Two efficient inexact algorithms for a class of large sparse complex linear systems

Vahid Edalatpour[†], Davod Hezari[‡] and Davod Khojasteh Salkuyeh[§*]

[†] *Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran*
E-mail: vedalat.math@gmail.com

[‡] *Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran*
E-mail: davodhezari@phd.guilan.ac.ir, d.hezari@gmail.com

[§] *Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran*
E-mail:khojasteh@guilan.ac.ir, salkuyeh@gmail.com

Abstract. Recently Salkuyeh et al. in (Int. J. Comput. Math. 92 (2015) 802-815) studied the generalized SOR (GSOR) iterative method for a class of complex symmetric linear system of equations. In this paper, we present an inexact variant of the GSOR (IGSOR) method in which the conjugate gradient (CG) and the preconditioned conjugate gradient (PCG) methods are regarded as its inner iteration processes at each step of the GSOR outer iteration. Moreover, we construct a new method called shifted GSOR iteration method (SGSOR) which is obtained from combination of a shift-splitting iteration scheme and the GSOR iteration method. The convergence analysis of the proposed methods are presented. Some numerical experiments are given to show the performance of the methods and are compared with those of the inexact MHSS (IMHSS) method.

Keywords: Complex symmetric systems, real equivalent form, inexact algorithm, shift splitting, GSOR method.
AMS Subject Classification: 65F10, 93C10.

1 Introduction

In the present paper, we consider nonsingular systems of linear equations of the form

$$Az = b, \quad \text{with} \quad A = W + iT, \quad z = x + iy \quad \text{and} \quad b = f + ig, \tag{1}$$

where $i = \sqrt{-1}$, $W, T \in \mathbb{R}^{n\times n}$ are large sparse symmetric matrices with at least one of them, e.g., W, being positive definite, and $x, y, f, g \in \mathbb{R}^n$. This class of systems of linear equations arise frequently from science and engineering applications such as computational electrodynamic [19, 25], diffuse optical tomography [1], FFT-based solution of certain time-dependent PDEs [14], quantum mechanics [15], molecular scattering [21], structural dynamics [16], and lattice quantum chromodynamics [17]. For more examples and applications see [12] and references therein.

In general, there are two approaches for solving complex systems, each of which has its own advantages and disadvantages. The first is to directly solve the $n \times n$ system (1) in complex

*Corresponding author.

arithmetic and the second is working with one of the several $(2n) \times (2n)$ equivalent real formulations. Some various equivalent formulations of a complex linear system can be found in [12].

To directly solve the complex system (1), we can use the Hermitian/skew-Hermitian splitting (HSS) iteration and its inexact variant, the inexact Hermitian/skew-Hermitian splitting (IHSS) iteration, presented by Bai et al. [7]. Recently, Bai et al. designed a modification of the HSS iteration method (MHSS) and a preconditioned variant of the MHSS iteration method (PMHSS) for solving the complex linear system (1), respectively, in [11] and [4] and investigated their performance. Moreover, lopsided version of the PMHSS method has been presented by Li et al. in [20].

It is possible to avoid complex arithmetic by rewriting the complex system (1) in the real-valued form. This can be done in several ways (see [12] or [2] for more details), however, in this paper we focus on the following equivalent real form

$$\mathcal{C}u \equiv \begin{bmatrix} W & -T \\ T & W \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \equiv d. \tag{2}$$

The above system is a special case of linear systems with 2×2 block structure. Such a structure also occurs in saddle point problems which appear in many scientific and engineering applications and a lot of research works have been done to obtain the solution of these types of systems and several methods have been proposed in this context, for example, Uzawa method [9], GSOR method [8], AHSS method [6] and so on. See [13] for a survey.

Bai et al. in [5] have accommodated the foregoing PMHSS method to solve the two-by-two block linear system (2) using real arithmetic. For more practical backgrounds of this class of problems, see [2, 12, 14]. Recently, Salkuyeh et al. in [24], applied the generalized successive overrelaxation (GSOR) iterative method to the equivalent real system (2). The GSOR iteration method can be described as follows.

The GSOR iteration method

Given an initial guess $(x_0; y_0) \in \mathbb{R}^{2n}$ *and positive constant* β, *for* $k = 0, 1, 2, \ldots$, *until* $\{(x_k; y_k)\}$ *converges, compute*

$$\begin{cases} Wx_{k+1} = (1-\beta)Wx_k + \beta T y_k + \beta f, \\ Wy_{k+1} = -\beta T x_{k+1} + (1-\beta)Wy_k + \beta g. \end{cases} \tag{3}$$

The authors showed that if $W, T \in \mathbb{R}^{n\times n}$ are symmetric positive definite and symmetric, respectively, then the GSOR method for solving the Eq. (2) is convergent if and only if

$$0 < \beta < \frac{2}{1+\rho(S)},$$

where $S = W^{-1}T$ and $\rho(S)$ is the spectral radius of S. Moreover, they introduced the optimal iteration parameter and the corresponding optimal convergence factor of the method. To accelerate the convergence rate of the GSOR method, they then proposed the preconditioned GSOR (PGSOR) iteration method [18], for the linear system (2) and established the conditions under which the PGSOR iteration method is more effective than the GSOR iteration method.

In this paper to improve the computing efficiency of the GSOR iteration method we employ a Krylov subspace iteration scheme such as the conjugate gradient (CG) method or preconditioned

conjugate gradient (PCG) method (see [22]), to solve inexactly the two linear subsystems at each step of the GSOR iteration. This results in the inexact GSOR (IGSOR) iteration method. We also present the shifted GSOR (SGSOR) iteration method which is based on the shift-splitting iteration scheme (see [10]) and the GSOR method for solving the linear system (2), which is suitable and efficient when the coefficient matrix of linear sub-systems arising at each inner iteration of the GSOR method are ill-conditioned. Moreover, similar to IGSOR, we establish the inexact variant of SGSOR (ISGSOR). To speed up the convergence of the GSOR method one may precondition the system (2) by the preconditioner proposed in [18].

Throughout this paper, we use $\rho(\cdot)$ and $\mathcal{I}_{\mathcal{C}}$ to denote the spectral radius of the corresponding matrix and the convergence interval of the GSOR method corresponding to the linear system (2), respectively.

