AIMS Mathematics, 6(4): 4105–4118.
DOI: 10.3934/math.2021243
Received: 01 December 2020
Accepted: 01 February 2021
Published: 05 February 2021

http://www.aimspress.com/journal/Math

Research article

Shift-splitting iteration methods for a class of large sparse linear matrix equations

Xu Li* **and Rui-Feng Li**

Department of Applied Mathematics, School of Science, Lanzhou University of Technology, Lanzhou 730050, P. R. China

* **Correspondence:** Email: lixu@lut.edu.cn; Tel: +869312976040.

Abstract: By utilizing an inner-outer iteration strategy, a shift-splitting (SS) iteration method to solve a class of large sparse linear matrix equation $AXB = C$ is proposed in this work. Two convergence theorems for differential forms are studied in depth. Moreover, the quasi-optimal parameters which minimize the upper bound for the spectral radius of SS iteration matrix are given. Two numerical examples illustrate the high-efficiency of SS iteration method, especially when coefficient matrices are ill-conditioned.

Keywords: linear matrix equation; shift-splitting iteration; inexact iteration; convergence analysis
Mathematics Subject Classification: 15A24, 15A30, 15A69, 65F10, 65F30, 65F50, 65H10

1. Introduction

Consider large sparse linear matrix equation

$$AXB = C, \tag{1.1}$$

where $A \in \mathbb{C}^{m\times m}$ and $B \in \mathbb{C}^{n\times n}$ are non-Hermitian positive definite matrices, $C \in \mathbb{C}^{m\times n}$ is a given complex matrix. In many areas of scientific computation and engineering applications, such as signal and image processing [9, 20], control theory [11], photogrammetry [19], we need to solve such matrix equations. Therefore, solving such matrix equations by efficient methods is a very important topic.

We often rewrite the above matrix Eq (1.1) as the following linear system

$$(B^T \otimes A)\mathbf{x} = \mathbf{c}, \tag{1.2}$$

where the vectors $\mathbf{x}$ and $\mathbf{c}$ contain the concatenated columns of the matrices X and C, respectively, $\otimes$ being the Kronecker product symbol and B^T representing the transpose of the matrix B. Although this

equivalent linear system can be applied in theoretical analysis, in fact, solving (1.2) is always costly and ill-conditioned.

So far there are many numerical methods to solve the matrix Eq (1.1). When the coefficient matrices are not large, we can use some direct algorithms, such as the QR-factorization-based algorithms [13, 28]. Iterative methods are usually employed for large sparse matrix Eq (1.1), for instance, least-squares-based iteration methods [26] and gradient-based iteration methods [25]. Moreover, the nested splitting conjugate gradient (NSCG) iterative method, which was first proposed by Axelsson, Bai and Qiu in [1] to solve linear systems, was considered for the matrix Eq (1.1) in [14].

Bai, Golub and Ng originally established the efficient Hermitian and skew-Hermitian splitting (HSS) iterative method [5] for linear systems with non-Hermitian positive definite coefficient matrices. Subsequently, some HSS-based methods were further considered to improve its robustness for linear systems; see [3, 4, 8, 17, 24, 27] and other literature. For solving the continuous Sylvester equation, Bai recently established the HSS iteration method [2]. Hereafter, some HSS-based methods were discussed for solving this Sylvester equation [12, 15, 16, 18, 22, 30–33]. For the matrix Eq (1.1), Wang, Li and Dai recently use an inner-outer iteration strategy and then proposed an HSS iteration method [23]. According to the discussion in [23], if the quasi-optimal parameter is employed, the upper bound of the convergence rate is equal to that of the CG method. After that, Zhang, Yang and Wu considered a more efficient parameterized preconditioned HSS (PPHSS) iteration method [29] to further improve the efficiency for solving the matrix Eq (1.1), and Zhou, Wang and Zhou presented a modified HSS (MHSS) iteration method [34] for solving a class of complex matrix Eq (1.1).

Moreover, the shift-splitting (SS) iteration method [7] was first presented by Bai, Yin and Su to solve the ill-conditioned linear systems. Then this splitting method was subsequently considered for solving saddle point problems due to its promising performance; see [10, 21] and other literature. In this paper, the SS technique is implemented to solve the matrix Eq (1.1). Some related convergence theorems of the SS method are discussed in detail. Numerical examples demonstrate that the SS is superior to the HSS and NSCG methods, especially when the coefficient matrices are ill-conditioned.

The content of this paper is arranged as follows. In Section 2 we establish the SS method for solving the matrix Eq (1.1), and then some related convergence properties are studied in Section 3. In Section 4, the effectiveness of our method is illustrated by two numerical examples. Finally, our brief conclusions are given in Section 5.

2. The shift-splitting (SS) iteration method

Based on the shift-splitting proposed by Bai, Yin and Su in [7], we have the shift-splitting of A and B as follows:

$$A = \frac{1}{2}(\alpha I_m + A) - \frac{1}{2}(\alpha I_m - A), \tag{2.1}$$

and

$$B = \frac{1}{2}(\beta I_n + B) - \frac{1}{2}(\beta I_n - B), \tag{2.2}$$

where α and β are given positive constants.

