Applications of Mathematics

Beibei Li; Jingjing Cui; Zhengge Huang; Xiaofeng Xie

A dual-parameter double-step splitting iteration method for solving complex symmetric linear equations

Applications of Mathematics, Vol. 69 (2024), No. 3, 311–337

Persistent URL: http://dml.cz/dmlcz/152352

A DUAL-PARAMETER DOUBLE-STEP SPLITTING ITERATION METHOD FOR SOLVING COMPLEX SYMMETRIC LINEAR EQUATIONS

Beibei Li, Jingjing Cui, Zhengge Huang, Xiaofeng Xie, Nanning

Received June 28, 2023. Published online April 5, 2024.

Abstract. We multiply both sides of the complex symmetric linear system $Ax = b$ by $1 - \mathrm{i}\omega$ to obtain a new equivalent linear system, then a dual-parameter double-step splitting (DDSS) method is established for solving the new linear system. In addition, we present an upper bound for the spectral radius of iteration matrix of the DDSS method and obtain its quasi-optimal parameter. Theoretical analyses demonstrate that the new method is convergent when some conditions are satisfied. Some tested examples are given to illustrate the effectiveness of the proposed method.

Keywords: DDSS iteration method; linear equations; SPD matrix; SPSD matrix; convergence property

MSC 2020: 65F10, 65H10

1. Introduction

In this paper, we consider to solve the linear equations

$$Ax = b,\ A \in \mathbb{C}^{n\times n} \quad \text{and} \quad x, b \in \mathbb{C}^n, \tag{1.1}$$

where $A = W + \mathrm{i}T$, and W and T are real symmetric positive semi-definite (SPSD) matrices such that at least one of them is positive definite. Throughout the paper, we assume that T is a symmetric positive definite (SPD) matrix. Here and in the sequel, we use $\mathrm{i} = \sqrt{-1}$ to stand for the imaginary unit.

This work was subsidized by the Guangxi Natural Science Foundations (No. 2021 GXNSFBA196064, GuikeAD21220129), the National Science Foundation of China (No. 12361078), and the Guangxi Natural Science Foundations (No. 2019GXNSFBA185014, Guike AD20159056).

DOI: 10.21136/AM.2024.0133-23

The above complex symmetric linear equations (1.1) are widely used in scientific and engineering calculations, such as structural dynamics [16], lattice quantum chromodynamics [17], FFT-based solution of certain time-dependent PDEs [13], diffuse optical tomography [1], molecular scattering [28], and quantum mechanics [31]. For more examples, we refer to [12].

The coefficient matrix A naturally has the Hermitian and skew-Hermitian splitting

$$A = H + S,$$

where $H = \frac{1}{2}(A + A^*) = W$ and $S = \frac{1}{2}(A - A^*) = \mathrm{i}T$ with $H = W$, $S = \mathrm{i}T$ and A^* being the Hermitian part, skew-Hermitian part and conjugate transpose of A, respectively.

Based on the above Hermite and skew-Hermite splitting of the coefficient matrix, Bai et al. developed the Hermitian and skew-Hermitian splitting (HSS) method in [9]. Due to the advantages of simple structure, easy program implementation, and unconditional convergence, the HSS iteration method has attracted the interest of many scholars. Based on the HSS iteration method, many improved and generalized corresponding iteration methods have been proposed.

The HSS iteration method [9]**:** Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the iterative sequence $\{x^{(k)}\}$ converges, calculate the program

$$\begin{cases} (\alpha I + W)x^{(k+1/2)} = (\alpha I - \mathrm{i}T)x^{(k)} + b, \\ (\alpha I + \mathrm{i}T)x^{(k+1)} = (\alpha I - W)x^{(k+1/2)} + b, \end{cases}$$

where $\alpha > 0$, and I is the identity matrix.

The HSS iteration method requires solving a skew-Hermitian linear system for each iteration, which is undoubtedly difficult and time-consuming. Then Bai et al. cleverly designed an improved HSS (MHSS) method [6]. For more details, we can refer to [11].

The MHSS iteration method [6]**:** Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the iterative sequence $\{x^{(k)}\}$ converges, calculate the program

$$\begin{cases} (\alpha I + W)x^{(k+1/2)} = (\alpha I - \mathrm{i}T)x^{(k)} + b, \\ (\alpha I + T)x^{(k+1)} = (\alpha I + \mathrm{i}W)x^{(k+1/2)} - \mathrm{i}b, \end{cases}$$

where $\alpha > 0$, and I is the identity matrix.

In order to further improve the convergence speed of the MHSS iteration method, Bai et al. combined preconditioning technology with the MHSS iteration method to establish a preconditioned MHSS (PMHSS) method [7]. This method can be expressed as:

The PMHSS iteration method [7]**:** Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the iterative sequence $\{x^{(k)}\}$ converges, calculate the program

$$\begin{cases} (\alpha V + W)x^{(k+1/2)} = (\alpha V - \mathrm{i}T)x^{(k)} + b, \\ (\alpha V + T)x^{(k+1)} = (\alpha V + \mathrm{i}W)x^{(k+1/2)} - \mathrm{i}b, \end{cases}$$

where $\alpha > 0$, and V and I are the symmetric positive definite matrix and identity matrix, respectively.

In order to achieve better numerical behaviors of the iteration methods, Zheng et al. proposed a two-step scale splitting (DSS) method in [44] for the complex symmetric linear system (1.1). The specific format is:

The DSS iteration method [44]**:** Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. Compute the next iterate $x^{(k+1)}$ according to the following program

$$\begin{cases} (\alpha W + T)x^{(k+1/2)} = \mathrm{i}(W - \alpha T)x^{(k)} + (\alpha - \mathrm{i})b, \\ (\alpha T + W)x^{(k+1)} = \mathrm{i}(\alpha W - T)x^{(k+1/2)} + (1 - \mathrm{i}\alpha)b, \end{cases}$$

and $\alpha > 0$.

In order to reduce complex operations in the iterative system and further accelerate the convergence speed of the DSS method, Zhang et al. used real value acceleration technology to establish a double-step scale splitting real-valued (DSS real-valued) method in [41]. The specific format is as follows:

The DSS real-valued iteration method [41]**:** Let $y^{(0)}$, $z^{(0)} \in \mathbb{R}^n$ be the arbitrary initial guesses. For $k = 0, 1, 2, \ldots$ until the iterative sequences ${y^{(k)}}^\top, {z^{(k)}}^\top$ converge, calculate the program

$$\begin{cases} \begin{bmatrix} \alpha T + W & 0 \\ 0 & \alpha T + W \end{bmatrix} \begin{bmatrix} y^{(k+1/2)} \\ z^{(k+1/2)} \end{bmatrix} = \begin{bmatrix} \alpha T & T \\ -T & \alpha T \end{bmatrix} \begin{bmatrix} y^{(k)} \\ z^{(k)} \end{bmatrix} + \begin{bmatrix} p \\ q \end{bmatrix}, \\ \begin{bmatrix} \alpha W + T & 0 \\ 0 & \alpha W + T \end{bmatrix} \begin{bmatrix} y^{(k+1)} \\ z^{(k+1)} \end{bmatrix} = \begin{bmatrix} \alpha W & -W \\ W & \alpha W \end{bmatrix} \begin{bmatrix} y^{(k+1/2)} \\ z^{(k+1/2)} \end{bmatrix} + \begin{bmatrix} q \\ -p \end{bmatrix}, \end{cases}$$

where $\alpha > 0$, and I is the identity matrix.

In addition, many effective iteration methods are established, such as accelerated HSS (AHSS) [8], quasi-HSS (QHSS) [5], lopsided HSS (LHSS) [25], single-step HSS (SHSS) [24], single-step preconditioned HSS (SPHSS) [34], modified QHSS (MQHSS) [14], preconditioned MQHSS (PMQHSS) [23], lopsided PMHSS (LPMHSS) [26], generalized PMHSS (GPMHSS) [15], efficient block splitting (PBS) [20], modified two-step scale-splitting (MTSS) [21], new double-step scale splitting (NDSS) [19], minimum residual HSS (MRHSS) [36], [37], minimum residual MHSS (MRMHSS) [43], new Hermitian and skew-Hermitian splitting

(NHSS) [27], preconditioned NHSS (P*NHSS) [35], real valued [2], preconditioned HSS (PHSS) [10], preconditioned accelerated generalized successive overrelaxation (PAGSOR) [22], single step [29], combination real part and imaginary (CRI) [32], simplified HNS (SHNS) [33], new block preconditioner [40], preconditioned SHNS (PSHNS) [39], relaxed block splitting preconditioner [18], improved block (IB) splitting preconditioner [42], inexact MQHSS (IMQHSS) method [38] and so forth. Now, we briefly introduce the CRI iteration method [32].

The CRI iteration method [32]**:** Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. Compute the next iterate $x^{(k+1)}$ from

$$\begin{cases} (\alpha T + W)x^{(k+1/2)} = (\alpha - \mathrm{i})Tx^{(k)} + b, \\ (\alpha W + T)x^{(k+1)} = (\alpha + \mathrm{i})Wx^{(k+1/2)} - \mathrm{i}b, \quad k = 0, 1, 2, \ldots, \end{cases}$$

where $\alpha > 0$.

We give an equivalent form of linear equations (1.1) below

$$(1 - \mathrm{i}\omega)Ax = (1 - \mathrm{i}\omega)b \Leftrightarrow (W + \omega T + \mathrm{i}T - \mathrm{i}\omega W)x = (1 - \mathrm{i}\omega)b.$$

After simple arrangements, a new linear system equivalent to linear equations (1.1) can be obtained

$$\widehat{A}x = \widehat{b}, \tag{1.2}$$

where $\widehat{A} = H_\omega + \mathrm{i}T$, $H_\omega = W + \omega T$, $\widehat{b} = \mathrm{i}\omega Wx + (1 - \mathrm{i}\omega)b$, $\omega \geqslant 0$.

In order to get an iteration method with better convergence to solve linear system (1.2), this paper applies the idea of the double-step splitting (DSS) method to obtain a dual-parameter double-step splitting (DDSS) iteration method. We theoretically analyze the convergence properties of the DDSS method. In addition, the experiments in Section 4 also illustrate the effectiveness of our method.

The paper is organized as follows. In Section 2, we apply the idea of the double-step splitting (DSS) method to propose a dual-parameter double-step splitting (DDSS) iteration method. In Section 3, convergence conditions making the upper bound of the spectral radius of the iteration matrix less than 1 are given. In addition, two tested problems are reported to verify the effectiveness of our method in Section 4. Finally, Section 5 draws a conclusion.

