N_{11} - \frac{\mathbf{s}\mathbf{s}^T}{\beta}. \tag{2.14}$$

Now, let M_{11} be decomposed into the following 2×2 block form

$$M_{11} = \begin{bmatrix} \gamma_1 - \mathbf{g}_1^T(\alpha I + T_C)^{-1}\mathbf{g}_1 & \mathbf{g}_1^T \\ 0 & \alpha I + T_C \end{bmatrix} \begin{bmatrix} 1 & 0 \\ (\alpha I + T_C)^{-1}\mathbf{g}_1 & I \end{bmatrix}$$

and let M_{11}^{-1} be denoted by

$$M_{11}^{-1} = \begin{bmatrix} \eta & \mathbf{q}^T \\ \mathbf{q} & N_{22} \end{bmatrix},$$

where $N_{22} \in \mathbb{R}^{n\times n}$, $\mathbf{q} \in \mathbb{R}^n$, $\eta \in \mathbb{R}$. We then have

$$\begin{cases} N_{22} = (\alpha I + T_C)^{-1} + (\alpha I + T_C)^{-1}\mathbf{g}_1(\gamma_1 - \mathbf{g}_1^T(\alpha I + T_C)^{-1}\mathbf{g}_1)^{-1}\mathbf{g}_1^T(\alpha I + T_C)^{-1} \\ \mathbf{q} = -(\alpha I + T_C)^{-1}\mathbf{g}_1(\gamma_1 - \mathbf{g}_1^T(\alpha I + T_C)^{-1}\mathbf{g}_1)^{-1} \\ \eta = (\gamma_1 - \mathbf{g}_1^T(\alpha I + T_C)^{-1}\mathbf{g}_1)^{-1} \end{cases},$$

which implies that

$$(\alpha I + T_C)^{-1} = N_{22} - \frac{\mathbf{q}^T\mathbf{q}}{\eta}. \tag{2.15}$$

Thus, the computation of $(\alpha I + T_C)^{-1}\mathbf{v}$ can be formulated as the Algorithm 2.

Next, we compute $(\alpha I - T_S)\mathbf{v}$. From the definition of T_S, we have

$$(\alpha I - T_S)\mathbf{v} = \frac{1}{2}[\mathcal{S}_n^I(2\alpha I - \Lambda)\mathcal{S}_n^I\mathbf{v} - \frac{\lambda_0(\mathbf{e}^T\mathbf{v})}{n+1}\mathbf{e} - \frac{\lambda_{n+1}(\mathbf{f}^T\mathbf{v})}{n+1}\mathbf{f}],$$

which can be formulated as the following algorithm 3.

Finally, we turn to compute $(\alpha I + T_S)^{-1}\mathbf{v}$. Note that

$$\alpha I + T_S = \frac{1}{2}\{[(\mathcal{S}_n^I(2\alpha I + \Lambda)\mathcal{S}_n^I) + \frac{\lambda_0}{n+1}\mathbf{e}\mathbf{e}^T] + \frac{\lambda_{n+1}}{n+1}\mathbf{f}\mathbf{f}^T\},$$

Algorithm 2 Calculate $(\alpha I + T_C)^{-1}\mathbf{v}$.

1: from right to left, to compute $\begin{bmatrix}\mathbf{s}\\ \beta\end{bmatrix} = 2\mathcal{C}^I_{n+2}(2\alpha I_{n+2} + \Lambda_{n+2})^{-1}\mathcal{C}^I_{n+2}\mathbf{e}_{n+2}$, by DCT I, where $\mathbf{e}_{n+2} = (0, \cdots, 1)^T \in \mathbb{R}^{n+2}$;
2: from right to left, to compute $\begin{bmatrix}N_{11}\mathbf{e}_1\\ \mathbf{s}^T\mathbf{e}_1\end{bmatrix} = 2\mathcal{C}^I_{n+2}(2\alpha I_{n+2} + \Lambda_{n+2})^{-1}\mathcal{C}^I_{n+2}\begin{bmatrix}\mathbf{e}_1\\ 0\end{bmatrix}$, by DCT I, where $\mathbf{e}_1 = (1, 0, \cdots, 0)^T \in \mathbb{R}^{n+1}$;
3: to compute $\begin{bmatrix}\eta\\ \mathbf{q}\end{bmatrix} = M_{11}^{-1}\mathbf{e}_1 = N_{11}\mathbf{e}_1 - \frac{\mathbf{s}^T\mathbf{e}_1}{\beta}\mathbf{s}$;
4: from right to left, to compute $\begin{bmatrix}N_{11}\mathbf{v}_1\\ \mathbf{s}^T\mathbf{v}_1\end{bmatrix} = 2\mathcal{C}^I_{n+2}(2\alpha I_{n+2} + \Lambda_{n+2})^{-1}\mathcal{C}^I_{n+2}\begin{bmatrix}\mathbf{v}_1\\ 0\end{bmatrix} \in \mathbb{R}^{n+2}$ by DCT I, where $\mathbf{v}_1 = \begin{bmatrix}0\\ \mathbf{v}\end{bmatrix} \in \mathbb{R}^{n+1}$, $\mathbf{v} \in \mathbb{R}^n$;
5: to compute $\begin{bmatrix}\mathbf{q}^T\mathbf{v}\\ N_{22}\mathbf{v}\end{bmatrix} = M_{11}^{-1}\mathbf{v}_1 = N_{11}\mathbf{v}_1 - \frac{\mathbf{s}^T\mathbf{v}_1}{\beta}\mathbf{s}$;
6: to compute $(\alpha I + T_C)^{-1}\mathbf{v} = N_{22}\mathbf{v} - \frac{\mathbf{q}^T\mathbf{v}}{\eta}\mathbf{q}$.

