Computing (2010) 87:93–111
DOI 10.1007/s00607-010-0077-0

Modified HSS iteration methods for a class of complex symmetric linear systems

Zhong-Zhi Bai · Michele Benzi · Fang Chen

Received: 20 October 2009 / Accepted: 21 December 2009 / Published online: 4 February 2010
© Springer-Verlag 2010

Abstract In this paper, we introduce and analyze a modification of the Hermitian and skew-Hermitian splitting iteration method for solving a broad class of complex symmetric linear systems. We show that the *modified Hermitian and skew-Hermitian splitting* (MHSS) iteration method is unconditionally convergent. Each iteration of

Communicated by C.C. Douglas.

This work was supported by The National Basic Research Program (No. 2005CB321702) and The National Outstanding Young Scientist Foundation (No. 10525102), P. R. China, and by The US National Science Foundation grants DMS-0511336 and DMS-0810862.

Z.-Z. Bai
School of Mathematics and Computer Science, Guizhou Normal University,
550001 Guiyang, People's Republic of China

Z.-Z. Bai (✉)
State Key Laboratory of Scientific/Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, P.O. Box 2719, 100190 Beijing, People's Republic of China
e-mail: bzz@lsec.cc.ac.cn

M. Benzi
Department of Mathematics and Computer Science, Emory University, Atlanta, GA 30322, USA
e-mail: benzi@mathcs.emory.edu

F. Chen
Department of Mathematical Sciences, Xi'an Jiaotong University,
710049 Xi'an, People's Republic of China

F. Chen
Department of Mathematics and Physics, Xi'an University of Post and Telecommunication,
710121 Xi'an, People's Republic of China
e-mail: chenfreesky@gmail.com

this method requires the solution of two linear systems with real symmetric positive definite coefficient matrices. These two systems can be solved inexactly. We consider acceleration of the MHSS iteration by Krylov subspace methods. Numerical experiments on a few model problems are used to illustrate the performance of the new method.

Keywords Complex symmetric matrix · Hermitian and skew-Hermitian splitting · Iteration method · Krylov subspace method · Convergence analysis · Preconditioning

Mathematics Subject Classification (2000) 65F10 · 65F50 · 65N22; CR: G1.3

1 Introduction

We consider the iterative solution of systems of linear equations of the form

$$Ax = b, \quad A \in \mathbb{C}^{n\times n} \quad \text{and} \quad x, b \in \mathbb{C}^n, \tag{1}$$

where $A \in \mathbb{C}^{n\times n}$ is a complex symmetric matrix of the form

$$A = W + \mathrm{i}\, T, \tag{2}$$

and $W, T \in \mathbb{R}^{n\times n}$ are real symmetric matrices, with W being positive definite and T positive semidefinite. Here and in the sequel we use $\mathrm{i} = \sqrt{-1}$ to denote the imaginary unit. We assume $T \neq 0$, which implies that A is non-Hermitian.

Complex symmetric linear systems of this kind arise in many problems in scientific computing and engineering applications, including diffuse optical tomography [1], FFT-based solution of certain time-dependent PDEs [6], quantum mechanics [15], molecular scattering [11], structural dynamics [8], and lattice quantum chromodynamics [9]. For more examples and additional references, we refer to [5].

The Hermitian and skew-Hermitian parts of the complex symmetric matrix $A \in \mathbb{C}^{n\times n}$ are given by

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

respectively, hence, $A \in \mathbb{C}^{n\times n}$ is a non-Hermitian, but positive definite matrix. Here A^* is used to denote the conjugate transpose of the matrix A. Based on the *Hermitian and skew-Hermitian splitting* (HSS)

$$A = H + S$$

of the matrix $A \in \mathbb{C}^{n\times n}$, we can straightforwardly employ the HSS iteration method, introduced in Bai et al. [3], to compute an approximate solution for the complex symmetric linear system (1–2). An algorithmic description of the correspondingly induced HSS iteration method is given as follows.

The HSS Iteration Method. Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the sequence of iterates $\{x^{(k)}\}_{k=0}^{\infty} \subset \mathbb{C}^n$ converges, compute the next iterate $x^{(k+1)}$ according to the following procedure:

$$\begin{cases} (\alpha I + W)x^{(k+\frac{1}{2})} = (\alpha I - \mathrm{i}\, T)x^{(k)} + b, \\ (\alpha I + \mathrm{i}\, T)x^{(k+1)} = (\alpha I - W)x^{(k+\frac{1}{2})} + b, \end{cases} \tag{3}$$

where α is a given positive constant and I is the identity matrix.

Note that this HSS iteration scheme is a specialized version of the original one, in which the complex symmetric structure of the coefficient matrix $A \in \mathbb{C}^{n\times n}$ is particularly highlighted. Since $W \in \mathbb{R}^{n\times n}$ is symmetric positive definite, we know from [3] that the HSS iteration method is convergent for any positive constant α.

Of course, at each step of the HSS iteration we need to solve two linear sub-systems with their coefficient matrices being the symmetric positive definite one $\alpha I + W$ and the shifted skew-Hermitian one $\alpha I + \mathrm{i}\, T$, respectively. The matrix $\alpha I + W$ can be treated in real arithmetic, so the iterate $x^{(k+\frac{1}{2})}$ in the first-half step may be effectively computed either exactly by the Cholesky factorization or inexactly by the conjugate gradient scheme; see [10]. The matrix $\alpha I + \mathrm{i}\, T$ is, however, complex and non-Hermitian, although it is positive definite and symmetric. This necessitates the use of complex arithmetic, and the iterate $x^{(k+1)}$ in the second-half step may be computed either exactly by some variant of Bunch–Parlett factorization or inexactly by certain Krylov subspace iteration schemes such as the Concus–Golub–Widlund (CGW) method or GMRES; see [10,13,14]. We refer to [3,4] for actual implementations and detailed analyses of inexact variants of the HSS iteration method.

A potential difficulty with the HSS iteration approach is the need to solve the shifted skew-Hermitian sub-system of linear equations at each iteration step. In some cases its solution is as difficult as that of the original problem, although there are situations where the matrix T is structured in such a way as to make linear systems involving $\alpha I + \mathrm{i}\, T$ easy to solve. In general, however, this will not be the case. A remedy may be using inner iterative schemes such as some preconditioned Krylov subspace iterations to solve this class of linear systems to relatively low accuracy. There is now considerable evidence that the good convergence properties of the HSS iteration method are preserved even when the inner solves are performed to rather low accuracy, resulting in significant savings, especially for very large problems. When applied to shifted skew-Hermitian linear systems, however, the Krylov subspace iterations need to be implemented in complex arithmetic. Moreover, their convergence rates tend to be considerably worse than the rapidly convergent conjugate gradient method applied to symmetric positive definite linear systems. This may result in slow convergence and low efficiency of the inexact HSS iteration methods.

