Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 408167, 16 pages
http://dx.doi.org/10.1155/2013/408167

Research Article

Exploiting the Composite Step Strategy to the Biconjugate A-Orthogonal Residual Method for Non-Hermitian Linear Systems

Yan-Fei Jing,[1] Ting-Zhu Huang,[1] Bruno Carpentieri,[2] and Yong Duan[1]

[1] *School of Mathematical Sciences, Institute of Computational Science, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[2] *Institute of Mathematics and Computing Science, University of Groningen, Nijenborgh 9, P.O. Box 407, 9700 AK Groningen, The Netherlands*

Correspondence should be addressed to Yan-Fei Jing; 00jyfvictory@163.com

Received 15 October 2012; Accepted 19 December 2012

Academic Editor: Zhongxiao Jia

The Biconjugate A-Orthogonal Residual (BiCOR) method carried out in finite precision arithmetic by means of the biconjugate A-orthonormalization procedure may possibly tend to suffer from two sources of numerical instability, known as two kinds of breakdowns, similarly to those of the Biconjugate Gradient (BCG) method. This paper naturally exploits the composite step strategy employed in the development of the composite step BCG (CSBCG) method into the BiCOR method to cure one of the breakdowns called as pivot breakdown. Analogously to the CSBCG method, the resulting interesting variant, with only a minor modification to the usual implementation of the BiCOR method, is able to avoid near pivot breakdowns and compute all the well-defined BiCOR iterates stably on the assumption that the underlying biconjugate A-orthonormalization procedure does not break down. Another benefit acquired is that it seems to be a viable algorithm providing some further practically desired smoothing of the convergence history of the norm of the residuals, which is justified by numerical experiments. In addition, the exhibited method inherits the promising advantages of the empirically observed stability and fast convergence rate of the BiCOR method over the BCG method so that it outperforms the CSBCG method to some extent.

1. Introduction

The computational cost of many simulations with integral equations or partial differential equations that model the process is dominated by the solution of systems of linear equations of the form $Ax = b$. The field of iterative methods for solving linear systems has been observed as an explosion of activity spurred by demand due to extraordinary technological advances in engineering and sciences in the past half century [1]. Krylov subspace methods belong to one of the most widespread and extensively accepted techniques for iterative solution of today's large-scale linear systems [2]. With respect to "the greatest influence on the development and practice of science and engineering in the 20th century" as written by Dongarra and Sullivan [3], Krylov subspace methods are considered as one of the "Top Ten Algorithms of the Century."

Many advances in the development of Krylov subspace methods have been inspired and made by the study of even more effective approaches to linear systems. Various methods differ in the way they extract information from Krylov spaces. A basis for the underlying Krylov subspace to form iterative recurrences involved is usually constructed based on the state-of-the-art Arnoldi orthogonalization procedure or Lanczos biorthogonalization procedure; see recent excellent and thorough review articles by Simoncini and Szyld [2] and by Philippe and Reichel [4] or monographs such as those of Greenbaum [5] and Saad [6]. A novel Lanczos-type biconjugate A-orthonormalization procedure has recently been established to give birth to a new family of efficient

short-recurrence methods for large real nonsymmetric and complex non-Hermitian systems of linear equations, named as the Lanczos biconjugate A-orthonormalization methods [7].

As observed from numerous numerical experiments carried out with the Lanczos biconjugate A-orthonormalization methods, it has been numerically demonstrated that this family of solvers shows competitive convergence properties, is cheap in memory as it is derived from short-term vector recurrences, is parameter-free, and does not require a symmetric preconditioner like other methods, if A is symmetric indefinite [8–10]. The efficiency of the BiCOR method is highlighted by the performance profiles on a set of 14 sparse matrix problems of size up to 1.5 M unknowns shown in [10].

On the other hand, this family of solvers is often faced with apparently irregular convergence behaviors appearing as "spikes" in the convergence history of the norm of the residuals, possibly leading to substantial build-up of rounding errors and worse approximate solutions, or possibly even overflow [7, 9]. Therefore, it is quite necessary to tackle their irregular convergence properties to obtain more stabilized variants so as to improve the accuracy of the desired numerical physical solutions. It is emphasized that our main attention in this paper is focused on the straightforward natural enhancement of the Biconjugate A-Orthogonal Residual (BiCOR) method, which is the basic underlying variant of the Lanczos biconjugate A-Orthonormalization methods [7]. The improvement of the BiCOR method will simultaneously result in analogous improvements for the other two evolving variants known as the Conjugate A-orthogonal Residual Squared (CORS) method and the Biconjugate A-Orthogonal Residual Stabilized (BiCORSTAB) method.