Algorithm 3 Calculate $(\alpha I - T_S)\mathbf{v}$.

1: to compute $\xi_1 = \mathbf{e}^T\mathbf{v}$ and $\xi_2 = \mathbf{f}^T\mathbf{v}$;
2: to compute $\hat{\mathbf{v}} = \mathcal{S}^I_n(2\alpha I - \Lambda)\mathcal{S}^I_n\mathbf{v}$, from right to left, by DST I;
3: to compute $(\alpha I - T_S)\mathbf{v} = \frac{1}{2}(\hat{\mathbf{v}} - \frac{\xi_1\lambda_0}{n+1}\mathbf{e} - \frac{\xi_2\lambda_{n+1}}{n+1}\mathbf{f})$.

we can solve $(\alpha I + T_S)^{-1}$ by the following Sherman-Morrison-Woodbury formula:

$$(B + UV^T)^{-1} = B^{-1} - B^{-1}U(I + V^TB^{-1}U)^{-1}V^TB^{-1}.$$

Setting

$$Z_1 = \mathcal{S}^I_n(2\alpha I + \Lambda)\mathcal{S}^I_n, \quad Z_2 = Z_1 + \frac{\lambda_0}{n+1}\mathbf{e}\mathbf{e}^T,$$
$$Z_3 = Z_2 + \frac{\lambda_{n+1}}{n+1}\mathbf{f}\mathbf{f}^T, \quad \alpha I + T_S = \frac{1}{2}Z_3,$$

we have

$$Z_1^{-1} = \mathcal{S}^I_n(2\alpha I + \Lambda)^{-1}\mathcal{S}^I_n, \quad Z_2^{-1} = Z_1^{-1} - \frac{\lambda_0 Z_1^{-1}\mathbf{e}\mathbf{e}^T Z_1^{-1}}{n+1+\lambda_0\mathbf{e}^T Z_1^{-1}\mathbf{e}},$$
$$Z_3^{-1} = Z_2^{-1} - \frac{\lambda_{n+1}Z_2^{-1}\mathbf{f}\mathbf{f}^T Z_2^{-1}}{n+1+\lambda_{n+1}\mathbf{f}^T Z_2^{-1}\mathbf{f}}, \quad (\alpha I + T_S)^{-1} = 2Z_3^{-1},$$

which can be simply summarized as the algorithm 4.

Algorithm 4 Calculate $(\alpha I + T_S)^{-1}\mathbf{v}$

1: to compute $\mathbf{s}_1 = Z_1^{-1}\mathbf{v}$, $\mathbf{s}_2 = Z_1^{-1}\mathbf{e}$ and $\mathbf{s}_3 = Z_1^{-1}\mathbf{f}$ by DST I;
2: to compute $\mathbf{s}_4 = Z_2^{-1}\mathbf{v} = \mathbf{s}_1 - \frac{\lambda_0(\mathbf{e}^T\mathbf{s}_1)\mathbf{s}_2}{n+1+\lambda_0\mathbf{e}^T s_2}$ and $\mathbf{s}_5 = Z_2^{-1}\mathbf{f} = \mathbf{s}_3 - \frac{\lambda_0(\mathbf{e}^T\mathbf{s}_3)\mathbf{s}_2}{n+1+\lambda_0\mathbf{e}^T\mathbf{s}_2}$;
3: to compute $\mathbf{s}_6 = \mathbf{s}_4 - \frac{\lambda_{n+1}(\mathbf{f}^T\mathbf{s}_4)\mathbf{s}_5}{n+1+\lambda_{n+1}\mathbf{f}^T\mathbf{s}_5}$;
4: to compute $(\alpha I + T_S)^{-1}\mathbf{v} = 2\mathbf{s}_6$.

The Algorithms 1-4 show that the exact solution to (2.9) at each step can be obtained efficiently by using 6 DCTs and 5 DSTs of n-vectors, which are highly parallelizable. Therefore, the proposed method is well-adapted for parallel computing. In particular, the number of real operations which are involved in each step of the TTS iterative method is $O(n \log n)$.

2.3. Convergence of the TTS iteration. In this section, we study the convergence rate of the TTS iteration. We first note that the TTS iterative method can be generalized to the two-step splitting iteration framework, and the following lemma describes a general convergence criterion for a two-step splitting iteration.

LEMMA 2.2. [4, Lemma 2.1] *Let $A \in \mathbb{C}^{n\times n}$, $A = M_i - N_i$ $(i = 1, 2)$ be two splittings and $x^{(0)} \in \mathbb{C}^n$ be a given initial vector. If $\{x^{(k)}\}$ is a two-step iteration sequence defined by*

$$\begin{cases} M_1 x^{(k+1/2)} = N_1 x^{(k)} + b \\ M_2 x^{(k+1)} = N_2 x^{(k+1/2)} + b \end{cases},$$

$k = 0, 1, 2 \ldots$, then

$$x^{(k+1)} = M_2^{-1}N_2M_1^{-1}N_1x^{(k)} + M_2^{-1}(I + N_2M_1^{-1})b, \quad k = 0, 1, 2, \ldots.$$

Moreover, if the spectral radius of the iteration matrix $M_2^{-1}N_2M_1^{-1}N_1$ is less than 1, then the iterative sequence $\{x^{(k)}\}$ converges to the unique solution $x^* \in \mathbb{C}^n$ of the system of linear equations $Ax = b$ for all initial vectors $x^{(0)} \in \mathbb{C}^n$.