In the following chapters, the symbol $\kappa(A)$ denotes the number of spectral conditions of a given matrix A. The notations $\varrho(A)$ and $\mathrm{sp}(A)$ indicate the spectral radius and spectrum of a given matrix A, respectively. In addition, $\|A\|_2$ and $\|x\|_2$ stand for the Euclidean norm of a matrix A and vector x, respectively.

2. The DDSS iteration method

In this section, to process complex linear equations (1.1), inspired by the DSS iteration method, we propose a dual-parameter double-step splitting iteration method, which will be simply called the DDSS iteration method.

Multiplying both sides of linear equation (1.2) by $\alpha - \mathrm{i}$ leads to

$$(\alpha H_\omega + T)x = \mathrm{i}(H_\omega - \alpha T)x + \mathrm{i}(\alpha - \mathrm{i})\omega W x + (\alpha - \mathrm{i})(1 - \mathrm{i}\omega)b.$$

Then we can construct the iterative scheme from the above equation as follows:

$$(\alpha H_\omega + T)x^{(k+1/2)} = \mathrm{i}(H_\omega - \alpha T)x^{(k)} + \mathrm{i}(\alpha - \mathrm{i})\omega W x^{(k)} + (\alpha - \mathrm{i})(1 - \mathrm{i}\omega)b. \tag{2.1}$$

Meanwhile, we multiply $1 - \mathrm{i}\alpha$ on both sides for (1.2) to have

$$(\alpha T + H_\omega)x = \mathrm{i}(\alpha H_\omega - T)x + \mathrm{i}(1 - \mathrm{i}\alpha)\omega W x + (1 - \mathrm{i}\alpha)(1 - \mathrm{i}\omega)b,$$

and the corresponding iterative scheme of the above equation is

$$(\alpha T + H_\omega)x^{(k+1)} = \mathrm{i}(\alpha H_\omega - T)x^{(k+1/2)} + \mathrm{i}(1 - \mathrm{i}\alpha)\omega W x^{(k)} + (1 - \mathrm{i}\alpha)(1 - \mathrm{i}\omega)b. \tag{2.2}$$

Combining the iterative schemes (2.1) and (2.2), we can get a dual-parameter double-step splitting (DDSS) iteration method. The specific representation of the DDSS method is given below.

The DDSS iteration method: Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. Compute the next iterate $x^{(k+1)}$ according to the following procedure:

$$\begin{cases} (\alpha H_\omega + T)x^{(k+1/2)} = \mathrm{i}(H_\omega - \alpha T)x^{(k)} + \mathrm{i}(\alpha - \mathrm{i})\omega W x^{(k)} + (\alpha - \mathrm{i})(1 - \mathrm{i}\omega)b, \\ (\alpha T + H_\omega)x^{(k+1)} = \mathrm{i}(\alpha H_\omega - T)x^{(k+1/2)} + \mathrm{i}(1 - \mathrm{i}\alpha)\omega 1 W x^{(k)} + (1 - \mathrm{i}\alpha)(1 - \mathrm{i}\omega)b, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad k = 0, 1, 2, \ldots. \end{cases} \tag{2.3}$$

Here, $H_\omega = W + \omega T$, $\alpha > 0$, $\omega \geqslant 0$.

By substituting the first half-iterate $x^{(k+1/2)}$ into the second half-iterate $x^{(k+1)}$ in (2.3), we can briefly rewrite the DDSS iteration method into a standard stationary matrix splitting iteration method as follows:

$$x^{k+1} = G_{(\alpha,\omega)}x^k + R_{(\alpha,\omega)}b.$$

Here,

$$\begin{aligned} G_{(\alpha,\omega)} = {} & (\alpha T + H_\omega)^{-1}(\alpha H_\omega - T)(\alpha H_\omega + T)^{-1}(\alpha T - H_\omega) \\ & + 2\alpha\omega(\alpha T + H_\omega)^{-1}(T + \mathrm{i}H_\omega)(\alpha H_\omega + T)^{-1}W \end{aligned} \tag{2.4}$$

and

$$R_{(\alpha,\omega)} = 2\alpha(1 - \mathrm{i}\omega)(\alpha T + H_\omega)^{-1}(H_\omega - \mathrm{i}T)(\alpha H_\omega + T)^{-1}.$$

The matrix $G_{(\alpha,\omega)}$ is the iteration matrix of the DDSS method. And the matrix

$$R^{-1}_{(\alpha,\omega)} = \frac{1}{2\alpha(1-\mathrm{i}\omega)}(\alpha H_\omega + T)(H_\omega - \mathrm{i}T)^{-1}(\alpha T + H_\omega)$$

can be viewed as a preconditioner for the coefficient matrix A. We call it the DDSS preconditioner.

From iterative scheme (2.3), we will give the algorithm version of the DDSS iteration method below.

Algorithm 2.1. The algorithm representation of the DDSS iteration method:

Step 1: Calculate $b_1 = (1-\mathrm{i}\omega)b$, $b_2 = (\alpha-\mathrm{i})b_1$, $b_3 = (1-\mathrm{i}\alpha)b_1$ and $H_\omega = W+\omega T$;
Step 2: Calculate $t^{(k)} = \mathrm{i}(H_\omega - \alpha T)x^{(k)} + \mathrm{i}(\alpha - \mathrm{i})\omega W x^{(k)} + b_2$;
Step 3: Solve $(\alpha H_\omega + T)x^{(k+1/2)} = t^{(k)}$;
Step 4: Calculate $s^{(k)} = \mathrm{i}(\alpha H_\omega - T)x^{(k+1/2)} + \mathrm{i}(1 - \mathrm{i}\alpha)\omega W x^{(k)} + b_3$;
Step 5: Solve $(\alpha T + H_\omega)x^{(k+1)} = s^{(k)}$.

3. Convergence property

In this section, we give the upper bound of the spectral radius of the iteration matrix of the DDSS method and investigate its convergence properties for solving the complex symmetric linear equations. In order to discuss the properties of the method more conveniently, we define a function as in [44] by

$$f(x) = x + \frac{1}{x}. \tag{3.1}$$

Now we present the following lemmas, which are used to prove the convergence of the DDSS method.

Lemma 3.1. *Let $A = W+\mathrm{i}T \in \mathbb{C}^{n\times n}$, $W \in \mathbb{R}^{n\times n}$ be SPSD matrix, $T \in \mathbb{R}^{n\times n}$ be SPD matrix, $\omega \geqslant 0$ and $\alpha > 0$. Let $\mu_1 \geqslant \mu_2 \ldots \geqslant \mu_n$ be the eigenvalues of the matrix $T^{-1/2}WT^{-1/2}$, $\lambda_1 \geqslant \lambda_2 \ldots \geqslant \lambda_n$ be the eigenvalues of the matrix $T^{-1/2}H_\omega T^{-1/2}$ with $H_\omega = W + \omega T$, and let $\lambda_{\max} = \lambda_1$, $\lambda_{\min} = \lambda_n$, $\mu_{\max} = \mu_1$, $\mu_{\min} = \mu_n$. Then $\lambda_j = \mu_j + \omega$, $\lambda_{\min} = \mu_{\min} + \omega$ and $\lambda_{\max} = \mu_{\max} + \omega$.*

Proof. The proof of this lemma is simple, and thus it is omitted. □

Exploiting a quite similar strategy utilized in the proof of Lemma 2.4 in [44], the following lemma can be established.

Lemma 3.2. *Assume that the conditions in Lemma* 3.1 *are satisfied. Let* $f_{\min}(\lambda)$ *and* $f_{\max}(\lambda)$ *be the smallest and largest values of the function* $f(\lambda)$*, respectively. Then*

$$f_{\max}(\lambda) = \begin{cases} \lambda_{\min} + \dfrac{1}{\lambda_{\min}}, & \lambda_{\max} < 1 \quad (0 \leqslant \omega < 1 - \mu_{\max}), \\ \max\left\{\lambda_{\min} + \dfrac{1}{\lambda_{\min}}, \lambda_{\max} + \dfrac{1}{\lambda_{\max}}\right\}, & \lambda_{\min} \leqslant 1 \leqslant \lambda_{\max} \\ & (1 - \mu_{\max} \leqslant \omega \leqslant 1 - \mu_{\min}), \\ \lambda_{\max} + \dfrac{1}{\lambda_{\max}}, & \lambda_{\min} > 1 \quad (\omega > 1 - \mu_{\min}), \end{cases}$$

$$f_{\min}(\lambda) = \begin{cases} \lambda_{\max} + \dfrac{1}{\lambda_{\max}}, & \lambda_{\max} < 1 \quad (0 \leqslant \omega < 1 - \mu_{\max}), \\ 2, \quad \lambda_{\min} \leqslant 1 \leqslant \lambda_{\max} & (1 - \mu_{\max} \leqslant \omega \leqslant 1 - \mu_{\min}), \\ \lambda_{\min} + \dfrac{1}{\lambda_{\min}}, & \lambda_{\min} > 1 \quad (\omega > 1 - \mu_{\min}). \end{cases}$$

P r o o f. By analogy with the definition of the function in (3.1), we give the function $f(\lambda)$ as

$$f(\lambda) = \lambda + \frac{1}{\lambda}.$$

Then

$$\frac{\mathrm{d}f(\lambda)}{\mathrm{d}\lambda} = \frac{\lambda^2 - 1}{\lambda^2}.$$

(1) If $\lambda_{\max} < 1$, then $\mathrm{d}f(\lambda)/\mathrm{d}\lambda = (\lambda^2 - 1)/\lambda^2 < 0$ with respect to all λ, so the function $f(\lambda)$ is monotonically decreasing function with respect to all λ. Then the function $f(\lambda)$ obtains the maximum and the minimum values at $\lambda_{\min}$ and $\lambda_{\max}$, respectively. Therefore, $f_{\max}(\lambda) = \lambda_{\min} + 1/\lambda_{\min}$ and $f_{\min}(\lambda) = \lambda_{\max} + 1/\lambda_{\max}$.

(2) If $\lambda_{\min} > 1$, then $\mathrm{d}f(\lambda)/\mathrm{d}\lambda = (\lambda^2 - 1)/\lambda^2 > 0$ with respect to all λ. Owing to the fact that the function $f(\lambda)$ is monotonically increasing function with respect to all λ, we have $f_{\max}(\lambda) = \lambda_{\max} + 1/\lambda_{\max}$ and $f_{\min}(\lambda) = \lambda_{\min} + 1/\lambda_{\min}$.