By comparison of the biconjugate A-orthonormalization procedure [7] and the Lanczos biorthogonalization procedure [6], respectively, for the BiCOR method [7] and the Biconjugate Gradient (BCG) method [11] as well as their corresponding implementation algorithms, it is obviously observed that when carried out in finite precision arithmetic, the BiCOR method does tend to suffer from two possible sources of numerical instability, known as two kinds of breakdowns, exactly similarly to those of the BCG method. We still call such two kinds of breakdowns as Lanczos breakdown and pivot breakdown, which will be analyzed and discussed in the following sections. As far as reported, a large number of strategies designed to handle such two kinds of breakdowns have been proposed in the literature [12–16], including composite step techniques [11, 17–21] for pivot breakdown and look-ahead techniques [22–30] for Lanczos breakdown or both types. For comprehensive reviews and discussions about the two kinds of breakdowns mentioned above, refer to [2, 6, 16] and the references therein. As shown and analyzed detailedly by Bank and Chan [18, 19], the 2×2 composite step strategy employed in the development of the composite step biconjugate gradient method (CSBCG) can eliminate (near) pivot breakdowns of the BCG method caused by (near) singularity of principal submatrices of the tridiagonal matrix generated by the underlying Lanczos biorthogonalization procedure, assuming that Lanczos breakdowns do not occur. Furthermore, besides simplifying implementations a great deal in comparison with those existing look-ahead techniques, the composite step strategy used there is able to provide some smoothing of the convergence history of the norm of the residuals without involving any arbitrary machine-dependent or user-supplied tolerances.

This paper revisits the composite step strategy taken for the CSBCG method [18, 19] to naturally stabilize the convergence performance of the BiCOR method from the pivot-breakdown treatment point of view. Therefore, suppose that the underlying biconjugate A-orthonormalization procedure does not break down; that is to assume that there is no occurrence of the so-called Lanczos breakdown encountered during algorithm implementations throughout. The resulting interesting algorithm, given a name as the composite step BiCOR (CSBiCOR) method, could be also considered as a relatively simple modification of the regular BiCOR method.

The main objectives are twofold. First, the CSBiCOR method is devised to be able to avoid near pivot breakdowns and compute all the well-defined BiCOR iterates stably with only minor modifications inspired by the advantages of the CSBCG method over the BCG method. Second, the CSBiCOR method can reduce the number of spikes in the convergence history of the norm of the residuals to the greatest extent, providing some further practically desired smoothing behavior towards stabilizing the behavior of the BiCOR method when it has erratic convergence behaviors. Additional purpose is that the CSBiCOR method inherits the promising advantages of the empirically observed stability and fast convergence rate of the BiCOR method over the BCG method so that it outperforms the CSBCG method to some extent.

The remainder of this work is organized as follows. A brief review of the biconjugate A-orthonormalization procedure is made and the factorization of the resulted nonsingular tridiagonal matrices in the aforementioned procedure is considered in Section 2 for the preparation of the theoretical basis and relevant background of the CSBiCOR method. And then the breakdowns of the BiCOR method are presented in Section 3 while the CSBiCOR method is derived in Section 4. Section 5 analyzes the properties of the CSBiCOR method as well as providing a best approximation result for the convergence of the CSBiCOR method. In order to conveniently and simply present the CSBiCOR method, some implementation issues on both algorithm simplification and stepping strategy are discussed detailedly in Section 6. The improved performance of the CSBiCOR method will be illustrated in Section 7 in the perspective that the CSBiCOR method could hopefully smooth the convergence history through the reduction of the number of spikes when the BiCOR method has irregular convergence behavior. Finally, concluding remarks are made in the last section.

Throughout the paper, we denote the overbar "–" the conjugate complex of a scalar, vector or matrix and the superscript "T" the transpose of a vector, or matrix. For a non-Hermitian matrix $A = (a_{ij})_{N\times N} \in \mathbb{C}^{N\times N}$, the Hermitian conjugate of A is denoted as

$$A^H \equiv \overline{A}^T = \left(\overline{a}_{ji}\right)_{N\times N}. \tag{1}$$

The standard Hermitian inner product of two complex vectors $u, v \in \mathbb{C}^N$ is defined as

$$\langle u, v\rangle = u^H v = \sum_{i=1}^{N} \overline{u}_i v_i. \tag{2}$$

The nested Krylov subspace of dimension t generated by A from v is of the form

$$\mathcal{K}_t(A, v) = \operatorname{span}\left\{v, Av, A^2 v, \ldots, A^{t-1} v\right\}. \tag{3}$$

In addition, e_i denotes the ith column of the appropriate identity matrix.

When it will be helpful, we will use the word "ideally" (or "mathematically") to refer to a result that could hold in exact arithmetic ignoring effects of rounding errors, and "numerically" (or "computationally") to a result of a finite precision computation.

2. Theoretical Basis for the CSBiCOR Method

For the sake of discussing the theoretical basis and relevant background of the CSBiCOR method, the biconjugate A-orthonormalization procedure [7] is first briefly recalled as in Algorithm 1, which can ideally build up a pair of biconjugate A-orthonormal bases for the dual Krylov subspaces $\mathcal{K}_m(A, v_1)$ and $\mathcal{K}_m(A^H, w_1)$, where v_1 and w_1 are chosen initially to satisfy certain conditions.

Observe that the above algorithm is possible to have Lanczos-type breakdown whenever δ_{j+1} vanishes while $\widehat{w}_{j+1}$ and $A\widehat{v}_{j+1}$ are not equal to $\mathbf{0} \in \mathbb{C}^N$ appearing in line 8. In the interest of counteraction against such breakdowns, refer oneself to remedies such as the so-called look-ahead strategies [22–30] which can enhance stability while increasing cost modestly. But that is outside the scope of this paper and we will not pursue that here. For more details, please refer to [2, 6] and the references therein. Throughout the rest of the present paper, suppose there is no such Lanczos-type breakdown encountered during algorithm implementations because most of our considerations concern the exploration of the composite step strategy [18, 19] to handle the pivot breakdown occurring in the BiCOR method for solving non-Hermitian linear systems.