The following fact mentioned in [17] is also needed in the convergence analysis of the TTS iteration. We reformulate them as the following lemma.

LEMMA 2.3. *Let $\boldsymbol{\lambda}$ be defined as in* (2.6). *If the generating function f in* (1.2) *of the Toeplitz matrix A is a real positive even function, then the matrices T_C and T_S in* (2.8) *are positive definite for sufficient large n.*

Proof. [*Stretch of the proof*] From the hypothesis, we have

$$\begin{aligned}
\lambda_0 &= \frac{a_0 + a_{n+1}}{2} + \sum_{k=1}^{n} a_k = (S_{f,n}(0) + S_{f,n+1}(0))/4 \approx f(0)/2, \\
\lambda_j &= a_0 + 2\sum_{k=1}^{n} a_k \cos(\frac{\pi jk}{n+1}) + (-1)^{n+1} a_{n+1} = \frac{1}{2}(S_{f,n} + S_{f,n+1})(\frac{\pi j}{n+1}) \\
&\approx f(\frac{\pi j}{n+1}), \\
\lambda_{n+1} &= \frac{a_0}{2} + \sum_{k=1}^{n+1}(-1)^k a_k = (S_{f,n}(\pi) + S_{f,n+1}(\pi))/4 \approx f(\pi)/2,
\end{aligned} \tag{2.16}$$

where $S_{f,n}$ is the partial sum of the Fourier series of f. If n is large enough, then $\lambda_j > 0$, for $j = 0, 1, \ldots, n+1$, which imply that T_C and T_S are positive definite. □

For the convergence property of the TTS iteration, we apply the above results to obtain the following main theorem.

THEOREM 2.4. *Let T_C and T_S be the matrices given in* (2.8) *and α be a positive constant. If the generating function f in* (1.2) *of the Toeplitz matrix A is a real positive even function, then the iteration matrix $\mathscr{H}(\alpha)$ of the TTS iteration is given in* (2.11) *and the spectral radius $\rho(\mathscr{H}(\alpha))$ is bounded by*

$$\sigma(\alpha) = \max_{\eta_j \in \lambda(T_C)} |\frac{\alpha - \eta_j}{\alpha + \eta_j}| \max_{\xi_j \in \lambda(T_S)} |\frac{\alpha - \xi_j}{\alpha + \xi_j}|. \tag{2.17}$$

Therefore, it holds that for large enough n

$$\rho(\mathscr{H}(\alpha)) \leq \sigma(\alpha) < 1, \quad \forall \alpha > 0, \tag{2.18}$$

i.e., the TTS iteration converges to the exact solution $x^* \in \mathbb{R}^n$ of the system of linear equations (1.1).

Proof. Similar to the proof of the Theorem 2.2 in [4], we can prove this theorem as follows.

From the hypothesis, we have that T_C and T_S are positive definite and so are $\alpha I + T_C$ and $\alpha I + T_S$ for any positive constant α. By putting

$$M_1 = \alpha I + T_C,\ N_1 = \alpha I - T_S,\ M_2 = \alpha I + T_S,\ N_2 = \alpha I - T_C,$$

we get (2.11).

By the similarity invariance of the matrix spectrum, we have

$$\begin{aligned}
\rho(\mathscr{H}(\alpha)) &= \rho((\alpha I - T_C)(\alpha I + T_C)^{-1}(\alpha I - T_S)(\alpha I + T_S)^{-1}) \\
&\leq \| (\alpha I - T_C)(\alpha I + T_C)^{-1}(\alpha I - T_S)(\alpha I + T_S)^{-1} \|_2 \\
&\leq \| (\alpha I - T_C)(\alpha I + T_C)^{-1} \|_2 \| (\alpha I - T_S)(\alpha I + T_S)^{-1} \|_2 .
\end{aligned}$$

It then follows that

$$\rho(\mathscr{H}(\alpha)) \leq \max_{\eta_j \in \lambda(T_C)} |\frac{\alpha - \eta_j}{\alpha + \eta_j}| \max_{\xi_j \in \lambda(T_S)} |\frac{\alpha - \xi_j}{\alpha + \xi_j}| \equiv \sigma(\alpha).$$

Since η_j and ξ_j are positive for $j = 1, 2, \ldots, n$, we have that $\sigma(\alpha)$ is strictly less than 1 and therefore $\rho(\mathscr{H}(\alpha)) < 1$, for all $\alpha > 0$. □

We remark that if the eigenvalues of the matrices T_C and T_S contain in $\Omega = [\gamma_{min}, \gamma_{max}]$, then $\sigma(\alpha)$ can be estimated by

$$\max_{\gamma \in \Omega} \frac{(\alpha - \gamma)^2}{(\alpha + \gamma)^2},$$

where γ_{min} and γ_{max} are the lower and the upper bounds of the eigenvalues of the matrices T_C and T_S. The optimal parameter α^* is chosen such that the above estimate can be minimized. This fact is precisely stated as the following theorem.