The rest of the paper is organized as follows. In Section 2, an inexact variant of the GSOR iteration method is described and the convergence analysis of this iteration method is established. In Section 3, shifted GSOR method is introduced, the convergence analysis and inexact variant of the method are also considered. An implementation aspect is briefly discussed in Section 4. Numerical experiments are given in Section 5 to illustrate the effectiveness of our iteration method. Finally, in Section 6, we end the paper with a brief conclusion.

2 The Inexact GSOR Method

At each step of the GSOR iteration two linear systems with coefficient matrix W need to be solved, which is very costly and impractical in actual implementations. To overcome this disadvantage, since W is a symmetric positive definite matrix, we can inexactly solve the two subproblems iteratively by suitable iteration methods. This results in the following inexact GSOR iteration for solving the system of linear equations (2).

The IGSOR iteration method

Given an initial guess $(x_0; y_0) \in \mathbb{R}^{2n}$ *and positive constant* β, *for* $k = 0, 1, 2, \ldots$, *until* $\{(x_k; y_k)\}$ *converges, compute* $(x_{k+1}; y_{k+1})$ *from*

$$\begin{cases} W\bar{x}_{k+1} \approx (1-\beta)W\bar{x}_k + \beta T\bar{y}_k + \beta f, \\ W\bar{y}_{k+1} \approx -\beta T\bar{x}_{k+1} + (1-\beta)W\bar{y}_k + \beta g, \end{cases} \tag{4}$$

by employing an inner solver (e.g., the CG method).

To simplify the convergence analysis of the above method, we may rewrite the IGSOR iteration method as the following equivalent scheme.

Given an initial guess $(x_0; y_0)$ *and positive constant* β, *for* $k = 0, 1, 2, \ldots$, *until* $\{(x_k; y_k)\}$ *converges,*

1. *Compute the approximate solution* $\bar{s}_k$ *for* $Ws = \hat{r}_k$ $(\hat{r}_k = \beta(f - W\bar{x}_k + T\bar{y}_k))$ *iteratively, such that*

$$\|\bar{p}_k\| \leq \epsilon_k \|\hat{r}_k\|,$$

where $\bar{p}_k = \hat{r}_k - W\bar{s}_k$. *Set* $\bar{x}_{k+1} = \bar{x}_k + \bar{s}_k$;

2. *Compute the approximate solution* $\bar{t}_k$ *for* $Wt = \tilde{r}_k$ *(*$\tilde{r}_k = \beta(g - T\bar{x}_{k+1} - W\bar{y}_k)$*) iteratively, such that*

$$\|\bar{q}_k\| \leq \epsilon_k \|\tilde{r}_k\|,$$

where $\bar{q}_k = \tilde{r}_k - W\bar{t}_k$*. Set* $\bar{y}_{k+1} = \bar{y}_k + \bar{t}_k$*.*

The IGSOR iteration can be equivalently written as follows. Let $\beta\mathcal{C} = \mathcal{M}_\beta - \mathcal{N}_\beta$ be the splitting of the coefficient matrix $\beta\mathcal{C}$ with

$$\mathcal{M}_\beta = \begin{bmatrix} W & 0 \\ \beta T & W \end{bmatrix} \quad \text{and} \quad \mathcal{N}_\beta = \begin{bmatrix} (1-\beta)W & \beta T \\ 0 & (1-\beta)W \end{bmatrix}. \tag{5}$$

Then, the IGSOR method computes the approximate solution $\bar{u}_{k+1} = \bar{u}_k + \bar{z}_k$ to the system at $(k+1)$th iteration such that $\bar{z}_k = (\bar{s}_k; \bar{t}_k)$ is the solution of the system

$$\mathcal{M}_\beta z = \bar{r}_k, \tag{6}$$

in which $\bar{r}_k = (\hat{r}_k; \tilde{r}_k) = \beta(b - \mathcal{C}\bar{u}_k)$, $\bar{u}_k = (\bar{x}_k; \bar{y}_k)$ and $b = (f; g)$. In addition, the stopping criterion is

$$\|\bar{v}_k\| \leq \epsilon_k \|\bar{r}_k\|$$

where $\bar{v}_k = \bar{r}_k - \mathcal{M}_\beta \bar{z}_k$.

Theorem 1. *[23] For every* $B \in \mathbb{C}^{n\times n}$ *and* $\epsilon > 0$*, there exists a norm* $\|.\|$ *on* $\mathbb{C}^n$ *such that for the corresponding induced norm,* $\|B\| \leq \rho(B) + \epsilon$*.*

The next theorem analyzes the convergence of the IGSOR iteration method.

Theorem 2. *Suppose that* $\mathcal{C}$ *is defined by (2), such that* W *and* T *are symmetric matrices and* W *being positive definite. Let also* $\beta\mathcal{C} = \mathcal{M}_\beta - \mathcal{N}_\beta$ *where* $\beta \in \mathcal{I}_\mathcal{C}$ *and* $\mathcal{M}_\beta$ *and* $\mathcal{N}_\beta$ *are defined by (5). There exists a norm* $\|.\|$ *on* $\mathbb{C}^{2n}$ *such that for the corresponding induced norm,* $\|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\| < 1$*. Also, if* $\{\bar{u}_k\}$ *is a sequence defined as*

$$\bar{u}_{k+1} = \bar{u}_k + \bar{z}_k, \quad \textit{with} \quad \mathcal{M}_\beta \bar{z}_k = \bar{v}_k + \bar{r}_k, \tag{7}$$

satisfying

$$\|\bar{v}_k\| \leq \epsilon_k \|\bar{r}_k\|, \tag{8}$$

where $\bar{r}_k = \beta(b - \mathcal{C}\bar{u}_k)$*, and* u^* *is the exact solution of the system (2), then we have*

$$\|\bar{u}_{k+1} - u^*\| \leq (\|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\| + \varepsilon\beta\|\mathcal{M}_\beta^{-1}\|\|\mathcal{C}\|)\|\bar{u}_k - u^*\|, \tag{9}$$

where $\varepsilon = \max_k \{\epsilon_k\}$*. In particular, if*

$$\varepsilon < \frac{1 - \|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\|}{\beta\|\mathcal{M}_\beta^{-1}\|\|\mathcal{C}\|}$$