In this paper, a modification of the HSS iteration scheme is presented and some of its basic properties are studied. A considerable advantage of the *modified HSS* (MHSS) iteration method consists in the fact that solution of linear system with coefficient matrix $\alpha I + \mathrm{i}\, T$ is avoided and only two linear sub-systems with coefficient matrices

$\alpha I + W$ and $\alpha I + T$, both being real and symmetric positive definite, need to be solved at each step. Therefore, operations on these two matrices, such as complete or incomplete factorizations, can be carried out using real arithmetic only. The computation of the iterates $x^{(k+\frac{1}{2})}$ requires only real arithmetic, whereas the computation of $x^{(k+1)}$ requires a modest amount of complex arithmetic due to the fact that the right-hand side in the corresponding system is complex. Both can be efficiently computed either exactly by a sparse Cholesky factorization, or inexactly by a preconditioned conjugate gradient scheme. For systems arising from elliptic PDEs, multigrid solvers may be very competitive. Moreover, like the HSS iteration method, the MHSS iteration method also converges unconditionally to the unique solution of the complex symmetric linear system (1–2). Theoretical analysis shows that an upper bound on the contraction factor of the MHSS iteration depends on the spectra of the symmetric positive definite matrices W and T, but is independent of the eigenvectors of the matrices W, T and A. In addition, the optimal value of the iteration parameter α for an upper bound of the contraction factor of the MHSS iteration can be determined by the lower and the upper eigenvalue bounds of the matrix W.

The remainder of the paper is organized as follows: in Sect. 2, the MHSS iteration method is described and its convergence properties are discussed. Some implementation aspects are briefly discussed in Sect. 3. The results of numerical experiments on a few model problems are discussed in Sect. 4. Finally, in Sect. 5 we offer brief concluding remarks to end the paper.

2 The MHSS iteration method

By making use of the special structure of the coefficient matrix $A \in \mathbb{C}^{n\times n}$ of the complex symmetric linear system (1–2), in this section we derive a modification of the HSS iteration method that was initially proposed in Bai et al. [3]. The new splitting iteration method will be referred to as the *modified HSS* (MHSS) iteration method or, in brief, the MHSS iteration method.

To this end, we first rewrite the complex symmetric linear system (1–2) into the system of fixed-point equations

$$(\alpha I + W)x = (\alpha I - \mathrm{i}\, T)x + b, \tag{4}$$

where α is a prescribed positive parameter. Because the complex symmetric linear system (1–2) is also equivalent to the one $-\mathrm{i}\, Ax = -\mathrm{i}\, b$, or more explicitly,

$$(T - \mathrm{i}\, W)x = -\mathrm{i}\, b,$$

it can be alternatively rewritten as the system of fixed-point equations

$$(\alpha I + T)x = (\alpha I + \mathrm{i}\, W)x - \mathrm{i}\, b. \tag{5}$$

Now, by alternately iterating between the two systems of fixed-point equations (4) and (5), we can establish the following modified HSS iteration method for solving the

complex symmetric linear system (1–2) in an analogous fashion to the HSS iteration scheme in [3].

The modified HSS iteration method. Let $x^{(0)} \in \mathbb{C}^n$ be an arbitrary initial guess. For $k = 0, 1, 2, \ldots$ until the sequence of iterates $\{x^{(k)}\}_{k=0}^{\infty} \subset \mathbb{C}^n$ converges, compute the next iterate $x^{(k+1)}$ according to the following procedure:

$$\begin{cases} (\alpha I + W)x^{(k+\frac{1}{2})} = (\alpha I - \mathrm{i}\, T)x^{(k)} + b, \\ (\alpha I + T)x^{(k+1)} \ = (\alpha I + \mathrm{i}\, W)x^{(k+\frac{1}{2})} - \mathrm{i}\, b, \end{cases} \tag{6}$$

where α is a given positive constant.

Evidently, each iterate of the MHSS iteration alternates between the two symmetric matrices W and T. As $W \in \mathbb{R}^{n\times n}$ is symmetric positive definite, $T \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite and $\alpha \in \mathbb{R}$ is positive, we see that both matrices $\alpha I + W$ and $\alpha I + T$ are symmetric positive definite. Hence, the two linear sub-systems involved in each step of the MHSS iteration can be solved effectively using mostly real arithmetic either exactly by a Cholesky factorization or inexactly by some conjugate gradient or multigrid scheme. This is different from the HSS iteration method, in which a shifted skew-Hermitian linear sub-system with coefficient matrix $\alpha I + \mathrm{i}\, T$ needs to be solved at every iteration step. If sparse triangular factorizations are used to solve the sub-systems involved at each step, the MHSS method is likely to require considerably less storage than HSS since only two triangular factors rather than three have to be computed and stored.

After straightforward derivations we can reformulate the MHSS iteration scheme into the standard form

$$x^{(k+1)} = M(\alpha)x^{(k)} + G(\alpha)b, \quad k = 0, 1, 2, \ldots,$$

where

$$M(\alpha) = (\alpha I + T)^{-1}(\alpha I + \mathrm{i}\, W)(\alpha I + W)^{-1}(\alpha I - \mathrm{i}\, T)$$

and

$$G(\alpha) = (1 - \mathrm{i})\alpha(\alpha I + T)^{-1}(\alpha I + W)^{-1}.$$

Note that $M(\alpha)$ is the iteration matrix of the MHSS iteration method.

In addition, if we introduce matrices

$$B(\alpha) = \frac{1+\mathrm{i}}{2\alpha}(\alpha I + W)(\alpha I + T) \quad \text{and} \quad C(\alpha) = \frac{1+\mathrm{i}}{2\alpha}(\alpha I + \mathrm{i}\, W)(\alpha I - \mathrm{i}\, T),$$

then it holds that

$$A = B(\alpha) - C(\alpha) \quad \text{and} \quad M(\alpha) = B(\alpha)^{-1}C(\alpha). \tag{7}$$

Therefore, the MHSS iteration scheme is induced by the matrix splitting $A = B(\alpha) - C(\alpha)$ defined in (7). It follows that the splitting matrix $B(\alpha)$ can be used as a preconditioning matrix for the complex symmetric matrix $A \in \mathbb{C}^{n\times n}$. Note that the multiplicative factor $(1 + \mathrm{i})/(2\alpha)$ has no effect on the preconditioned system and therefore it can be dropped. Hence, the preconditioning matrix is just the real matrix $B(\alpha) = (\alpha I + W)(\alpha I + T)$. Note that $B(\alpha)$, being the product of two symmetric and positive definite matrices, is diagonalizable and has real positive eigenvalues. Matrix $B(\alpha)$ will be referred to as the MHSS preconditioner.