LEMMA 2.5. *The minimizer of $\sigma(\alpha)$ over all positive α is attained at*

$$\alpha^* = \sqrt{\gamma_{min}\gamma_{max}}, \tag{2.19}$$

and the corresponding minimum value is equal to

$$\sigma(\alpha^*) = (\frac{\sqrt{\gamma_{max}} - \sqrt{\gamma_{min}}}{\sqrt{\gamma_{max}} + \sqrt{\gamma_{min}}})^2 \leq \frac{\sqrt{\gamma_{max}} - \sqrt{\gamma_{min}}}{\sqrt{\gamma_{max}} + \sqrt{\gamma_{min}}}.$$

The proof is a verbatim of one of Corollary 2.3 in [4] and therefore omitted.

We emphasize that, the optimal parameter α^* minimizes only the upper bound $\sigma(\alpha)$ of the spectral radius of the iteration matrix, i.e.

$$\alpha^* = arg \min_{\alpha > 0} \sigma(\alpha) = \sqrt{\gamma_{min}\gamma_{max}}$$

but does not minimize the spectral radius itself, i.e.,

$$\alpha^* \neq arg \min_{\alpha > 0} \rho(\mathscr{H}(\alpha)) \equiv \alpha_T.$$

We illustrate this fact by Example 3.1 ($p = 0.8$) and Example 3.2 ($\varphi_1 = 0.9$) in Table 2.1-2.2, where α^* is defined in (2.19), α_T is the spectral radius optimal parameters, $\rho(\mathscr{H}(\alpha^*))$ and $\rho(\mathscr{H}(\alpha_T))$ are the spectral radii of the corresponding iteration matrices.

From the tables, we can see that $\rho(\mathscr{H}(\alpha_T)) < \rho(\mathscr{H}(\alpha^*))$ in all cases. Therefore, it is important to get α_T or a good approximation of α_T for improving the convergence speed of the TTS iterative method. But it is a hard task that needs further in-depth study from the viewpoint of both theory and computations.

Table 2.1 The comparison of $\rho(\mathscr{H}(\alpha^*))$ and $\rho(\mathscr{H}(\alpha_T))$ for Example 3.1

n	α^*	$\rho(\mathscr{H}(\alpha^*))$	α_T	$\rho(\mathscr{H}(\alpha_T))$
64	0.68	0.6041	0.78	0.4003
128	0.77	0.6365	0.87	0.4383
256	0.85	0.6624	0.95	0.4717
512	0.93	0.6864	1.03	0.5017
1024	1.01	0.7068	1.11	0.5282

Table 2.2 The comparison of $\rho(\mathscr{H}(\alpha^*))$ and $\rho(\mathscr{H}(\alpha_T))$ for Example 3.2

n	α^*	$\rho(\mathscr{H}(\alpha^*))$	α_T	$\rho(\mathscr{H}(\alpha_T))$
64	1.28	0.5136	1.43	0.3135
128	1.27	0.5202	1.44	0.3207
256	1.27	0.5250	1.45	0.3229
512	1.27	0.5274	1.46	0.3247
1024	1.27	0.5286	1.46	0.3248

3. Numerical examples. To verify the effectiveness of the TTS iteration, four kinds of generating functions were tested and they are as follows:

EXAMPLE 3.1. $a_j = (1 + |j|)^{-p}$, $j = 0, \pm 1, \cdots, \pm n - 1$, *where* $p = 0.9$, 1.0 *and* 1.1, *respectively.*

EXAMPLE 3.2. $f(x) = x^2 + \varphi_1$.

EXAMPLE 3.3. $f(x) = |x|$.

EXAMPLE 3.4. $f(x) = \pi^2 - x^2$.

We remark here that the Toeplitz matrices generated by the functions in Examples 3.1-3.2 are well-conditioned and the Toeplitz matrices generated by the functions in Examples 3.3-3.4 are ill-conditioned. Also, we observe that the matrices T_C and T_S defined in (2.8) in Examples 3.1-3.4 are all positive definite for any positive integer n.

3.1. Numerical results. All the numerical tests were done on a Founder desktop PC with Intel(R) Core(TM) i3-2310M CPU @2.10 by Matlab R2009(b) with a machine precision of 10^{-16}.

In our numerical experiments, we test the TTS iteration in the cases of $a_n = a_{n+1} = 0$ and of a_n and a_{n+1} obtained from (1.2), and denote them by TTS_1 and TTS_2, respectively.

In all test, we use the vector of all ones as the right-hand-side vector and the initial guess. The stopping criteria is $\tau = \frac{||r^{(k)}||_2}{||r^{(0)}||_2} \leq 10^{-6}$, where $r^{(k)}$ is the residual vector at the k-th iteration.

For comparison, we also test the CSCS iterative method, the SGS iterative method and the PSS iterative method.

By taking $A = (D - L) - U = (D - U) - L$, we can obtain the following known SGS

iteration.

The SGS iteration. *Given an initial guess* $x^{(0)}$. *For* $k = 0, 1, 2, \ldots$ *until* $\{x^{(k)}\}$ *converges, compute*

$$\begin{cases} (D-L)x^{(k+\frac{1}{2})} = Ux^{(k)} + b \\ (D-U)x^{(k+1)} = Lx^{(k+\frac{1}{2})} + b \end{cases}. \tag{3.1}$$

It is known that the SGS is always convergent if A is Hermitian positive definite. Furthermore, if A is an Hermitian positive definite Toeplitz matrix, then $D-U{=}D-L^*$ and L are Toeplitz matrices. In this case, the main operations in (3.1) are to calculate $(D-L)^{-1}$, $(D-L)^{-1}\mathbf{u}$ and $L^*\mathbf{v}$. A fast algorithm in [13, 18] for computing $(D-L)^{-1}$ requires about 10 FFTs of n-vectors. Furthermore, to compute $(D-L)^{-1}\mathbf{u}$, $(D-L^*)^{-1}\mathbf{w}$, $L\mathbf{z}$ and $L^*\mathbf{v}$ needs 12 FFTs of $2n$-vectors, see, e.g., [11]. That is to say that the exact solutions with lower triangular Toeplitz matrices can be obtained by using 34 FFTs of n-vectors.