Therefore, using the splitting of the matrix A in (2.1), the following splitting iteration method to

solve (1.1) can be defined:

$$(\alpha I_m + A)X^{(k+1)}B = (\alpha I_m - A)X^{(k)}B + 2C. \tag{2.3}$$

Then, from the splitting of the matrix B in (2.2), we can solve each step of (2.3) iteratively by

$$(\alpha I_m + A)X^{(k+1,j+1)}(\beta I_n + B) = (\alpha I_m + A)X^{(k+1,j)}(\beta I_n - B) + 2(\alpha I_m - A)X^{(k)}B + 4C. \tag{2.4}$$

Therefore, we can establish the following shift-splitting (SS) iteration method to solve (1.1).

Algorithm 1 (The SS iteration method)**.** *Given an initial guess $X^{(0)} \in \mathbb{C}^{m\times n}$, for $k = 0, 1, 2, \ldots$, until $X^{(k)}$ converges.*
Approximate the solution of

$$(\alpha I_m + A)Z^{(k)}B = 2R^{(k)} \tag{2.5}$$

with $R^{(k)} = C - AX^{(k)}B$, i.e., let $Z^{(k)} := Z^{(k,j+1)}$ and compute $Z^{(k,j+1)}$ iteratively by

$$(\alpha I_m + A)Z^{(k,j+1)}(\beta I_n + B) = (\alpha I_m + A)Z^{(k,j)}(\beta I_n - B) + 4R^{(k)}, \tag{2.6}$$

once the residual $P^{(k)} = 2R^{(k)} - (\alpha I_m + A)Z^{(k,j+1)}B$ of the outer iteration (2.5) *satisfies*

$$\|P^{(k)}\|_F \le \varepsilon_k \|R^{(k)}\|_F,$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Then compute

$$X^{(k+1)} = X^{(k)} + Z^{(k)}.$$

Here, $\{\varepsilon_k\}$ is a given tolerance. In addition, we can choose efficient methods in the process of computing $Z^{(k,j+1)}$ in (2.6).

The pseudo-code of this algorithm is shown as following:

The pseudo-code of the SS algorithm for matrix equation $AXB = C$
1. Given an initial guess $X^{(0)} \in \mathbb{C}^{m\times n}$
2. $R^{(0)} = C - AX^{(0)}B$
3. For $k = 0, 1, 2, \ldots, k_{\max}$ Do:
4. Given an initial guess $Z^{(k,0)} \in \mathbb{C}^{m\times n}$
5. $P^{(k,0)} = 2R^{(k)} - (\alpha I_m + A)Z^{(k,0)}B$
6. For $j = 0, 1, 2, \ldots, j_{\max}$ Do:
7. Compute $Z^{(k,j+1)}$ iteratively by
$(\alpha I_m + A)Z^{(k,j+1)}(\beta I_n + B) = (\alpha I_m + A)Z^{(k,j)}(\beta I_n - B) + 4R^{(k)}$
8. $P^{(k,j+1)} = 2R^{(k)} - (\alpha I_m + A)Z^{(k,j+1)}B$
9. If $\|P^{(k,j+1)}\|_F \le \varepsilon_k \|R^{(k)}\|_F$ Go To 11
10. End Do
11. $X^{(k+1)} = X^{(k)} + Z^{(k)}$
12. $R^{(k+1)} = C - AX^{(k+1)}B$
13. If $\|R^{(k+1)}\|_F \le \text{tol}\|R^{(0)}\|_F$ Stop
14. End Do

Remark 1. *Because the SS iteration scheme is only a single-step method, a considerable advantage is that it costs less computing workloads than the two-step iteration methods such as the HSS iteration [23] and the modified HSS (MHSS) iteration [34].*

3. Convergence analysis of the SS method

In this section, we denote by

$$H = \frac{1}{2}(A + A^*) \quad \text{and} \quad S = \frac{1}{2}(A - A^*)$$

the Hermitian and skew-Hermitian parts of the matrix A, respectively. Moreover, $\lambda_{\min}$ and $\lambda_{\max}$ represent the smallest and the largest eigenvalues of H, respectively, and $\kappa = \lambda_{\max}/\lambda_{\min}$.

Firstly, the unconditional convergence property of the SS iteration (2.3) is given as follows.

Theorem 1. *Let $A \in \mathbb{C}^{m\times m}$ be positive definite, and α be a positive constant. Denote by*

$$M(\alpha) = I_n \otimes \Big((\alpha I_m + A)^{-1}(\alpha I_m - A)\Big). \tag{3.1}$$

Then the convergence factor of the SS iteration method (2.3) *is given by the spectral radius $\rho(M(\alpha))$ of the matrix $M(\alpha)$, which is bounded by*

$$\varphi(\alpha) := \|(\alpha I_m + A)^{-1}(\alpha I_m - A)\|_2. \tag{3.2}$$

Consequently, we have

$$\rho(M(\alpha)) \le \varphi(\alpha) < 1, \qquad \forall \alpha > 0, \tag{3.3}$$

i.e., the SS iteration (2.3) *is unconditionally convergent to the exact solution $X^\star \in \mathbb{C}^{m\times n}$ of the matrix Eq* (1.1).