Next, some properties of the vectors produced by Algorithm 1 are reviewed [7] in the following proposition for the preparation of the theoretical basis of the composite step method.

Proposition 1. *If Algorithm 1 proceeds m steps, then the right and left Lanczos-type vectors v_j, $j = 1, 2, \ldots, m$ and w_i, $i = 1, 2, \ldots, m$ form a biconjugate A-orthonormal system in exact arithmetic, that is,*

$$\left\langle \omega_i, Av_j \right\rangle = \delta_{i,j}, \quad 1 \le i, j \le m. \tag{4}$$

Furthermore, denote by $V_m = [v_1, v_2, \ldots, v_m]$ and $W_m = [w_1, w_2, \ldots, w_m]$ the $N \times m$ matrices and by $\underline{T_m}$ the extended tridiagonal matrix of the form

$$\underline{T_m} = \begin{bmatrix} T_m \\ \delta_{m+1} e_m^T \end{bmatrix}, \tag{5}$$

(1) Choose v_1, ω_1, such that $\langle \omega_1, Av_1\rangle = 1$
(2) Set $\beta_1 = \delta_1 \equiv 0$, $\omega_0 = v_0 \equiv \mathbf{0} \in \mathbb{C}^N$
(3) **for** $j = 1, 2, \ldots$ **do**
(4) $\quad \alpha_j = \left\langle \omega_j, A(Av_j)\right\rangle$
(5) $\quad \widehat{v}_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1}$
(6) $\quad \widehat{\omega}_{j+1} = A^H \omega_j - \overline{\alpha}_j \omega_j - \delta_j \omega_{j-1}$
(7) $\quad \delta_{j+1} = \left|\left\langle \widehat{\omega}_{j+1}, A\widehat{v}_{j+1}\right\rangle\right|^{1/2}$
(8) $\quad \beta_{j+1} = \left\langle \widehat{\omega}_{j+1}, A\widehat{v}_{j+1}\right\rangle / \delta_{j+1}$
(9) $\quad v_{j+1} = \widehat{v}_{j+1}/\delta_{j+1}$
(10) $\quad \omega_{j+1} = \widehat{\omega}_{j+1}/\overline{\beta}_{j+1}$
(11) **end for**

Algorithm 1: Biconjugate A-orthonormalization procedure.

where

$$T_m = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \delta_2 & \alpha_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \delta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \delta_m & \alpha_m \end{bmatrix}, \tag{6}$$

whose entries are the coefficients generated during the algorithm implementation, and in which $\alpha_1, \ldots, \alpha_m, \beta_2, \ldots, \beta_m$ are complex while $\delta_2, \ldots, \delta_m$ are positive. Then with the biconjugate A-orthonormalization procedure, the following four relations hold:

$$AV_m = V_m T_m + \delta_{m+1} v_{m+1} e_m^T, \tag{7}$$

$$A^H W_m = W_m T_m^H + \overline{\beta}_{m+1} \omega_{m+1} e_m^T, \tag{8}$$

$$W_m^H A V_m = I_m, \tag{9}$$

$$W_m^H A^2 V_m = T_m. \tag{10}$$

It is well known that the Lanczos biorthogonalization procedure can mathematically build up a pair of biorthogonal bases for the dual Krylov subspaces $\mathcal{K}_m(A, v_1)$ and $\mathcal{K}_m(A^H, w_1)$, where v_1 and w_1 are initially selected such that $\langle \omega_1, v_1\rangle = 1$ [6]. While the biconjugate A-orthonormalization procedure presented as in Algorithm 1 can ideally build up a pair of biconjugate A-orthonormal bases for the dual Krylov subspaces $\mathcal{K}_m(A, v_1)$ and $\mathcal{K}_m(A^H, w_1)$, where v_1 and w_1 are chosen initially to satisfy $\langle \omega_1, Av_1\rangle = 1$. Keeping in mind the above differences in terms of different Krylov subspace bases constructed by the two different underlying procedures, we will in parallel borrow some of the results of the CSBCG method [18] for establishing the following counterparts for the CSBiCOR method; see [18] for relevant proofs.

It should be emphasized that the present CSBiCOR method will be derived in exactly the same way the CSBCG method is obtained [19]. Hence, the analysis and theoretical background for the CSBiCOR method in this section are the

counterparts of the CSBCG method in a different context, that is, the biconjugate A-orthonormalization procedure as presented in Algorithm 1. They are included as follows for the purpose of theoretical completeness and can be proved in the same way for the corresponding results of the CSBCG method analyzed in [18].

The following proposition states the tridiagonal matrix T_m formed in Proposition 1 cannot have two successive singular leading principal submatrices if the coefficient matrix A is nonsingular.

Proposition 2. *Let T_k $(1 \le k \le m)$ be the upper left principal submatrices of the nonsingular tridiagonal matrix T_m appeared as in Proposition 1. Then T_{k-1} and T_k cannot be both singular.*

It is noted that there may exist possible breakdown in the factorization without pivoting of the tridiagonal matrix T_m formed in Proposition 1. The following results illustrate how to correct such problem with the occasional use of 2×2 block pivots.