(3) If $\lambda_{\min} \leqslant 1 \leqslant \lambda_{\max}$, when $\lambda_{\min} \leqslant \lambda \leqslant 1$, then $\mathrm{d}f(\lambda)/\mathrm{d}\lambda = (\lambda^2 - 1)/\lambda^2 < 0$, and the function $f(\lambda)$ is monotonically decreasing function for $\lambda_{\min} \leqslant \lambda \leqslant 1$. When $1 \leqslant \lambda \leqslant \lambda_{\max}$, then $\mathrm{d}f(\lambda)/\mathrm{d}\lambda = (\lambda^2 - 1)/\lambda^2 > 0$, and the function $f(\lambda)$ is monotonically increasing function for $1 \leqslant \lambda \leqslant \lambda_{\max}$. Therefore, the function $f(\lambda)$ obtains the minimum value at $\lambda = 1$. Moreover, the minimum value is $f_{\min}(\lambda) = 2$. Furthermore, it is not difficult to see that $f_{\max}(\lambda) = \max\{\lambda_{\min} + 1/\lambda_{\min}, \lambda_{\max} + 1/\lambda_{\max}\}$. □

Theorem 3.1. *Assume that the conditions in Lemmas* 3.1 *and* 3.2 *are satisfied. Then*

$$\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)}, \tag{3.2}$$

where

$$\sigma_{(\alpha,\omega)} = \begin{cases} \dfrac{f(\alpha) - f_{\min}(\lambda)}{f(\alpha) + f_{\min}(\lambda)} + \dfrac{\eta}{f(\alpha) + f_{\min}(\lambda)}, & f(\alpha) \geqslant \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}, \\ \dfrac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \dfrac{\eta}{f(\alpha) + f_{\min}(\lambda)}, & f(\alpha) \leqslant \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} \end{cases}$$

with the function $f(\alpha)$ *being defined in* (3.1) *and* $\eta = \lambda_{\max}\sqrt{1 + \lambda_{\max}^2}$.

P r o o f. From (2.4) we have

$$\begin{aligned}
\varrho(G_{(\alpha,\omega)}) &= \varrho((\alpha T + H_\omega)^{-1}(\alpha H_\omega - T)(\alpha H_\omega + T)^{-1}(\alpha T - H_\omega) \\
&\quad + 2\alpha\omega(\alpha T + H_\omega)^{-1}(T + \mathrm{i}H_\omega)(\alpha H_\omega + T)^{-1}W) \\
&= \varrho(T^{-1/2}(\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(\alpha T^{-1/2}H_\omega T^{-1/2} - I) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}(\alpha I - T^{-1/2}H_\omega T^{-1/2})T^{1/2} \\
&\quad + 2\alpha\omega T^{-1/2}(\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(I + \mathrm{i}T^{-1/2}H_\omega T^{-1/2}) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}T^{-1/2}WT^{-1/2}T^{1/2}) \\
&= \varrho((\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(\alpha T^{-1/2}H_\omega T^{-1/2} - I) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}(\alpha I - T^{-1/2}H_\omega T^{-1/2}) \\
&\quad + 2\alpha\omega(\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(I + \mathrm{i}T^{-1/2}H_\omega T^{-1/2}) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}T^{-1/2}WT^{-1/2}) \\
&\leqslant \|(\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(\alpha T^{-1/2}H_\omega T^{-1/2} - I) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}(\alpha I - T^{-1/2}H_\omega T^{-1/2})\|_2 \\
&\quad + 2\alpha\omega\|(\alpha I + T^{-1/2}H_\omega T^{-1/2})^{-1}(I + \mathrm{i}T^{-1/2}H_\omega T^{-1/2}) \\
&\quad \times (\alpha T^{-1/2}H_\omega T^{-1/2} + I)^{-1}T^{-1/2}WT^{-1/2}\|_2 \\
&= \max_{\lambda_l \in \mathrm{sp}(T^{-1}H_\omega)} \frac{|(\alpha - \lambda_l)(\alpha\lambda_l - 1)|}{(\alpha + \lambda_l)(\alpha\lambda_l + 1)} + \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{2\alpha\omega\mu_j\sqrt{1 + \lambda_j^2}}{(\alpha + \lambda_j)(\alpha\lambda_j + 1)} \\
&= \max_{\lambda_l \in \mathrm{sp}(T^{-1}H_\omega)} \frac{|(\alpha + 1/\alpha) - (\lambda_l + 1/\lambda_l)|}{(\alpha + 1/\alpha) + (\lambda_l + 1/\lambda_l)} + \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{2\alpha\omega\mu_j\sqrt{1 + \lambda_j^2}}{(\alpha + \lambda_j)(\alpha\lambda_j + 1)} \\
&= \max_{\lambda_l \in \mathrm{sp}(T^{-1}H_\omega)} \frac{|f(\alpha) - f(\lambda_l)|}{f(\alpha) + f(\lambda_l)} + \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{2\alpha\omega\mu_j\sqrt{1 + \lambda_j^2}}{(\alpha + \lambda_j)(\alpha\lambda_j + 1)},
\end{aligned}$$

where $\lambda_j = \omega + \mu_j$. In addition, it is not difficult to verify that the inequality

$$2\omega\mu_j \leqslant \omega^2 + \mu_j^2 \leqslant (\omega + \mu_j)^2 = \lambda_j^2$$

holds. Thus, we have

$$\begin{aligned}(3.3)\qquad \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{2\alpha\omega\mu_j\sqrt{1+\lambda_j^2}}{(\alpha+\lambda_j)(\alpha\lambda_j+1)} &\leqslant \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{\alpha\lambda_j^2\sqrt{1+\lambda_j^2}}{(\alpha^2\lambda_j+\lambda_j)+(\alpha\lambda_j^2+\alpha)} \\ &= \max_{\mu_j \in \mathrm{sp}(T^{-1}W)} \frac{\lambda_j\sqrt{1+\lambda_j^2}}{(\alpha+1/\alpha)+(\lambda_j+1/\lambda_j)} \\ &\leqslant \frac{\lambda_{\max}\sqrt{1+\lambda_{\max}^2}}{f(\alpha)+f_{\min}(\lambda)} = \frac{\eta}{f(\alpha)+f_{\min}(\lambda)},\end{aligned}$$

where $\eta = \lambda_{\max}\sqrt{1+\lambda_{\max}^2}$. According to the results in [9] and [44], one has

$$(3.4)\qquad \max_{\mu_l \in \mathrm{sp}(T^{-1}W)} \frac{|f(\alpha)-f(\lambda_l)|}{f(\alpha)+f(\lambda_l)} = \begin{cases} \dfrac{f(\alpha)-f_{\min}(\lambda)}{f(\alpha)+f(\lambda)_{\min}}, & f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}, \\ \dfrac{f_{\max}(\lambda)-f(\alpha)}{f(\alpha)+f_{\max}(\lambda)}, & f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}. \end{cases}$$

The following results can be obtained by sorting out equations (3.3) and (3.4) above:

$$\varrho(G_{(\alpha,\omega)}) \leqslant \begin{cases} \dfrac{f_{\min}(\alpha)-f(\lambda)}{f(\alpha)+f_{\min}(\lambda)} + \dfrac{\eta}{f(\alpha)+f_{\min}(\lambda)}, & f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} \\ \dfrac{f_{\max}(\lambda)-f(\alpha)}{f(\alpha)+f(\lambda)_{\max}} + \dfrac{\eta}{f(\alpha)+f_{\min}(\lambda)}, & f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} \end{cases} \triangleq \sigma_{(\alpha,\omega)}.$$

To sum up, Theorem 3.1 can be obtained. □

Theorem 3.2. *Assume that the conditions in Theorem* 3.1 *are satisfied. Then:*

(a) *If* $\eta < 2f_{\min}(\lambda)$ *and* $f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, *then* $\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} < 1$.

(b) *If* $\eta < 2f_{\min}(\lambda)$ *and* $(\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4 < f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, *then* $\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} < 1$, *where* $\triangle = (2f_{\min}(\lambda) - \eta)^2 + 8\eta f_{\max}(\lambda)$.

P r o o f. It follows from Theorem 3.1 that

$$\begin{aligned}\varrho(G_{(\alpha,\omega)}) &\leqslant \sigma_{(\alpha,\omega)} \\ &= \begin{cases} \dfrac{f(\alpha)-f(\lambda)_{\min}}{f(\alpha)+f(\lambda)_{\min}} + \dfrac{\eta}{f(\alpha)+f(\lambda)_{\min}}, & f(\alpha) \geqslant \sqrt{f(\lambda)_{\min}f(\lambda)_{\max}}, \\ \dfrac{f(\lambda)_{\max}-f(\alpha)}{f(\alpha)+f(\lambda)_{\max}} + \dfrac{\eta}{f(\alpha)+f(\lambda)_{\min}}, & f(\alpha) \leqslant \sqrt{f(\lambda)_{\min}f(\lambda)_{\max}}. \end{cases}\end{aligned}$$

(1) If $f(\alpha) \geqslant \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}$, then

$$\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} = \frac{f(\alpha) - f_{\min}(\lambda)}{f(\alpha) + f_{\min}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

Letting

$$\sigma_{(\alpha,\omega)} = \frac{f(\alpha) - f_{\min}(\lambda)}{f(\alpha) + f_{\min}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)} < 1,$$

we can get $\eta < 2f_{\min}(\lambda)$.

(2) If $f(\alpha) \leqslant \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}$, one has

$$\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} = \frac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

Let

$$\sigma_{(\alpha,\omega)} = \frac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)} < 1.$$

Then

$$\begin{aligned} &\frac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)} < 1 \\ &\quad \Leftrightarrow (f_{\max}(\lambda) - f(\alpha))(f(\alpha) + f_{\min}(\lambda)) + \eta(f(\alpha) + f_{\max}(\lambda)) \\ &\qquad\qquad < (f(\alpha) + f_{\max}(\lambda))(f(\alpha) + f(\lambda)_{\min}) \\ &\quad \Leftrightarrow 2f(\alpha)^2 + (2f_{\min}(\lambda) - \eta) f(\alpha) - \eta f_{\max}(\lambda) > 0. \end{aligned} \tag{3.5}$$

In order to make inequality (3.5) valid, we consider the corresponding quadratic equation with respect to $f(\alpha)$ as follows

$$2f(\alpha)^2 + (2f_{\min}(\lambda) - \eta) f(\alpha) - \eta f_{\max}(\lambda) = 0. \tag{3.6}$$

The discriminant of the root of the equation is

$$\triangle = (2f_{\min}(\lambda) - \eta)^2 + 8\eta f_{\max}(\lambda) > 0.$$

Therefore, it can be obtained that the two roots of the quadratic equation (3.6) with respect to $f(\alpha)$ are

$$f_1(\alpha) = \frac{\eta - 2f_{\min}(\lambda) - \sqrt{\triangle}}{4} \quad \text{and} \quad f_2(\alpha) = \frac{\eta - 2f_{\min}(\lambda) + \sqrt{\triangle}}{4}.$$

Then the solution set of inequality (3.5) is $f(\alpha) > (\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4$. Next, we discuss the conditions under which the inequality $(\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4 <$

$\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$ holds. If the value of the function $2f(\alpha)^2 + (2f_{\min}(\lambda) - \eta)f(\alpha) - \eta f_{\max}(\lambda)$ on the left-hand side of inequality (3.5) at $f(\alpha) = \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$ is greater than 0, then it yields $\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} > (\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4$. So substituting $f(\alpha) = \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$ into the left-hand side of inequality (3.5) leads to

$$\begin{aligned} &2f_{\min}(\lambda)f_{\max}(\lambda) + (2f_{\min}(\lambda) - \eta)\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} - \eta f_{\max}(\lambda) \\ &\qquad = (2f_{\min}(\lambda) - \eta)(f_{\max}(\lambda) + \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}). \end{aligned}$$

(a) If $2f_{\min}(\lambda) - \eta \leqslant 0$, that is, $2f(\lambda)_{\min} \leqslant \eta$, we have $(2f_{\min}(\lambda) - \eta)(f_{\max}(\lambda) + \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}) \leqslant 0$. Then the parameters α and ω making inequality (3.5) valid do not exist.