Proof. The SS iteration (2.3) can be reformulated as

$$X^{(k+1)} = (\alpha I_m + A)^{-1}(\alpha I_m - A)X^{(k)} + 2(\alpha I_m + A)^{-1}CB^{-1}. \tag{3.4}$$

Using the Kronecker product, we can rewrite (3.4) as follows:

$$\mathbf{x}^{(k+1)} = M(\alpha)\mathbf{x}^{(k)} + N(\alpha)\mathbf{c},$$

where $M(\alpha)$ is the iteration matrix defined in (3.1), and $N(\alpha) = 2B^{-T} \otimes (\alpha I_m + A)^{-1}$.

We can easily see that $\rho(M(\alpha)) \le \varphi(\alpha)$ holds for all $\alpha > 0$. From Lemma 2.1 in [8], we can obtain that $\varphi(\alpha) < 1$, $\forall \alpha > 0$. This completes the proof. □

Noting that the matrix $(\alpha I_m + A)^{-1}(\alpha I_m - A)$ is an extrapolated Cayley transform of A, from [6], we can obtain another upper bound for the convergence factor of $\rho(M(\alpha))$, as well as the minimum point and the corresponding minimal value of this upper bound.

Theorem 2. *Let the conditions of Theorem 1 be satisfied. Denote by*

$$\sigma(\alpha) = \max_{\lambda\in[\lambda_{\min},\lambda_{\max}]} \frac{|\alpha - \lambda|}{\alpha + \lambda}, \quad \zeta(\alpha) = \frac{\|S\|_2}{\alpha + \lambda_{\min}}.$$

Then for the convergence factor of $\rho(M(\alpha))$ it holds that

$$\rho(M(\alpha)) \le \left(\frac{(\sigma(\alpha))^2 + (\zeta(\alpha))^2}{1 + (\zeta(\alpha))^2}\right)^{1/2} \equiv \phi(\alpha) < 1. \tag{3.5}$$

Moreover, at

$$\alpha_\star = \begin{cases} \sqrt{\lambda_{\min}\lambda_{\max}}, & for\ \|S\|_2 \le \lambda_{\min}\sqrt{\kappa-1}, \\ \sqrt{\lambda_{\min}^2 + \|S\|_2^2}, & for\ \|S\|_2 \ge \lambda_{\min}\sqrt{\kappa-1}, \end{cases} \tag{3.6}$$

the function $\phi(\alpha)$ attains its minimum

$$\phi(\alpha_\star) = \begin{cases} (\frac{\eta^2+\tau^2}{1+\tau^2})^{1/2}, & for\ \|S\|_2 \le \lambda_{\min}\sqrt{\kappa-1}, \\ (\frac{1-\upsilon}{1+\upsilon})^{1/2}, & for\ \|S\|_2 \ge \lambda_{\min}\sqrt{\kappa-1}, \end{cases} \tag{3.7}$$

where

$$\eta = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}, \quad \tau = \frac{\|S\|_2}{(\sqrt{\kappa}+1)\lambda_{\min}} \quad and \quad \upsilon = \frac{\lambda_{\min}}{\sqrt{\lambda_{\min}^2 + \|S\|_2^2}}.$$

Proof. From Theorem 3.1 in [6], we can directly obtain (3.5)–(3.7). □

Remark 2. *$\alpha_\star$ is called the theoretical quasi-optimal parameter of the SS iteration method. Similarly, the theoretical quasi-optimal parameter $\beta_\star$ of the inner iterations* (2.6) *can be also obtained, which has the same form as $\alpha_\star$.*

In the following, we present another convergence theorem for a new form.

Theorem 3. *Let the conditions of Theorem 1 be satisfied. If $\{X^{(k)}\}_{k=0}^{\infty} \subseteq \mathbb{C}^{m\times n}$ is an iteration sequence generated by Algorithm 1 and if $X^\star \in \mathbb{C}^{m\times n}$ is the exact solution of the matrix Eq* (1.1)*, then it holds that*

$$\|X^{(k+1)} - X^\star\|_F \le (\varphi(\alpha) + \mu\theta\varepsilon_k)\|X^{(k)} - X^\star\|_F, \quad k = 0, 1, 2, \ldots$$

where the constants μ and θ are given by

$$\mu = \|B^{-T} \otimes (\alpha I_m + A)^{-1}\|_2, \quad \theta = \|B^T \otimes A\|_2.$$

In particular, when

$$\varphi(\alpha) + \mu\theta\varepsilon_{\max} < 1, \tag{3.8}$$

the iteration sequence $\{X^{(k)}\}_{k=0}^{\infty}$ converges to $X^\star$, where $\varepsilon_{\max} = \max_k\{\varepsilon_k\}$.

Proof. We can rewrite the SS iteration in Algorithm 1 as the following form:

$$(B^T \otimes (\alpha I_m + A))\mathbf{z}^{(k)} = 2\mathbf{r}^{(k)}, \qquad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{z}^{(k)}, \tag{3.9}$$

with $\mathbf{r}^{(k)} = \mathbf{c} - (B^T \otimes A)\mathbf{x}^{(k)}$, where $\mathbf{z}^{(k)}$ is such that the residual

$$\mathbf{p}^{(k)} = 2\mathbf{r}^{(k)} - (B^T \otimes (\alpha I_m + A))\mathbf{z}^{(k)}$$

satisfies $\|\mathbf{p}^{(k)}\|_2 \le \varepsilon_k\|\mathbf{r}^{(k)}\|_2$.