Concerning the convergence of the stationary MHSS iteration method, we have the following theorem.

Theorem 2.1 *Let $A = W + \mathrm{i}\,T \in \mathbb{C}^{n\times n}$, with $W \in \mathbb{R}^{n\times n}$ and $T \in \mathbb{R}^{n\times n}$ symmetric positive definite and symmetric positive semidefinite, respectively, and let α be a positive constant. Then the spectral radius $\rho(M(\alpha))$ of the MHSS iteration matrix $M(\alpha) = (\alpha I + T)^{-1}(\alpha I + \mathrm{i}\,W)(\alpha I + W)^{-1}(\alpha I - \mathrm{i}\,T)$ satisfies $\rho(M(\alpha)) \le \sigma(\alpha)$, where*

$$\sigma(\alpha) \equiv \max_{\lambda_j \in \mathrm{sp}(W)} \frac{\sqrt{\alpha^2 + \lambda_j^2}}{\alpha + \lambda_j},$$

where $\mathrm{sp}(W)$ denotes the spectrum of the matrix W. Therefore, it holds that

$$\rho(M(\alpha)) \le \sigma(\alpha) < 1, \quad \forall \alpha > 0,$$

i.e., the MHSS iteration converges to the unique solution $x_\star \in \mathbb{C}^n$ of the complex symmetric linear system (1–2) for any initial guess.

Proof By direct computations we have

$$\begin{aligned}
\rho(M(\alpha)) &= \rho((\alpha I + T)^{-1}(\alpha I + \mathrm{i}\,W)(\alpha I + W)^{-1}(\alpha I - \mathrm{i}\,T)) \\
&= \rho((\alpha I + \mathrm{i}\,W)(\alpha I + W)^{-1}(\alpha I - \mathrm{i}\,T)(\alpha I + T)^{-1}) \\
&\le \|(\alpha I + \mathrm{i}\,W)(\alpha I + W)^{-1}(\alpha I - \mathrm{i}\,T)(\alpha I + T)^{-1}\|_2 \\
&\le \|(\alpha I + \mathrm{i}\,W)(\alpha I + W)^{-1}\|_2 \|(\alpha I - \mathrm{i}\,T)(\alpha I + T)^{-1}\|_2.
\end{aligned}$$

Because $W \in \mathbb{R}^{n\times n}$ and $T \in \mathbb{R}^{n\times n}$ are symmetric, there exist orthogonal matrices $U, V \in \mathbb{R}^{n\times n}$ such that

$$U^T W U = \Lambda_W, \quad V^T T V = \Lambda_T,$$

where

$$\Lambda_W = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

and

$$\Lambda_T = \mathrm{diag}(\mu_1, \mu_2, \ldots, \mu_n),$$

with λ_j $(1 \le j \le n)$ and μ_j $(1 \le j \le n)$ being the eigenvalues of the matrices W and T, respectively. By assumption, it holds that

$$\lambda_j > 0 \quad \text{and} \quad \mu_j \ge 0, \quad j = 1, 2, \ldots, n.$$

Now, based on the orthogonal invariance of the Euclidean norm $\|\cdot\|_2$, we can further obtain the following upper bound on $\rho(M(\alpha))$:

$$\begin{aligned}\rho(M(\alpha)) &\le \|(\alpha I + \mathrm{i}\,\Lambda_W)(\alpha I + \Lambda_W)^{-1}\|_2 \|(\alpha I - \mathrm{i}\,\Lambda_T)(\alpha I + \Lambda_T)^{-1}\|_2 \\ &= \max_{\lambda_j \in \mathrm{sp}(W)} \left|\frac{\alpha + \mathrm{i}\,\lambda_j}{\alpha + \lambda_j}\right| \cdot \max_{\mu_j \in \mathrm{sp}(T)} \left|\frac{\alpha - \mathrm{i}\,\mu_j}{\alpha + \mu_j}\right| \\ &= \max_{\lambda_j \in \mathrm{sp}(W)} \frac{\sqrt{\alpha^2 + \lambda_j^2}}{\alpha + \lambda_j} \cdot \max_{\mu_j \in \mathrm{sp}(T)} \frac{\sqrt{\alpha^2 + \mu_j^2}}{\alpha + \mu_j}.\end{aligned}$$

Recalling that $\mu_j \ge 0$ holds for all $\mu_j \in \mathrm{sp}(T)$ $(1 \le j \le n)$, we see that $\sqrt{\alpha^2 + \mu_j^2} \le \alpha + \mu_j$. It then follows that

$$\rho(M(\alpha)) \le \max_{\lambda_j \in \mathrm{sp}(W)} \frac{\sqrt{\alpha^2 + \lambda_j^2}}{\alpha + \lambda_j} = \sigma(\alpha).$$

Obviously, $\sigma(\alpha) < 1$ holds true for any $\alpha > 0$, therefore the MHSS iteration converges to the unique solution of the complex symmetric linear system (1–2). □

Theorem 2.1 shows that the convergence rate of the MHSS iteration method is bounded by $\sigma(\alpha)$, which depends on the eigenvalues of the symmetric positive definite matrix W.

If the extreme eigenvalues of the matrix W are known, then the value of α which minimizes the upper bound $\sigma(\alpha)$ can be obtained. This fact is precisely stated as the following corollary.

Corollary 2.1 *Let the conditions of Theorem 2.1 be satisfied. Let $\gamma_{\min}$ and $\gamma_{\max}$ be the extreme eigenvalues of the symmetric positive definite matrix $W \in \mathbb{R}^{n\times n}$, respectively. Then*

$$\alpha_\star \equiv \arg\min_{\alpha} \left\{ \max_{\gamma_{\min} \le \lambda \le \gamma_{\max}} \frac{\sqrt{\alpha^2 + \lambda^2}}{\alpha + \lambda} \right\} = \sqrt{\gamma_{\min}\gamma_{\max}}$$

and

$$\sigma(\alpha_\star) = \frac{\sqrt{\gamma_{\min} + \gamma_{\max}}}{\sqrt{\gamma_{\max}} + \sqrt{\gamma_{\min}}} = \frac{\sqrt{\kappa(W) + 1}}{\sqrt{\kappa(W)} + 1}, \tag{8}$$

where $\kappa(W) = \gamma_{\max}/\gamma_{\min}$ is the spectral condition number of the matrix W.