Similarly, by taking $A = (\alpha I + P) - (\alpha I - S){=}(\alpha I + S) - (\alpha I - P)$ with P positive definite and S non-Hermitian, we can obtain the following known PSS iteration [3].

The PSS iteration. *Given an initial guess* $x^{(0)}$. *For* $k = 0, 1, 2, \ldots$ *until* $\{x^{(k)}\}$ *converges, compute*

$$\begin{cases} (\alpha I + P)x^{(k+\frac{1}{2})} = (\alpha I - S)x^{(k)} + b \\ (\alpha I + S)x^{(k+1)} = (\alpha I - P)x^{(k+\frac{1}{2})} + b \end{cases}, \tag{3.2}$$

where α is a positive constant.

It is shown in [3] that the PSS iteration (3.2) converges unconditionally to the unique solution of the system of linear equations (1.1), if the coefficient matrix A is a non-Hermitian positive definite matrix.

In our numerical tests, we take $P = D - 2L$ and $S = L - U = L - L^*$, which result in a practical PSS iteration (referred to TSS iteration in [3]).

Since $\alpha I + P$ is a lower triangular Toeplitz matrix, $(\alpha I + P)^{-1}\mathbf{u}$ and $(\alpha I - P)\mathbf{v}$ can be fast obtained by employing FFTs, as discussed above for SGS iteration. Moreover, S is a skew-symmetric Toeplitz matrix, by using an unitary similarity transformation, the second system of (3.2) can be reduced into two subsystems with about half sizes. Thus, one need to solve a Toeplitz-plus-Hankel subsystem, see [12, 15] for details. The complexity of each iteration is at least $O(n^2)$ if the direct method is employed, and may be reduced to $O(\frac{n}{2}\log\frac{n}{2})$ if a preconditioned conjugate gradient method [12] is used. However, a good preconditioner is not easy to get. Moreover, the use of FFTs makes the complex operations be involved for real system (1.1).

The experimentally found optimal values of the iteration parameters are used for the TTS, CSCS and PSS iterative methods. In particular, the optimal parameters of the TTS iteration (2.9) are selected in a neighbourhood of $\sqrt{\lambda_1\lambda_n}$, where λ_1 and λ_n are defined as in (2.6).

Table 3.1 The optimal iteration parameter α versus the No. of iterations for Example 3.1 with p=0.9

n	SGS	PSS		CSCS		TTS_1		TTS_2	
	IT	IT	α	IT	α	IT	α	IT	α
64	32	21	1.84	11	1.00	10	1.08	10	1.08
128	38	22	2.16	12	1.16	11	1.20	11	1.20
256	45	23	2.48	13	1.48	11	1.48	11	1.48
512	53	24	2.80	13	1.64	11	1.76	11	1.76
1024	61	25	3.16	14	1.80	12	1.84	12	1.84

Table 3.2 The optimal iteration parameter α versus the No. of iterations for Example 3.1 with p=1.0

n	SGS	PSS		CSCS		TTS_1		TTS_2	
	IT	IT	α	IT	α	IT	α	IT	α
64	26	18	1.80	9	1.04	8	1.08	8	1.08
128	30	19	2.00	10	1.16	8	1.32	8	1.32
256	34	19	2.16	11	1.28	8	1.52	8	1.52
512	38	20	2.48	11	1.48	8	1.68	8	1.68
1024	43	20	2.88	11	1.72	8	1.84	8	1.84

Table 3.3 The optimal iteration parameter α versus the No. of iterations for Example 3.1 with p=1.1

n	SGS	PSS		CSCS		TTS_1		TTS_2	
	IT	IT	α	IT	α	IT	α	IT	α
64	21	16	1.76	8	1.00	6	1.12	6	1.12
128	24	16	1.84	9	1.08	6	1.24	6	1.24
256	27	17	2.12	9	1.24	6	1.40	6	1.40
512	29	17	2.20	9	1.40	6	1.56	6	1.56
1024	32	17	2.40	9	1.56	7	1.48	7	1.48

Table 3.4 The optimal iteration parameter α versus the No. of iterations for Example 3.2 when $\varphi_1 = 0.8$

n	SGS	PSS		CSCS		TTS_1		TTS_2	
	IT	IT	α	IT	α	IT	α	IT	α
64	16	22	2.64	11	1.24	10	1.32	10	1.32
128	16	23	2.64	11	1.24	10	1.28	10	1.28
256	16	23	2.76	11	1.20	10	1.28	10	1.28
512	17	23	2.76	11	1.20	10	1.24	10	1.24
1024	17	23	2.76	10	1.20	10	1.24	10	1.24

It is shown in Tables 3.1 – 3.4 that the TTS iterative methods perform more efficiently than the CSCS iterative method and much more efficiently than the PSS and SGS iterative methods, respectively.