Proposition 3. *Let T_m be the nonsingular tridiagonal matrix formed in Proposition 1. Then T_m can be factored as*

$$T_m = L_m D_m U_m, \tag{11}$$

where L_m is unit lower block bidiagonal, U_m is unit upper block bidiagonal, and D_m is block diagonal, with 1×1 and 2×2 diagonal blocks.

With the above two propositions, one can prove the following result insuring that if the upper left principal submatrices T_k $(1 \le k \le m)$ of T_m is singular, it can still be factored.

Corollary 4. *Suppose one upper left principal submatrix T_k $(1 \le k \le m)$ of the nonsingular tridiagonal matrix T_m formed in Proposition 1 is singular, but T_{k-1} must be nonsingular as stated in Proposition 2. Then*

$$T_k = L_k D_k U_k, \tag{12}$$

where L_k is unit lower block bidiagonal, U_k is unit upper block bidiagonal, and D_k is block diagonal, with 1×1 and 2×2 diagonal blocks, and in particular, the last block of D_k is the 1×1 zero matrix.

From Corollary 4, the singularity of T_k $(1 \le k \le m)$ can be recognized by only examining the last diagonal element (and next potential pivot) d_k. If $d_k = 0$, then the 2×2 block

$$\begin{bmatrix} d_k & \beta_{k+1} \\ \delta_{k+1} & \alpha_{k+1} \end{bmatrix} \tag{13}$$

will be nonsingular and can be used as a 2×2 pivot, where β_{k+1}, δ_{k+1}, and α_{k+1} are the corresponding elements of T_m as shown in Proposition 1.

It can be concluded that appropriate combinations of 1×1 and 2×2 steps for the BiCOR method can skip over the breakdowns caused by the singular principal submatrices of the tridiagonal matrix T_m generated by the underlying biconjugate A-orthonormalization procedure presented as in Algorithm 1 since the BiCOR method pivots implicitly with those submatrices. The next section will have a detailed investigation of breakdowns possibly occurring in the BiCOR method.

3. Breakdowns of the BiCOR Method

Given an initial guess x_0 to the non-Hermitian linear system $Ax = b$ associated with the initial residual $r_0 = b - Ax_0$, define a Krylov subspace $\mathscr{L}_m \equiv A^H \operatorname{span}(W_m) = A^H \mathscr{K}_m(A^H, w_1)$, where W_m is defined in Proposition 1, $v_1 = r_0/||r_0||_2$ and w_1 is chosen arbitrarily such that $\langle w_1, Av_1 \rangle \neq 0$. But w_1 is often chosen to be equal to $Av_1/||Av_1||_2^2$ subjecting to $\langle w_1, Av_1 \rangle = 1$. It is worth noting that this choice for w_1 plays a significant role in establishing the empirically observed superiority of the BiCOR method to the BiCR [31] method as well as to the BCG method [7]. Thus running Algorithm 1 m steps, we can seek an mth approximate solution x_m from the affine subspace $x_0 + \mathscr{K}_m(A, v_1)$ of dimension m, by imposing the Petrov-Galerkin condition

$$b - Ax_m \perp \mathscr{L}_m, \tag{14}$$

which can be mathematically written in matrix formulation as

$$\left(A^H W_m\right)^H (b - Ax_m) = 0. \tag{15}$$

Analogously, an mth dual approximation x_m^* of the corresponding dual system $A^H x^* = b^*$ is sought from the affine subspace $x_0^* + \mathscr{K}_m(A^H, w_1)$ of dimension m by satisfying

$$b^* - A^H x_m^* \perp A\mathscr{K}_m(A, v_1), \tag{16}$$

which can be mathematically written in matrix formulation as

$$(AV_m)^H \left(b^* - A^H x_m\right) = 0, \tag{17}$$

where x_0^* is an initial dual approximate solution and V_m is defined in Proposition 1 with $v_1 = r_0/||r_0||_2$.

Consequently, the BiCOR iterates x_j's can be computed by the coming Algorithm 2, which is just the unpreconditioned BiCOR method with the preconditioner M there taken as the identity matrix [7] and has been rewritten with the algorithmic scheme of the unpreconditioned BCG method as presented in [6, 19].

Suppose Algorithm 2 runs successfully to step n, that is, $\sigma_i \neq 0$, $\rho_i \neq 0$, $i = 0, 1, \ldots, n-1$. The BiCOR iterates satisfy the following properties [7].

Proposition 5. *Let $R_{n+1} = [r_0, r_1, \ldots, r_n]$, $R_{n+1}^* = [r_0^*, r_1^*, \ldots, r_n^*]$ and $P_{n+1} = [p_0, p_1, \ldots, p_n]$, $P_{n+1}^* = [p_0^*, p_1^*, \ldots, p_n^*]$. One has the following.*

(1) *$\mathrm{Range}(R_{n+1}) = \mathrm{Range}(P_{n+1}) = \mathscr{K}_{n+1}(A, r_0)$, $\mathrm{Range}(R_{n+1}^*) = \mathrm{Range}(P_{n+1}^*) = \mathscr{K}_{n+1}(A^H, r_0^*)$.*