(b) If $2f_{\min}(\lambda) - \eta > 0$, that is, $2f_{\min}(\lambda) > \eta$, we have $(2f_{\min}(\lambda) - \eta)(f_{\max}(\lambda) + \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}) > 0$. Then we deduce that

$$\frac{\eta - 2f_{\min}(\lambda) + \sqrt{\triangle}}{4} < \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}.$$

Thus, for this case that $(\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4 < f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, inequality (3.5) holds, that is, $\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} < 1$. □

The result of Theorem 3.2 can be obtained by sorting out the results of (1.1) and (1.2) above.

Theorem 3.3. *Assume that the conditions in Theorem* 3.1 *are satisfied. Then the quasi-optimal value of the parameter α which minimizes the upper bound $\sigma_{(\alpha,\omega)}$ is*

$$\alpha_1^* = \frac{2}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + \sqrt{(f_{\min}(\lambda)f_{\max}(\lambda))^2 - 4}} \quad \text{or } \alpha_2^* = \frac{1}{\alpha_1^*}.$$

And the corresponding optimal upper bound of spectral radius is

$$\sigma_{(\alpha_1^*,\omega)} = \sigma_{(\alpha_2^*,\omega)} = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} + \frac{\eta}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\min}(\lambda)} \quad \text{with } \kappa = \frac{f_{\max}(\lambda)}{f_{\min}(\lambda)}.$$

P r o o f. It follows from Theorems 3.1 and 3.2 that:

(1) If $f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, then

$$\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} = \frac{f(\alpha) - f_{\min}(\lambda)}{f(\alpha) + f_{\min}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

When $\eta < 2f_{\min}(\lambda)$ and $f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, then $\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} < 1$. Let

$$g(x) = \frac{x - f_{\min}(\lambda)}{x + f_{\min}(\lambda)} + \frac{\eta}{x + f_{\min}(\lambda)}.$$

Then

$$\sigma_{(\alpha,\omega)} = g(f(\alpha)) = \frac{f(\alpha) - f_{\min}(\lambda)}{f(\alpha) + f_{\min}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

By simple calculations, we have

$$\frac{\mathrm{d}g(f(\alpha))}{\mathrm{d}f(\alpha)} = \frac{2f_{\min}(\lambda) - \eta}{(f(\alpha) + f_{\min}(\lambda))^2} > 0.$$

Then the function $g(f(\alpha))$ is monotonic increasing about the variable $f(\alpha)$ for $\eta < 2f_{\min}(\lambda)$ and $f(\alpha) \geqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$. So the function $g(f(\alpha))$ gets the minimum value at $f(\alpha) = \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, and the minimum value is

$$\begin{aligned}(3.7)\qquad &\frac{\sqrt{f_{\min}(\lambda)f(\lambda)_{\max}} - f_{\min}(\lambda)}{\sqrt{f_{\min}(\lambda)f(\lambda)_{\max}} + f_{\min}(\lambda)} + \frac{\eta}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\min}(\lambda)} \\ &\qquad = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} + \frac{\eta}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\min}(\lambda)}.\end{aligned}$$

(2) If $f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, then

$$\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} = \frac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

When $\eta < 2f_{\min}(\lambda)$ and

$$\frac{\eta - 2f_{\min}(\lambda) + \sqrt{\triangle}}{4} < f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)},$$

then $\varrho(G_{(\alpha,\omega)}) \leqslant \sigma_{(\alpha,\omega)} < 1$. Letting

$$h(x) = \frac{f_{\max}(\lambda) - x}{x + f_{\max}(\lambda)} + \frac{\eta}{x + f_{\min}(\lambda)},$$

we have

$$\sigma_{(\alpha,\omega)} = h(f(\alpha)) = \frac{f_{\max}(\lambda) - f(\alpha)}{f(\alpha) + f_{\max}(\lambda)} + \frac{\eta}{f(\alpha) + f_{\min}(\lambda)}.$$

Then

$$\frac{\mathrm{d}h(f(\alpha))}{\mathrm{d}f(\alpha)} = -\frac{2f_{\max}(\lambda)}{(f(\alpha) + f_{\max}(\lambda))^2} - \frac{\eta}{(f(\alpha) + f_{\min}(\lambda))^2} < 0.$$

Thus the function $h(f(\alpha))$ decreases monotonically for $(\eta - 2f_{\min}(\lambda) + \sqrt{\triangle})/4 < f(\alpha) \leqslant \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$. For this case, the function $h(f(\alpha))$ gets the minimum value at $f(\alpha) = \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}$, and the minimum value is

$$\begin{aligned}(3.8)\qquad &\frac{f_{\max}(\lambda) - \sqrt{f_{\min}(\lambda)f_{\max}(\lambda)}}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\max}(\lambda)} + \frac{\eta}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\min}(\lambda)} \\ &\qquad = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} + \frac{\eta}{\sqrt{f_{\min}(\lambda)f_{\max}(\lambda)} + f_{\min}(\lambda)}.\end{aligned}$$

From equations (3.7) and (3.8), we deduce that the optimal upper bound of spectral radius which can be obtained at $f(\alpha) = \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}$ is

$$\begin{aligned}
&\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} + \frac{\eta}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + f_{\min}(\lambda)} \\
&\quad = \frac{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} - f_{\min}(\lambda)}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + f_{\min}(\lambda)} + \frac{\eta}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + f_{\min}(\lambda)} \\
&\quad = \frac{f_{\min}(\lambda) - \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + f_{\max}(\lambda)} + \frac{\eta}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + f_{\min}(\lambda)}.
\end{aligned}$$

For the equation $f(\alpha^*) = \alpha^* + 1/\alpha^* = \sqrt{f_{\min}(\lambda) f_{\max}(\lambda)}$, after simple calculations, we can get that the quasi-optimal parameter is

$$\alpha_1^* = \frac{2}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + \sqrt{(f_{\min}(\lambda) f_{\max}(\lambda))^2 - 4}} \quad \text{or } \alpha_2^* = \frac{1}{\alpha_1^*}.$$

The proof of Theorem 3.3 is completed. □

4. Test problems

In this section, two numerical examples are given to show the effectiveness of the proposed method and preconditioner. We compare the numerical behaviors of the DDSS method with the DSS real-valued, CRI, PMHSS, DSS methods in terms of both the number of iteration steps (denoted by "IT"), the elapsed CPU time in seconds (denoted by "CPU") and the relative residual (denoted by "RES"). The numerical experiment results are run in MATLAB2018b on PC with Intel(R) Core(TM) i5 CPU 1.19Hz and 8GB.

In all calculations below, we set the initial vector as $x^{(0)} = \mathbf{0}$. All iterations are terminated once the current relative residual norm satisfies

$$\frac{\|r^{(k)}\|_2}{\|r^{(0)}\|_2} = \frac{\|b - Ax^{(k)}\|_2}{\|b\|_2} < 10^{-8},$$

or the number of the prescribed iteration step $k_{\max} = 10m$ (m is the problem size of the following numerical examples) is exceeded.

Example 4.1 ([6], [7]). We give a linear system of equations in the form of system (1.1):

$$Ax = (W + \mathrm{i}T)x = b.$$

Let $b = (1 + \mathrm{i})A\mathbf{1}$ and $\mathbf{1}$ be the vector whose all entries equal to 1, and

$$T = I \otimes B_m + B_m \otimes I, \quad W = 10(I \otimes B + B \otimes I) + 9(e_1 e_m^\top + e_m e_1^\top) \otimes I,$$

$B_m = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{n\times n}$, $B = B_m - e_1 e_m^\top - e_m e_1^\top \in \mathbb{R}^{n\times n}$, where e_1, e_m are the 1th and the mth unity basis vectors in $\mathbb{R}^m$, respectively.

E x a m p l e 4.2 ([6], [2]). We consider a linear system of equations in the form of system (1.1):

$$[(V - \kappa I) + \mathrm{i}(V + \varepsilon I)]x = b,$$

where κ is the time step-size and V is the five-point centered difference matrix obtained by approximating the negative Laplacian operator $L = -\triangle$ with homogeneous Dirichlet boundary conditions, on a uniform mesh in the unit square $[0, 1] \times [0, 1]$ with the mesh-size $h = (m + 1)^{-1}$. Let $V = I \otimes V_m + V_m \otimes I \in \mathbb{R}^{n\times n}$, with $V_m = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$. From the calculation form of V, it is easy to conclude that V is an $n \times n$ block-tridiagonal matrix. We set $\kappa = 50$, $\varepsilon = 400$ and the right-hand side vector $b = (1+\mathrm{i})A\mathbf{1}$, with $\mathbf{1}$ being the vector of all entries equal to 1.

4.1. The parameter selection strategy of the DDSS iteration method. Deriving the efficient selection strategy of parameters involved in the DDSS method is significant to obtain fast convergence rate. This subsection discusses the choice strategy of the feasible parameter α of the DDSS method.