In Fig.3.1, We depict the convergence behavior of the SGS, PPS, CSCS, TTS_1 and TTS_2 for Example 3.3-3.4, which are denoted by the $***$, $\times\times\times$, $+++$, $\circ\circ\circ$ and $\cdots$ curves, respectively. From Fig.3.1, we can see that the TTS_1 and TTS_2 iterations have the same convergence behavior, we therefore refer to them as TTS iteration. Furthermore, we see that the TTS iteration have a better convergence behavior than the CSCS iteration and a much better convergence behavior than the PSS and SGS iterations.

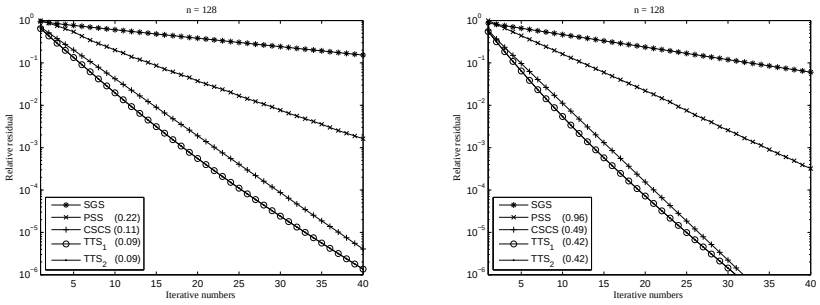

Fig. 3.1 Convergence curves for SGS, PSS, CSCS and TTS for Example 3.3(left) and Example 3.4(right).

3.2. Spectral radius. In this subsection, we focus on the Example 3.1 to further explain why the TTS iterative methods work better than the CSCS and PSS iterative methods, by comparing the spectral radii of the iteration matrices of the TTS iteration and CSCS and PSS iterations, denoted by $\mathscr{H}_T(\alpha)$, $\mathscr{H}_C(\alpha)$ and $\mathscr{H}_P(\alpha)$, respectively. Note in this example that the TTS_1 and TTS_2 are the same effect, therefore, we only describe the TTS_1 which are briefly denoted by TTS.

We plot the varying behavior of four functions of $\rho(\mathscr{H}_T(\alpha))$, $\rho(\mathscr{H}_C(\alpha))$, $\rho(\mathscr{H}_P(\alpha))$ and $\sigma(\alpha)$ according to α in Fig. 3.2, which are denoted by $***$, $\cdots$, $\circ\circ\circ$ and $+++$ curves respectively.

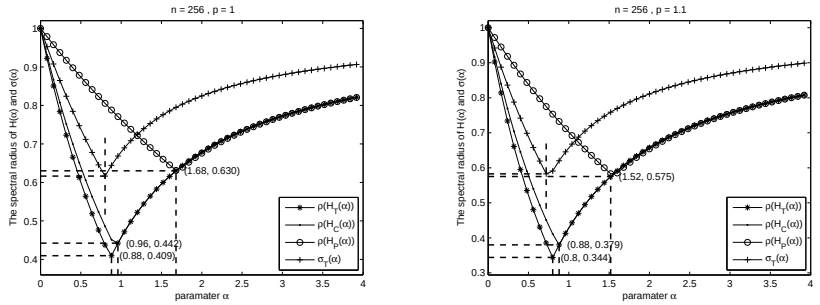

Fig. 3.2 Curves of $\rho(\mathscr{H}_T(\alpha))$, $\rho(\mathscr{H}_C(\alpha))$ $\rho(\mathscr{H}_P(\alpha))$, and $\sigma(\alpha)$ for Example 3.1.

From the Fig.3.2, we can see that $\rho(\mathscr{H}_T(\alpha_T)) < \rho(\mathscr{H}_C(\alpha_T)) < \rho(\mathscr{H}_P(\alpha_T))$ for both cases. This numerically answers the question proposed in the beginning of this subsection.

4. Conclusion. In this paper we have exploited the TTS of the real symmetric Toeplitz matrix to develop some efficient iterative methods for real symmetric positive definite Toeplitz systems. We have shown that the TTS methods converge to the unique solution of the linear system of equations for sufficient large n, if the generating function f of the $n \times n$ Toeplitz matrix is a real positive even function. Moreover, we have derived an upper bound of the contraction factor of the TTS iteration which is dependent solely on the spectra of the two TTS matrices involved.

Our Algorithms 1-4 show that the exact solution to (2.9) at each step can be obtained efficiently by using DCT I and DST I which are highly parallelizable. In particular, the number of real operations which are involved in each step of the TTS iterative method is $O(n \log n)$.

Theoretically, we can choose α to be any positive constant. However, as it is shown in the previous sections, the convergence rate is seriously dependent on the choices of α. In our numerical tests, we take experimentally optimal iterative parameters and find that our method works better than the CSCS iterative method and much better than the PSS and the SGS iterative methods. It is an important and hard task to find a good approximation α_{exp} of α_T which strongly depend on the concrete structures and properties of the coefficient matrix A and needs further in-depth study from the viewpoint of both theory and computations.

Finally, we remark that we can extend our methods in parallel to symmetric block-Toeplitz-symmetric-Toeplitz-block (SBTSTB) matrices, see Appendix.

5. Acknowledgement. The authors are grateful to the anonymous referees for their valuable comments and suggestions which improved the quality of this paper. Also, the authors would like to thank the supports of the National Natural Science Foundation of China under Grant No. 11371075, the Portuguese Funds through FCT–Fundac$\tilde{a}$o para a Ciência e a Tecnologia, within the Project UID/MAT/00013/2013.

REFERENCES

[1] G. Ammar and W. Gragg, *Superfast solution of real positive definite Toeplitz systems*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 61-76.