Proof One can easily show that

$$\sigma(\alpha) = \max\left\{\frac{\sqrt{\alpha^2+\gamma_{\min}^2}}{\alpha+\gamma_{\min}}, \quad \frac{\sqrt{\alpha^2+\gamma_{\max}^2}}{\alpha+\gamma_{\max}}\right\}. \tag{9}$$

A simple monotonicity argument shows that the minimizer $\alpha_\star$ of the function $\sigma(\alpha)$ is the unique positive root of the algebraic equation in one unknown

$$\frac{\sqrt{\alpha^2+\gamma_{\min}^2}}{\alpha+\gamma_{\min}} = \frac{\sqrt{\alpha^2+\gamma_{\max}^2}}{\alpha+\gamma_{\max}}.$$

After solving this equation we find

$$\alpha_\star = \sqrt{\gamma_{\min}\gamma_{\max}}. \tag{10}$$

By substituting this $\alpha_\star$ into the expression of $\sigma(\alpha)$ in (9), we easily obtain the formula (8). □

An immediate implication of this theorem is that all the eigenvalues of the MHSS-preconditioned matrix lie in the interior of the disk of radius 1 centered at the point $(1, 0)$. In particular, the preconditioned matrix is positive stable. If the spectral radius is small, the eigenvalues are clustered around $(1, 0)$. As is well known, these are desirable properties when a Krylov subspace method like GMRES is used to accelerate the basic iteration.

We emphasize that the iteration parameter $\alpha_\star$ in Corollary 2.1 only minimizes the upper bound $\sigma(\alpha)$ on the spectral radius $\rho(M(\alpha))$ of the MHSS iteration matrix $M(\alpha)$, but not $\rho(M(\alpha))$ itself. Usually, computing the optimal iteration parameter which minimizes $\rho(M(\alpha))$ is not an easy task.

We further observe that when the matrix W is just the $n \times n$ identity matrix, the upper bound on the spectral radius of the iteration matrix attains its smallest possible value, which is equal to $\frac{\sqrt{2}}{2} \approx 0.7071$. In this case, convergence will be quite fast. Hence, if a good preconditioner $\widehat{W} = \widehat{L}\widehat{L}^T$ is available for W, it may be useful to precondition the original system $Ax = b$ symmetrically with $\widehat{W}$. That is, the original system should be replaced by $\widehat{A}\widehat{x} = \widehat{b}$, where $\widehat{A} = \widehat{L}^{-1}A\widehat{L}^{-T}$, $\widehat{x} = \widehat{L}^T x$, and $\widehat{b} = \widehat{L}^{-1}b$.

Finally, we remark that the MHSS iteration scheme and the foregoing convergence result are equally applicable to the case where matrix W is symmetric positive semidefinite and matrix T is symmetric positive definite. More generally, if there exist real numbers β and δ such that both matrices $\widetilde{W} := \beta W + \delta T$ and $\widetilde{T} := \beta T - \delta W$ are symmetric positive semidefinite with at least one of them positive definite, we can first scale the complex symmetric linear system (1–2) by the complex number $\beta - \mathrm{i}\,\delta$ to obtain the equivalent system

$$(\widetilde{W} + \mathrm{i}\,\widetilde{T})x = \widetilde{b}, \quad \text{with} \quad \widetilde{b} := (\beta - \mathrm{i}\,\delta)b,$$

and then apply the MHSS iteration scheme to compute an approximate solution.

3 Implementation aspects

In the MHSS method, the two half-steps comprising each iteration require the 'exact' solution of two symmetric positive definite systems with matrices $\alpha I + W$ and $\alpha I + T$. However, this may be very costly and impractical in actual implementations, particularly when the original problem arises from the discretization of a three-dimensional partial differential equation. To improve the computing efficiency of the MHSS iteration method we can employ an inner iterative method, such as the conjugate gradient (CG) method, to solve the two linear systems to some prescribed accuracies at each step of the MHSS iteration. This results in the *inexact modified Hermitian and skew-Hermitian splitting* (IMHSS) iteration method, or shortly, the IMHSS iteration method. Its convergence can be established in an analogous fashion to that of the *inexact HSS* (IHSS) iteration method, by making use of Theorem 3.1 in [3].

The tolerances (or the numbers of inner iteration steps) for the inner iterative methods may be different and in general they vary according to the accuracy attained in the outer iteration scheme. Therefore, the IMHSS iteration is actually a non-stationary iterative method for solving the system of linear equations (1–2). This means that IMHSS cannot be used as a preconditioner for GMRES, and flexible GMRES [12] should be used instead.

It is worth mentioning that when the tolerances of the inner iterations tend to zero with increasing of the outer iterate index, the asymptotic convergence rate of the IMHSS iteration approaches that of the MHSS iteration. Moreover, if good preconditioners for the matrices $\alpha I + W$ and $\alpha I + T$ are available, we can use the preconditioned conjugate gradient (PCG) method instead of CG at each iteration step so that the computational efficiency of the IMHSS iteration may be considerably improved. For example, if either W or T (or both) have Toeplitz or block Toeplitz structure, fast algorithms are available for the solution of the corresponding sub-systems. For more details, we refer to [3,4,7].

4 Numerical experiments

In this section, we use three different types of test problems to assess the feasibility and effectiveness of the MHSS iteration method when it is used either as a solver or as a preconditioner for solving the system of linear equations (1–2) with complex symmetric coefficient matrix. We also compare MHSS with HSS both as iterative solvers and as preconditioners for the (full) GMRES method and its restarted variant, GMRES(m). We experiment with inexact solves and compare the two methods from the point of view of the number of inner iterations (denoted as $\mathrm{IT}_{\mathrm{int}}$) and in terms of the total CPU time (denoted as CPU).[1]

In our implementations, the initial guess is chosen to be $x^{(0)} = 0$ and the iteration is terminated once the current iterate $x^{(k)}$ satisfies

$$\frac{\|b - Ax^{(k)}\|_2}{\|b\|_2} \leq 10^{-6}.$$

[1] $\mathrm{IT}_{\mathrm{int}}$ denotes the average number of inner iteration steps per outer iteration.

For the inexact HSS and the inexact MHSS iteration methods, the stopping criteria for the inner CG and the inner GMRES iterations are set to be

$$\frac{\|r^{(k,\ell_k)}\|_2}{\|b - Ax^{(k)}\|_2} \leq 10^{-2}, \tag{11}$$

where $r^{(k,\ell_k)}$ represents the residual of the ℓ_kth inner iterate in the kth outer iterate. In addition, all codes were run in MATLAB [version 7.4.0.336 (R2007a)] in double precision and all experiments were performed on a personal computer with 2.66 GHz central processing unit [Intel(R) Core(TM) Duo E6750], 1.97G memory and Linux operating system (2.6.23.9-85.fc8).