It follows from Lemma 3.2 and Theorem 3.3 that the quasi-optimal value α^* of the parameter α involved in the DDSS method is

$$\alpha_1^* = \frac{2}{\sqrt{f_{\min}(\lambda) f_{\max}(\lambda)} + \sqrt{(f_{\min}(\lambda) f_{\max}(\lambda))^2 - 4}} \quad \text{or } \alpha_2^* = \frac{1}{\alpha_1^*},$$

where

$$f_{\max}(\lambda) = \begin{cases} \lambda_{\min} + \dfrac{1}{\lambda_{\min}}, & \lambda_{\max} < 1 \quad (0 \leqslant \omega < 1 - \mu_{\max}), \\ \max\left\{\lambda_{\min} + \dfrac{1}{\lambda_{\min}}, \lambda_{\max} + \dfrac{1}{\lambda_{\max}}\right\}, & \lambda_{\min} \leqslant 1 \leqslant \lambda_{\max} \\ & (1 - \mu_{\max} \leqslant \omega \leqslant 1 - \mu_{\min}), \\ \lambda_{\max} + \dfrac{1}{\lambda_{\max}}, & \lambda_{\min} > 1 \quad (\omega > 1 - \mu_{\min}), \end{cases}$$

$$f_{\min}(\lambda) = \begin{cases} \lambda_{\max} + \dfrac{1}{\lambda_{\max}}, & \lambda_{\max} < 1 \quad (0 \leqslant \omega < 1 - \mu_{\max}), \\ 2, \quad \lambda_{\min} \leqslant 1 \leqslant \lambda_{\max} & (1 - \mu_{\max} \leqslant \omega \leqslant 1 - \mu_{\min}), \\ \lambda_{\min} + \dfrac{1}{\lambda_{\min}}, & \lambda_{\min} > 1 \quad (\omega > 1 - \mu_{\min}). \end{cases}$$

Note that $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum eigenvalues of the matrix $T^{-1/2} H_\omega T^{-1/2}$, respectively, and $\mu_{\min}$ and $\mu_{\max}$ are the minimum and maximum eigenvalues of the matrix $T^{-1/2} W T^{-1/2}$, respectively. Moreover, $\lambda_{\min} = \mu_{\min} + \omega$ and $\lambda_{\max} = \mu_{\max} + \omega$.

It is noteworthy that the quasi-optimal parameter α^* relies on the eigenvalues of the matrix $T^{-1}W$ and the parameter ω, but according to [30], [20], we see that in Examples 4.1–4.2, the minimum and maximum eigenvalues of the matrix $T^{-1}W$ can be computed. The details are as follows:

1. For Example 4.1, we have

$$T = I \otimes B_m + B_m \otimes I, \quad W = 10(I \otimes B + B \otimes I) + 9(e_1 e_m^\top + e_m e_1^\top) \otimes I,$$

$B_m = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{n\times n}$, $B = B_m - e_1 e_m^\top - e_m e_1^\top \in \mathbb{R}^{n\times n}$, where e_1, e_m are the 1th and the mth unity basis vectors in $\mathbb{R}^m$, respectively. Let $\zeta_1 \geqslant \zeta_2 \ldots \geqslant \zeta_n$ be the eigenvalues of the matrix $(I \otimes B_m + B_m \otimes I)^{-1}[10I \otimes (e_1 e_m^\top - e_m e_1^\top) + (e_1 e_m^\top - e_m e_1^\top) \otimes I]$, $\zeta_{\max} = \zeta_1$, $\zeta_{\min} = \zeta_n$. Then

$$\begin{aligned}
T^{-1}W &= (I \otimes B_m + B_m \otimes I)^{-1}[10(I \otimes B + B \otimes I) + 9(e_1 e_m^\top + e_m e_1^\top) \otimes I] \\
&= (I \otimes B_m + B_m \otimes I)^{-1}[10(I \otimes (B_m - e_1 e_m^\top - e_m e_1^\top) \\
&\quad + (B_m - e_1 e_m^\top - e_m e_1^\top) \otimes I) + 9(e_1 e_m^\top + e_m e_1^\top) \otimes I] \\
&= (I \otimes B_m + B_m \otimes I)^{-1}[10(I \otimes B_m + B_m \otimes I) \\
&\quad - 10I \otimes (e_1 e_m^\top - e_m e_1^\top) - (e_1 e_m^\top - e_m e_1^\top) \otimes I] \\
&= 10I - (I \otimes B_m + B_m \otimes I)^{-1}[10I \otimes (e_1 e_m^\top - e_m e_1^\top) + (e_1 e_m^\top - e_m e_1^\top) \otimes I].
\end{aligned}$$

Then $\mu_{\max} = 10 - \zeta_{\min}$, $\mu_{\min} = 10 - \zeta_{\max}$, $\lambda_{\max} = 10 - \zeta_{\min} + \omega$ and $\lambda_{\min} = 10 - \zeta_{\max} + \omega$.

2. For Example 4.2, we have

$$W = V - \kappa I, \quad T = V + \varepsilon I.$$

Let $V = I \otimes V_m + V_m \otimes I \in \mathbb{R}^{n\times n}$, with $V_m = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$. We set $\kappa = 50$ and $\varepsilon = 400$. Let $\xi_1 \geqslant \xi_2 \ldots \geqslant \xi_n$ be the eigenvalues of the matrix V, $\xi_{\max} = \zeta_1$, $\xi_{\min} = \zeta_n$. Then

$$\begin{aligned}
T^{-1}W &= (V + 400I)^{-1}(V - 50I) = (V + 400I)^{-1}(V + 400I - 450I) \\
&= I - (V + 400I)^{-1}450I = I - \Big(\frac{V}{450} + \frac{8}{9}I\Big)^{-1}, \\
\mu_{\max} &= 1 - \Big(\frac{\xi_{\max}}{450} + \frac{8}{9}\Big)^{-1}, \quad \mu_{\min} = 1 - \Big(\frac{\xi_{\min}}{450} + \frac{8}{9}\Big)^{1}, \\
\lambda_{\max} &= 1 - \Big(\frac{\xi_{\max}}{450} + \frac{8}{9}\Big)^{-1} + \omega, \quad \text{and} \quad \lambda_{\min} = 1 - \Big(\frac{\xi_{\min}}{450} + \frac{8}{9}\Big)^{-1} + \omega.
\end{aligned}$$

Based on the conclusions of [30], [20], it can be seen that the maximum and minimum eigenvalues of the matrix V are

$$\xi_{\max} = 8\cos^2\frac{\pi}{2(m+1)} \quad \text{and} \quad \xi_{\min} = 8\sin^2\frac{\pi}{2(m+1)},$$

respectively. Thus, from the forms of the matrices W and T, we deduce that

$$\mu_{\max} = 1 - \Big(\frac{\xi_{\max}}{450} + \frac{8}{9}\Big)^{-1} = 1 - \Big(\frac{8\cos^2\frac{1}{2}\pi/(m+1)}{450} + \frac{8}{9}\Big)^{-1},$$
$$\lambda_{\max} = 1 - \Big(\frac{\xi_{\max}}{450} + \frac{8}{9}\Big)^{-1} + \omega = 1 - \Big(\frac{8\cos^2\frac{1}{2}\pi/(m+1)}{450} + \frac{8}{9}\Big)^{-1} + \omega,$$
$$\mu_{\min} = 1 - \Big(\frac{\xi_{\min}}{450} + \frac{8}{9}\Big)^{-1} = 1 - \Big(\frac{8\sin^2\frac{1}{2}\pi/(m+1)}{450} + \frac{8}{9}\Big)^{-1},$$
$$\lambda_{\min} = 1 - \Big(\frac{\xi_{\min}}{450} + \frac{8}{9}\Big)^{-1} + \omega = 1 - \Big(\frac{8\sin^2\frac{1}{2}\pi/(m+1)}{450} + \frac{8}{9}\Big)^{-1} + \omega.$$

4.2. Test result. In the following numerical experiments, we use α^* and ω^* to represent the theoretical quasi-optimal parameters, use α_{eps} and ω_{eps} to represent the experimental optimal parameters, and α_{left} and ω_{left} to represent the ones which are selected as the leftmost values of the parameter intervals.

m		128	256	512	1024
DSS [44]	α	[4.33,4.58]	[4.3,4.67]	[5.88,6.41]	[8.8,9]
CRI [32]	α	[0.91,1.09]	[0.78,1.29]	[0.96,1.04]	[0.77,1.3]
PMHSS [7] ($V = T$)	α	[1.25,1.82]	[1.05,1.49]	[0.96,1.21]	[0.83,1.17]
DSS real-valued [41]	α	[0.91,1.09]	[0.78,1.29]	[0.96,1.04]	[0.77,1.3]
PMHSS [7] ($V = W$)	α	[0.55,0.8]	[0.68,0.95]	[0.83,1.04]	[0.85,1.21]
		$\{0.23\}$	$\{0.23\}$	$\{0.22\}$	$\{0.22\}$
DDSS	α	$\cup$	$\cup$	$\cup$	$\cup$
		[4.32,4.51]	[4.32,4.51]	[4.47,4.54]	[4.44,4.58]
	ω	[0.01,0.02]	[0.01,0.04]	[0.04,0.06]	[0.07,0.08]

Table 1. The parameters ranges that minimize the iterations number of the tested methods for Example 4.1.

In Table 1, we show the parameters intervals with the least IT when different iteration methods are used to solve Example 4.1. The parameter α_{esp} with the least IT and CPU time, the corresponding IT, CPU time and RES are shown in Table 2. In addition, Table 2 also shows the IT, CPU time and RES of the DDSS method with the parameters α_{left} and ω_{left} to solve Example 4.1. As observed in Table 2, the proposed method performs better than the other test methods in terms of both the IT and CPU times when the matrix order is large. Furthermore, it can also be seen that the IT of the DDSS iteration method are unchanged with respect to m, and our method is stable for solving Example 4.1.

m		128	256	512	1024
DSS [44]	α_{eps}	4.5	4.64	6.41	8.98
	IT	18	18	24	33
	CPU	0.1606	0.8035	4.6843	28.3876
	RES	8.22E − 9	9.04E − 9	9.90E − 9	9.88E − 9
CRI [32]	α_{eps}	0.96	0.97	1.03	0.85
	IT	23	23	22	22
	CPU	0.1915	0.9597	4.3991	21.7686
	RES	9.60E − 9	6.93E-9	9.92E − 9	8.03E − 9
PMHSS [7]	α_{eps}	1.35	1.16	1.09	1.1
$(V = T)$	IT	41	42	43	45
	CPU	0.2587	1.3640	6.3111	32.1875
	RES	8.76E − 9	7.70E − 9	8.43E − 9	8.61E − 9
DSS real-valued [41]	α_{eps}	1.03	1.21	1.04	0.77
	IT	23	23	22	22
	CPU	0.2345	1.1643	5.4555	25.9714
	RES	9.56E − 9	8.43E − 9	9.96E − 9	9.97E − 9
	RES	8.22E − 9	9.04E − 9	9.90E − 9	9.88E − 9
PMHSS [7]	α_{eps}	0.8	0.73	0.89	0.88
$(V = W)$	IT	41	42	43	45
	CPU	0.2966	1.5516	7.3983	37.9046
	RES	9.97E − 9	7.48E − 9	8.61E − 9	9.19E − 9
DDSS	α_{eps}	4.35	4.33	0.22	0.22
	ω_{eps}	0.02	0.04	0.04	0.07
	IT	18	18	18	18
	CPU	0.1656	0.8309	3.9115	18.4456
	RES	9.26E − 9	9.64E − 9	9.49E − 9	9.07E − 9
	α_{left}	0.23	0.23	0.22	0.22
	ω_{left}	0.01	0.01	0.04	0.07
	IT	18	18	18	18
	CPU	0.2031	0.8985	3.9115	18.4456
	RES	9.26E − 9	9.64E − 9	9.49E − 9	9.07E − 9

Table 2. Numerical results of different iteration methods for Example 4.1.