(1) Compute $r_0 = b - Ax_0$ for some initial guess x_0.
(2) Choose $r_0^* = P(A)\, r_0$ such that $\langle r_0^*, Ar_0 \rangle \neq 0$, where $P(t)$ is a polynomial in t.
(e.g., $r_0^* = Ar_0$).
(3) Set $p_0 = r_0$, $p_0^* = r_0^*$, $q_0 = Ap_0$, $q_0^* = A^H p_0^*$, $\widehat{r}_0 = Ar_0$, $\rho_0 = \langle r_0^*, \widehat{r}_0 \rangle$.
(4) **for** $n = 0, 1, \ldots$ **do**
(5) $\sigma_n = \langle q_n^*, q_n \rangle$
(6) $\alpha_n = \rho_n / \sigma_n$
(7) $x_{n+1} = x_n + \alpha_n p_n$
(8) $r_{n+1} = r_n - \alpha_n q_n$
(9) $x_{n+1}^* = x_n^* + \overline{\alpha}_n p_n^*$
(10) $r_{n+1}^* = r_n^* - \overline{\alpha}_n q_n^*$
(11) $\widehat{r}_{n+1} = Ar_{n+1}$
(12) $\rho_{n+1} = \langle r_{n+1}^*, \widehat{r}_{n+1} \rangle$
(13) **if** $\rho_{n+1} = 0$, **method fails**
(14) $\beta_{n+1} = \rho_{n+1} / \rho_n$
(15) $p_{n+1} = r_{n+1} + \beta_{n+1} p_n$
(16) $p_{n+1}^* = r_{n+1}^* + \overline{\beta}_{n+1} p_n^*$
(17) $q_{n+1} = \widehat{r}_{n+1} + \beta_{n+1} q_n$
(18) $q_{n+1}^* = A^H p_n^*$
(19) check convergence; continue if necessary
(20) **end for**

Algorithm 2: Algorithm BiCOR.

(2) $R_{n+1}^{*H} A R_{n+1}$ *is diagonal.*

(3) $P_{n+1}^{*H} A^2 P_{n+1}$ *is diagonal.*

Similarly to the breakdowns of the BCG method [19], it is observed from Algorithm 2 that there also exist two possible kinds of breakdowns for the BiCOR method:

(1) $\rho_n \equiv \langle r_n^*, \widehat{r}_n \rangle \equiv \langle r_n^*, Ar_n \rangle = 0$ but r_n^* and Ar_n are not equal to $\mathbf{0} \in \mathbb{C}^N$ appearing in line 14;

(2) $\sigma_n \equiv \langle q_n^*, q_n \rangle \equiv \langle A^H p_n^*, Ap_n \rangle = 0$ appearing in line 6.

Although the computational formulae for the quantities where the breakdowns reside are different between the BiCOR method and the BCG method, we do not have a better name for them. And therefore, we still call the two cases of breakdowns described above as Lanczos breakdown and pivot breakdown, respectively.

The Lanczos breakdown can be cured using look-ahead techniques [22–30] as mentioned in the first section, but such techniques require a careful and sophisticated way so as to make them become necessarily quite complicated to apply. This aspect of applying look-ahead techniques to the BiCOR method demands further research.

In this paper, we attempt to resort to the composite step idea employed for the CSBCG method [18, 19] to handle the pivot breakdown of the BiCOR method with the assumption that the underlying biconjugate A-orthonormalization procedure depicted as in Algorithm 1 does not break down; that is the situation where $\sigma_n = 0$ while $\rho_n \neq 0$.

4. The Composite Step BiCOR Method

Suppose Algorithm 2 comes across a situation where $\sigma_n = 0$ after successful algorithm implementation up to step n with the assumption that $\rho_n \neq 0$, which indicates that the updates of x_{n+1}, r_{n+1}, x_{n+1}^*, r_{n+1}^* are not well defined. Taking the composite step idea, we will avoid division by $\sigma_n = 0$ via skipping this $(n+1)$th update and exploiting a composite step update to directly obtain the quantities in step $(n+2)$ with scaled versions of r_{n+1} and r_{n+1}^* as well as with the previous primary search direction vector p_n and shadow search direction vector p_n^*. The following process for deriving the CSBiCOR method is the same as that of the derivation of the CSBCG method [19] except for the different underlying procedures involved to correspondingly generate different Krylov subspace bases.

Analogously, define auxiliary vectors $z_{n+1} \in \mathcal{K}_{n+2}(A, r_0)$ and $z_{n+1}^* \in \mathcal{K}_{n+2}(A^H, r_0^*)$ as follows:

$$\begin{aligned} z_{n+1} &= \sigma_n r_{n+1} \\ &= \sigma_n r_n - \rho_n A p_n, \end{aligned} \tag{18}$$

$$\begin{aligned} z_{n+1}^* &= \overline{\sigma}_n r_{n+1}^* \\ &= \overline{\sigma}_n r_n^* - \overline{\rho}_n A^H p_n^*, \end{aligned} \tag{19}$$

which are then used to look for the iterates $x_{n+2} \in x_0 + \mathcal{K}_{n+2}(A, r_0)$ and $x_{n+2}^* \in x_0^* + \mathcal{K}_{n+2}(A^H, r_0^*)$ in step $(n+2)$ as follows:

$$\begin{aligned} x_{n+2} &= x_n + [p_n, z_{n+1}]\, f_n, \\ x_{n+2}^* &= x_n^* + [p_n^*, z_{n+1}^*]\, f_n^*, \end{aligned} \tag{20}$$