4.1 Example descriptions

In this section, we describe the numerical examples used to assess the performance of the MHSS and the HSS iteration methods.

Example 4.1 (See [2]) The system of linear equations (1–2) is of the form

$$\left[\left(K + \frac{3-\sqrt{3}}{\tau} I\right) + \mathrm{i}\left(K + \frac{3+\sqrt{3}}{\tau} I\right)\right] x = b, \tag{12}$$

where τ is the time step-size and K is the five-point centered difference matrix approximating the negative Laplacian operator $L = -\Delta$ with homogeneous Dirichlet boundary conditions, on a uniform mesh in the unit square $[0, 1] \times [0, 1]$ with the mesh-size $h = \frac{1}{m+1}$. The matrix $K \in \mathbb{R}^{n\times n}$ possesses the tensor-product form $K = I \otimes V_m + V_m \otimes I$, with $V_m = h^{-2}\mathrm{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$. Hence, K is an $n \times n$ block-tridiagonal matrix, with $n = m^2$. We take

$$W = K + \frac{3-\sqrt{3}}{\tau} I \quad \text{and} \quad T = K + \frac{3+\sqrt{3}}{\tau} I,$$

and the right-hand side vector b with its jth entry b_j being given by

$$b_j = \frac{(1-\mathrm{i})\, j}{\tau (j+1)^2}, \quad j = 1, 2, \ldots, n.$$

This complex symmetric system of linear equations arises in centered difference discretizations of the R_{22}-Padé approximations in the time integration of parabolic partial differential equations [2]. The R_{22}-Padé approximants satisfy the operator equation

$$\left(I + \frac{1}{2}\tau L + \frac{1}{12}\tau^2 L^2\right) u = -\frac{\mathrm{i}\tau^2}{24}\phi,$$

which can be reformulated in factorized form as

$$\left[\left(L+\frac{3+\sqrt{3}}{\tau}I\right)+\mathrm{i}\left(L+\frac{3-\sqrt{3}}{\tau}I\right)\right]\times\left[\left(L+\frac{3-\sqrt{3}}{\tau}I\right)+\mathrm{i}\left(L+\frac{3+\sqrt{3}}{\tau}I\right)\right]u=\phi.$$

Hence, the above fourth-order partial differential equation can be reduced to the two second-order equations

$$\left[\left(L+\frac{3+\sqrt{3}}{\tau}I\right)+\mathrm{i}\left(L+\frac{3-\sqrt{3}}{\tau}I\right)\right]v=\phi$$

and

$$\left[\left(L+\frac{3-\sqrt{3}}{\tau}I\right)+\mathrm{i}\left(L+\frac{3+\sqrt{3}}{\tau}I\right)\right]u=v. \tag{13}$$

We see that the complex symmetric linear system (12) is exactly the discretized form of the linear operator equation (13). In our tests, we take $\tau = h$. Furthermore, we normalize coefficient matrix and right-hand side by multiplying both by h^2. For more details, we refer to [2].

Example 4.2 The system of linear equations (1–2) is of the form

$$[(-\omega^2 M + K) + \mathrm{i}\,(\omega C_V + C_H)]x = b, \tag{14}$$

where M and K are the inertia and the stiffness matrices, C_V and C_H are the viscous and the hysteretic damping matrices, respectively, and ω is the driving circular frequency. We take $C_H = \mu K$ with μ a damping coefficient, $M = I$, $C_V = 10I$, and K the five-point centered difference matrix approximating the negative Laplacian operator with homogeneous Dirichlet boundary conditions, on a uniform mesh in the unit square $[0, 1] \times [0, 1]$ with the mesh-size $h = \frac{1}{m+1}$. The matrix $K \in \mathbb{R}^{n\times n}$ possesses the tensor-product form $K = I \otimes V_m + V_m \otimes I$, with $V_m = h^{-2}\mathrm{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$. Hence, K is an $n \times n$ block-tridiagonal matrix, with $n = m^2$. In addition, we set $\omega = \pi$, $\mu = 0.02$, and the right-hand side vector b to be $b = (1 + \mathrm{i})A\mathbf{1}$, with $\mathbf{1}$ being the vector of all entries equal to 1. As before, we normalize the system by multiplying both sides through by h^2.

This complex symmetric system of linear equations arises in direct frequency domain analysis of an n-degree-of-freedom (n-DOF) linear system. In fact, the equations of motion of an n-DOF linear system can be written in matrix form as

$$M\ddot{q} + \left(C_V + \frac{1}{\omega}C_H\right)\dot{q} + Kq = p,$$

where M, K, C_V, C_H and ω are the same as above, q is the configuration vector and p is the vector of generalized components of dynamic forces. Complex harmonic excitation at the driving circular frequency ω, i.e., of the type $p(t) = fe^{\mathrm{i}\omega t}$, admits the steady-state solution $q(t) = \tilde{q}(\omega)e^{\mathrm{i}\omega t}$, where $\tilde{q}$ solves the linear system $E(\omega)\tilde{q}(\omega) = f$ and $E(\omega)$ is the dynamic impedance matrix. Substituting $q(t) = \tilde{q}(\omega)e^{\mathrm{i}\omega t}$ and its derivatives into the equation of motion, the matrix $E(\omega)$ can be directly evaluated as

$$E(\omega) = -\omega^2 M + \mathrm{i}\omega\left(C_V + \frac{1}{\omega}C_H\right) + K.$$

This leads to the complex symmetric linear system (14). For more details, we refer to [8,5].

Example 4.3 The system of linear equations (1–2) is of the form $(W + \mathrm{i}\,T)x = b$, with

$$T = I \otimes V + V \otimes I \quad \text{and} \quad W = 10(I \otimes V_c + V_c \otimes I) + 9(e_1 e_m^T + e_m e_1^T) \otimes I,$$

where $V = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$, $V_c = V - e_1 e_m^T - e_m e_1^T \in \mathbb{R}^{m\times m}$, and e_1 and e_m are the first and the last unit vectors in $\mathbb{R}^m$, respectively. We take the right-hand side vector b to be $b = (1 + \mathrm{i})A\mathbf{1}$, with $\mathbf{1}$ being the vector of all entries equal to 1.

