Proceedings of the Pakistan Academy of Sciences: A Physical and Computational Sciences 62(4): 301-312 (2025)

ISSN (Print): 2518-4245; ISSN (Online): 2518-4253
http://doi.org/10.53560/PPASA(62-4)697

Pakistan Academy of Sciences

Research Article

A Flexible-Scalar Splitting Iterative Method for Linear Inverse Problems with Complex Symmetric Matrix

Ruiping Wen[1], Dongqi Li[1], Zubair Ahmed[2], Jinrui Guan[1], and Owais Ali*[2, 3]

[1]School of Mathematics and Statistics, Taiyuan Normal University, Jinzhong, China
[2]Institute of Mathematics and Computer Science, University of Sindh, Jamshoro, Pakistan
[3]Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan

Abstract: This paper introduces a flexible scalar-splitting (f-SCSP) iterative scheme and examines its convergence properties. The approach also yields a straightforward matrix-splitting preconditioner for the original linear system. To confirm the theoretical results and evaluate practical performance, comprehensive numerical examinations are performed on various test cases. The findings indicate that the proposed method is practical, reliable, and more efficient than existing techniques for handling demanding classes of complex symmetric linear systems.

Keywords: Inverse Problem, Complex Symmetric Matrix, Splitting; Iterative Method, Flexible-scalar.

1. INTRODUCTION

We focus on the iterative resolution of linear systems.

$$Ax = b \tag{1}$$

where $A \in C^{n\times n}$ and $x,\ b \in C^n$. In Equation (1), $A = W + iT$ is a matrix which is non-Hermitian and symmetric $(A \neq A^*, A = A^T)$ with $W, T \in \mathbb{R}^{n\times m}$ are real and symmetric, and W and T are positive definite and positive semidefinite matrices, respectively. In this text, the imaginary quantity iota, $i^2 = -1$, is denoted by the symbol i. Let there be a splitting $A = M - N$ of the matrix $A \in \mathbb{C}^{n\times n}$ i.e., $M \in \mathbb{C}^{n\times n}$ is nonsingular and $N \in \mathbb{C}^{n\times n}$. This splitting gives rise to a fixed-point iterative method of the following form.

$$x^{k+1} = M^{-1}Nx^k + M^{-1}b, \qquad k = 0,1,2,\ldots \tag{2}$$

where $x^0 \in \mathbb{C}^n$ is a given starting vector.

Systems corresponding to Equation (1) appear frequently throughout computational science and in numerous branches of engineering, where they form a core component of many modelling and simulation tasks. A few notable examples include Diffuse Optical Tomography (DOT); very helpful for small animal imaging, breast cancer detection, and functional brain imaging [1]. Because of the nature of light propagation in scattering media and the usage of complex coefficients to simulate absorption and diffusion, the mathematical modelling and numerical computation required in DOT frequently result in complex symmetric linear systems. When time-dependent PDEs are treated with FFT-driven schemes, the resulting discretisations commonly lead to complex symmetric linear algebraic systems, particularly in frequency-domain formulations or in spectral and pseudo-spectral frameworks [2].

Advanced scientific applications in structural dynamics, especially those involving damping, frequency-domain analysis, or non-proportional damping models, the governing equations lead to complex symmetric linear systems [3]. Lattice Quantum Chromo Dynamics (Lattice-QCD)

Received: July 2025; Revised: November 2025; Accepted: December 2025
* Corresponding Author: Owais Ali <owais@g.ecc.u-tokyo.ac.jp>

[4] is a computational approach for examining QCD. Complex symmetric linear systems emerge naturally in various formulations of Lattice-QCD, particularly in fermion discretization such as staggered fermions or twisted mass fermions [5]. Numerical computations in molecular scattering is a crucial subject in quantum chemistry, chemical physics, and dynamics. The foundational theory relies on quantum scattering theory, resulting in extensive linear algebraic systems that are frequently complex and occasionally symmetrical under certain conditions [6].

Recently, Ahmed *et al.* [7] and Kanwal *et al.* [8] suggested that if the forward operator A is symmetric, iterative over-relaxation can solve (1) efficiently. Axelsson and Kucherov [9] presented an iterative method for real matrices, Benzi and Bertaccini [10] proposed a block preconditioning for real-valued iterative algorithms, Bai [11] and Bai *et al.* [12, 13] introduced a modified Hermitian and skew-Hermitian splitting (MHSS) as well as preconditioned-MHSS (PMHSS) iterative methods and Wang *et al.* [14] improved the PMHSS method. Various preconditioning techniques have been developed to enhance the convergence rate of these iterative methods. For instance, Salkuyeh *et al.* [15], Hezari *et al.* [16], Axelsson and Salkuyeh [17], Xie and Li [18], Xiang and Zhang [19], and Salkuyeh [20], Zhao *et al*. [21] put forward a Single-Step-MHSS method (SMHSS) and its variants with a flexible-shift (f-SMHSS). Wen *et al*. [22, 23] also suggested some iterative methods and respective preconditioning techniques. Vorst and Melissen [24], Freund [25], while, Bunse-Gerstner and Stöver [26] presented the conjugate gradient-type methods; Clements *et al*. [27] introduced Krylov-type methods. In particular, Hezari *et al*. [28] proposed the Scale-Spliting (SCSP) method employing a scaling approach. Later Salkuyeh [29] suggested a two-step SCSP method, while Salkuyeh and Siahkolaei [30] introduced a two-parameter SCSP (TSCSP). Zheng *et al.* [31] also introduced a double-step scale splitting iterative method. Li *et al*. [32, 33] put forward a dual-parameter double-step splitting iteration method, and two iterative methods with quasi-combining real and imaginary parts. However, the scaled parameters mentioned above are given in advance. Motivated by the optimization models given by Zhao *et al*. [21], this study introduced a flexible-scalar strategy based on the SCSP iterative method, which the scaled parameters α are determined by minimizing the residuals at each iteration.

Following we present the essential notations. The set of $p \times p$ real (complex) arrays and the p -dimensional real (complex) vector space are represented as $\mathbb{R}^{p\times p}$ and $\mathbb{R}^{p}$ ($\mathbb{C}^{p\times p}$ and $\mathbb{C}^{p}$) respectively. The conjugate and transpose of a matrix or a vector x is x^{*} and x^{T} repectively. A matrix $A \in \mathbb{C}^{p\times p}$ ($A \in \mathbb{R}^{p\times p}$) is said to be Hermitian (symmetric) positive definite (or semidefinite), denoted by $A > 0$ (or ≥ 0); if it is Hermitian (or symmetric) and for all $x \in \mathbb{C}^{n},\ x \neq \bar{0},\ x^{*}Ax > 0$ ($x^{*}Ax \geq 0$) holds true. The real and imaginary parts of a complex number x are denoted by $\mathfrak{Re}(x)$ and $\mathfrak{Im}(x)$, respectively. $\rho(A)$ is used to represent the spectral radius of a matrix A and $\Sigma(A)$ represents the spectrum set of the matrix. The condition number of a matrix A is denoted by $\kappa(A)$. The splitting of A , defined as $A = M - N,$ is said to be convergent if $\rho(M^{-1}N) < 1$.