where $f_n, f_n^* \in \mathbb{C}^2$. Correspondingly, the $(n+2)$th primary residual $r_{n+2} \in \mathcal{K}_{n+3}(A, r_0)$ and shadow residual $r_{n+2}^* \in \mathcal{K}_{n+3}(A^H, r_0^*)$ are, respectively, computed as

$$r_{n+2} = r_n - A\left[p_n, z_{n+1}\right] f_n, \tag{21}$$

$$r_{n+2}^* = r_n^* - A^H\left[p_n^*, z_{n+1}^*\right] f_n^*. \tag{22}$$

The biconjugate A-orthogonality condition between the BiCOR primary residuals and shadow residuals shown as Property (2) in Proposition 5 requires

$$\begin{aligned} \left\langle \left[p_n^*, z_{n+1}^*\right], Ar_{n+2} \right\rangle &= 0, \\ \left\langle \left[p_n, z_{n+1}\right], A^H r_{n+2}^* \right\rangle &= 0, \end{aligned} \tag{23}$$

combining with (21) and (22) gives rise to the following two 2×2 systems of linear equations for, respectively, solving f_n and f_n^* :

$$\begin{aligned} &\begin{bmatrix} \left\langle A^H p_n^*, Ap_n \right\rangle & \left\langle A^H p_n^*, Az_{n+1} \right\rangle \\ \left\langle A^H z_{n+1}^*, Ap_n \right\rangle & \left\langle A^H z_{n+1}^*, Az_{n+1} \right\rangle \end{bmatrix} \begin{bmatrix} f_n^{(1)} \\ f_n^{(2)} \end{bmatrix} \\ &\quad = \begin{bmatrix} \left\langle p_n^*, Ar_n \right\rangle \\ \left\langle z_{n+1}^*, Ar_n \right\rangle \end{bmatrix}, \end{aligned} \tag{24}$$

$$\begin{aligned} &\begin{bmatrix} \left\langle Ap_n, A^H p_n^* \right\rangle & \left\langle Ap_n, A^H z_{n+1}^* \right\rangle \\ \left\langle Az_{n+1}, A^H p_n^* \right\rangle & \left\langle Az_{n+1}, A^H z_{n+1}^* \right\rangle \end{bmatrix} \begin{bmatrix} f_n^{*(1)} \\ f_n^{*(2)} \end{bmatrix} \\ &\quad = \begin{bmatrix} \left\langle Ap_n, r_n^* \right\rangle \\ \left\langle Az_{n+1}, r_n^* \right\rangle \end{bmatrix}. \end{aligned} \tag{25}$$

Similarly, the $(n+2)$th primary search direction vector $p_{n+2} \in \mathcal{K}_{n+3}(A, r_0)$ and shadow search direction vector $p_{n+2}^* \in \mathcal{K}_{n+3}(A^H, r_0^*)$ in a composite step are computed with the following form:

$$p_{n+2} = r_{n+2} + \left[p_n, z_{n+1}\right] g_n, \tag{26}$$

$$p_{n+2}^* = r_{n+2}^* + \left[p_n^*, z_{n+1}^*\right] g_n^*, \tag{27}$$

where $g_n, g_n^* \in \mathbb{C}^2$.

The biconjugate A^2-orthogonality condition between the BiCOR primary search direction vectors and shadow search direction vectors shown as Property (3) in Proposition 5 requires

$$\begin{aligned} \left\langle \left[p_n^*, z_{n+1}^*\right], A^2 p_{n+2} \right\rangle &= 0, \\ \left\langle \left[p_n, z_{n+1}\right], \left(A^H\right)^2 p_{n+2}^* \right\rangle &= 0, \end{aligned} \tag{28}$$

combining with (26) and (27) results in the following two 2×2 systems of linear equations for respectively solving g_n and g_n^* :

$$\begin{aligned} &\begin{bmatrix} \left\langle A^H p_n^*, Ap_n \right\rangle & \left\langle A^H p_n^*, Az_{n+1} \right\rangle \\ \left\langle A^H z_{n+1}^*, Ap_n \right\rangle & \left\langle A^H z_{n+1}^*, Az_{n+1} \right\rangle \end{bmatrix} \begin{bmatrix} g_n^{(1)} \\ g_n^{(2)} \end{bmatrix} \\ &\quad = -\begin{bmatrix} \left\langle A^H p_n^*, Ar_{n+2} \right\rangle \\ \left\langle A^H z_{n+1}^*, Ar_{n+2} \right\rangle \end{bmatrix}, \end{aligned} \tag{29}$$

$$\begin{aligned} &\begin{bmatrix} \left\langle Ap_n, A^H p_n^* \right\rangle & \left\langle Ap_n, A^H z_{n+1}^* \right\rangle \\ \left\langle Az_{n+1}, A^H p_n^* \right\rangle & \left\langle Az_{n+1}, A^H z_{n+1}^* \right\rangle \end{bmatrix} \begin{bmatrix} g_n^{*(1)} \\ g_n^{*(2)} \end{bmatrix} \\ &\quad = -\begin{bmatrix} \left\langle Ap_n, A^H r_{n+2}^* \right\rangle \\ \left\langle Az_{n+1}, A^H r_{n+2}^* \right\rangle \end{bmatrix}. \end{aligned} \tag{30}$$

Therefore, it could be able to advance from step n to step $(n+2)$ to provide x_{n+2}, r_{n+2}, x_{n+2}^*, r_{n+2}^*, p_{n+2}, p_{n+2}^* by solving the above four 2×2 linear systems represented as in (24), (25), (29), and (30). With an appropriate combination of 1×1 and 2×2 steps, the CSBiCOR method can be simply obtained with only a minor modification to the usual implementation of the BiCOR method. Before explicitly presenting the algorithm, some properties of the CSBiCOR method will be given in the following section.