Here T and W correspond to the five-point centered difference matrices approximating the negative Laplacian operator with homogeneous Dirichlet boundary conditions and periodic boundary conditions, respectively, on a uniform mesh in the unit square $[0, 1] \times [0, 1]$ with the mesh-size $h = \frac{1}{m+1}$. Although this problem is an artificially constructed one, it is quite challenging for iterative solvers and therefore we include it in our tests.

4.2 Experimental results

For the tests reported in this section we used the optimal values of the parameter α (denoted by α_{exp}) for both the MHSS and the HSS iteration methods. The experimentally found optimal parameters α_{exp} are the ones resulting in the least numbers of iterations for the two methods for each of the numerical examples and for each choice of the spatial mesh-sizes.

From Table 1, we see that for each example α_{exp} decreases with the mesh-size h. For Example 4.1, the optimal α decreases approximately by a factor of $\sqrt{2}$ each time m is doubled. For the other two problems, α_{exp} is roughly halved each time m is doubled. Note that this behavior is qualitatively in agreement with the expression (10).

We further note that α_{exp} for MHSS is always larger than that for HSS for Example 4.1, while for Examples 4.2 and 4.3, α_{exp} for MHSS is always smaller than that for HSS; see Fig. 1 for a graph illustrating the convergence behavior of the two solvers with respect to α. In the case of Example 4.2 (with $m = 64$), the graphs are rather flat near the minimum, which shows that the rate of convergence is not overly sensitive

Table 1 The experimentally optimal parameters $\alpha_{\exp}$ for HSS and MHSS iteration methods

Example	Method	Grid				
		16×16	32×32	64×64	128×128	256×256
No. 4.1	HSS	0.81	0.55	0.37	0.28	0.20
	MHSS	1.06	0.75	0.54	0.40	0.30
No. 4.2	HSS	0.42	0.23	0.12	0.07	0.04
	MHSS	0.21	0.08	0.04	0.02	0.01
No. 4.3	HSS	4.41	2.71	1.61	0.93	0.53
	MHSS	1.61	1.01	0.53	0.26	0.13

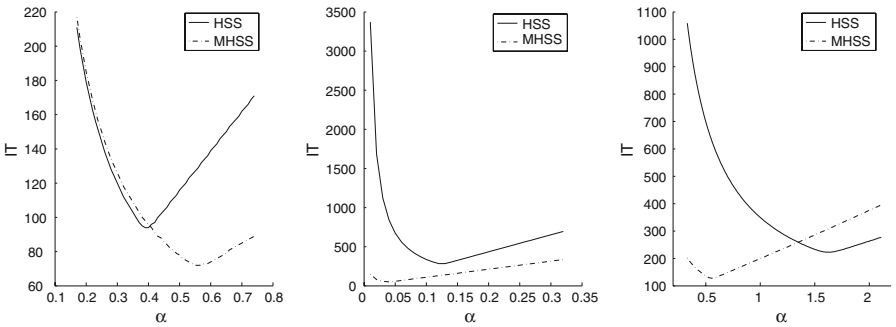

Fig. 1 Pictures of IT versus α for HSS and MHSS iteration methods; *left* Example 4.1, *middle* Example 4.2, and *right* Example 4.3

Table 2 IT and CPU for HSS, MHSS, GMRES(m) and GMRES methods for Example 4.1

Method	$m \times m$	16×16	32×32	64×64	128×128	256×256
HSS	IT	44	65	97	136	191
	CPU	0.06	0.18	1.78	16.88	188.76
MHSS	IT	40	54	73	98	133
	CPU	0.03	0.14	0.99	8.24	58.53
GMRES	IT	97	169	283	421	607
	CPU	0.07	0.69	9.53	83.25	837.84
GMRES(10)	IT	217	503	805	1199	2034
	CPU	0.03	0.15	0.85	7.35	80.13
GMRES(20)	IT	151	347	911	1465	2367
	CPU	0.03	0.15	1.62	15.51	146.19

to the choice of α. This sensitivity is somewhat more pronounced for the other two examples.

Numerical results for Example 4.1 are listed in Tables 2 and 3.

Table 3 IT and CPU for HSS- and MHSS-preconditioned GMRES(m) and GMRES methods for Example 4.1

Method	Prec		16×16	32×32	64×64	128×128	256×256
GMRES	HSS	IT	26	38	52	67	86
		CPU	0.02	0.15	1.35	12.00	126.77
	MHSS	IT	14	17	20	24	29
		CPU	0.01	0.06	0.34	2.51	15.83
GMRES(10)	HSS	IT	29	43	58	72	102
		CPU	0.02	0.15	1.31	11.46	132.68
	MHSS	IT	14	17	21	26	28
		CPU	0.01	0.06	0.37	2.68	15.00
GMRES(20)	HSS	IT	27	40	56	71	90
		CPU	0.02	0.14	1.27	11.21	120.75
	MHSS	IT	14	17	20	25	29
		CPU	0.01	0.06	0.34	2.61	15.48

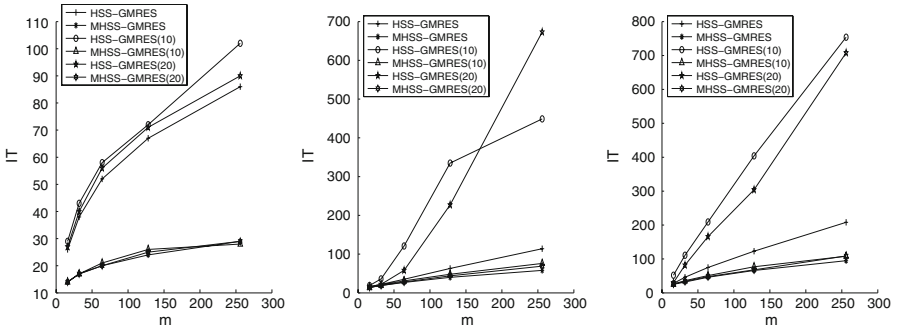

Fig. 2 Pictures of IT versus m for HSS- and MHSS-preconditioned GMRES(10), GMRES(20) and GMRES methods; *left* Example 4.1, *middle* Example 4.2, and *right* Example 4.3

In Table 2 we show IT and CPU for HSS, MHSS, GMRES(10), GMRES(20) and GMRES methods, while in Table 3 we show results for HSS- and MHSS-preconditioned GMRES(10), GMRES(20) and GMRES methods, respectively. Not surprisingly, we see from Table 2 that for all methods, the number of iterations grows with problem size. However, this growth is slower for MHSS than for all other methods, and the results show that in almost all cases MHSS is superior to the other methods in terms of both iteration count and CPU time.