[2] Z.-Z. Bai, M. Benzi, F. Chen and Z.-Q. Wang, *Preconditioned MHSS iteration methods for a class of block two-by-two linear systems with applications to distributed control problems*, IMA J. Numer. Anal., 33 (2013), pp. 343-369.

[3] Z.-Z. Bai, G. H. Golub, L.-Z. Lu, and J.-F. Yin, *Block triangular and skew-Hemitian spiltting methods for Hemitian positive definite systems,* SIAM J. Sci. Comput., 26 (2005), pp. 844-863.

[4] Z.-Z. Bai, G. H. Golub, and M. K. Ng, *Hermitian and skew-Hemitian spiltting methods for non-Hemitian positive definite linear systems*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 603-626.

[5] Z.-Z. Bai, G. H. Golub, and M. K. Ng, *On successive-overrelaxation acceleration of the Hermitian and skew-Hermitian splitting iterations*, Numer. Linear Algebra Appl., 14 (2007), pp. 319-335.

[6] Z.-Z. Bai, G. H. Golub, and M. K. Ng, *On inexact Hermitian and skew-Hermitian splitting methods*

for non-Hermitian positive definite linear systems, Linear Algebra Appl., 428 (2008), pp. 413-440.
[7] Z.-Z. Bai and M. K. Ng, *Erratum*, Numer. Linear Algebra Appl., 19 (2012), pp. 891.
[8] Z.-Z. Bai, G. H. Golub, J.-Y. Pan, *Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems*, Numer. Math., 98 (2004), pp. 1-32.
[9] M. Benzi, *A generalization of the Hermitian and skew-Hemitian spiltting iteration*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 360-374.
[10] M. Benzi and D. Szyld, *Existence and uniqueness of splittings of stationary iterative methods with applications to alternating methods*, Numer. Math., 76 (1997), pp. 309-321.
[11] R. Chan and M. K. Ng, *Conjugate gradient methods for Toeplitz systems,* SIAM Rev., 38 (1996), pp. 427-482.
[12] F. Chen and Y.-L. Jiang, *On HSS and AHSS iterative methods for nonsymmetric positive definite Toeplitz systems*, J. Comput. Applied Math., 234 (2010), pp. 2432-2440.
[13] D. Commges and M. Monsion, *Fast inversion of triangular Toeplitz matrices*, lEEE Transactions on automatic control, 29 (1984), pp. 250-251.
[14] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore and London, 3rd edition, 1996.
[15] C.-Q. Gu and Z.-L. Tian, *On the HSS iterative methods for positive definite Toeplitz linear system*, J. Comput. Applied Math., 224 (2009), pp. 709-718.
[16] G. Heinig and K. Rost, *Representations of Toeplitz-plus-Hankel matrices using trigonometric transformations with application to fast matrix-vector multiplication*, Linear Algebra Appl., 275-276 (1998), pp. 225-248.
[17] T. Huckle, *Iteration mathods for ill-conditioned Toeplitz matrices*, Calcolo, 33 (1996), pp. 177-190.
[18] F.-R. Lin, W. K. Ching and M. K. Ng, *Fast inversion of triangular Toeplitz matrices*, Theoretical Computer Science, 315 (2004), pp. 511-523.
[19] M. K. Ng, *Circulant and skew-circulant splitting methods for Toeplitz systems,* J. Comput. Applied Math., 159 (2003), pp. 101-108.
[20] D. W. Peaceman and H. H. Rachford, *The numerical solution of parabolic and elliptic differential equations*, J. SIAM., 3 (1955), pp. 28-41.
[21] Y. Saad, *Iterative methods for sparse Linear systems*, 2nd edition, SIAM, Philadelphia, PA, USA, 2000.
[22] G. Strang, *The Discrete Cosine Transform,* SIAM Rev., 41 (1999), pp. 135-147.
[23] Z. D. Wang, *Fast algorithms for the discrete W transform and for the discrete Fourier transform,* IEEE Transactions On Acoustics, Speech, and Signal Processing, 32 (1984), pp. 803-816.
[24] Y.-J. Xie and C.-F. Ma, *A modified positive-definite and skew-Hermitian splitting preconditioner for generalized saddle point problems from the Navier-Stokes equation*, Numer Algor., 72 (2016), pp. 243-258.

Appendix.

Let A be a symmetric block-Toeplitz-symmetric-Toeplitz-block (SBTSTB) matrix of the following form:

$$A = \begin{bmatrix} A_0 & A_1 & \cdots & A_{m-1} \\ A_1 & A_0 & \ddots & A_{m-2} \\ \vdots & \ddots & \ddots & \vdots \\ A_{m-1} & A_{m-2} & \cdots & A_0 \end{bmatrix} \quad \text{with} \quad A_k = \begin{bmatrix} a_{k,0} & a_{k,1} & \cdots & a_{k,n-1} \\ a_{k,1} & a_{k,0} & \ddots & a_{k,n-2} \\ \vdots & \ddots & \ddots & \vdots \\ a_{k,n-1} & a_{k,n-2} & \cdots & a_{k,0} \end{bmatrix},$$

where the blocks A_k are themselves symmetric Toeplitz matrices of order n.