We provide the maximum eigenvalue $\lambda_{\max}$ and minimum eigenvalue $\lambda_{\min}$ of the matrix $T^{-1}H_\omega$ for different ω in Table 3, and calculate the maximum value $f_{\max}(\lambda)$ and minimum value $f_{\min}(\lambda)$ of the function $f(\lambda)$ based on Lemma 3.2. Then we

use Theorem 3.3 to calculate the two quasi-optimal parameters α_1^* and α_2^* of the DDSS iteration method. Table 4 lists the numerical results of the DDSS iteration method with quasi-optimal parameters α_1^* and α_2^* for difficult parameter ω. From the results of Table 4, we can observe that the IT of the DDSS method with quasi-optimal parameters are more than those of the DDSS method with the experimental optimal parameters. The reason for this phenomenon is that quasi-optimal parameters are the ones making the upper bound of the spectral radius of the iteration matrix minimized, not spectral radius itself. When $m = 1024$, the amount of eigenvalue calculation exceeds the memory of computer, so there are no results in Tables 3 and 4.

ω \ m		128	256	512	1024
$\omega = 0.5$	$\lambda_{\max}$	20.3450	20.4222	20.4610	-
	$\lambda_{\min}$	0.5000	0.5000	0.5000	-
	$f_{\max}(\lambda)$	20.3941	20.4711	20.5099	-
	$f_{\min}(\lambda)$	2	2	2	-
$\omega = 0.7$	$\lambda_{\max}$	20.5450	20.6222	20.6610	-
	$\lambda_{\min}$	0.7000	0.7000	0.7000	-
	$f_{\max}(\lambda)$	20.5936	20.6708	20.7094	-
	$f_{\min}(\lambda)$	2	2	2	-
$\omega = 0.9$	$\lambda_{\max}$	20.7450	20.8222	20.8610	-
	$\lambda_{\min}$	0.9000	0.9000	0.9000	-
	$f_{\max}(\lambda)$	20.7932	20.8702	20.9089	-
	$f_{\min}(\lambda)$	2	2	2	-
$\omega = 1.11$	$\lambda_{\max}$	20.9550	21.0322	21.0710	-
	$\lambda_{\min}$	1.1100	1.1100	1.1100	-
	$f_{\max}(\lambda)$	21.0027	21.0797	21.1185	-
	$f_{\min}(\lambda)$	2.0109	2.0109	2.0109	-

Table 3. The corresponding eigenvalues and results of the function about eigenvalue for Example 4.1.

Table 5 shows the parameters intervals that minimize the IT when using different iteration methods for Example 4.2. On the basis of the parameters intervals given in Table 5 and similar to Table 2, we display the parameters, IT, CPU time, and RES in Table 6. In addition, Table 6 also shows the IT, CPU time and RES of the DDSS methods with the parameters α_{left} and ω_{left} to solve Example 4.2. The results shown in Table 6 indicate that the DDSS iteration method with the parameters α_{left},

ω_{eps} and α_{left}, ω_{left} outperforms the other test methods in terms of the IT and CPU times. The DSS iteration method is not converget when solving Example 4.2, so the corresponding results are not shown in Tables 5 and 6.

ω \ m		128	256	512	1024
$\omega = 0.5$	α_1^*	0.04	0.04	0.04	-
	IT	113	111	109	-
	CPU	0.8075	3.8511	17.1759	-
	RES	9.80E − 9	9.94E − 9	9.30E − 9	-
	α_2^*	23.56	23.65	23.69	-
	IT	107	106	103	-
	CPU	0.7679	3.6655	16.1576	-
	RES	9.30E − 9	8.75E − 9	9.89E − 9	-
$\omega = 0.7$	α_1^*	0.04	0.04	0.04	-
	IT	115	113	110	-
	CPU	0.7793	3.8698	17.3027	-
	RES	8.73E − 9	8.90E − 9	9.75E − 9	-
	α_2^*	23.78	23.86	23.90	-
	IT	108	108	106	-
	CPU	0.7622	3.6581	16.5863	-
	RES	9.43E − 9	8.66E − 9	8.76E − 9	-
$\omega = 0.9$	α_1^*	0.04	0.04	0.04	-
	IT	113	111	109	-
	CPU	0.7802	3.8639	16.9809	-
	RES	9.86E − 9	9.82E − 9	9.85E − 9	-
	α_2^*	23.99	24.08	24.12	-
	IT	109	107	106	-
	CPU	0.7721	3.6779	16.5600	-
	RES	9.19E − 9	9.82E − 9	8.77E − 9	-
$\omega = 1.11$	α_1^*	0.04	0.04	0.04	-
	IT	109	108	106	-
	CPU	0.7463	3.6974	16.4676	-
	RES	9.93E − 9	8.99E − 9	9.73E − 9	-
	α_2^*	24.34	24.43	24.47	-
	IT	107	105	104	-
	CPU	0.7275	3.6065	16.2140	-
	RES	8.75E − 9	9.85E − 9	9.42E − 9	-

Table 4. Numerical results of the DDSS iteration method with quasi-optimal parameter α^* for Example 4.1.

m		128	256	512	1024
CRI [32]	α	[0.8,1.26]	[0.8,1.26]	[0.8,1.26]	[0.8,1.26]
PMHSS [7] ($V = T$)	α	[1.45,1.61]	[1.46,1.61]	[1.46,1.61]	[1.46,1.61]
DSS real-valued [41]	α	[0.8,1.26]	[0.8,1.26]	[0.8,1.26]	[0.8,1.26]
PMHSS [7]($V = W$)	α	[0.62,0.69]	[0.63,0.68]	[0.63,0.68]	[0.63,0.68]
		[0.44,0.46]	[0.44,0.46]	[0.44,0.46]	[0.44,0.46]
DDSS	α	$\cup$	$\cup$	$\cup$	$\cup$
		[2.16,2.31]	[2.16,2.31]	[2.16,2.31]	[2.16,2.31]
	ω	[0.63,0.71]	[0.63,0.71]	[0.63,0.71]	[0.63,0.71]

Table 5. The parameters ranges that minimize the iterations number of the tested methods for Example 4.2.

m		128	256	512	1024
CRI [32]	α_{eps}	0.86	0.81	0.86	0.97
	IT	18	18	18	18
	CPU	0.1256	0.7017	3.3880	15.3411
	RES	6.97E − 9	8.93E − 9	7.03E − 9	5.50E − 9
PMHSS [7]	α_{eps}	1.48	1.61	1.5	1.58
($V = T$)	IT	79	79	79	79
	CPU	0.3988	2.2694	10.9420	48.2985
	RES	9.78E − 9	9.98E − 9	9.74E − 9	9.82E − 9
DSS real-valued [41]	α_{eps}	0.9	1.19	0.81	0.82
	IT	18	18	18	18
	CPU	0.1534	0.8388	4.1566	17.8286
	RES	6.11E − 9	7.62E − 9	8.96E − 9	8.48E − 9
PMHSS [7]	α_{eps}	0.63	0.65	0.66	0.68
($V = W$)	IT	79	79	79	79
	CPU	0.4112	2.6085	15.2430	82.7916
	RES	9.82E − 9	9.69E − 9	9.70E − 9	9.87E − 9
DDSS	α_{eps}	2.24	0.46	2.16	0.44
	ω_{eps}	0.69	0.69	0.68	0.64
	IT	7	7	7	7
	CPU	0.0828	0.4375	2.1264	9.6575
	RES	2.03E − 9	1.75E − 9	1.86E − 9	1.22E − 9
	α_{left}	0.44	0.44	0.44	0.44
	ω_{left}	0.63	0.63	0.63	0.63
	IT	7	7	7	7
	CPU	0.0892	0.4543	2.1535	9.7008
	RES	2.03E − 9	1.75E − 9	1.86E − 9	1.22E − 9

Table 6. Numerical results of different iteration methods for Example 4.2.

Similarly to Table 3, we provide the maximum eigenvalue $\lambda_{\max}$ and minimum eigenvalue $\lambda_{\min}$ of the matrix $T^{-1}H_\omega$, the maximum value $f_{\max}(\lambda)$ and minimum value $f_{\min}(\lambda)$ of the function $f(\lambda)$ for different values of ω for Example 4.2 in Table 7. Based on $f_{\max}(\lambda)$ and $f_{\min}(\lambda)$ of $f(\lambda)$ given in Table 7, the numerical results of the DDSS iteration method with quasi-optimal parameters α_1^* and α_2^* for different values of the parameter ω are presented in Table 8. It can be seen that the IT of the DDSS method with quasi-optimal parameters are slightly more than those of the DDSS method with the experimental optimal parameters. This phenomenon indicates that the quasi-optimal parameters in Theorem 3.3 can be used in practical computation and make the DDSS iteration method effective for Example 4.2.

In order to more intuitively show the influence of parameters on different iteration methods, we describe the change of the IT of different iteration ones with the parameter α in Figure 1. Figure 1 intuitively shows that the minimum IT of the DDSS method is less than those of other existing methods. We describe the change of the IT of the DDSS method with the parameter ω in Figure 2. We observe that the IT of the DDSS iteration method decreases first and then increases with the increasing of the single parameter ω, and the parameter value of the minimum IT is basically consistent with the information shown in Tables 1 and 3.