In Table 3 we report results for full GMRES, GMRES(10) and GMRES(20) preconditioned with HSS and MHSS. We use again the values of α_{exp} given in Table 1, since we found that they are very close, for the test problems considered in this paper, to the values of α which minimize the number of preconditioned iterations. From these results we observe that when used as a preconditioner, MHSS performs much better than HSS in both iteration steps and CPU times, especially when the mesh-size h becomes small; see Fig. 2. Hence, for this example, also as a preconditioner MHSS

Table 4 IT and CPU for HSS, MHSS, GMRES(m) and GMRES methods for Example 4.2

Method	$m \times m$	16×16	32×32	64×64	128×128	256×256
HSS	IT	86	153	284	540	1084
	CPU	0.05	0.40	4.81	59.75	860.29
MHSS	IT	34	38	50	81	139
	CPU	0.02	0.10	0.69	6.83	60.86
GMRES	IT	42	83	161	308	593
	CPU	0.01	0.14	2.98	45.44	778.17
GMRES(10)	IT	128	346	973	3096	9979
	CPU	0.02	0.11	1.08	19.68	427.16
GMRES(20)	IT	93	233	632	1704	5202
	CPU	0.03	0.11	1.50	21.54	388.81

Table 5 IT and CPU for HSS- and MHSS-preconditioned GMRES(m) and GMRES methods for Example 4.2

Method	Prec		16×16	32×32	64×64	128×128	256×256
GMRES	HSS	IT	16	22	35	63	114
		CPU	0.02	0.09	0.93	11.34	160.23
	MHSS	IT	14	19	27	40	58
		CPU	0.01	0.07	0.46	4.30	33.65
GMRES(10)	HSS	IT	19	36	121	335	449
		CPU	0.02	0.13	2.47	42.45	428.02
	MHSS	IT	14	20	31	48	76
		CPU	0.01	0.07	0.52	4.75	38.99
GMRES(20)	HSS	IT	16	22	58	227	673
		CPU	0.02	0.09	1.30	29.57	606.33
	MHSS	IT	14	19	28	44	69
		CPU	0.01	0.06	0.47	4.42	35.36

shows much higher quality than HSS. While the number of iterations is still growing with problem size, the growth is slow and the number of iterations on the finer grid is quite acceptable with MHSS preconditioning.

Numerical results for Example 4.2 are listed in Tables 4 and 5.

In Table 4 we show IT and CPU for HSS, MHSS, GMRES(10), GMRES(20) and GMRES methods, and in Table 5 we show results for HSS- and MHSS-preconditioned GMRES(10), GMRES(20) and GMRES methods, respectively. From Table 4 we see that almost without exceptions, the stationary MHSS iteration vastly outperforms both the stationary HSS iteration and the nonstationary full and restarted GMRES iterations.

From Table 5 we observe that when used as a preconditioner for GMRES(10), GMRES(20) and GMRES, MHSS results in fast convergence and performs much better than HSS in both iteration steps and CPU times, especially when the mesh-size h

Table 6 IT and CPU for HSS, MHSS, GMRES(m) and GMRES methods for Example 4.3

Method	$m \times m$	16×16	32×32	64×64	128×128	256×256
HSS	IT	84	137	223	390	746
	CPU	0.25	0.63	4.71	49.37	674.05
MHSS	IT	53	76	130	246	468
	CPU	0.06	0.24	2.27	24.24	239.53
GMRES	IT	56	113	221	385	721
	CPU	0.02	0.35	6.54	81.04	1174.11
GMRES(10)	IT	213	546	1468	4228	14520
	CPU	0.03	0.17	1.65	26.62	607.30
GMRES(20)	IT	151	335	778	2406	7935
	CPU	0.03	0.15	1.49	25.98	492.40

Table 7 IT and CPU for HSS- and MHSS-preconditioned GMRES(m) and GMRES methods for Example 4.3

Method	Prec		16×16	32×32	64×64	128×128	256×256
GMRES	HSS	IT	28	46	75	123	208
		CPU	0.04	0.20	2.34	25.39	311.13
	MHSS	IT	25	32	46	66	95
		CPU	0.07	0.13	1.08	9.16	71.19
GMRES(10)	HSS	IT	52	111	209	404	754
		CPU	0.04	0.40	5.02	57.93	753.48
	MHSS	IT	26	36	51	77	108
		CPU	0.02	0.14	1.08	9.01	65.06
GMRES(20)	HSS	IT	30	82	166	304	708
		CPU	0.03	0.31	4.05	43.92	696.58
	MHSS	IT	26	34	48	68	109
		CPU	0.02	0.13	1.01	7.95	65.35

becomes small; see Fig. 2. Hence, also for this example MHSS is much better than HSS as a preconditioner.

Numerical results for Example 4.3 are listed in Tables 6 and 7.

In Table 6 we show IT and CPU for the HSS, MHSS, GMRES(10), GMRES(20) and GMRES methods, and in Table 7 we show results for HSS- and MHSS-preconditioned GMRES(10), GMRES(20) and GMRES methods, respectively. From Table 6 we see again that when used as stationary iterative solvers, MHSS requires less iteration steps and computing times than HSS for achieving the prescribed stopping criterion, especially when the mesh-size h is small. In iteration steps, MHSS performs much better than GMRES(10), GMRES(20) and GMRES. In terms of computing times, MHSS costs less CPU than GMRES except for $m = 16$, but more CPU than GMRES(10) and GMRES(20) except for $m = 128$ and 256. Hence, for this example, as iterative

Table 8 Average number of inner iterations for IHSS and IMHSS methods for Examples 4.1–4.3

Example	Method		16×16	32×32	64×64	128×128	256×256
No. 4.1	IHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	4.8	6.2	8.5	10.7	10.7
		$\mathrm{IT}_{\mathrm{int}}$(GMRES)	4.1	5.0	6.2	7.8	7.3
	IMHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	5.3	6.3	7.3	9.1	8.3
		$\mathrm{IT}_{\mathrm{int}}$(CG)	5.0	5.9	7.1	8.9	8.1
No. 4.2	IHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	5.1	6.2	5.8	–	–
		$\mathrm{IT}_{\mathrm{int}}$(GMRES)	2.2	3.5	5.0	–	–
	IMHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	10.5	13.0	15.6	16.2	20.6
		$\mathrm{IT}_{\mathrm{int}}$(CG)	2.0	3.9	5.0	7.0	9.9
No. 4.3	IHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	6.8	8.5	8.8	10.6	13.5
		$\mathrm{IT}_{\mathrm{int}}$(GMRES)	2.9	3.2	4.0	5.0	6.0
	IMHSS	$\mathrm{IT}_{\mathrm{int}}$(CG)	12.2	14.7	15.4	17.6	22.8
		$\mathrm{IT}_{\mathrm{int}}$(CG)	5.8	6.6	8.0	10.4	14.1

solvers MHSS outperforms GMRES and behaves much better than HSS, and is much faster than GMRES(10) and GMRES(20) for large enough problem sizes.