A broad range of preconditioning strategies has been introduced in past to accelerate the convergence behavior of such iterative schemes. For instance, a double-step scale splitting iterative method employing a scaling approach given by Salkuyeh and Siahkolaei [30]. By multiplying two parameters $(\alpha - i)$ and $(1 - i\alpha)$ both sides of the Equation (1), two equivalent systems can be respectively yielded, i.e., $(\alpha - i)Ax = (\alpha - i)b$ and $(1 - i\alpha)Ax = (1 - i\alpha)b$, where α is a real positive number. Then two fixed-point equations can be generated as follows:

$$\big((\alpha W + T) + i(\alpha T - W)\big)x = (\alpha - i)b\,, \qquad (3)$$

$$\big((\alpha W + T) + i(\alpha T - W)\big)x = (1 - i\alpha)b. \qquad (4)$$

Zheng *et al.* [31] expanded on the PMHSS iterative method, suggested by Bai *et al*. [13], and proposed the following alternative iterative scheme:

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

whereas the Equations (3) and (4) are in fact two preconditioned systems in Equation (2) when $P = (\alpha - 1)I$ and $P = (1 - i\alpha)I$, that is to say, the preconditioned matrices are both the scalar matrices. Equations (3) and (4) are one when $\alpha = 1$, therefore, the alternation of the DSS iterative method

was only carried out in twins of two preconditioned systems. This work focuses on linear systems whose coefficient matrices are complex symmetric yet not Hermitian. We focus on the scaled preconditioned splitting iterative methods generally and consider the systems in Equation (2) when $P = (\alpha - \beta i)I$ with α, β are both real numbers in this study.

2. MATERIALS AND METHODS

To provide context and completeness, this section begins with a brief overview of existing methods for solving linear systems whose coefficient matrices are complex symmetric but non-Hermitian, as in Equation (1). We then introduce the Flexible-Scalar Splitting (f-SCSP) scheme.

2.1. The Relevant Methods

2.1.1. MHSS method [12, 13]:

The MHSS iteration method: Let $x^{(0)} \in \mathbb{C}^n$ be an initial guess. For $k = 0, 1, 2, \cdots$, until $\{x^{(k)}\}$ converges, compute $x^{(k+1)}$ according to the following sequence:

$$\begin{cases} (\alpha I + W)x^{k+\frac{1}{2}} = (\alpha I - iT)x^k + b, \\ (\alpha I + T)x^{k+1} = (\alpha I + iW)x^{k+\frac{1}{2}} - ib, \end{cases}$$

where α is a given positive constant.

2.1.2. The SMHSS and f-SMHSS methods [21]:

(1) The SMHSS iteration method: Let $x^{(0)} \in \mathbb{C}^n$ be an initial guess. For $k = 0, 1. 2. \cdots$, until $\{x^{(k)}\}$ converges, compute $x^{(k+1)}$ according to the following sequence $(\alpha I + W)x^{k+1} = (\alpha I - iT)x^k + b.$

(2) The f-SMHSS iteration method: Let $x^{(0)} \in \mathbb{C}^n$ be an initial guess, for $\varepsilon > 0$, $k = 0, 1, 2, \cdots$, until $\{x^{(k)}\}$ converges, the single-step iteration formula for computing the next $x^{(k+1)}$ is as follows.

Step 1: Compute $r_k = b - Ax_k$.

Step 2: Solve the equation
$(\alpha_{k+1} I + W)x^{k+1} = (\alpha_{k+1} I - iT)x^k + b,$
where the flexible shift α_{k+1} is the solution to the following optimization problem: $\min_{\alpha} \|(\alpha I + W)^{-1} r_{k+1}\|_2^2$, with $r_{k+1} = b - Ax_{k+1}$.

Step 3: If $\|r_k\|_2 \leq \epsilon$, stop; otherwise, set $k = k + 1$ and return to Step 1.

2.1.3. The scale-splitting (SCSP) method [28]:

Let α be a real positive constant and the matrix $\alpha W + T$ be nonsingular. By multiplying the complex number $(\alpha - i)$ through both sides of Equation (1), the following equivalent system can be obtained.

$$A_\alpha x = (\alpha - i)b \tag{5}$$

Where $A_\alpha = (\alpha W + T) + i(\alpha T - W)$. By rewriting it as the system of fixed-point equations:
$(\alpha W + T)x = i(W - \alpha T)x + (\alpha - i)b,$
the SCSP iteration method can be summarized as follows.

The SCSP iteration method: Let $x^{(0)} \in \mathbb{C}^n$ be an initial guess. For $k = 0, 1, 2, \cdots$, until $\{x^{(k)}\}$ converges, compute $x^{(k+1)}$ according to the following sequence:

$$(\alpha W + T)x^{k+1} = i(W - \alpha T)x^k + (\alpha - i)b, \tag{6}$$

where α is a given positive constant.

2.2. Proposed Iterative Method: The Flexible-Scalar Splitting (f-SCSP)

The variant system can be obtained by multiplying the complex number $\alpha - i$,
$[(\alpha W + T) - i(W - \alpha T)]x = (\alpha - i)b.$
To use the flexible-scalar strategy, the f-SCSP method is formulated as follows:

$$(\alpha_{k+1} W + T)x^{k+1} = i(W - \alpha_{k+1} T)x^k + (\alpha_{k+1} - i)b \tag{7}$$

where,

$$\alpha_{k+1} = \arg\min_{\alpha} \frac{1}{2} r_k^* (\alpha W + T)^{-1} r_k \tag{8}$$

with $r_k = b - Ax^k$, $k = 0,1,2, \ldots$.
Remark: In fact, the exact solutions of the quadratic programming models in Equation (8) can be given theoretically by simple computing. To avoid the tedious computation of $(\alpha_k W + T)^{-1}$, we can use the inexact line search to find the approximations of α.
In matrix-vector form, the scheme presented in Equation (7) can be equivalently rewritten as:

$$x^{k+1} = T_{\alpha_{k+1}} x^k + G_{\alpha_{k+1}} b, \quad k = 0,1,2, \ldots \tag{9}$$

where,

$$T_{\alpha_{k+1}} = i(\alpha_{k+1}W + T)^{-1}(W - \alpha_{k+1}T), \text{ and } G_{\alpha_{k+1}} = (\alpha - i)(\alpha W + T)^{-1} \quad (10)$$

Here, $T_{\alpha_{k+1}}$ is the iteration matrix of the f-SCSP method. In fact, Equation (9) is also generated by the splitting, $A_{\alpha_k} = M_{\alpha_k} - N_{\alpha_k}$, with

$$M_{\alpha_k} = \frac{\alpha + i}{\alpha^2 + 1}(\alpha W + T), \quad \text{and } N_{\alpha_k} = \frac{-1 + i\alpha}{\alpha^2 + 1}(W - iT)$$

Moreover,

$T_{\alpha_{k+1}} = M_{\alpha_{k+1}}^{-1} N_{\alpha_{k+1}}$, and M_{α_k} can be identified as a preconditioner to all linear systems of type Equation (1).