then the sequence $\{\bar{u}_k\}$ *converges to* u^**.*

Proof. Since $\mathcal{C} = \frac{1}{\beta}\mathcal{M}_\beta - \frac{1}{\beta}\mathcal{N}_\beta$ is the GSOR splitting of $\mathcal{C}$ and $\beta \in \mathcal{I}_\mathcal{C}$, we have $\rho(\mathcal{M}_\beta^{-1}\mathcal{N}_\beta) < 1$, and from Theorem 1, for $\gamma = (1 - \rho(\mathcal{M}_\beta^{-1}\mathcal{N}_\beta))/2$, there exists a norm $\|.\|$ on $\mathbb{C}^{2n}$ such that for the corresponding induced norm we get

$$\|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\| \leq \rho(\mathcal{M}_\beta^{-1}\mathcal{N}_\beta) + \gamma < 1.$$

From (7), we have

$$\begin{aligned}\bar{u}_{k+1} &= \bar{u}_k + \mathcal{M}_\beta^{-1}(\bar{v}_k + \bar{r}_k)\\ &= (I - \beta\mathcal{M}_\beta^{-1}\mathcal{C})\bar{u}_k + \beta\mathcal{M}_\beta^{-1}b + \mathcal{M}_\beta^{-1}\bar{v}_k\\ &= \mathcal{M}_\beta^{-1}\mathcal{N}_\beta\bar{u}_k + \beta\mathcal{M}_\beta^{-1}b + \mathcal{M}_\beta^{-1}\bar{v}_k. \end{aligned} \tag{10}$$

Since u^* is the exact solution of the system (2), it follows that

$$u^* = \mathcal{M}_\beta^{-1}\mathcal{N}_\beta u^* + \beta\mathcal{M}_\beta^{-1}b. \tag{11}$$

Subtracting Eq. (11) from Eq. (10), yields

$$\bar{u}_{k+1} - u^* = \mathcal{M}_\beta^{-1}\mathcal{N}_\beta(\bar{u}_k - u^*) + \mathcal{M}_\beta^{-1}\bar{v}_k. \tag{12}$$

The latter equation results in

$$\|\bar{u}_{k+1} - u^*\| \leq \|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\|\|\bar{u}_k - u^*\| + \|\mathcal{M}_\beta^{-1}\|\|\bar{v}_k\|. \tag{13}$$

On the other hand, from Eq. (8) we see that

$$\|\bar{v}_k\| \leq \epsilon_k\|\bar{r}_k\| = \epsilon_k\beta\|b - \mathcal{C}u_k\| = \epsilon_k\beta\|\mathcal{C}(u^* - \bar{u}_k)\| \leq \epsilon_k\beta\|\mathcal{C}\|\|u^* - \bar{u}_k\|. \tag{14}$$

Therefore, from Eqs. (13) and (14) we obtain

$$\begin{aligned}\|\bar{u}_{k+1} - u^*\| &\leq \|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\|\|\bar{u}_k - u^*\| + \|\mathcal{M}_\beta^{-1}\|(\epsilon_k\beta\|\mathcal{C}\|\|u^* - \bar{u}_k\|)\\ &= (\|\mathcal{M}_\beta^{-1}\mathcal{N}_\beta\| + \epsilon_k\beta\|\mathcal{M}_\beta^{-1}\|\|\mathcal{C}\|)\|\bar{u}_k - u^*\|, \end{aligned} \tag{15}$$

which completes the proof. □

3 The shifted GSOR Method

In the GSOR (or IGSOR) method two sub-systems with the coefficient matrix W, which is symmetric positive definite, should be solved exactly (or inexactly). When the coefficient matrix W is ill-conditioned, as $\mathcal{C}$ defined in (2) is a non-Hermitian positive definite matrix, we apply the following shift-splitting iteration method (see [10])

$$(\mathcal{C} + \alpha I)u_{k+1} = \alpha u_k + d, \quad k = 1, 2, \ldots, \tag{16}$$

where α is a real nonnegative number and at each step of iteration, the shifted linear system itself is iteratively solved either by the GSOR or IGSOR iteration method. Therefore, the method is actually an inner/outer iterative method with a standard splitting iteration as its outer iteration, and the GSOR (or IGSOR) iteration as its inner iteration. Then, depending on the use of the GSOR or IGSOR method as inner solver, the above method is called shifted

GSOR (SGSOR) or inexact shifted GSOR (ISGSOR), respectively. Clearly, when $\alpha = 0$, the SGSOR (resp. ISGSOR) iteration method reduces to GSOR (resp. IGSOR) iteration method. The shifted GSOR iteration method can be described as follows.

The shifted GSOR iteration method

Given an initial guess $u_0 = (x_0; y_0)$, and a sequence $\{l_k\}_{k=0}^{\infty}$ of positive integers, compute $u_{k+1} = (x_{k+1}; y_{k+1})$ for $k = 0, 1, 2, \ldots$, until $\{u_k\}$ converges, using the following iteration scheme:

(a) *Set $u_k^0 := u_k$;*

(b) *For $l = 0, 1, \ldots, l_k - 1$, solve the following linear systems to obtain u_{k+1}:*

$$\begin{cases} (W + \alpha I)x_k^{l+1} = (1-\beta)(W+\alpha I)x_k^l + \beta T y_k^l + \beta(\alpha x_k + f), \\ (W + \alpha I)y_k^{l+1} = -\beta T x_k^{l+1} + (1-\beta)(W+\alpha I)y_k^l + \beta(\alpha y_k + g), \end{cases}$$

where α and β are two given real positive constant.

(c) *Set $u_{k+1} = (x_k^{l_k}; y_k^{l_k})$.*

When $\alpha = 0$, the shifted GSOR iteration method reduces to the GSOR iteration method which is convergent for the parameters in the convergence region.