From Table 7 we observe that as preconditioners for GMRES(10), GMRES(20) and GMRES methods, MHSS performs much better than HSS in both iteration steps and CPU times, especially when the mesh-size h becomes small; see also Fig. 2.

We conclude this section with numerical results for the inexact variants IHSS and IMHSS on all three Examples. In Table 8 we report the average numbers of CG and GMRES inner iterations per outer iteration of IHSS, where CG is used to solve the linear system with matrix $\alpha I + W$ and GMRES for the system with matrix $\alpha I + \mathrm{i}\, T$. For IMHSS we report the average number of inner CG iterations for each of the two sub-systems with matrices $\alpha I + W$ and $\alpha I + T$. The stopping criterion for the inner iterations is (11). No preconditioning is used for these inner iterations. We can see that the average number of inner iterations per outer iteration is quite small for both IHSS and IMHSS, and it grows slowly with problem size. This growth can probably be eliminated using a suitable preconditioner, or (for the model problems considered here) using a suitable multigrid V-cycle.

The fact that the two sub-systems arising at each HSS/MHSS iteration are solved inexactly has a different impact on the number of outer iterations for different test problems. We found that for Example 4.1, the number of outer iterations is essentially the same for the exact and inexact versions of these two methods. On the other hand, for Examples 4.2 and 4.3 this is true only for the smaller values of m. For the finer meshes we found that solving the sub-systems inexactly causes a significant growth in the number of outer iterations, even causing IHSS to converge extremely slowly on the two larger instances of Example 4.2. This phenomenon can be alleviated by using a tighter tolerance in the inner stopping criterion (11). Of course, this causes an increase in the number of inner iterations. It also seems likely that using a (flexible) Krylov subspace method like FGMRES [12] to accelerate convergence could be beneficial. We leave a detailed investigations of these issues for future work.

5 Concluding remarks

Based on the Hermitian and skew-Hermitian splitting of the coefficient matrix, we have established and analyzed a class of alternating splitting iteration methods, including the modified HSS methods and the corresponding inexact variants, for solving an important class of complex symmetric linear systems. Numerical experiments have shown that these methods may yield satisfactory results when applied to linear systems of practical interest.

The basic ideas of designing the MHSS iteration method and its inexact variants can be straightforwardly extended to more general non-Hermitian linear systems. More specifically, we consider the system of linear equations

$$\widehat{A}\widehat{x} = \widehat{b}, \quad \text{with} \quad \widehat{A} \in \mathbb{C}^{n\times n} \quad \text{nonsingular and} \quad \widehat{x}, \widehat{b} \in \mathbb{C}^n.$$

Let

$$\widehat{H} = \frac{1}{2}(\widehat{A} + \widehat{A}^*) \quad \text{and} \quad \widehat{Z} = \frac{1}{2\mathrm{i}}(\widehat{A} - \widehat{A}^*).$$

Then it holds that

$$\widehat{A} = \widehat{H} + \mathrm{i}\widehat{Z},$$

with both $\widehat{H}$ and $\widehat{Z}$ being Hermitian matrices. Now, the correspondingly induced MHSS iteration method can be described as follows:

$$\begin{cases} (\alpha I + \widehat{H})\widehat{x}^{(k+\frac{1}{2})} = (\alpha I - \mathrm{i}\widehat{Z})\widehat{x}^{(k)} + \widehat{b}, \\ (\alpha I + \widehat{Z})\widehat{x}^{(k+1)} \ = (\alpha I + \mathrm{i}\widehat{H})\widehat{x}^{(k+\frac{1}{2})} - \mathrm{i}\widehat{b}, \end{cases}$$

where α is a prescribed positive constant. Note that the coefficient matrices of both linear sub-systems of this iteration scheme are Hermitian, and they are positive definite when both matrices $\widehat{H}$ and $\widehat{Z}$ are positive semidefinite. Therefore, we can demonstrate the convergence property of the above-described MHSS iteration method in an analogous manner to Theorem 2.1 and those in [3], with only slight and technical modifications.

Acknowledgments Part of this work was done when Dr. Fang Chen was visiting the State Key Laboratory of Scientific/Engineering Computing, Chinese Academy of Sciences, during September 2008–August 2009. She is very much indebted to Prof. Zhong-Zhi Bai for his kind invitation to visit.

References

1. Arridge SR (1999) Optical tomography in medical imaging. Inverse Probl 15:R41–R93
2. Axelsson O, Kucherov A (2000) Real valued iterative methods for solving complex symmetric linear systems. Numer Linear Algebra Appl 7:197–218
3. Bai Z-Z, Golub GH, Ng MK (2003) Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. SIAM J Matrix Anal Appl 24:603–626

4. Bai Z-Z, Golub GH, Ng MK (2008) On inexact Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. Linear Algebra Appl 428:413–440
5. Benzi M, Bertaccini D (2008) Block preconditioning of real-valued iterative algorithms for complex linear systems. IMA J Numer Anal 28:598–618
6. Bertaccini D (2004) Efficient solvers for sequences of complex symmetric linear systems. Electr Trans Numer Anal 18:49–64
7. Chan RH, Ng MK (1996) Conjugate gradient methods for Toeplitz systems. SIAM Rev 38:427–482
8. Feriani A, Perotti F, Simoncini V (2000) Iterative system solvers for the frequency analysis of linear mechanical systems. Comput Methods Appl Mech Eng 190:1719–1739
9. Frommer A, Lippert T, Medeke B, Schilling K (eds) (2000) Numerical challenges in lattice quantum chromodynamics. Lecture notes in computational science and engineering, vol 15. Springer, Heidelberg
10. Golub GH, Van Loan CF (1996) Matrix computations, 3rd edn. The Johns Hopkins University Press, Baltimore
11. Poirier B (2000) Efficient preconditioning scheme for block partitioned matrices with structured sparsity. Numer Linear Algebra Appl 7:715–726
12. Saad Y (1993) A flexible inner–outer preconditioned GMRES algorithm. SIAM J Sci Comput 14:461–469
13. Saad Y, Schultz MH (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 7:856–869
14. van der Vorst HA (2003) Iterative Krylov methods for large linear systems. Cambridge University Press, Cambridge
15. van Dijk W, Toyama FM (2007) Accurate numerical solutions of the time-dependent Schrödinger equation. Phys Rev E 75:036707-1–036707-10