5. Some Properties of the CSBiCOR Method

In order to show the use of 2×2 steps is sufficient for CSBiCOR method to compute exactly all those well-defined BiCOR iterates stably provided that the underlying biconjugate A-orthonormalization procedure depicted as in Algorithm 1 does not break down, it is necessary to establish some lemmas similar to those for the CSBCG method [19].

Lemma 6. *Using either a 1×1 or a 2×2 step to obtain p_n, p_n^*, the following hold:*

$$\begin{aligned} \left\langle A^H r_n^*, Ap_n \right\rangle &= \left\langle A^H p_n^*, Ar_n \right\rangle \\ &= \left\langle A^H p_n^*, Ap_n \right\rangle = \left\langle q_n^*, q_n \right\rangle \equiv \sigma_n. \end{aligned} \tag{31}$$

Proof. If p_n, p_n^* are obtained via a 1×1 step, that is, $p_n = r_n + \beta_n p_{n-1}$, $p_n^* = r_n^* + \overline{\beta}_n p_{n-1}^*$ as shown in lines 15 and 16 of Algorithm 2, then with the biconjugate A^2-orthogonality condition between the BiCOR primary search direction vectors and shadow search direction vectors shown as Property (3) in Proposition 5, it follows that

$$\begin{aligned} \left\langle A^H r_n^*, Ap_n \right\rangle &= \left\langle A^H \left(p_n^* - \overline{\beta}_n p_{n-1}^*\right), Ap_n \right\rangle \\ &= \left\langle A^H p_n^*, Ap_n \right\rangle - \beta_n \left\langle A^H p_{n-1}^*, Ap_n \right\rangle \\ &= \left\langle A^H p_n^*, Ap_n \right\rangle \\ &= \left\langle q_n^*, q_n \right\rangle \equiv \sigma_n, \end{aligned}$$

$$\begin{aligned}\left\langle A^H p_n^*, Ar_n\right\rangle &= \left\langle A^H p_n^*, A\left(p_n - \beta_n p_{n-1}\right)\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle - \beta_n \left\langle A^H p_n^*, Ap_{n-1}\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle \\ &= \left\langle q_n^*, q_n\right\rangle \equiv \sigma_n.\end{aligned} \tag{32}$$

If p_n, p_n^* are obtained via a 2×2 step, that is, $p_n = r_n + g_{n-2}^{(1)} p_{n-2} + g_{n-2}^{(2)} z_{n-1}$, $p_n^* = r_n^* + g_{n-2}^{*(1)} p_{n-2}^* + g_{n-2}^{*(2)} z_{n-1}^*$ as presented in (26) and (27), then according to the biconjugate A^2-orthogonality conditions mentioned above for deriving p_n, p_n^* with a 2×2 step, it follows that

$$\begin{aligned}\left\langle A^H r_n^*, Ap_n\right\rangle &= \left\langle A^H \left(p_n^* - g_{n-2}^{*(1)} p_{n-2}^* - g_{n-2}^{*(2)} z_{n-1}^*\right), Ap_n\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle - \overline{g_{n-2}^{*(1)}} \left\langle A^H p_{n-2}^*, Ap_n\right\rangle \\ &\quad - \overline{g_{n-2}^{*(2)}} \left\langle A^H z_{n-1}^*, Ap_n\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle \\ &= \left\langle q_n^*, q_n\right\rangle \equiv \sigma_n, \\ \left\langle A^H p_n^*, Ar_n\right\rangle &= \left\langle A^H p_n^*, A\left(p_n - g_{n-2}^{(1)} p_{n-2} - g_{n-2}^{(2)} z_{n-1}\right)\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle - g_{n-2}^{(1)} \left\langle A^H p_n^*, Ap_{n-2}\right\rangle \\ &\quad - g_{n-2}^{(2)} \left\langle A^H p_n^*, Az_{n-1}\right\rangle \\ &= \left\langle A^H p_n^*, Ap_n\right\rangle \\ &= \left\langle q_n^*, q_n\right\rangle \equiv \sigma_n.\end{aligned} \tag{33}$$

□

With the above lemma, we can show the relationships among the four 2×2 coefficient matrices of the linear systems represented as in (24), (25), (29), and (30).

Lemma 7. *The four 2×2 coefficient matrices of the linear systems represented as in* (24), (25), (29), *and* (30) *have the following properties and relationships.*

(1) *The coefficient matrices in* (24) *and* (29) *are identical, so are those matrices in* (25) *and* (30).

(2) *All the coefficient matrices involved are symmetric.*

(3) *The coefficient matrices in* (24) *and* (25) *(correspondingly in* (29) *and* (30)*) are Hermitian to each other.*

Proof. The first relation is obvious by observation of the corresponding coefficient matrices in (24) and (29) (correspondingly in (25) and (30)).