In the matrix-vector form the $(k+1)$th iterate of the shifted GSOR iteration method can be written as

$$u_{k+1} = \mathcal{T}_\beta u_k^{l_k - 1} + p_k. \quad k = 0, 1, 2, \ldots \tag{17}$$

where $\mathcal{T}_\beta = \mathcal{F}_\beta^{-1}\mathcal{G}_\beta$ and $p_k = \beta\mathcal{F}_\beta^{-1}(\alpha u_k + b)$ with

$$\mathcal{F}_\beta = \begin{bmatrix} W + \alpha I & 0 \\ \beta T & W + \alpha I \end{bmatrix} \quad \text{and} \quad \mathcal{G}_\beta = \begin{bmatrix} (1-\beta)(W+\alpha I) & \beta T \\ 0 & (1-\beta)(W+\alpha I) \end{bmatrix}.$$

Since for any $k = 0, 1, 2, \ldots$,

$$u_k^{l+1} = \mathcal{T}_\beta u_k^l + p_k, \quad l = 0, 1, \ldots, l_k - 1,$$

the relation (17) is equivalent to

$$u_{k+1} = \mathcal{T}_\beta^{l_k} u_k + \sum_{i=0}^{l_k - 1} \mathcal{T}_\beta^i p_k. \quad k = 0, 1, 2, \ldots. \tag{18}$$

The next theorem provides sufficient conditions for convergence of the shifted GSOR iteration method.

Theorem 3. *Suppose that $\mathcal{C}$ is defined by (2), such that W and T are symmetric matrices with W being positive definite. Let $\alpha > 0$ and $\beta \in \mathcal{I}_{\mathcal{C}+\alpha I}$. If u^* is the exact solution of the system of linear equations (2), then for any initial guess u_0 and any sequence of positive integers l_k,*

$k = 0, 1, 2, \ldots$, the sequence $\{u_k\}$ produced by the shifted GSOR iteration method converges to u^* provided that $l = \liminf_{k\to\infty} l_k \geq N$, where N is a natural number satisfying

$$\|\mathcal{T}_\beta^s\| < \frac{1-\eta}{1+\eta} \quad \forall s \geq N.$$

for some $0 < \eta < 1$.

Proof. Since u^* is the solution of the system (2), from (18) we deduce that

$$u^* = \mathcal{T}_\beta^{l_k} u^* + \beta \sum_{i=0}^{l_k-1} \mathcal{T}_\beta^i \mathcal{F}_\beta^{-1}(\alpha u^* + b). \tag{19}$$

Subtracting Eq. (19) from Eq. (18), yields

$$u_{k+1} - u^* = \mathcal{T}_\beta^{l_k}(u_k - u^*) + \alpha\beta \sum_{i=0}^{l_k-1} \mathcal{T}_\beta^i \mathcal{F}_\beta^{-1}(u_k - u^*). \tag{20}$$

Since $\mathcal{T}_\beta$ is the iteration matrix of the GSOR method for the system (16) and $\beta \in \mathcal{I}_{\mathcal{C}+\alpha I}$, we get $\rho(\mathcal{T}_\beta) < 1$. Thanks to the relation

$$\begin{bmatrix} W+\alpha I & -T \\ T & W+\alpha I \end{bmatrix} = \alpha I + \mathcal{C} = \frac{1}{\beta}(\mathcal{F}_\beta - \mathcal{G}_\beta),$$

we obtain

$$\begin{aligned} \sum_{i=0}^{l_k-1} \mathcal{T}_\beta^i \mathcal{F}_\beta^{-1} &= (I - \mathcal{T}_\beta^{l_k})(I - \mathcal{T}_\beta)^{-1}\mathcal{F}_\beta^{-1} \\ &= (I - \mathcal{T}_\beta^{l_k})(I - \mathcal{F}_\beta^{-1}\mathcal{G}_\beta)^{-1}\mathcal{F}_\beta^{-1} \\ &= (I - \mathcal{T}_\beta^{l_k})(\mathcal{F}_\beta - \mathcal{G}_\beta)^{-1} \\ &= \frac{1}{\beta}(I - \mathcal{T}_\beta^{l_k})(\mathcal{C} + \alpha I)^{-1}. \end{aligned}$$

Substituting this in Eq. (20) results in

$$u_{k+1} - u^* = \mathcal{T}_\beta^{l_k}(u_k - u^*) + \alpha(I - \mathcal{T}_\beta^{l_k})(\mathcal{C} + \alpha I)^{-1}(u_k - u^*).$$

Letting $\mathcal{H}_\alpha = \alpha(\mathcal{C} + \alpha I)^{-1}$, we get

$$u_{k+1} - u^* = (\mathcal{H}_\alpha + (I - \mathcal{H}_\alpha)\mathcal{T}_\beta^{l_k})(u_k - u^*), \tag{21}$$

On the other hand we have

$$\rho(\mathcal{H}_\alpha) = \max_{\lambda\in\sigma(\mathcal{C})} \frac{\alpha}{|\,\lambda + \alpha\,|} = \frac{\alpha}{|\,\hat{\lambda} + \alpha\,|},$$

for some $\hat{\lambda} \in \sigma(\mathcal{C})$. Since $\mathcal{C}$ is positive definite, $\Re(\hat{\lambda}) > 0$ and for $\alpha \geq 0$, $\rho(\mathcal{H}_\alpha) < 1$. Hence, from Theorem 1 we deduce that there exists a norm on $\mathbb{C}^{2n}$ such that for the corresponding induced norm

$$\|\mathcal{H}_\alpha\| \leq \rho(\mathcal{H}_\alpha) + \epsilon = \eta < 1, \tag{22}$$

where $\epsilon = (1 - \rho(\mathcal{H}_\alpha))/2$. Now, from Eqs. (21) and (22) we obtain

$$\|u_{k+1} - u^*\| \leq (\eta + (1+\eta)\|\mathcal{T}_\beta^{l_k}\|)\|u_k - u^*\|.$$

Since $\rho(\mathcal{T}_\alpha) < 1$, $\mathcal{T}_\alpha^s \to 0$ as s tends to infinity. Therefore, there is a natural number N such that

$$\|\mathcal{T}_\beta^s\| < \frac{1-\eta}{1+\eta} \quad \forall s \geq N.$$

Hence, if we assume $l = \liminf_{k\to\infty} l_k \geq N$, then the desired result is obtained. □

In the sequel, we consider the shifted GSOR method when the inner loop of the shifted GSOR algorithm involves only one iteration. In this case, the shifted GSOR method is rewritten as follows.