Consequently, the preconditioned system can be expressed as follows.

$$M_{\alpha_k}^{-1} A x = M_{\alpha_k}^{-1} b. \quad (11)$$

We now investigate the optimal parameter selection and the spectral radius characteristics of the iteration matrix, and assess the convergence behavior of the previously described f-SCSP method.

Theorem 2.1: Let be a non-Hermitian but symmetric matrix $A = W + iT \in \mathbb{C}^{n\times n}$, $(A \neq A^*,\ A = A^T)$ with both $W, T \in \mathbb{R}^{n\times n}$ being symmetric, W and T being both positive definite positive. Let α be positive real numbers and $\lambda_{\min}$ and $\lambda_{\max}$ be the extremal eigenvalues of the matrix $W^{-1}T$. Then the following statements hold true:

(i) In the f-SCSP method, the upper bound of the spectral radius $\rho(T_{\alpha_k})$ is:

$$\delta_{\alpha_k} = \max\left\{\frac{-\alpha_k\lambda_{\min}}{\alpha_k+\lambda_{\min}}, \frac{\alpha_k\lambda_{\max}}{\alpha_k+\lambda_{\max}}\right\} \quad (12)$$

(ii) The sequence $\{x^k\}$ produced by Method 2.1

$$\begin{cases} \dfrac{1-\lambda_{\min}}{1+\lambda_{\min}} < \alpha_k < \dfrac{1-\lambda_{\max}}{1+\lambda_{\max}}, & \lambda_{\max} \in (1, +\infty) \\ \alpha_k > \dfrac{1-\lambda_{\min}}{1+\lambda_{\min}}, & \Sigma(W^{-1}T) \subset [0, 1] \end{cases}$$

In particular, the iterative scheme presented in Equation (6) is convergent if α for the case that T is a positive semidefinite matrix.

Proof (i): By Equation (12) and direct calculations, we have:

$$\begin{aligned} &\rho(T_{\alpha_k}) = \rho\left(i(\alpha_k W + T)^{-1}(W - \alpha_k T)\right) \\ &\le \|i(\alpha_k W + T)^{-1}(W - \alpha_k T)\|_2 \\ &\le \|(\alpha_k W + T)^{-1}\|_2 \|W - \alpha_k T\|_2 \\ &= \|(\alpha_k I + W^{-1}T)^{-1}\|_2 \|I - \alpha_k W^{-1}T\|_2 \\ &= \max_{\lambda\in\Sigma(W^{-1}T)} \left|\frac{-\alpha_k\lambda}{\alpha_k + \lambda}\right|. \end{aligned}$$

In the last step, the equality holds since $W^{-1}T$ is a symmetric positive definite matrix, and then so is $(\alpha I + W^{-1}T)^{-1}$.

It is known that λ is positive. By introducing the following function:

$$f(\lambda) = \frac{-\alpha\lambda}{\alpha + \lambda},$$

it is obtained that $f(\lambda)$ is a decreasing function

with respect to λ since $f'(\lambda) = -\frac{1+\alpha^2}{(\alpha+\lambda)^2} < 0$.

Thus Equation (12) provides the upper bound of $\rho(T_{\alpha_k})$.

Proof (ii): For the case that $\lambda_{\max} > 1$, $\delta_{\alpha_k} < 1$ is equivalent to $\alpha > \frac{1-\lambda_{\min}}{1+\lambda_{\min}}$ by simple calculations.

And then $\rho(T_{\alpha_k}) < 1$, so the sequence $\{x^k\}$ produced by the f-SCSP method converges to the unique solution to Equation (1) for any initial guess x^*.

For the case that $\Sigma(W^{-1}T) \subset [0,\ 1]$, then $\lambda_{\max} < 1$ at that time. Thus, $\delta_{\alpha_k} < 1$ is only equivalent to $\alpha_k > \frac{1-\lambda_{\min}}{1+\lambda_{\min}}$.

It is well-known that $\lambda_{\min} = 0$ if T is a positive semidefinite matrix. And then $\rho(T_{\alpha_k}) \le \alpha^{-1}$, the iterative scheme in Equation (6) is convergent if $\alpha > 1$. The proof is completed.

Corollary 2.1: Assuming the conditions of Theorem 2.1 hold, the optimal the parameters α that minimises the upper bound δ_{α_k} of the spectral radius $\rho(T_{\alpha_k})$ is given by:

$$\alpha = \frac{1-\lambda_{\min}\lambda_{\max}+\sqrt{(1+\lambda_{\min}^2)(1+\lambda_{\max}^2)}}{\lambda_{\min}+\lambda_{\max}} \quad (13)$$

A similar proof is presented in [28, theorem 1], which is omitted here.

Theorem 2.2: Let be a non-Hermitian, symmetric matrix $A = W + iT \in \mathbb{C}^{n\times n}$, with $W, T \in \mathbb{R}^{n\times n}$ being both symmetric, also, W being positive-definite and T positive definite or semidefinite. Then $\rho(T_{\alpha_k}) < 1$ if for all $x \in \mathbb{C}^n$, it holds that $\alpha > \frac{x^*Wx - x^*Tx}{x^*Wx + x^*Tx}$.

Proof: Let an eigenvalue of the matrix T_{α_k}

be λ with the corresponding eigenvector x, i.e., $M_{\alpha_k}^{-1}N_{\alpha_k}x = \lambda x$, which means, $\lambda(\alpha W + T)x = i(W - \alpha T)x$. Then we have from the assumptions that:

$$|\lambda| = \left|\frac{x^*Wx - \alpha x^*Tx}{\alpha x^*Wx + x^*Tx}\right|$$

We obtain $\alpha > \frac{x^*Wx - x^*Tx}{x^*Wx + x^*Tx}$ by direct calculations under $|\lambda| < 1$. The theorem is proved.

Remark: Theorem 2.2 implies that all eigenvalues of the matrix T_{α_k} lie along the imaginary axis.

The last of this section, a property of the matrix $M_{\alpha_k}^{-1}A$ can be given.

Theorem 2.3: Let $A = W + iT \in \mathbb{C}^{n\times n}$ be a non-Hermitian but symmetric matrix ($A \neq A^*$, $A = A^T$) with $W, T \in \mathbb{R}^{n\times n}$ be real, symmetric, and W being positive-definite and T positive definite or semidefinite. Assuming that μ is any eigenvalue of the matrix $M_{\alpha_k}^{-1}A$ defined by Theorem (2.2), the $\Re e(\mu) = 1$.