ω \ m		128	256	512	1024
$\omega = 0.5$	$\lambda_{\max}$	0.3971	0.3971	0.3971	0.3971
	$\lambda_{\min}$	0.3750	0.3750	0.3750	0.3750
	$f_{\max}(\lambda)$	3.0416	3.0416	3.0416	3.0416
	$f_{\min}(\lambda)$	2.9156	2.9156	2.9156	2.9156
$\omega = 0.7$	$\lambda_{\max}$	0.5971	0.5971	0.5971	0.5971
	$\lambda_{\min}$	0.5750	0.5750	0.5750	0.5750
	$f_{\max}(\lambda)$	2.3141	2.3141	2.3141	2.3141
	$f_{\min}(\lambda)$	2.2719	2.2719	2.2719	2.2719
$\omega = 0.9$	$\lambda_{\max}$	0.7971	0.7971	0.7971	0.7971
	$\lambda_{\min}$	0.7750	0.7750	0.7750	0.7750
	$f_{\max}(\lambda)$	2.0653	2.0653	2.0653	2.0653
	$f_{\min}(\lambda)$	2.0517	2.0517	2.0517	2.0517
$\omega = 1.11$	$\lambda_{\max}$	1.0071	1.0071	1.0071	1.0071
	$\lambda_{\min}$	0.9850	0.9850	0.9850	0.9850
	$f_{\max}(\lambda)$	2.0324	2.0324	2.0324	2.0324
	$f_{\min}(\lambda)$	2	2	2	2
$\omega = 3$	$\lambda_{\max}$	2.8971	2.8971	2.8971	2.8971
	$\lambda_{\min}$	2.8750	2.8750	2.8750	2.8750
	$f_{\max}(\lambda)$	3.2422	3.2422	3.2422	3.2422
	$f_{\min}(\lambda)$	3.2228	3.2228	3.2228	3.2228

Table 7. The corresponding eigenvalues and results of the function about eigenvalue for Example 4.2.

ω \ m		128	256	512	1024
$\omega = 0.5$	α_1^*	0.17	0.17	0.17	0.17
	IT	17	17	17	17
	CPU	0.2733	0.7151	3.5416	15.1949
	RES	3.27E − 9	3.25E − 9	3.25E − 9	3.24E − 9
	α_2^*	5.81	5.81	5.81	5.81
	IT	16	16	16	16
	CPU	0.1353	0.6885	3.3435	14.6647
	RES	8.36E − 9	8.33E − 9	8.31E − 9	8.31E − 9
$\omega = 0.7$	α_1^*	0.28	0.28	0.28	0.28
	IT	13	13	13	13
	CPU	0.1227	0.6078	2.9843	12.9766
	RES	4.63E − 9	4.62E − 9	4.62E − 9	4.61E − 9
	α_2^*	3.58	3.58	3.58	3.58
	IT	13	13	13	13
	CPU	0.1204	0.5974	2.9267	13.3153
	RES	4.92E − 9	4.91E − 9	4.90E − 9	4.90E − 9
$\omega = 0.9$	α_1^*	0.35	0.35	0.35	0.35
	IT	12	12	12	12
	CPU	0.1134	0.5921	2.9806	12.3257
	RES	2.84E − 9	2.83E − 9	2.83E − 9	2.83E − 9
	α_2^*	2.90	2.90	2.90	2.90
	IT	12	12	12	12
	CPU	0.1109	0.5932	2.7792	12.2942
	RES	4.10E − 9	4.09E − 9	4.09E − 9	4.08E − 9
$\omega = 1.11$	α_1^*	0.37	0.37	0.37	0.37
	IT	12	12	12	12
	CPU	0.1105	0.5992	2.7999	12.2977
	RES	2.45E − 9	2.45E − 9	2.45E − 9	2.45E − 9
	α_2^*	2.73	2.73	2.73	2.73
	IT	12	12	12	12
	CPU	0.1140	0.5720	2.7922	12.2411
	RES	3.12E − 9	3.11E − 9	3.11E − 9	3.11E − 9
$\omega = 3$	α_1^*	0.23	0.23	0.23	0.23
	IT	12	12	12	12
	CPU	0.1185	0.6003	2.7203	12.2361
	RES	3.43E − 9	3.44E − 9	3.44E − 9	3.44E − 9
	α_2^*	4.33	4.33	4.33	4.33
	IT	12	12	12	12
	CPU	0.1238	0.5994	2.613	12.1480
	RES	3.13E − 9	3.13E − 9	3.14E − 9	3.14E − 9

Table 8. Numerical results of the DDSS iteration method with quasi-optimal parameter α^* for Example 4.2.

m		128	256	512	1024
DSS-GMRES	α	[0.65,1.54]	[0.58,1.75]	[0.77,1.3]	[0.51,1.99]
CRI-GMRES	α	[0.65,1.54]	[0.58,1.75]	[0.78,1.29]	[0.51,1.98]
PMHSS-GMRES ($V = T$)	α	[0.01,0.8]	[0.08,1.14]	[0.25,1.17]	[0.69,1]
PMHSS-GMRES ($V = W$)	α	[1.25,1000]	[0.88,14.03]	[0.85,4.02]	[1,1.46]
		[0.94,1.06]	[0.77,1.3]	[0.92,1.09]	[0.75,1.34]
DDSS-GMRES	α	$\cup$		$\cup$	
		[0.01,0.1]		[0.01,0.09]	
	ω	0.12	[0.03,0.04]	[2.11,0.14]	0.04

Table 9. The experimental optimal parameters for preconditioned GMRES methods by minimizing iteration steps for Example 4.1.

m		128	256	512	1024
DSS-GMRES	α_{eps}	0.86	1.09	0.8	0.52
	IT	9	9	9	9
	CPU	0.1176	0.5838	2.7571	13.4856
	RES	6.17E − 9	4.62E − 9	9.44E − 9	8.92E − 9
CRI-GMRES	α_{eps}	1.18	1.2	0.78	0.51
	IT	9	9	9	9
	CPU	0.1176	0.5838	2.7571	13.6676
	RES	6.25E − 9	4.93E − 9	9.78E − 9	9.52E − 9
PMHSS-GMRES	α_{eps}	0.23	0.17	0.27	0.69
($V = T$)	IT	19	24	30	35
	CPU	0.1227	0.8219	4.5287	24.2613
	RES	5.17E − 9	3.57E − 9	5.44E − 9	9.80E − 9
PMHSS-GMRES	α_{eps}	228	1.19	0.87	1.06
($V = W$)	IT	19	24	30	35
	CPU	0.1207	0.8240	4.5057	24.1491
	RES	7.44E − 9	4.22E − 9	9.19E − 9	9.12E − 9
DDSS	α_{eps}	0.99	0.94	0.97	0.84
	ω_{eps}	0.12	0.04	0.07	0.04
	IT	13	12	13	12
	CPU	0.1498	0.7121	3.4763	15.9780
	RES	9.92E − 9	8.19E − 9	7.56E − 9	9.47E − 9

Table 10. Numerical results of preconditioned GMRES methods for Example 4.1.

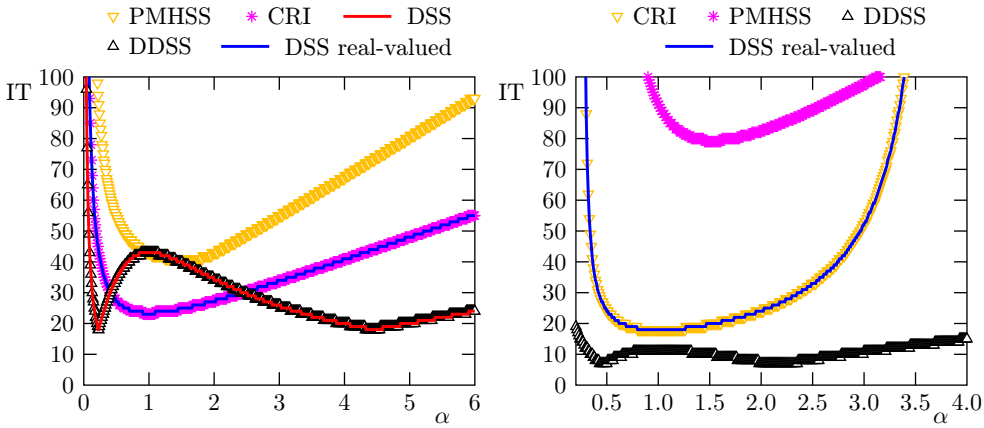

Figure 1. Effect of parameter α on iterations number (IT) of the CRI, PMHSS, DSS, DSS real-valued and DDSS methods (left: DDSS method with $\omega = 0.01$, PMHSS method with $V = T$ for Example 4.1 for $m = 128$; right: DDSS method with $\omega = 0.7$, PMHSS method with $V = T$ for Example 4.2 for $m = 128$).

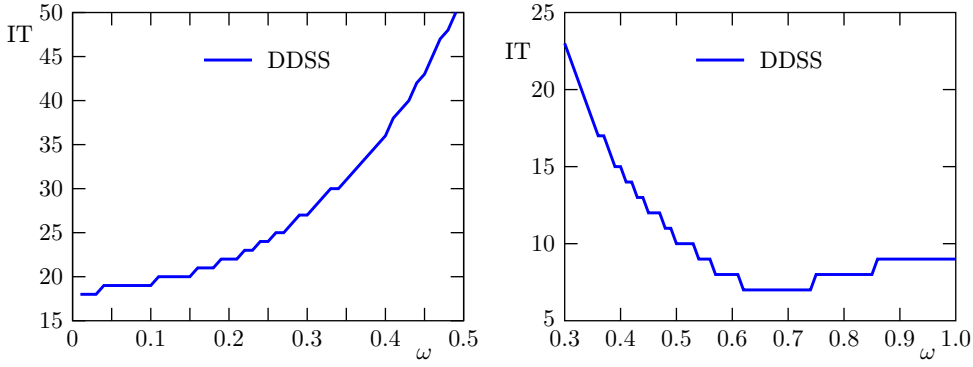

Figure 2. Effect of parameter ω on iterations number (IT) of the DDSS method for Example 4.1 ($\alpha = 4.4$ with $m = 128$) on the left and Example 4.2 ($\alpha = 2.2$ with $m = 128$) on the right.

Inspired by the preconditioning ideas in [3] and [4], the DSS, PMHSS, CRI and DDSS iteration methods can be used as preconditioners to process linear system (1.1), and the sublinear systems involved in the tested methods are solved by using the GMRES iteration method. The detailed results are shown in Tables 9 and 10.

Table 9 shows the ranges of the optimal parameters of the DSS, CRI, PMHSS and DDSS preconditioned GMRES iteration methods for Example 4.1. In Table 10 we report results for the DSS, CRI, PMHSS and DDSS preconditioned GMRES methods. From the results shown in Table 10, it can be seen that when used as a preconditioner, the DDSS iteration method performs better than the PMHSS iteration method with respect to the IT and CPU time. But it is not as effective as the

DSS method. For Example 4.2, the DDSS preconditioned GMRES method is not convergent, so it is not listed here. In addition, the results given in Tables 1–10 show that the DDSS iteration method performs better when used is to solve the complex symmetric linear equations, but when used as a preconditioner it takes slightly more computational time than the other tested methods.