The shifted GSOR iteration method (special case)

Given an initial guess $u_0 = (x_0; y_0)$ and positive real constants α and β. for $k = 0, 1, 2, \ldots$, until $u_{k+1} = (x_{k+1}; y_{k+1})$ converges, compute

$$\begin{cases} (W + \alpha I)x_{k+1} = (1-\beta)(W+\alpha I)x_k + \beta T y_k + \beta(\alpha x_k + f), \\ (W + \alpha I)y_{k+1} = -\beta T x_{k+1} + (1-\beta)(W+\alpha I)y_k + \beta(\alpha y_k + g), \end{cases}$$

We have the following theorem concerning the convergence of the above iteration scheme:

Corollary 1. *Suppose that $\mathcal{C}$ is defined by (2), such that W and T are symmetric matrices and W being positive definite. Let $\alpha > 0$ be sufficiently small and $\beta \in \mathcal{I}_{\mathcal{C}+\alpha I}$. If u^* is the exact solution of the system of linear equations (2), then for any initial guess u_0, the sequence of $\{u_k\}$ in the shifted GSOR iteration method (special case), converges to u^*.*

Proof. The proof is similar to that of Theorem 3, with this difference that here $l_k = 1$, for $k \geq 0$. In this case Eq. (20) is simplified to

$$u_{k+1} - u^* = (\mathcal{H}_\alpha + (I - \mathcal{H}_\alpha)\mathcal{T}_\beta)(u_k - u^*),$$

or equivalently

$$u_{k+1} - u^* = (\mathcal{T}_\beta + (I - \mathcal{T}_\beta)\mathcal{H}_\alpha)(u_k - u^*). \tag{23}$$

Since $\beta \in \mathcal{I}_{\mathcal{C}+\alpha I}$, we have $\rho(\mathcal{T}_\beta) < 1$ and from Theorem 1 for $\epsilon = \frac{1}{2}(1 - \rho(\mathcal{T}_\beta))$ there exists a norm on $\mathbb{C}^{2n}$ such that for the corresponding induced norm

$$\|\mathcal{T}_\alpha\| \leq \rho(\mathcal{T}_\alpha) + \epsilon = \eta < 1. \tag{24}$$

Therefore, from Eqs. (23) and (24) we have

$$\|u_{k+1} - u^*\| \leq (\eta + (1+\eta)\|\mathcal{H}_\alpha\|)\|u_k - u^*\|. \tag{25}$$

Let $f(\alpha) = \|\mathcal{H}_\alpha\| = \|\alpha(\mathcal{C} + \alpha I)^{-1}\|$. Since $\mathcal{C}$ is nonsingular and $\alpha > 0$, $f(\alpha)$ is well-defined, continues and $f(0) = 0$. So, for $\gamma = (1-\eta)/(1+\eta)$ there exists a $\delta > 0$ such that if $\alpha < \delta$, then $f(\alpha) < \gamma$ and with consider of Eq. (25), the sequence $\{u_k\}$ converges to u^* □

Since the coefficient matrix of the inner sub-systems is symmetric positive definite, to improve the computational efficiency of the shifted GSOR iteration method, we can solve the inner sub-systems of the SGSOR method approximatively by the CG method or the preconditioned CG method. We call this method inexact shifted GSOR method. Its convergence analysis can be established in an analogous way to that of the IGSOR iteration method.

4 The preconditioned GSOR method

If there exists a real number ω such that both matrices $\omega W + T$ and $\omega T - W$ are symmetric positive semidefinite with at least one of them being positive definite, we first use the preconditioner

$$\mathcal{P} = \begin{bmatrix} \omega I & I \\ -I & \omega I \end{bmatrix},$$

for the linear system (2) to obtain the equivalent system

$$\tilde{\mathcal{C}} u = \tilde{b}, \tag{26}$$

where

$$\tilde{\mathcal{C}} = \mathcal{C}\mathcal{P} = \begin{bmatrix} \omega W + T & W - \omega T \\ \omega T - W & \omega W + T \end{bmatrix} \quad \text{and} \quad \tilde{b} = \begin{bmatrix} \omega p + q \\ \omega q - p \end{bmatrix},$$

and then apply the GSOR iteration method to obtain an approximate solution. In [18], it has been proved that for $\omega = 1$, if we set the iteration parameter of the method to be $\beta = 0.828$, then the convergence factor would be $\rho(G_\beta) = 0.172$, where G_β is the iteration matrix of the method with the iteration parameter β. In the section of numerical experiments we apply the IGSOR and the ISGSOR for solving the equivalent system (26) with $\omega = 1$ instead of solving the system (2), and they are called inexact preconditioned GSOR (IPGSOR) and inexact shifted preconditioned GSOR (ISPGSOR), respectively. In this case, we set $\beta = 0.828$ as the relaxation parameter of the IGSOR and the ISGSOR methods.

5 Numerical experiments

In this section, we choose some examples from [11] and present some numerical experiments to illustrate the effectiveness of our proposed methods. The numerical results are compared with the Inexact Modified Hermitian and Skew-Hermitian Splitting (IMHSS) method. We examine the test problems by the ISPGSOR method, the IPGSOR method and the IMHSS method with the optimal parameters extracted from [11].

For the IMHSS, the ISPGSOR and the IPGSOR iteration methods, the stopping criteria for the inner iterative method is set to be

$$\frac{\|r_k^{l_k}\|_2}{\|b - \mathcal{C}u_k\|_2} \leq 10^{-2},$$

where $u_k = (x_k; y_k)$ and $r_k^{l_k}$ represents the residual of the l_kth inner iterate in the kth outer iterate. All the numerical experiments presented in this section were computed in double precision and the algorithms were implemented in MATLAB 8.0.0.783 (64-bit) and tested on a 64-bit 1.73 GHz intel Q740 core i7 processor and 4GB RAM running Windows 7. We use a null vector as an initial guess and the stopping criterion

$$\frac{\|b - \mathcal{C}u_k\|_2}{\|b\|_2} < 10^{-6}.$$

Also, we used the modified incomplete Cholesky factorization with dropping tolerance 10^{-3} as preconditioner for CG. This preconditioner for given symmetric positive definite matrix W, is constructed as follows (in MATLAB notation):

```
L=ichol(W,struct('michol','on','type','ict','droptol',1e-3));
```

Finally, the reported CPU times are the sum of the CPU times for the convergence of the method and the CPU times for computing the incomplete Cholesky factorization.