Proof: Let λ be an eigenvalue of the matrix $M_{\alpha_k}^{-1}A$ and x be the corresponding eigenvector of the eigenvalue λ with $\|x\|_2 = 1$. It is known that:

$$(\alpha_k - i)(W + iT)x = \lambda(\alpha_k W + T)x.$$

So, we have:

$$\lambda = \frac{\alpha_k x^*Wx + x^*Tx + i(\alpha_k x^*Tx - x^*Wx)}{\alpha_k x^*Wx + x^*Tx}.$$

From assumptions, $x^*Wx = c > 0$, $x^*Tx = d \geq 0$. Then we yield $\Re e(\mu) = 1$.

3. RESULTS AND DISCUSSION

This section presents a series of numerical experiments designed to evaluate the practicality, reliability, and computational efficiency of the proposed f-SCSP method in comparison with existing approaches. The evaluation is based on three key performance metrics: the number of iterations to convergence (IT), the total processing time taken by our computer in seconds for convergence (CPU), and the final residual norm (RES). These measures provide a comprehensive assessment of both the convergence characteristics and computational cost of each method.

The performance of f-SCSP is assessed in comparison with four well-known iterative techniques. The MHSS method [12, 13], SMHSS method [21], the f-SMHSS method [21], and the SCSP method [28], which were introduced and discussed in Section 2. In all numerical experiments, the initial guess is taken as the zero vector, and the iterations are terminated once the relative residual norm meets the predefined stopping criterion, set here as an ℓ_2-norm of the residual $\leq 10^{-6}$. The iteration process is considered unsuccessful if convergence is not achieved within a maximum of 8000 iterations. This limit guarantees an equitable assessment among all techniques and aids in avoiding excessive computation time when convergence is not reached as expected. All these experiments are done with different vector space sizes m given $A : \mathbb{C}^m \to \mathbb{C}^m$; the results provide empirical validation of the theoretical analysis and demonstrate the performance of the proposed method.

Example 3.1 [28]: The linear system of equations in (1) represents the form $(W + iT)x = b$, with

$$W = 10(I \otimes V_c + V_c \otimes I) + 9(e_1 e_m^T + e_m^T e_1) \otimes I$$

and $T = I \otimes V + V \otimes I$ where $V = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$, $V_c = V - e_1 e_m^T + e_m^T e_1 \in \mathbb{R}^{m\times m}$, $e_1 = (1, 0, \dots, 0)^T \in \mathbb{R}^m$ and $e_m = (0, 0, \dots, 1)^T \in \mathbb{R}^m$. The vector b on the right-hand side can be choosen as $b = (1 + i)A1$, where 1 is the vector with all entries equal to 1.

Example 3.2 [28]**:** The complex linear systems (1) is of the form:

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

where ω denote the driving circular frequency, with M and K representing the inertia and stiffness matrices, and C_V and C_H are denoting the viscous and hysteretic damping matrices. The viscous damping is modelled as $C_H = \mu K$ where μ is given as the damping coefficient, $M = I$, $C_V = 10I$, $K = I \otimes B_m + B_m \otimes I$, with $B_m = \frac{1}{h^2}\text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m\times m}$, and mesh size $h = \frac{1}{m+1}$. Accordingly, K takes the form of an $n \times n$ block-tridiagonal matrix with block dimension $n = m^2$. We further specify $\omega = \pi$, $\mu = 0.02$, and construct the right-hand vector $b = (1 + i)A1$, where 1 denotes the vector with all components equal to 1. To standardise the system, we pre-multiply both sides by h^2thereby obtaining a normalised formulation.

Example 3.3: Consider the two-dimensional

convection-diffusion equation:

$$-(u_{xx} + u_{yy}) + \eta(u_x + u_y) = g(x, y),$$

the region of interest is considered over the unit square domain $[0,\ 1] \times [0,\ 1]$ assuming constant coefficient η and imposing Dirichlet boundary conditions. Employing the five-point central difference discretisation leads to the linear system (1), characterised by the following coefficient matrix:

$W = T_1 \otimes I + I \otimes T_1$ and $T = I \otimes V + V \otimes I$,

where the matrices T_1 and V are given by:

$$T_1 = \text{tridiag}(-1 - R_e, 2, -1 + R_e), V = \text{tridiag}(2, -1, -1)$$

with $R_e = \mu h/2$, being the mesh Reynolds number, and $h = 1/(m + 1)$ being the equidistant step-size. Moreover, the right-hand side vector b is taken to be $b = Ax$, with $x^* = (1, 1, 1, \ldots, 1)^T \in \mathbb{R}^n$ being the true solution.

In the conducted experiments, matrices with dimensions approaching $270{,}000$ (i.e., $n = m^2 = 512 \times 512 = 262{,}144$) were examined. The numerical results are summarized in Tables 1–3. Evidently, the SCSP and f-SCSP methods perform commendably; the f-SCSP method achieves convergence in the fewest iterations, whereas the SCSP method demonstrates superior computational efficiency in most tests. The challenge of balancing iteration count and execution time to develop an enhanced method constitutes a key direction for forthcoming research.

When compared against its counterparts, SCSP, f-SMHSS, SMHSS and MHSS, the proposed f-SCSP method exhibits a compelling balance between iteration count and computational cost. Table 1 shows results from Example 3.1, and that SCSP is achieving convergence in 10-103 iterations across increasing problem sizes, closely matching the iteration efficiency of flexible f-SCSP but requiring only approximately half the CPU time (e.g., 0.0153s vs. 0.0592s for $m = n = 16$), highlighting its lower overhead in parameter selection. Although f-SCSP attains marginally fewer iterations in some cases, its per-iteration optimization of α_k sustain a significant time penalty. In contrast, classical SMHSS and MHSS methods demand up to an order of magnitude more iterations and substantially longer runtimes, often exceeding SCSP by factors of 5-10, reflecting the superior conditioning induced by the scaled preconditioning. Overall, f-SCSP converges in fewer iterations with better efficiency in all system sizes compared to MHSS, SMHSS, and f-SMHSS. The comparison between f-SCSP and SCSP is however subtle; f-SCSP converges with fewer iterations and a slightly better relative residual in larger system sizes, but the CPU time shows that SPSC is the most efficient throughout. Similarly, Table 2 shows results from Example 3.2, and again f-SPSC and SPSC are very close, with f-SPSC convergeing in fewer iterations and with better relative residual, and SPSC being faster in terms of CPU computational time. All the other methods follow f-SCSP and SCSP. In Table 3, we see results from Example 3, which show that f-SCSP performs superior to all of the existing methods, including SCSP, in terms of all, number of iterations required to converge, the relative residual, and the required CPU time for computation, while SMHSS variants exceed hundreds to thousands of iterations. This consistent performance highlights SCSP's robustness and its practical advantage for large-scale complex symmetric systems.

A catch is the use of the initial guess. All our experiments use $x_0 = \vec{0}$, but many practical solvers benefit from warm starts. Finally, while the convergence proofs (Theorems 2.1-2.3) guarantee $\rho(T_\alpha) < 1$ under stated assumptions, the potential for combining f-SCSP with Krylov acceleration can be addressed, representing an opportunity for further speed-ups in challenging regimes.