In fact, the inexact variant of the SS iteration method for solving the linear system (1.2) is just the above iteration scheme (3.9). From (3.9), we obtain

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} + (B^T \otimes (\alpha I_m + A))^{-1}(2\mathbf{r}^{(k)} - \mathbf{p}^{(k)}) \\ &= \mathbf{x}^{(k)} + (B^T \otimes (\alpha I_m + A))^{-1}\left(2\mathbf{c} - 2(B^T \otimes A)\mathbf{x}^{(k)} - \mathbf{p}^{(k)}\right) \\ &= \left(I_n \otimes \left((\alpha I_m + A)^{-1}(\alpha I_m - A)\right)\right)\mathbf{x}^{(k)} + 2\left(B^{-T} \otimes (\alpha I_m + A)^{-1}\right)\mathbf{c} \\ &\quad - \left(B^{-T} \otimes (\alpha I_m + A)^{-1}\right)\mathbf{p}^{(k)}. \end{aligned} \tag{3.10}$$

Because $\mathbf{x}^\star \in \mathbb{C}^n$ is the exact solution of the linear system (1.2), it must satisfy

$$\mathbf{x}^\star = \Big(I_n \otimes \big((\alpha I_m + A)^{-1}(\alpha I_m - A)\big)\Big)\mathbf{x}^\star + 2\Big(B^{-T} \otimes (\alpha I_m + A)^{-1}\Big)\mathbf{c}. \tag{3.11}$$

By subtracting (3.11) from (3.10), we have

$$\mathbf{x}^{(k+1)} - \mathbf{x}^\star = \Big(I_n \otimes \big((\alpha I_m + A)^{-1}(\alpha I_m - A)\big)\Big)(\mathbf{x}^{(k)} - \mathbf{x}^\star) - \Big(B^{-T} \otimes (\alpha I_m + A)^{-1}\Big)\mathbf{p}^{(k)}. \tag{3.12}$$

Taking norms on both sides from (3.12), then

$$\begin{aligned} &\|\mathbf{x}^{(k+1)} - \mathbf{x}^\star\|_2 \\ &\le \|I_n \otimes \big((\alpha I_m + A)^{-1}(\alpha I_m - A)\big)\|_2 \|\mathbf{x}^{(k)} - \mathbf{x}^\star\|_2 + \|B^{-T} \otimes (\alpha I_m + A)^{-1}\|_2 \|\mathbf{p}^{(k)}\|_2 \\ &\le \varphi(\alpha)\|\mathbf{x}^{(k)} - \mathbf{x}^\star\|_2 + \mu\varepsilon_k \|\mathbf{r}^{(k)}\|_2. \end{aligned} \tag{3.13}$$

Noticing that

$$\|\mathbf{r}^{(k)}\|_2 = \|\mathbf{c} - (B^T \otimes A)\mathbf{x}^{(k)}\|_2 = \|(B^T \otimes A)(\mathbf{x}^\star - \mathbf{x}^{(k)})\|_2 \le \theta\|\mathbf{x}^{(k)} - \mathbf{x}^\star\|_2,$$

by (3.13) the estimate

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^\star\|_2 \le (\varphi(\alpha) + \mu\theta\varepsilon_k)\|\mathbf{x}^{(k)} - \mathbf{x}^\star\|_2, \quad k = 0, 1, 2, \ldots \tag{3.14}$$

can be obtained. Note that for a matrix $Y \in \mathbb{C}^{m\times n}$, $\|Y\|_F = \|\mathbf{y}\|_2$, where the vector $\mathbf{y}$ contains the concatenated columns of the matrix Y. Then the estimate (3.14) can be equivalently rewritten as

$$\|X^{(k+1)} - X^\star\|_F \le (\varphi(\alpha) + \mu\theta\varepsilon_k)\|X^{(k)} - X^\star\|_F, \quad k = 0, 1, 2, \ldots.$$

So we can easily get the above conclusion. □

Remark 3. *From Theorem 3 we know that, in order to guarantee the convergence of the SS iteration, it is not necessary for the condition* $\varepsilon_k \to 0$ *. All we need is that the condition* (3.8) *is satisfied.*

4. Numerical results

In this section, two different matrix equations are solved by the HSS, SS and NSCG iteration methods. The efficiencies of the above iteration methods are examined by comparing the number of outer iteration steps (denoted by IT-out), the average number of inner iteration steps (denoted by IT-in-1 and IT-in-2 for the HSS, IT-in for the SS), and the elapsed CPU times (denoted by CPU). The notation "–" shows that no solution has been obtained after 1000 outer iteration steps.

The initial guess is the zero matrix. All iterations are terminated once $X^{(k)}$ satisfies

$$\frac{\|C - AX^{(k)}B\|_F}{\|C\|_F} \le 10^{-6}.$$

We set $\varepsilon_k = 0.01$, $k = 0, 1, 2, \ldots$ to be the tolerances for all the inner iteration schemes.

Moreover, in practical computation, we choose direct algorithms to solve all sub-equations involved in each step. We use Cholesky and LU factorization for the Hermitian and non-Hermitian coefficient matrices, respectively.