5. Conclusions

In this paper, inspired by the idea of the DSS method [44], we establish a dual-parameter double-step splitting (DDSS) iteration method for solving the complex symmetric linear system (1.1). We study the convergence properties of the DDSS iteration method. Finally, the numerical results illustrate the effectiveness and robustness of our method. Compared with the DSS, CRI, PMHSS and DSS real-valued methods, it can be found that the DDSS method requires less IT and CPU times. In addition, this paper also analyzes the influence of parameters α and ω on the iteration times of the DDSS method. In general, the DDSS method is competitive compared with the DSS, CRI, PMHSS and DSS real-valued methods.

References

[1] *S. R. Arridge*: Optical tomography in medical imaging. Inverse Probl. *15* (1999), R41–R93. zbl MR doi

[2] *O. Axelsson, A. Kucherov*: Real valued iterative methods for solving complex symmetric linear systems. Numer. Linear Algebra Appl. *7* (2000), 197–218. zbl MR doi

[3] *Z.-Z. Bai*: Sharp error bounds of some Krylov subspace methods for non-Hermitian linear systems. Appl. Math. Comput. *109* (2000), 273–285. zbl MR doi

[4] *Z.-Z. Bai*: Motivations and realizations of Krylov subspace methods for large sparse linear systems. J. Comput. Appl. Math. *283* (2015), 71–78. zbl MR doi

[5] *Z.-Z. Bai*: Quasi-HSS iteration methods for non-Hermitian positive definite linear systems of strong skew-Hermitian parts. Numer. Linear Algebra Appl. *25* (2018), Article ID e2116, 19 pages. zbl MR doi

[6] *Z.-Z. Bai, M. Benzi, F. Chen*: Modified HSS iteration methods for a class of complex symmetric linear systems. Computing *87* (2010), 93–111. zbl MR doi

[7] *Z.-Z. Bai, M. Benzi, F. Chen*: On preconditioned MHSS iteration methods for complex symmetric linear systems. Numer. Algorithms *56* (2011), 297–317. zbl MR doi

[8] *Z.-Z. Bai, G. H. Golub*: Accelerated Hermitian and skew-Hermitian splitting iteration methods for saddle-point problems. IMA J. Numer. Anal. *27* (2007), 1–23. zbl MR doi

[9] *Z.-Z. Bai, G. H. Golub, M. K. Ng*: Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. SIAM J. Matrix Anal. Appl. *24* (2003), 603–626. zbl MR doi

[10] *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), 1–32. zbl MR doi

[11] *Z.-Z. Bai, J.-Y. Pan*: Matrix Analysis and Computations. Other Titles in Applied Mathematics 173. SIAM, Philadelphia, 2021. zbl MR doi

[12] *M. Benzi, D. Bertaccini*: Block preconditioning of real-valued iterative algorithms for complex linear systems. IMA J. Numer. Anal. *28* (2008), 598–618. zbl MR doi
[13] *D. Bertaccini*: Efficient preconditioning for sequences of parametric complex symmetric linear systems. ETNA, Electron. Trans. Numer. Anal. *18* (2004), 49–64. zbl MR
[14] *F. Chen, T.-Y. Li, K.-Y. Lu, G. V. Muratova*: Modified QHSS iteration methods for a class of complex symmetric linear systems. Appl. Numer. Math. *164* (2021), 3–14. zbl MR doi
[15] *M. Dehghan, M. Dehghani-Madiseh, M. Hajarian*: A generalized preconditioned MHSS method for a class of complex symmetric linear systems. Math. Model. Anal. *18* (2013), 561–576. zbl MR doi
[16] *A. Feriani, F. Perotti, V. Simoncini*: Iterative system solvers for the frequency analysis of linear mechanical systems. Comput. Methods Appl. Mech. Eng. *190* (2000), 1719–1739. zbl doi
[17] *A. Frommer, T. Lippert, B. Medeke, K. Schilling* (Eds.): Numerical Challenges in Lattice Quantum Chromodynamics. Lecture Notes in Computational Science and Engineering 15. Springer, Berlin, 2000. zbl MR doi
[18] *Y. Huang, G. Chen*: A relaxed block splitting preconditioner for complex symmetric indefinite linear systems. Open Math. *16* (2018), 561–573. zbl MR doi
[19] *Z.-G. Huang*: A new double-step splitting iteration method for certain block two-by-two linear systems. Comput. Appl. Math. *39* (2020), Article ID 193, 42 pages. zbl MR doi
[20] *Z.-G. Huang*: Efficient block splitting iteration methods for solving a class of complex symmetric linear systems. J. Comput. Appl. Math. *395* (2021), Article ID 113574, 21 pages. zbl MR doi
[21] *Z.-G. Huang*: Modified two-step scale-splitting iteration method for solving complex symmetric linear systems. Comput. Appl. Math. *40* (2021), Article ID 122, 35 pages. zbl MR doi
[22] *Z.-G. Huang, L.-G. Wang, Z. Xu, J.-J. Cui*: Preconditioned accelerated generalized successive overrelaxation method for solving complex symmetric linear systems. Comput. Math. Appl. *77* (2019), 1902–1916. zbl MR doi
[23] *B. Li, J. Cui, Z. Huang, X. Xie*: On preconditioned MQHSS iterative method for solving a class of complex symmetric linear systems. Comput. Appl. Math. *41* (2022), Article ID 250, 23 pages. zbl MR doi
[24] *C.-X. Li, S.-L. Wu*: A single-step HSS method for non-Hermitian positive definite linear systems. Appl. Math. Lett. *44* (2015), 26–29. zbl MR doi
[25] *L. Li, T.-Z. Huang, X.-P. Liu*: Modified Hermitian and skew-Hermitian splitting methods for non-Hermitian positive-definite linear systems. Numer. Linear Algebra Appl. *14* (2007), 217–235. zbl MR doi
[26] *X. Li, A.-L. Yang, Y.-J. Wu*: Lopsided PMHSS iteration method for a class of complex symmetric linear systems. Numer. Algorithms *66* (2014), 555–568. zbl MR doi
[27] *H. Noormohammadi Pour, H. Sadeghi Goughery*: New Hermitian and skew-Hermitian splitting methods for non-Hermitian positive-definite linear systems. Numer. Algorithms *69* (2015), 207–225. zbl MR doi
[28] *B. Poirier*: Efficient preconditioning scheme for block partitioned matrices with structured sparsity. Numer. Linear Algebra Appl. *7* (2000), 715–726. zbl MR doi
[29] *A. Shirilord, M. Dehghan*: Single step iterative method for linear system of equations with complex symmetric positive semi-definite coefficient matrices. Appl. Math. Comput. *426* (2022), Article ID 127111, 17 pages. zbl MR doi
[30] *T. S. Siahkoalaei, D. K. Salkuyeh*: A new double-step method for solving complex Helmholtz equation. Hacet. J. Math. Stat. *49* (2020), 1245–1260. zbl MR doi
[31] *W. van Dijk, F. M. Toyama*: Accurate numerical solutions of the time-dependent Schrödinger equation. Phys. Rev. E *75* (2007), Article ID 036707, 10 pages. MR doi
[32] *T. Wang, Q. Zheng, L. Lu*: A new iteration method for a class of complex symmetric linear systems. J. Comput. Appl. Math. *325* (2017), 188–197. zbl MR doi

[33] *S.-L. Wu*: Several variants of the Hermitian and skew-Hermitian splitting method for a class of complex symmetric linear systems. Numer. Linear Algebra Appl. *22* (2015), 338–356. zbl MR doi
[34] *X.-Y. Xiao, X. Wang, H.-W. Yin*: Efficient single-step preconditioned HSS iteration methods for complex symmetric linear systems. Comput. Math. Appl. *74* (2017), 2269–2280. zbl MR doi
[35] *X.-Y. Xiao, X. Wang, H.-W. Yin*: Efficient preconditioned NHSS iteration methods for solving complex symmetric linear systems. Comput. Math. Appl. *75* (2018), 235–247. zbl MR doi
[36] *A.-L. Yang*: On the convergence of the minimum residual HSS iteration method. Appl. Math. Lett. *94* (2019), 210–216. zbl MR doi
[37] *A.-L. Yang, Y. Cao, Y.-J. Wu*: Minimum residual Hermitian and skew-Hermitian splitting iteration method for non-Hermitian positive definite linear systems. BIT *59* (2019), 299–319. zbl MR doi
[38] *M.-L. Zeng*: Inexact modified QHSS iteration methods for complex symmetric linear systems of strong skew-Hermitian parts. IAENG, Int. J. Appl. Math. *51* (2021), 109–115.
[39] *J. Zhang, H. Dai*: A new splitting preconditioner for the iterative solution of complex symmetric indefinite linear systems. Appl. Math. Lett. *49* (2015), 100–106. zbl MR doi
[40] *J.-H. Zhang, H. Dai*: A new block preconditioner for complex symmetric indefinite linear systems. Numer. Algorithms *74* (2017), 889–903. zbl MR doi
[41] *J. Zhang, Z. Wang, J. Zhao*: Double-step scale splitting real-valued iteration method for a class of complex symmetric linear systems. Appl. Math. Comput. *353* (2019), 338–346. zbl MR doi
[42] *J.-L. Zhang, H.-T. Fan, C.-Q. Gu*: An improved block splitting preconditioner for complex symmetric indefinite linear systems. Numer. Algorithms *77* (2018), 451–478. zbl MR doi
[43] *W.-H. Zhang, A.-L. Yang, Y.-J. Wu*: Minimum residual modified HSS iteration method for a class of complex symmetric linear systems. Numer. Algorithms *86* (2021), 1543–1559. zbl MR doi
[44] *Z. Zheng, F.-L. Huang, Y.-C. Peng*: Double-step scale splitting iteration method for a class of complex symmetric linear systems. Appl. Math. Lett. *73* (2017), 91–97. zbl MR doi

Authors' address: *Beibei Li*, *Jingjing Cui* (corresponding author), *Zhengge Huang*, *Xiaofeng Xie*, College of Mathematics and Physics, Center for Applied Mathematics of Guangxi, Guangxi Minzu University, No. 188, East Daxue Road, Xixiangtang District, 530006, Nanning, P. R. China, e-mail: `beibeili1997@163.com`, `jingjingcui1990@163.com`, `ZhenggeHuang@163.com`, `xiexiaofeng2022@163.com`.