Example 1. Consider the system of linear equations

$$\left[\left(K+\frac{3-\sqrt{3}}{\tau}I\right)+i\left(K+\frac{3+\sqrt{3}}{\tau}I\right)\right]x=b. \tag{27}$$

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 $V_m=h^{-2}\text{tridiag}(-1,2,-1)\in\mathbb{R}^{m\times m}$. 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

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

In our tests, we take $\tau=h$. Furthermore, we normalize the coefficient matrix and the right-hand side vector by multiplying both by h^2.

Example 2. Consider the system of linear equations $(W+iT)x=b$, with $T=I\otimes V+V\otimes I$ and $W=10(I\otimes V_c+V_c\otimes I)+9(e_1e_m^T+e_me_1^T)\otimes I$, where $V=\text{tridiag}(-1,2,-1)\in\mathbb{R}^{m\times m}$, $V_c=V-e_1e_m^T-e_me_1^T\in\mathbb{R}^{m\times m}$ and e_1 and e_m are the first and last unit vectors in $\mathbb{R}^m$, respectively. We take the right-hand side vector b to be $b=(1+i)A\mathbf{1}$, 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 $h=1/(m+1)$. Although this problem is an artificially constructed one, it is quite challenging for iterative solvers and therefore we include it in our tests.

Example 3. Consider the system of linear equations

$$\left[(-\omega^2M+K)+i(\omega C_V+C_H)\right]x=b,$$

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 $h=1/(m+1)$. The matrix $K\in\mathbb{R}^{n\times n}$ possesses the tensor-product form $K=I\otimes V_m+V_m\otimes I$, with $V_m=h^{-2}\text{tridiag}(-1,2,-1)\in\mathbb{R}^{m\times m}$. Hence, K is an $n\times n$ block-tridiagonal matrix, with $n=m^2$. In this example for $\mu=0.02$, we set $\omega=\pi$ and the right-hand side vector b to be $b=(1+i)A\mathbf{1}$, 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.

Table 1: The experimentally optimal parameters for the IMHSS iteration method for Examples 1-3.

Examples	Grid					
	32×32	64×64	128×128	256×256	512×512	1024×1024
No. 1	0.75	0.54	0.40	0.30	0.22	0.16
No. 2	0.08	0.04	0.02	0.01	0.005	0.002
No. 3	1.01	0.53	0.26	0.13	0.06	0.03

Example 4. (see [14]) Helmholtz equations are of fundamental importance in the modeling of wave propagation phenomena. In this example, we consider the finite-difference discrtization of the partial differential equation

$$-\triangle u - \sigma_1 u + i\sigma_2 u = f, \tag{28}$$

where the coefficients c, σ_1 and σ_2 are real-valued functions, u satisfies Dirichlet boundary conditions and $i = \sqrt{-1}$. We consider Eq. (28) on the 2D domain $[0,1] \times [0,1]$ with $\sigma_1 = 35$ and $\sigma_2 = 100$. We discretize the problem with finite differences on a $m \times m$ grid with mesh size $h = 1/(m+1)$. This leads to a system of linear equations

$$(K - \sigma_1 I + i\sigma_2 I)\, x = b,$$

where $K = I \otimes V_m + V_m \otimes I$ is a standard second-order finite-difference discretization of the diffusion operator $-\triangle u$, wherein $V_m = h^{-2}\text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$. The right-hand side vector b is taken to be $b = (1+i)A\mathbf{1}$, with $\mathbf{1}$ being the vector of all entries equal to 1. Furthermore, before solving the system we normalize the coefficient matrix and the right-hand side vector by multiplying both by h^2.

In Tables 2-8 we have compared the efficiency of the IMHSS, the ISPGSOR and the IPGSOR iteration methods for solving Examples 1-4, in terms of number of both iterations (IT) and CPU time (denoted by CPU). In all the tables, the average number of iterations for solving the inner systems are given in parentheses. The value $\alpha = 0.001$ for Example 1 and $\alpha = 0.0001$ for the other examples are chosen as the parameter of the ISPGSOR method. Also, the optimal parameter of the IMHSS iteration for Examples 1-3 were found experimentally and are listed in Table 1. For more details about how to get this values experimentally, see [11].

For various matrix sizes, in Tables 2-4, we show IT and CPU for the IMHSS, the ISPGSOR and the IPGSOR methods where at each iteration step of these methods the inner sub-systems are solved by the CG algorithm, while in Tables 5-7 we show the results for the above methods where at each iteration step the inner sub-systems are solved using the preconditioned conjugate gradient (PCG) method. The obtained results show that the implementation of the methods with the preconditioned CG as the inner solver, behave much better than the CG as inner solver. Although this is not the case for the IMHSS method in Example 3.

The reported results in Tables 2-7 show that the IPGSOR and the ISPGSOR methods have almost the same performance for Examples 1-3. Although the IPGSOR method behaves better

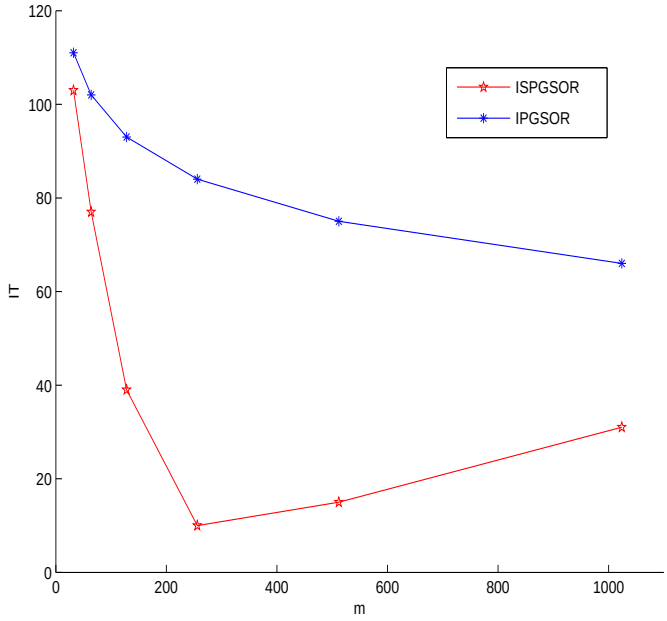

Figure 1: Graph of IT versus m for the IPGSOR and ISPGSOR methods for Example 4 using CG or PCG as the inner solver.