Example 1 ([2]) We consider the matrix Eq (1.1) with $m = n$ and

$$A = M + 5qN + \frac{100}{(n+1)^2}I \quad \text{and} \quad B = M + 2qN + \frac{100}{(n+1)^2}I,$$

where $M, N \in \mathbb{R}^{n\times n}$ are two tridiagonal matrices as follows:

$$M = \text{tridiag}(-1, 2, -1) \quad \text{and} \quad N = \text{tridiag}(0.5, 0, -0.5).$$

In Tables 1 and 2, the theoretical quasi-optimal parameters and experimental optimal parameters of HSS and SS are listed, respectively. In Tables 3 and 4, the numerical results of HSS and SS are listed.

From Tables 3 and 4 it can be observed that, the SS outperforms the HSS for various n and q, especially when q is small (the coefficient matrices are ill-conditioned).

Table 1. The theoretical quasi-optimal parameters of HSS and SS for Example 1.

Method		HSS		SS	
n	q	α_{quasi}	β_{quasi}	α_{quasi}	β_{quasi}
$n = 16$	$q = 0.1$	1.28	1.28	1.28	1.28
	$q = 0.3$	1.28	1.28	1.52	1.28
	$q = 1$	1.28	1.28	4.93	2.00
$n = 32$	$q = 0.1$	0.64	0.64	0.64	0.64
	$q = 0.3$	0.64	0.64	1.50	0.64
	$q = 1$	0.64	0.64	4.98	1.99
$n = 64$	$q = 0.1$	0.32	0.32	0.50	0.32
	$q = 0.3$	0.32	0.32	1.50	0.60
	$q = 1$	0.32	0.32	4.99	2.00
$n = 128$	$q = 0.1$	0.16	0.16	0.50	0.20
	$q = 0.3$	0.16	0.16	1.50	0.60
	$q = 1$	0.16	0.16	5.00	2.00

Table 2. The experimental optimal iteration parameters of HSS and SS for Example 1.

Method		HSS		SS	
n	q	α_{exp}	β_{exp}	α_{exp}	β_{exp}
$n = 16$	$q = 0.1$	1.22	1.14	1.14	0.98
	$q = 0.3$	1.40	1.12	1.66	1.16
	$q = 1$	1.96	1.30	0.36	1.74
$n = 32$	$q = 0.1$	1.84	0.72	0.70	0.66
	$q = 0.3$	1.04	0.72	1.12	0.68
	$q = 1$	1.70	0.90	3.02	0.84
$n = 64$	$q = 0.1$	3.00	0.40	0.20	0.40
	$q = 0.3$	1.10	0.40	0.90	0.50
	$q = 1$	1.30	0.60	2.30	0.70
$n = 128$	$q = 0.1$	3.00	0.30	0.30	0.20
	$q = 0.3$	1.10	0.30	0.60	0.30
	$q = 1$	1.10	0.80	2.90	0.60

Table 3. Numerical results of HSS and SS with the theoretical quasi-optimal parameters for Example 1.

Method		HSS				SS		
n	q	IT-out	IT-in-1	IT-in-2	CPU	IT-out	IT-in	CPU
$n = 16$	$q = 0.1$	22	7.2	7.0	0.0151	11	4.0	0.0036
	$q = 0.3$	16	7.0	7.0	0.0104	9	4.0	0.0029
	$q = 1$	20	6.0	6.0	0.0117	17	5.0	0.0078
$n = 32$	$q = 0.1$	36	14.0	14.0	0.1610	19	6.9	0.0295
	$q = 0.3$	33	14.0	14.0	0.1216	15	7.0	0.0269
	$q = 1$	39	11.0	11.0	0.1168	24	10.0	0.0472
$n = 64$	$q = 0.1$	68	28.3	28.3	1.6490	30	13.0	0.2677
	$q = 0.3$	74	24.7	24.8	1.5486	27	16.0	0.2906
	$q = 1$	87	25.0	25.0	1.8381	35	20.0	0.4377
$n = 128$	$q = 0.1$	144	54.5	54.7	34.394	57	21.2	4.3253
	$q = 0.3$	188	45.1	45.9	36.729	48	35.0	6.2253
	$q = 1$	465	52.0	52.0	104.331	52	38.0	6.7005

Table 4. Numerical results of HSS and SS with the experimental optimal iteration parameters for Example 1.

Method		HSS				SS		
n	q	IT-out	IT-in-1	IT-in-2	CPU	IT-out	IT-in	CPU
$n = 16$	$q = 0.1$	19	7.0	7.0	0.0096	11	4.0	0.0023
	$q = 0.3$	15	8.0	8.0	0.0093	8	4.0	0.0017
	$q = 1$	16	6.0	6.0	0.0070	11	4.0	0.0022
$n = 32$	$q = 0.1$	29	13.0	13.0	0.0853	18	7.0	0.0227
	$q = 0.3$	23	13.0	13.0	0.0667	12	7.0	0.0162
	$q = 1$	25	11.0	11.0	0.0637	20	7.0	0.0244
$n = 64$	$q = 0.1$	118	24.0	24.0	2.0849	30	10.5	0.1952
	$q = 0.3$	41	23.0	23.0	0.6952	16	11.1	0.1008
	$q = 1$	37	18.0	18.0	0.4733	30	10.0	0.1650
$n = 128$	$q = 0.1$	229	39.7	39.7	32.032	40	20.5	2.4942
	$q = 0.3$	70	35.0	35.0	8.6951	22	18.0	1.2280
	$q = 1$	53	38.6	38.6	7.9165	45	14.0	1.7385

Moreover, as two single-step methods, the numerical results of NSCG and SS are compared in Table 5. From Table 5 we see that the SS method has better computing efficiency than the NSCG method.