Our numerical results presented in the tables are given in line plots. Figure 1 shows the CPU time of taken by the respective methods plotted vs the vector space size m in Example 3.1. The f-SCSP is much faster than most other methods, and it performs very close to the existing SCSP. Similarly, Figure 2 show that in 3.2, as the system size increases, the SCSP performs better than the proposed method. However, it can be seen in Figure 3 for Example 3.3 that both methods perform equally well for all system sizes. Figure 4 show the convergence behavior of the proposed method in Example 3.1 with different system sizes. The residual error is plotted vs the number of iterations, and f-SCSP outperforms the existing methods in all tests, as demonstrated. Similarly, Figure 5 shows how f-SCSP outperforms all of the existing methods in convergence in Example 3.2. In Figure 6, the difference in convergence between f-SCSP and SCSP looks tight, especially in figure 6(b),

Table 1. Tests from Example 3.1. The first column lists the system sizes in $\mathbb{C}^m$. The second column shows iteration count, CPU time, and residual error. Columns 3-7 present the results from SCSP, f-SCSP, f-SMHSS, and MHSS respectively.

m		SCSP	f-SCSP	f-SMHSS	SMHSS	MHSS
16	Iter. Count	10	10	16	18	54
	CPU Time (s)	0.015	0.059	0.048	0.026	0.160
	Res. Err.	5.218e-7	9.694e-7	8.918e-7	6.845e-7	8.238e-7
32	Iter. Count	16	16	26	24	131
	CPU Time (s)	0.150	0.243	0.306	0.202	1.840
	Res. Err.	9.321e-7	4.443e-7	9.043e-7	7.460e-7	9.525e-7
48	Iter. Count	22	20	32	36	171
	CPU Time (s)	0.419	0.608	1.113	0.683	5.053
	Res. Err.	5.287e-07	8.892e-07	7.711e-07	9.641e-07	9.716e-07
64	Iter. Count	28	26	49	55	191
	CPU Time (s)	0.809	1.063	2.681	1.680	5.954
	Res. Err.	6.803e-07	8.043e-07	9.987e-07	8.918e-07	9.875e-07
128	Iter. Count	63	46	119	108	306
	CPU Time (s)	5.971	6.999	13.862	9.638	52.724
	Res. Err.	8.541e-07	9.464e-07	9.601e-07	8.168e-07	9.893e-07
256	Iter. Count	63	60	325	332	997
	CPU Time (s)	28.805	42.680	199.335	302.205	804.894
	Res. Err.	8.258e-07	8.122e-07	9.949e-07	9.929e-07	9.981e-07
512	Iter. Count	103	84	1093	7080	3345
	CPU Time (s)	252.411	510.790	3640.200	17965.00	22926.00
	Res. Err.	9.731e-07	9.537e-07	9.962e-07	9.995e-07	9.993e-07

Table 2. Tests from Example 3.2. The first column lists the system sizes in $\mathbb{C}^m$. The third column shows iteration count, CPU time, and residual error. Columns 3-7 present the results from SCSP, f-SCSP, f-SMHSS, and MHSS respectively.

m		SCSP	f-SCSP	f-SMHSS	SMHSS	MHSS
16	Iter. Count	37	40	268	268	34
	CPU Time (s)	0.053	0.104	0.772	0.372	0.094
	Res. Err.	8.345e-07	8.514e-07	9.782e-07	9.667e-07	9.539e-07
32	Iter. Count	42	38	245	244	49
	CPU Time (s)	0.243	0.364	1.729	1.107	0.557
	Res. Err.	8.969e-07	9.367e-07	9.600e-07	9.878e-07	8.624e-07
48	Iter. Count	44	39	231	231	82
	CPU Time (s)	0.584	0.808	2.900	1.795	1.310
	Res. Err.	8.230e-07	9.204e-07	9.940e-07	9.771e-07	8.920e-07
64	Iter. Count	45	40	222	222	128
	CPU Time (s)	1.147	1.101	6.738	3.955	6.312
	Res. Err.	7.628e-07	7.895e-07	9.781e-07	9.625e-07	9.766e-07
128	Iter. Count	46	41	200	199	440
	CPU Time (s)	4.321	5.710	49.653	30.106	138.168
	Res. Err.	7.429e-07	7.176e-07	9.574e-07	9.870e-07	9.928e-07
256	Iter. Count	46	41	177	177	835
	CPU Time (s)	18.428	26.657	225.347	143.796	1118.5
	Res. Err.	8.145e-07	7.801e-07	9.778e-07	9.643e-07	9.998e-07
512	Iter. Count	46	41	153	152	3160
	CPU Time (s)	140.608	186.084	581.892	355.640	17228.00
	Res. Err.	8.371e-07	7.998e-07	9.613e-07	9.838e-07	9.987e-07

Table 3. Tests from Example 3.3. The first column lists the system sizes in $\mathbb{C}^m$. The third column shows iteration count, CPU time, and residual error. Columns 3-7 present the results from SCSP, f-SCSP, f-SMHSS, and MHSS respectively.

m		SCSP	f-SCSP	f-SMHSS	SMHSS	MHSS
16	Iter. Count	6	3	131	131	150
	CPU Time (s)	0.009	0.010	0.468	0.201	0.443
	Res. Err.	5.645e-07	2.396e-07	9.467e-07	9.374e-07	9.449e-07
32	Iter. Count	6	3	226	226	238
	CPU Time (s)	0.040	0.036	1.938	1.241	1.934
	Res. Err.	5.645e-07	2.396e-07	9.974e-07	9.955e-07	9.782e-07
48	Iter. Count	6	3	345	323	347
	CPU Time (s)	0.080	0.080	5.208	3.519	5.710
	Res. Err.	5.645e-07	2.396e-07	9.934e-07	9.809e-07	9.956e-07
64	Iter. Count	6	3	437	428	624
	CPU Time (s)	0.158	0.155	11.377	7.933	15.952
	Res. Err.	5.645e-07	2.396e-07	9.876e-07	9.973e-07	9.848e-07
128	Iter. Count	6	3	815	751	912
	CPU Time (s)	0.887	0.852	121.434	83.248	177.660
	Res. Err.	5.645e-07	2.396e-07	9.942e-07	9.898e-07	9.914e-07
256	Iter. Count	6	3	1426	1350	1905
	CPU Time (s)	3.062	2.553	1091.90	814.111	1906.40
	Res. Err.	5.645e-07	2.396e-07	9.955e-07	9.950e-07	9.973e-07
512	Iter. Count	6	3	4712	4421	5233
	CPU Time (s)	15.015	13.637	11507.4	17269.0	42689.00
	Res. Err.	5.645e-07	2.396e-07	9.9966e-07	9.994e-07	9.9989e-07

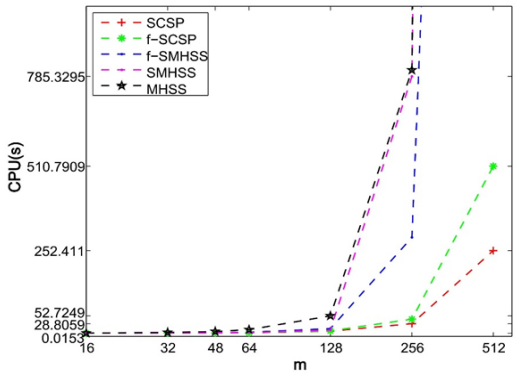

Fig. 1. Comparing f-SCSP with the existing methods in terms of CPU time from Example 3.1. f-SCSP performs better than its most counterparts.