Table 2: IT and CPU for IMHSS, ISPGSOR when $\alpha = 0.001$ and IPGSOR, for Example 1 using CG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	57	77	103	135	184	254
	CPU	0.21	1.37	6.21	32.13	152.88	1180.80
ISPGSOR	IT	9(19.66)	9(29.44)	8(42.13)	8(56.20)	9(79.33)	10(109.21)
	CPU	0.04	0.16	0.64	3.07	18.72	83.31
IPGSOR	IT	9(19.66)	9(29.50)	8(42.56)	8(59.87)	8(84.06)	8(118.38)
	CPU	0.04	0.16	0.65	3.08	19.98	85.25

Table 3: IT and CPU for IMHSS, ISPGSOR when $\alpha = 0.0001$ and IPGSOR, for Example 2, using CG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	51	80	185	541	> 1000	> 1000
	CPU	0.27	1.85	11.83	121.38	–	–
ISPGSOR	IT	9(28)	9(48.44)	9(91.63)	9(162.23)	13(243.50)	30(269.59)
	CPU	0.05	0.24	1.31	7.34	76.45	913.33
IPGSOR	IT	9(28)	9(48.94)	9(93.11)	9(170.17)	9(301.83)	9(592.72)
	CPU	0.05	0.26	1.32	7.50	65.29	602.86

Table 4: IT and CPU for IMHSS, ISPGSOR with $\alpha = 0.0001$ and IPGSOR, for Example 3, using CG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	89	169	357	726	> 1000	> 1000
	CPU	0.45	3.65	23.70	233.19	–	–
ISPGSOR	IT	9(39.94)	9(73.83)	9(130.54)	9(246.11)	9(425.50)	9(764.88)
	CPU	0.07	0.43	2.13	12.64	91.63	788.44
IPGSOR	IT	9(39.94)	9(74.05)	9(130.77)	9(242.27)	9(461.4)	9(882.50)
	CPU	0.07	0.47	2.15	12.65	103.32	880.98

Table 5: IT and CPU for IMHSS, ISPGSOR with $\alpha = 0.001$ and IPGSOR, for Example 1, using PCG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	60	79	102	136	184	251
	CPU	0.20	1.02	5.11	33.31	169.61	19.42 *min*
ISPGSOR	IT	9(1.44)	9(1.72)	8(2.06)	9(2.56)	9(3.17)	10(3.90)
	CPU	0.02	0.06	0.22	1.00	5.12	26.15
IPGSOR	IT	9(1.44)	9(1.67)	8(1.94)	8(2.36)	8(2.69)	8(3.13)
	CPU	0.02	0.06	0.21	0.94	4.31	20.60

Table 6: IT and CPU for IMHSS, ISPGSOR with $\alpha = 0.0001$ and IPGSOR, for Example 2, using PCG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	41	53	86	157	352	–
	CPU	0.28	1.48	7.98	56.78	712.84	> 60 *min*
ISPGSOR	IT	9(2.06)	9(3.06)	9(4.06)	9(6.06)	13(7.38)	30(4.78)
	CPU	0.02	0.09	0.41	2.11	17.12	186.12
IPGSOR	IT	9(2.06)	9(3.06)	9(4.06)	9(5.83)	9(8.22)	9(11.56)
	CPU	0.02	0.09	0.41	2.08	11.33	61.32

Table 7: IT and CPU for IMHSS, ISPGSOR when $\alpha = 0.0001$ and IPGSOR, for Example 3, using PCG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
IMHSS	IT	137	289	691	> 1000	> 1000	> 1000
	CPU	0.51	3.15	33.71	–	–	–
ISPGSOR	IT	9(2.33)	9(3)	9(4.16)	9(5.83)	9(8.61)	9(11.39)
	CPU	0.02	0.09	0.41	1.95	10.81	59.82
IPGSOR	IT	9(2.33)	9(3.06)	9(4.11)	9(5.94)	9(8.72)	9(12.50)
	CPU	0.02	0.09	0.40	1.97	10.96	67.24

Table 8: IT and CPU for IMHSS, ISPGSOR with $\alpha = 0.0001$ and IPGSOR when $\sigma_1 = 35$ and $\sigma_2 = 100$, for Example 4, using CG or PCG as the inner solver.

Method		Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
CG as the inner solver							
ISPGSOR	IT	103(9.37)	77(18.90)	39(40.70)	10(117.80)	15(185.56)	31(243.54)
	CPU	0.21	0.96	2.15	3.89	56.23	6.91 *min*
IPGSOR	IT	111(9.43)	102(18.68)	93(36.87)	84(72.99)	75(145.87)	66(290.74)
	CPU	0.26	1.39	6.20	35.35	339.16	41.72 *min*
PCG as inner the solver							
ISPGSOR	IT	103(2)	77(3)	39(4.92)	10(5.25)	15(7.57)	31(9.50)
	CPU	0.16	0.64	1.19	1.66	14.80	169.95
IPGSOR	IT	111(2)	102(3)	93(4.97)	84(6.92)	75(10.75)	66(15.53)
	CPU	0.20	0.88	4.36	19.98	118.23	599.12

Table 9: Average time of every outer iteration for IMHSS, ISPGSOR with $\alpha = 0.001$ for Example 1 and $\alpha = 0.0001$ for the other examples, and IPGSOR methods for Examples 1-4 with CG as the inner solver; the IMHSS method (M1), the ISPGSOR method (M2) and the IPGSOR method (M3).