Table 5. Numerical results of NSCG and SS for Example 1.

Method		NSCG			SS		
n	q	IT-out	IT-in	CPU	IT-out	IT-in	CPU
$n = 16$	$q = 0.1$	15	25.4	0.0230	11	4.0	0.0023
	$q = 0.3$	291	30.5	0.2445	8	4.0	0.0017
	$q = 1$	90	170.9	0.4020	11	4.0	0.0022
$n = 32$	$q = 0.1$	–	–	–	18	7.0	0.0227
	$q = 0.3$	45	488.6	1.9451	12	7.0	0.0162
	$q = 1$	75	493.3467	2.9591	20	7.0	0.0244
$n = 64$	$q = 0.1$	–	–	–	30	10.5	0.1952
	$q = 0.3$	77	497.7	9.5250	16	11.1	0.1008
	$q = 1$	62	494.5	7.5782	30	10.0	0.1650
$n = 128$	$q = 0.1$	74	493.3	48.129	40	20.5	2.4942
	$q = 0.3$	69	492.9	44.699	22	18.0	1.2280
	$q = 1$	69	492.8	45.885	45	14.0	1.7385

Example 2 ([2]) We consider the matrix Eq (1.1) with $m = n$ and

$$\begin{cases} A = \text{diag}(1, 2, \cdots, n) + rL^T, \\ B = 2^{-t} I_n + \text{diag}(1, 2, \cdots, n) + rL^T + 2^{-t} L, \end{cases}$$

where L is a strictly lower triangular matrix and all the elements in the lower triangle part are ones, and t is a specified problem parameter. In our tests, we take $t = 1$.

In Tables 6 and 7, for various n and r, we list the theoretical quasi-optimal parameters and experimental optimal parameters of HSS and SS, respectively. In Tables 8 and 9, the numerical results of HSS and SS are listed. Moreover, the numerical results of NSCG and SS are compared in Table 10.

From Tables 8–10 we get the same conclusion as example 1.

Therefore, for large sparse matrix equation $AXB = C$, the SS method is an effective iterative approach.

Table 6. The theoretical quasi-optimal parameters of HSS and SS for Example 2.

Method		HSS		SS	
n	q	$\alpha_{\exp}$	$\beta_{\exp}$	$\alpha_{\exp}$	$\beta_{\exp}$
$n = 32$	$r = 0.01$	5.66	6.75	5.66	6.75
	$r = 0.1$	5.63	6.71	5.63	6.71
	$r = 1$	4.94	6.36	10.20	6.36
$n = 64$	$r = 0.01$	8.00	9.47	8.00	10.07
	$r = 0.1$	7.96	9.41	7.96	9.41
	$r = 1$	6.90	8.89	20.38	10.22
$n = 128$	$r = 0.01$	11.31	13.31	11.31	20.01
	$r = 0.1$	11.25	13.23	11.25	16.35
	$r = 1$	9.66	12.46	40.75	20.39
$n = 256$	$r = 0.01$	16.00	18.75	16.00	39.95
	$r = 0.1$	15.91	18.63	15.91	32.62
	$r = 1$	13.55	17.50	81.49	40.75

Table 7. The experimental optimal iteration parameters of HSS and SS for Example 2.

Method		HSS		SS	
n	q	$\alpha_{\exp}$	$\beta_{\exp}$	$\alpha_{\exp}$	$\beta_{\exp}$
$n = 32$	$r = 0.01$	7	10	7	13
	$r = 0.1$	7	9	7	14
	$r = 1$	7	6	30	10
$n = 64$	$r = 0.01$	10	11	10	25
	$r = 0.1$	11	12	10	26
	$r = 1$	10	1	60	15
$n = 128$	$r = 0.01$	16	10	15	49
	$r = 0.1$	16	12	15	53
	$r = 1$	16	2	120	23
$n = 256$	$r = 0.01$	24	16	22	98
	$r = 0.1$	24	16	24	104
	$r = 1$	24	4	239	34

Table 8. Numerical results of HSS and SS with the theoretical quasi-optimal parameters for Example 2.