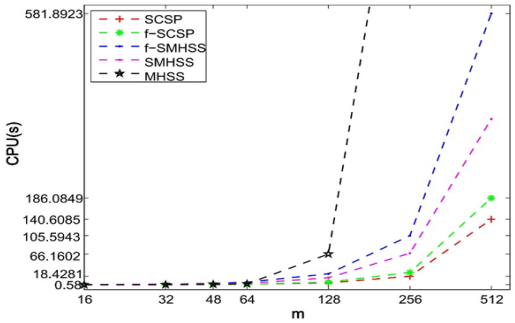

Fig. 2. Comparing f-SCSP with the existing methods in terms of CPU time from Example 3.2. f-SCSP performs better than its counterparts, and is close to SCSP, if not matches its performance. f-SCSP takes a little longer to converge for larger system sizes.

but f-SCSP outperforms SCSP both in number of iterations and final residual error, taking half the number of iterations.

Moreover, Figure 7 shows the eigenvalues spread of the preconditioned matrix vs the actual system matrix in Examples 3.1 for a system size of 48×48 . The real part of an eigenvalue is directly related to how a system behaves over time. If the real part is positive, the system grows exponentially, meaning it becomes unstable over time. If the real part is negative, the system decays exponentially, meaning it settles down to zero. In all preconditioned cases, we see that the eigenvalues have a real part of one and that the system has no fast growing or decaying. Instead, it might oscillate or stay at a constant amplitude. This doesn't guarantee that the matrix is strictly stable, but it demonstrates that the matrix is not unstable either. The same behaviour of strong clustering of the spectrum resulting due to preconditioning can also be observed in Figures 8 and 9 for Example 3.2 and 3.3, respectively, where the preconditioned matrix $M_{\alpha_k}^{-1}A$ evidently has a faster convergence compared to the original matrix A.

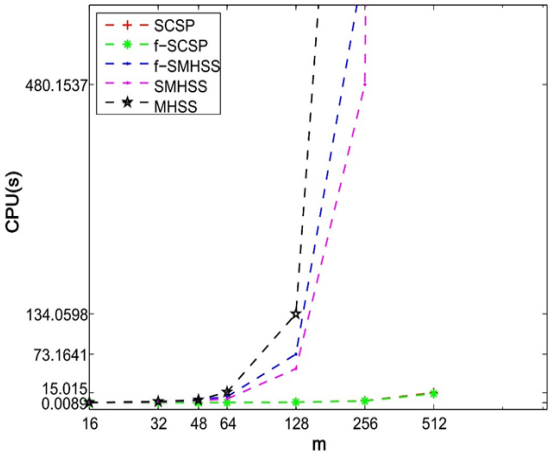

Fig. 3. Comparing f-SCSP with the existing methods in terms of CPU time from Example 3.3. f-SCSP performs better than its counterparts, and performs equally well as SCSP, matching its performance.

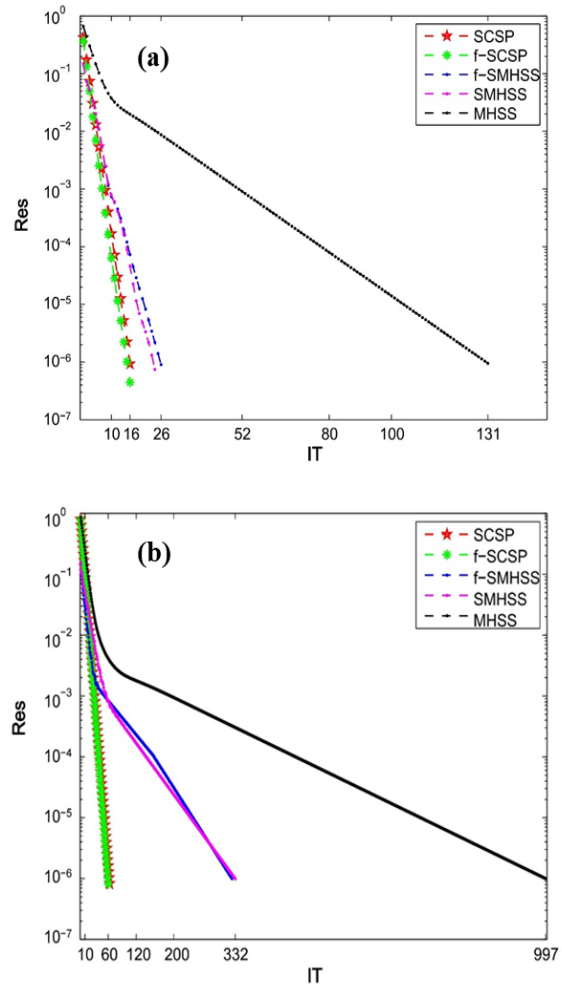

Fig. 4. The convergence behavior of f-SCSP vs its counterparts. (a) show tests from Example 3.1 with vector space $\mathbb{C}^{32}$ and (b) shows $\mathbb{C}^{256}$. Clearly, the convergence in f-SCSP dominates others with a margin.

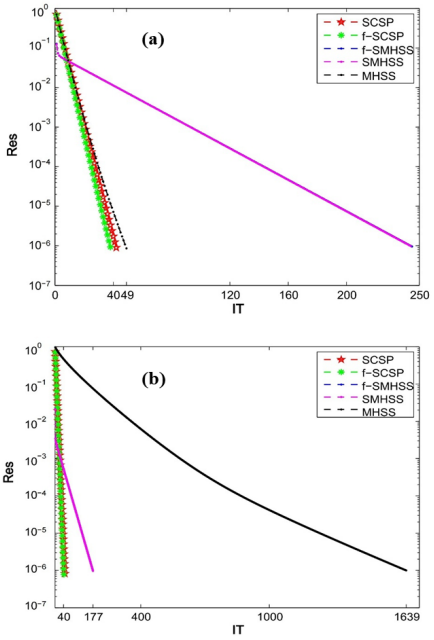

Fig. 5. The convergence behavior of f-SCSP vs its counterparts. (a) test results from Example 3.2 with vector space $\mathbb{C}^{32}$ and (b) shows results with vector space $\mathbb{C}^{256}$. f-SCSP dominates others in convergence with a margin. (a) show the dominance of f-SCSP clearly; whereas (b) shows convergence line of f-SCSP close to SCSP; however, f-SCSP convergence in fewer iterations and with lower residual error.