As mentioned in the previous sections, we can choose $a_n^{(k)}$ and $a_{n+1}^{(k)}$ such that $\lambda_0^{(k)} = \lambda_{n+1}^{(k)} = 0$. In this case, we have

$$A_k = \frac{1}{2}(\mathcal{S}_n^I \Lambda_k \mathcal{S}_n^I + \hat{\mathcal{C}}_n \Lambda_k \hat{\mathcal{C}}_n)$$

for a special diagonal matrix $\Lambda_k \ \ (k = 1, 2, \cdots, m-1)$.

Then, by Theorem 2.1, the SBTSTB matrix A possesses the following splitting:

$$A = \frac{1}{2}((I_m \otimes \mathcal{S}_n^I) T (I_m \otimes \mathcal{S}_n^I) + (I_m \otimes \hat{\mathcal{C}}_n) T (I_m \otimes \hat{\mathcal{C}}_n)),$$

where the notation $\otimes$ is the Kronecker product, I_n is the $n \times n$ identity matrix, and

$$T = \begin{bmatrix} \Lambda_0 & \Lambda_1 & \cdots & \Lambda_{m-1} \\ \Lambda_1 & \Lambda_0 & \ddots & \Lambda_{m-2} \\ \vdots & \ddots & \ddots & \vdots \\ \Lambda_{m-1} & \Lambda_{m-2} & \cdots & \Lambda_0 \end{bmatrix},$$

with $\Lambda_k = diag(\lambda_{0,k}, \lambda_{1,k}, \cdots, \lambda_{n-1,k})$.

Let e_k be the k-th row of I_n. Defining

$$E_{k-1} = \overbrace{\left[\begin{array}{c|c|c} e_k & \cdots & \\ \hline & \ddots & \\ \hline & \cdots & e_k \end{array}\right]}^{\text{m blocks}} \begin{array}{l} \text{row } 1 \\ \vdots \\ \text{row } m \end{array}$$

which is an $m \times mn$ matrix, and

$$E = \begin{bmatrix} E_0 \\ \vdots \\ E_{n-1} \end{bmatrix} \in \mathbb{R}^{mn \times mn}$$

yields a permutation matrix.

A straightforward computation shows that

$$T = E^T \begin{bmatrix} T_0 & & & \\ & T_1 & & \\ & & \ddots & \\ & & & T_{n-1} \end{bmatrix} E, \ T_k = \begin{bmatrix} \lambda_{k,0} & \lambda_{k,1} & \cdots & \lambda_{k,m-1} \\ \lambda_{k,1} & \lambda_{k,0} & \ddots & \lambda_{k,m-2} \\ \vdots & \ddots & \ddots & \vdots \\ \lambda_{k,m-1} & \lambda_{k,m-2} & \cdots & \lambda_{k,0} \end{bmatrix},$$

where T_k is an $m \times m$ real symmetric Toeplitz matrix.

Again, by Theorem 2.1, we have

$$T_k = \frac{1}{2}(\mathcal{S}_m^I \tilde{\Lambda}_k \mathcal{S}_m^I + \hat{\mathcal{C}}_m \tilde{\Lambda}_k \hat{\mathcal{C}}_m),$$

where all $\tilde{\Lambda}_k$, for $k = 0, 1, \cdots, n-1$, are diagonal matrices of order m.

In this case, we further obtain

$$T = \frac{1}{2} E^T [(I_n \otimes \mathcal{S}_m^I)\tilde{\Lambda}(I_n \otimes \mathcal{S}_m^I) + (I_n \otimes \hat{\mathcal{C}}_m)\tilde{\Lambda}(I_n \otimes \hat{\mathcal{C}}_m)]E,$$

where $\tilde{\Lambda} = diag(\tilde{\Lambda}_0, \tilde{\Lambda}_1, \cdots, \tilde{\Lambda}_{n-1})$.

Thus, a real SBTSTB admits a splitting

$$A = T_c^c + T_c^s + T_s^c + T_s^s,$$

where

$$\begin{aligned}
T_c^c &= \frac{1}{4}((I_m \otimes \hat{\mathcal{C}}_n)E^T(I_n \otimes \hat{\mathcal{C}}_m)\tilde{\Lambda}(I_n \otimes \hat{\mathcal{C}}_m)E(I_m \otimes \hat{\mathcal{C}}_n)),\\
T_c^s &= \frac{1}{4}((I_m \otimes \hat{\mathcal{C}}_n)E^T(I_n \otimes \mathcal{S}_m^I)\tilde{\Lambda}(I_n \otimes \mathcal{S}_m^I)E(I_m \otimes \hat{\mathcal{C}}_n)),\\
T_s^c &= \frac{1}{4}((I_m \otimes \mathcal{S}_n^I)E^T(I_n \otimes \hat{\mathcal{C}}_m)\tilde{\Lambda}(I_n \otimes \hat{\mathcal{C}}_m)E(I_m \otimes \mathcal{S}_n^I)),\\
T_s^s &= \frac{1}{4}((I_m \otimes \mathcal{S}_n^I)E^T(I_n \otimes \mathcal{S}_m^I)\tilde{\Lambda}(I_n \otimes \mathcal{S}_m^I)E(I_m \otimes \mathcal{S}_n^I).
\end{aligned}$$

Therefore, the systems of linear equations with coefficient matrices $\alpha I + T_c^c$, $\alpha I + T_c^s$, $\alpha I + T_s^c$ and $\alpha I + T_s^c$ of order mn can be solved efficiently using DCTs and DSTs. The total number of operations required for each step of the method is $O(mn \log mn)$ real arithmetics. By using the similar arguments in Theorem 2.4, we can show that the resulting method converges if T_c^c, T_c^s, T_s^c and T_s^s are positive definite.