Method		HSS				SS		
n	q	IT-out	IT-in-1	IT-in-2	CPU	IT-out	IT-in	CPU
$n = 32$	$r = 0.01$	37	10.3	10.3	0.1743	18	6.0	0.0373
	$r = 0.1$	37	10.3	10.3	0.1380	18	7.0	0.0388
	$r = 1$	39	10.4	10.5	0.1376	11	9.0	0.0306
$n = 64$	$r = 0.01$	60	12.9	12.9	0.9061	25	8.0	0.2025
	$r = 0.1$	56	12.9	12.9	0.8133	25	9.0	0.2330
	$r = 1$	69	18.6	18.7	1.4371	11	12.0	0.1388
$n = 128$	$r = 0.01$	95	19.7	19.7	9.9527	35	8.0	1.2843
	$r = 0.1$	96	20.2	20.3	10.807	35	10.0	1.4921
	$r = 1$	100	27.3	27.4	14.947	11	12.0	0.5410
$n = 256$	$r = 0.01$	100	28.4	28.4	93.739	49	8.0	10.863
	$r = 0.1$	100	29.2	29.2	92.406	49	10.0	13.902
	$r = 1$	100	39.0	38.8	124.37	11	12.0	3.8920

Table 9. Numerical results of HSS and SS with the experimental optimal iteration parameters for Example 2.

Method		HSS				SS		
n	q	IT-out	IT-in-1	IT-in-2	CPU	IT-out	IT-in	CPU
$n = 32$	$r = 0.01$	32	7.7	7.7	0.0787	16	3.1	0.0130
	$r = 0.1$	31	8.2	8.2	0.0783	16	3.2	0.0127
	$r = 1$	30	7.0	7.0	0.0646	2	6.0	0.0028
$n = 64$	$r = 0.01$	43	12.2	12.2	0.5301	21	3.2	0.0510
	$r = 0.1$	42	11.4	11.4	0.4874	21	3.3	0.0647
	$r = 1$	39	55.9	55.9	2.1008	2	8.0	0.0117
$n = 128$	$r = 0.01$	57	24.3	24.3	5.8351	28	3.4	0.3985
	$r = 0.1$	54	20.3	20.3	4.8781	27	3.3	0.3821
	$r = 1$	53	55.2	55.2	12.885	2	10.8	0.0902
$n = 256$	$r = 0.01$	75	30.9	30.9	59.116	36	3.1	2.5999
	$r = 0.1$	70	30.1	30.1	64.391	34	3.2	3.3392
	$r = 1$	66	52.3	52.3	85.011	2	14.0	0.7423

Table 10. Numerical results of NSCG and SS for Example 2.

Method		NSCG			SS		
n	q	IT-out	IT-in	CPU	IT-out	IT-in	CPU
$n = 32$	$r = 0.01$	17	25.4	0.0467	16	3.1	0.0130
	$r = 0.1$	18	50.1	0.0717	16	3.2	0.0127
	$r = 1$	100	87.3	0.6921	2	6.0	0.0028
$n = 64$	$r = 0.01$	21	36.2	0.2206	21	3.2	0.0510
	$r = 0.1$	23	86.9	0.4631	21	3.3	0.0647
	$r = 1$	100	99.5	2.4361	2	8.0	0.0117
$n = 128$	$r = 0.01$	25	51.0	1.7965	28	3.4	0.3985
	$r = 0.1$	29	96.5	3.4998	27	3.3	0.3821
	$r = 1$	100	100	12.997	2	10.8	0.0902
$n = 256$	$r = 0.01$	31	64.0	17.264	36	3.1	2.5999
	$r = 0.1$	37	98.1	32.679	34	3.2	3.3392
	$r = 1$	100	100	96.801	2	14.0	0.7423

5. Conclusions

By utilizing an inner-outer iteration strategy, we established a shift-splitting (SS) iteration method for large sparse linear matrix equations $AXB = C$. Two different convergence theories were analysed in depth. Furthermore, the quasi-optimal parameters of SS iteration matrix are given. Numerical experiments illustrated that, the SS method can always outperform the HSS and NSCG methods both in outer and inner iteration numbers and computing time, especially for the ill-conditioned coefficient matrices.

Acknowledgments

The authors are very grateful to the anonymous referees for their helpful comments and suggestions on the manuscript. This research is supported by the Natural Science Foundation of Gansu Province (No. 20JR5RA464), the National Natural Science Foundation of China (No. 11501272), and the China Postdoctoral Science Foundation funded project (No. 2016M592858).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. O. Axelsson, Z. Z. Bai, S. X. Qiu, A class of nested iteration schemes for linear systems with a coefficient matrix with a dominant positive definite symmetric part, *Numer. Algorithms*, **35** (2004), 351–372.

2. Z. Z. Bai, On Hermitian and skew-Hermitian splitting iteration methods for continuous Sylvester equations, *J. Comput. Math.*, **29** (2011), 185–198.

3. Z. Z. Bai, M. Benzi, F. Chen, On preconditioned MHSS iteration methods for complex symmetric linear systems, *Numer. Algorithms*, **56** (2011), 297–317.

4. Z. Z. Bai, G. H. Golub, C. K. Li, Convergence properties of preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite matrices, *Math. Comput.*, **76** (2007), 287–298.

5. 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.

6. Z. Z. Bai, A. Hadjidimos, Optimization of extrapolated Cayley transform with non-Hermitian positive definite matrix, *Linear Algebra Appl.*, **463** (2014), 322–339.

7. Z. Z. Bai, J. F. Yin, Y. F. Su, A shift-splitting preconditioner for non-Hermitian positive definite matrices, *J. Comput. Math.*, **24** (2006), 539–552.