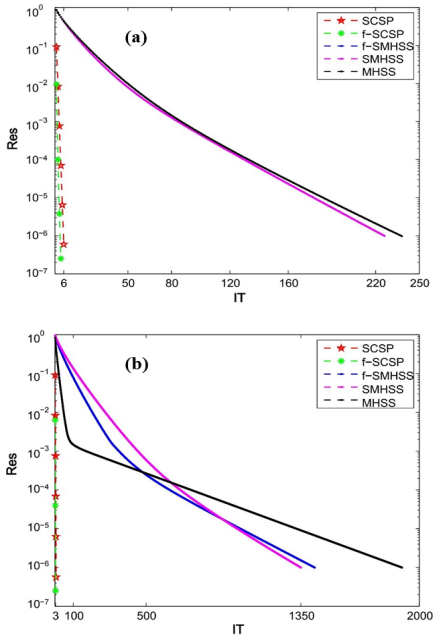

Fig. 6. The convergence behavior of f-SCSP vs its counterparts. (a) test results from Example 3.3 with vector space $\mathbb{C}^{32}$ and (b) shows results with vector space $\mathbb{C}^{256}$. f-SCSP dominates others in convergence with a margin. (a) show the dominance of f-SCSP clearly; however, (b) shows almost overlapping lines for f-SCSP and SCSP; but f-SCSP convergence in half the number of iterations required by SCSP and with lower residual error.

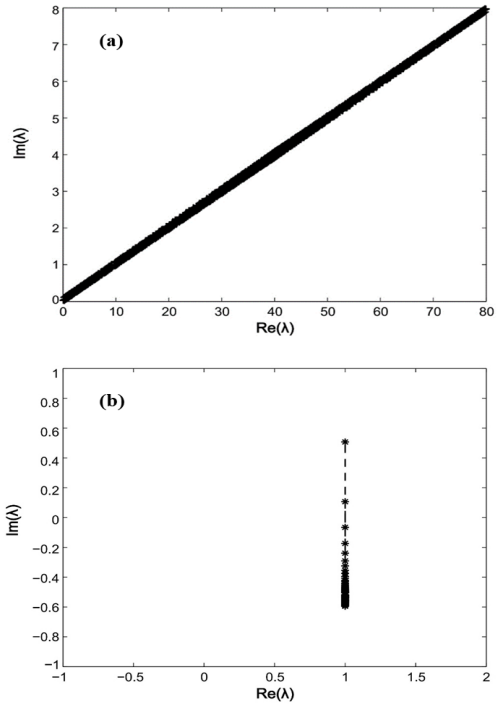

Fig. 7. The eigenvalues of the matrices $\boldsymbol{A}$ compared (a), and the preconditioned matrix $\boldsymbol{M}_{\alpha_k}^{-1}\boldsymbol{A}$ (b), from the system matrix in Example 3.1. The eigenvalues spread in preconditioned system matrix (b) shows the eigenvalues clustered much closer compared to the original matrices (a). Note that the axes ranges are not consistent.

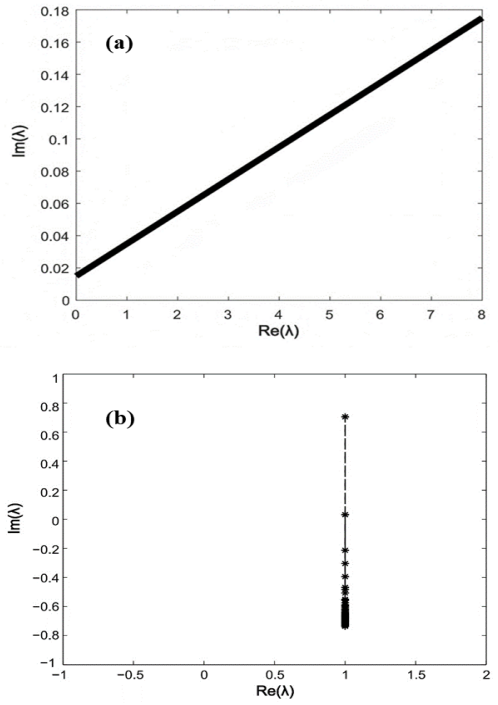

Fig. 8. The eigenvalues of the matrices $\boldsymbol{A}$ compared (a), and the preconditioned matrix $\boldsymbol{M}_{\alpha_k}^{-1}\boldsymbol{A}$ (b), from the system matrix in Example 3.2. The eigenvalues spread in preconditioned system matrix (b) shows the eigenvalues clustered much closer compared to the original matrices (a). Note that the axes ranges are not consistent.

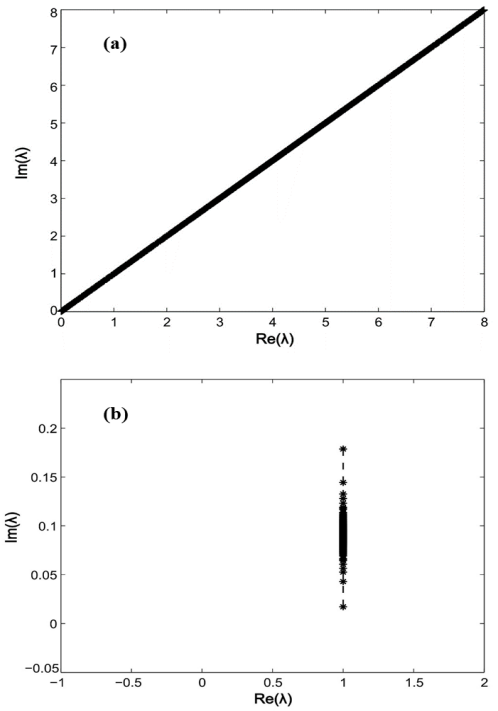

Fig. 9. The eigenvalues of the matrices $\boldsymbol{A}$ compared (a), and the preconditioned matrix $\boldsymbol{M}_{\alpha_k}^{-1}\boldsymbol{A}$ (b), from the system matrix in Example 3.3. The eigenvalues spread in preconditioned system matrix (b) shows the eigenvalues clustered much closer compared to the original matrices (a). Note that the axes ranges are not consistent.

4. CONCLUSIONS

In this paper, we have presented a flexible-scalar splitting iterative methods based on the SCSP method for effectively solving a broad category of complex symmetric linear systems. Special attention is given to the structure and properties of the equivalent systems $(\alpha - i)Ax = (\alpha - i)b$ particularly in cases where the parameters α is chosen to preserve the symmetry and improve the conditioning of the original system. Theoretical analyses have been conducted to demonstrate that the proposed method is convergent under reasonable and practically relevant assumptions. Moreover, explicit expressions linking the optimal parameters α to the spectral radius of the associated iteration matrix have been established, offering a rigorous theoretical basis for parameter tuning and enhanced convergence rates.

To evaluate the practical efficacy of the proposed approaches, extensive numerical experiments were performed comparing the f-SCSP method against four established algorithms from the literature [28]. The findings consistently highlight the proposed

method' reliability, robustness, and computational efficiency. Notably, the f-SCSP method exhibit equal or superior convergence rates and iteration counts, thereby confirming their suitability for tackling complex symmetric linear systems.