Example	Method	Grid					
		32×32	64×64	128×128	256×256	512×512	1024×1024
No. 1	M1	0.004	0.018	0.060	0.238	0.827	4.649
	M2	0.004	0.018	0.080	0.384	2.478	10.431
	M3	0.004	0.017	0.081	0.384	2.497	10.656
No. 2	M1	0.005	0.023	0.064	0.224	–	–
	M2	0.006	0.027	0.146	0.815	5.880	30.433
	M3	0.006	0.027	0.147	0.833	7.254	66.984
No. 3	M1	0.005	0.023	0.064	0.224	–	–
	M2	0.008	0.047	0.236	1.404	10.181	87.604
	M3	0.008	0.052	0.238	1.404	11.480	97.886
No. 4	M2	0.002	0.012	0.055	0.389	3.749	27.640
	M3	0.002	0.014	0.067	0.421	4.522	33.376

than the ISPGSOR method in terms of IT, in the same iterations the ISPGSOR method has better relative performance of the IPGSOR method which is expectable since the condition number of the coefficient matrix of the inner systems in each iteration of ISPGSOR method is slightly smaller than those of the IPGSOR method. Also, for all examples both the IPGSOR and the ISPGSOR methods are more efficient than the IMHSS methods in terms of iterations count and CPU time.

Table 8 compares the performance of the IPGSOR method and the ISPGSOR method with $\alpha = 0.0001$, for Example 4 when $\sigma_1 = 35$ and $\sigma_2 = 100$. We observe that the ISPGSOR method behaves much better than the IPGSOR method both in terms of IT and CPU. Also, in Fig 1 we illustrate the number of iterations for the methods versus size of grid (m) for Example 4, where we use CG or PCG as inner solvers. As seen, iteration numbers of the IPGSOR method decreases with increasing m while this is not the case for the ISPGSOR method.

In Table 9, we present the average time for outer iteration of the methods for Examples 1-4, where the two inner sub-systems are solved by the CG method. As we observe, in most cases the average time per iteration for the IMHSS method is the least and in the IPGSOR method is the greatest among the methods. This is expectable, since the condition number of the coefficient matrix of the inner sub-systems in the IMHSS method is the least and for the IPGSOR method is the greatest.

6 Conclusion

For an equivalent real formulation of complex linear system of equations of two-by-two block structures, we have presented an inexact variant of the GSOR method named IGSOR, where in each iteration of the method the sub-systems are solved inexactly by CG and preconditioned CG. Also, based on shift-splitting of the main system we have constructed a new method named SGSOR, where at each iteration of the method instead of solving the inner sub-systems of the GSOR method, the shifted sub-systems are solved. Similar to the inexact GSOR method, we have also established an inexact variant of the SGSOR method. Convergence properties of the methods have been investigated, and to expedite the convergence of the methods the primary system has been preconditioned by a suitable preconditioner. Finally, the methods have been tested numerically by four examples and compared with the inexact MHSS method. The numerical results show that the IPGSOR and the ISPGSOR methods are suitable for large and sparse matrices and are significantly superior to the IMHSS method in both iteration steps and CPU times. Moreover, when the coefficient matrix of the inner sub-systems is ill-conditioned, the ISPGSOR method with a small shift parameter α is more efficient than the IPGSOR method.

Acknowledgements

The authors would like to thank the anonymous reviewer for careful reading of the paper and giving helpful comments and suggestions. The work of Davod Khojasteh Salkuyeh is supported by University of Guilan.

References

[1] S. R. Arridge, *Optical tomography in medical imaging*, Inverse Probl. **15** (1999) 41–93.

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

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

[4] Z. -Z. Bai, M. Benzi, F. Chen, *On preconditioned MHSS iteration methods for complex symmetric linear systems*, Numer. Algor. **56** (2011) 297–317.

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

[6] 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.

[7] 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.

[8] Z. -Z. Bai, B. N. Parlett, Z. -Q. Wang, *On generalized successive overrelaxation methods for augmented linear systems*, Numer. Math. **102** (2005) 1–38.

[9] Z. -Z. Bai, Z. -Q. Wang, *On parameterized inexact Uzawa methods for generalized saddle point problems*, Numer. Linear Algebra Appl. **428** (2008) 2900-2932.

[10] 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.

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

[12] M. Benzi, D. Bertaccini, *Block preconditioning of real-valued iterative algorithms for complex linear systems*, IMA J. Numer. Anal. **28** (2008) 598–618.

[13] M. Benzi, G.H. Golub, J. Liesen, *Numerical solution of saddle point problems*, Acta Numer. **14** (2005) 1–137.

[14] D. Bertaccini, *Efficient solvers for sequences of complex symmetric linear systems*, Electr. Trans. Numer. Anal. **18** (2004) 49–64.

[15] W. V. Dijk, F. M. Toyama, *Accurate numerical solutions of the time-dependent Schrdinger equation*, Phys. Rev. E **75** (2007).

[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.

[17] A. Frommer, T. Lippert, B. Medeke, K. Schilling (eds), *Numerical challenges in lattice quantum chromodynamics*, Lecture notes in computational science and engineering, **15** (2000) 66–83.

[18] D. Hezari, V. Edalatpour, D. K. Salkuyeh, *Preconditioned GSOR iteration method for a class of complex symmetric linear system*, Numer. Linear Algebra Appl., DOI: 10.1002/nla.1987, to appear, arXiv:1403.5902.

[19] R. Hiptmair, *Finite elements in computational electromagnetism*, Acta Numer. **11** (2002) 237–339.

[20] X. Li, A. L. Yang, Y. J. Wu, *Lopsided PMHSS iteration method for a class of complex symmetric linear systems*, Numer. Algor. **66** (2014) 555–568.

[21] B. Poirier, *Efficient preconditioning scheme for block partitioned matrices with structured sparsity*, Numer. Linear Algebra Appl. **7** (2000) 715–726.

[22] Y. Saad, *Iterative methods for sparse linear systems, 2nd Edition*, SIAM, Philadelphia, 2003.

[23] D. Serre, *Matrices: Theory and Applications*, Springer, New York, 2002.

[24] D. K. Salkuyeh, D. Hezari, V. Edalatpour, *Generalized SOR iterative method for a class of complex symmetric linear system of equations*, Int. J. Comput. Math. **92** (2015) 802-815.

[25] U. Van Rienen, *Numerical methods in computational electrodynamics: linear systems in practical applications*, Springer, Berlin, 2001.