8. Z. Z. Bai, G. H. Golub, L. Z. Lu, J. F. Yin, Block triangular and skew-Hermitian splitting methods for positive-definite linear systems, *SIAM J. Sci. Comput.*, **26** (2005), 844–863.

9. D. Calvetti, L. Reichel, Application of ADI iterative methods to the restoration of noisy images, *SIAM J. Matrix Anal. Appl.*, **17** (1996), 165–186.

10. Y. Cao, S. Li, L. Q. Yao, A class of generalized shift-splitting preconditioners for nonsymmetric saddle point problems, *Appl. Math. Lett.*, **49** (2015), 20–27.

11. B. N. Datta, *Numerical Methods for Linear Control Systems*, San Diego: Elsevier Academic Press, 2004.

12. Y. X. Dong, C. Q. Gu, On PMHSS iteration methods for continuous Sylvester equations, *J. Comput. Math.*, **35** (2017), 600–619.

13. D. W. Fausett, C. T. Fulton, Large least squares problems involving Kronecker products, *SIAM J. Matrix Anal. Appl.*, **15** (1994), 219–227.

14. M. Khorsand Zak, F. Toutounian, Nested splitting conjugate gradient method for matrix equation $AXB = C$ and preconditioning, *Comput. Math. Appl.*, **66** (2013), 269–278.

15. M. Khorsand Zak, F. Toutounian, An iterative method for solving the continuous Sylvester equation by emphasizing on the skew-hermitian parts of the coefficient matrices, *Int. J. Comput. Math.*, **94** (2017), 633–649.

16. X. Li, H. F. Huo, A. L. Yang, Preconditioned HSS iteration method and its non-alternating variant for continuous Sylvester equations, *Comput. Math. Appl.*, **75** (2018), 1095–1106.

17. 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.

18. X. Li, Y. J. Wu, A. L. Yang, J. Y. Yuan, A generalized HSS iteration method for continuous Sylvester equations, *J. Appl. Math.*, **2014** (2014), 578102.

19. U. A. Rauhala, Introduction to array algebra, *Photogramm. Eng. Remote Sens.*, **46** (1980), 177–192.

20. P. A. Regalia, S. K. Mitra, Kronecker products, unitary matrices and signal processing applications, *SIAM Rev.*, **31** (1989), 586–613.

21. D. K. Salkuyeh, M. Masoudi, D. Hezari, On the generalized shift-splitting preconditioner for saddle point problems, *Appl. Math. Lett.*, **48** (2015), 55–61.

22. X. Wang, W. W. Li, L. Z. Mao, On positive-definite and skew-Hermitian splitting iteration methods for continuous Sylvester equation $AX + XB = C$, *Comput. Math. Appl.*, **66** (2013), 2352–2361.

23. X. Wang, Y. Li, L. Dai, On Hermitian and skew-Hermitian splitting iteration methods for the linear matrix equation $AXB = C$, *Comput. Math. Appl.*, **65** (2013), 657–664.

24. Y. J. Wu, X. Li, J. Y. Yuan, A non-alternating preconditioned HSS iteration method for non-Hermitian positive definite linear systems, *Comput. Appl. Math.*, **36** (2017), 367–381.

25. L. Xie, J. Ding, F. Ding, Gradient based iterative solutions for general linear matrix equations, *Comput. Math. Appl.*, **58** (2009), 1441–1448.

26. L. Xie, Y. J. Liu, H. Z. Yang, Gradient based and least squares based iterative algorithms for matrix equations $AXB + CX^TD = F$, *Appl. Math. Comput.*, **217** (2010), 2191–2199.

27. A. L. Yang, J. An, Y. J. Wu, A generalized preconditioned HSS method for non-Hermitian positive definite linear systems, *Appl. Math. Comput.*, **216** (2010), 1715–1722.

28. H. Y. Zha, Comments on large least squares problems involving Kronecker products, *SIAM J. Matrix Anal. Appl.*, **16** (1995), 1172–1172.

29. W. H. Zhang, A. L. Yang, Y. J. Wu, Parameterized preconditioned Hermitian and skew-Hermitian splitting iteration method for a class of linear matrix equations, *Comput. Math. Appl.*, **70** (2015), 1357–1367.

30. Q. Q. Zheng, C. F. Ma, On normal and skew-Hermitian splitting iteration methods for large sparse continuous Sylvester equations, *J. Comput. Appl. Math.*, **268** (2014), 145–154.

31. D. M. Zhou, G. L. Chen, Q. Y. Cai, On modified HSS iteration methods for continuous Sylvester equations, *Appl. Math. Comput.*, **263** (2015), 84–93.

32. R. Zhou, X. Wang, X. B. Tang, A generalization of the Hermitian and skew-Hermitian splitting iteration method for solving Sylvester equations, *Appl. Math. Comput.*, **271** (2015), 609–617.

33. R. Zhou, X. Wang, X. B. Tang, Preconditioned positive-definite and skew-Hermitian splitting iteration methods for continuous Sylvester equations $AX + XB = C$, *East Asian J. Appl. Math.*, **7** (2017), 55–69.

34. R. Zhou, X. Wang, P. Zhou, A modified HSS iteration method for solving the complex linear matrix equation $AXB = C$, *J. Comput. Math.*, **34** (2016), 437–450.

AIMS Press