5. ACKNOWLEDGEMENTS

This research was supported by the National Natural Science Foundation of China under Grant No. 12001395, and by the Natural Science Foundation of Shanxi Province, China under Grant No. 202403021222270.

6. CONFLICT OF INTEREST

The authors declare that they have no financial or personal conflicts of interest that could have influenced the research presented in this paper. This study was carried out independently, with funding sources having no role in the design, execution, or interpretation of the results.

7. REFERENCES

1. S.R. Arridge. Optical tomography in medical imaging. *Inverse Problems* 15(2): R41-R93 (1999).
2. D. Bertaccini. Efficient preconditioning for sequences of parametric complex symmetric linear systems. *Electronic Transactions on Numerical* 18: 49-64 (2004).
3. A. Feriani, F. Perotti, and V. Simoncini. Iterative system solvers for the frequency analysis of linear mechanical systems. *Computational Methods in Applied Mechanics and Engineering* 190: 1719-1739 (2000).
4. A. Frommer, T. Lippert, B. Medeke, and K. Schilling (Eds.). Numerical challenges in lattice quantum chromodynamics. *Proceedings, Joint Interdisciplinary Workshop, Wuppertal, Germany* (2000).
5. B. Poirier. Efficient preconditioning scheme for block partitioned matrices with structured sparsity. *Numerical Linear Algebra with Applications* 7: 715-726 (2000).
6. W.V. Dijk and F.M. Toyama. Accurate numerical solutions of the time-dependent Schrödinger equation. *Physical Review E* 75: 036707 (2007).
7. Z. Ahmed, Z.A. Kalhoro, A.W. Shaikh, M.S.R. Baloch, and O.A. Rajput. An improved iterative scheme using successive over-relaxation for solution of linear system of equations. *Proceedings of the Pakistan Academy of Sciences: Part A. Physical and Computational Sciences* 59(3): 35-43 (2022).
8. M. Kanwal, Z. Ahmed, and S. Jamali. Development of generalized refinement strategies in composite stationary iterative solvers for linear systems. *Southern Journal of Research* 5(02(01)): 1-13 (2025).
9. O. Axelsson and A. Kucherov. Real valued iterative methods for solving complex symmetric linear systems. *Numerical Linear Algebra with Applications* 7: 197-218 (2000).
10. M. Benzi and D. Bertaccini. Block preconditioning of real valued iterative algorithms for complex linear systems. *IMA Journal of Numerical Analysis* 28: 598-618 (2007).
11. Z.Z. Bai. On preconditioned iteration methods for complex linear systems. *Journal of Engineering Mathematics* 93: 41-60 (2014).
12. Z.Z. Bai, M. Benzi, and F. Chen. Modified HSS iteration methods for a class of complex symmetric linear systems. *Computing* 87: 93-111 (2010).
13. Z.Z. Bai, M. Benzi, and F. Chen. On preconditioned MHSS iteration methods for complex symmetric linear systems. *Numerical Algorithms* 56: 297-317 (2011).
14. T. Wang, Q. Zheng, and L. Lu. A new iteration method for a class of complex symmetric linear systems. *Journal of Computational and Applied Mathematics* 325: 188-197 (2017).
15. D.K. Salkuyeh, D. Hezari, and V. Edalatpour. Generalized successive overrelaxation iterative method for a class of complex symmetric linear system of equations. *International Journal of Computer Mathematics* 92: 802-815 (2015).
16. D. Hezari, V. Edalatpour, and D.K. Salkuyeh. Preconditioned GSOR iterative method for a class of complex symmetric system of linear equations. *Numerical Linear Algebra with Applications* 22: 761-776 (2015).
17. O. Axelsson and D.K. Salkuyeh. A new version of a preconditioning method for certain two-by-two block matrices with square blocks. *BIT Numerical Mathematics* 59: 321-342 (2019).
18. X. Xie and H. Li. On preconditioned Euler-extrapolated single-step Hermitian and skew-Hermitian splitting method for complex symmetric linear systems. *Japan Journal of Industrial and Applied Mathematics* 8: 503-518 (2020).
19. Y. Xiang and N.M. Zhang. On the preconditioned conjugate gradient method for complex symmetric systems. *Applied Mathematics Letters* 120: 107250 (2021).
20. D.K. Salkuyeh. A Preconditioner for Complex

Symmetric System of Linear Equations with Indefinite Hermitian Part. *Bulletin of the Iranian Mathematical Society* 51: 25-25 (2025).

21. P.P. Zhao, S.D. Li, and R.P. Wen. Single-step HSS methods for a class of complex symmetric linear systems. *Communications in Applied Mathematics and Computation* 31: 200-212 (2017).
22. R.P. Wen, F.J. Ren, and Y.Q. Gao. On convergence of splitting iteration methods for the complex symmetric linear systems. *Mathematica Applicanda* 29: 173-182 (2016).
23. R.P. Wen, S.D. Li, and F.J. Ren. A new splitting and preconditioner for iteratively solving a class of complex symmetric linear systems. *Mathematica Applicanda* 27: 65-72 (2014).
24. H.A. van-der-Vorst and J.B.M. Melissen. A Petrov–Galerkin type method for solving $Ax = b$, where A is symmetric complex. *IEEE Transactions on Magnetics* 26: 706-708 (1990).
25. R.W. Freund. Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices. *SIAM Journal on Scientific and Statistical Computing* 13: 425-448 (1992).
26. A. Bunse-Gerstner and R. Stöver. On a conjugate gradient–type method for solving complex symmetric linear systems. *Linear Algebra and Its Applications* 287: 105-123 (1999).
27. M. Clemens, T. Weiland, and U. Van-Rienen. Comparison of Krylov-type methods for complex linear systems applied to high-voltage problems. *IEEE Transactions on Magnetics* 34: 3335-3338 (1998).
28. D. Hezari, D.K. Salkuyeh, and V. Edalatpour. A new iterative method for solving a class of complex symmetric system of linear equations. *Numerical Algorithms* 73: 927-955 (2016).
29. D.K. Salkuyeh. Two-step scale-splitting method for solving complex symmetric system of linear equations. *Arxiv Preprint Arxiv*: 1705.02468 (2017).
30. D.K. Salkuyeh and T.S. Siahkolaei. Two-parameter TSCSP method for solving complex symmetric system of linear equations. *Calcolo* 55: 8 (2018).
31. Z. Zheng, F.L. Huang, and Y.C. Peng. Double-step scale splitting iteration method for a class of complex symmetric linear systems. *Applied Mathematics Letters* 73: 91-97 (2017).
32. B. Li, J. Cui, Z. Huang, and X. Xie. A dual-parameter double-step splitting iteration method for solving complex symmetric linear equations. *Applications of Mathematics* 69: 311-337 (2024).
33. B. Li, J. Cui, Z. Huang, and X. Xie. Two Quasi-combining Real and Imaginary Parts Iteration Methods for Solving Complex Symmetric System of Linear Equations. *Communications on Applied Mathematics and Computation* (2024). https://doi.org/10.1007/s42967-024-00448-0