For proving the second property, it suffices to show $\langle A^H z_{n+1}^*, Ap_n\rangle = \langle A^H p_n^*, Az_{n+1}\rangle$. From the presentations of z_{n+1} and z_{n+1}^* as respectively defined in (18) and (19), and following from Lemma 6, we have

$$\begin{aligned}\left\langle A^H z_{n+1}^*, Ap_n\right\rangle &= \left\langle A^H \left(\overline{\sigma}_n r_n^* - \overline{\rho}_n A^H p_n^*\right), Ap_n\right\rangle \\ &= \sigma_n \left\langle A^H r_n^*, Ap_n\right\rangle - \rho_n \left\langle \left(A^H\right)^2 p_n^*, Ap_n\right\rangle \\ &= \sigma_n^2 - \rho_n \left\langle \left(A^H\right)^2 p_n^*, Ap_n\right\rangle, \\ \left\langle A^H p_n^*, Az_{n+1}\right\rangle &= \left\langle A^H p_n^*, A\left(\sigma_n r_n - \rho_n Ap_n\right)\right\rangle \\ &= \sigma_n \left\langle A^H p_n^*, Ar_n\right\rangle - \rho_n \left\langle A^H p_n^*, A^2 p_n\right\rangle \\ &= \sigma_n^2 - \rho_n \left\langle \left(A^H\right)^2 p_n^*, Ap_n\right\rangle.\end{aligned} \tag{34}$$

The third fact directly comes from the Hermitian property of the standard Hermitian inner product of two complex vectors defined at the end of the first section; see, for instance, [6]. □

It is noticed that similar Hermitian relationships also hold for the corresponding matriced involved in (1), (2), (3) and (4) in [19] when the CSBCG method is applied in complex cases.

Now, an alternative result stating that a 2×2 step is always sufficient to skip over the pivot breakdown of the BiCOR method when $\sigma_n = 0$, just as stated at the end of Section 2, can be shown in the next lemma.

Lemma 8. *Suppose that the biconjugate A-orthonormalization procedure depicted as in Algorithm 1 underlying the BiCOR method does not breakdown; that is the situation where $\rho_i \neq 0$, $i = 0, 1, \ldots, n$ and $\langle z_{n+1}^*, Az_{n+1}\rangle \neq 0$. If $\sigma_n \equiv \langle q_n^*, q_n\rangle \equiv \langle A^H p_n^*, Ap_n\rangle = 0$, then the 2×2 coefficient matrices in* (24), (25), (29), *and* (30) *are nonsingular.*

Proof. It suffices to show that if $\sigma_n \equiv \langle q_n^*, q_n\rangle \equiv \langle A^H p_n^*, Ap_n\rangle = 0$, then $\langle A^H z_{n+1}^*, Ap_n\rangle \neq 0$. From Lemma 7 and (18), we have

$$\begin{aligned}\left\langle A^H z_{n+1}^*, Ap_n\right\rangle &= \left\langle A^H p_n^*, Az_{n+1}\right\rangle, \\ \left\langle A^H z_{n+1}^*, Ap_n\right\rangle &= \left\langle A^H z_{n+1}^*, -\frac{z_{n+1}}{\rho_n}\right\rangle \\ &= -\frac{1}{\rho_n} \left\langle z_{n+1}^*, Az_{n+1}\right\rangle \neq 0.\end{aligned} \tag{35}$$

□

The next proposition shows some properties of the CSBiCOR iterates similar to those of the BiCOR method presented in Proposition 5 with some minor changes.

Proposition 9. *For the CSBiCOR algorithm, let $R_{n+1} = [r_0, r_1, \ldots, r_n]$, $R_{n+1}^* = [r_0^*, r_1^*, \ldots, r_n^*]$, where r_i, r_i^* are replaced by z_i, z_i^* appropriately if the ith step is a composite 2×2 step; and let $P_{n+1} = [p_0, p_1, \ldots, p_n]$, $P_{n+1}^* = [p_0^*, p_1^*, \ldots, p_n^*]$. Assuming the underlying biconjugate A-orthonormalization procedure does not break down, one has the following properties.*

(1) $Range(R_{n+1}) = Range(P_{n+1}) = \mathscr{K}_{n+1}(A, r_0)$, $Range(R^*_{n+1}) = Range(P^*_{n+1}) = \mathscr{K}_{n+1}(A^H, r^*_0)$.

(2) $R^{*H}_{n+1} A R_{n+1}$ *is diagonal.*

(3) $P^{*H}_{n+1} A^2 P_{n+1} = \mathrm{diag}(D_k)$, *where* D_k *is either of order* 1×1 *or* 2×2 *and the sum of the dimensions of* D_k*'s is* $(n+1)$.

Proof. The properties can be proved with the same frame for those of Theorem 4.4 in [19]. □

Therefore, we can show with the above properties that the use of 2×2 steps is sufficient for the CSBiCOR method to compute exactly all those well-defined BiCOR iterates stably provided that the underlying biconjugate A-orthonormalization procedure depicted as in Algorithm 1 does not break down.

Proposition 10. *The CSBiCOR method is able to compute exactly all those well-defined BiCOR iterates stably without pivot breakdown assuming that the underlying biconjugate A-orthonormalization procedure does not break down.*

Proof. By construction, $x_{n+2} = x_n + f^{(1)}_n p_n + f^{(2)}_n z_{n+1}$; by induction, $x_n \in x_0 + \mathscr{K}_n(A, r_0)$ and $p_n \in \mathscr{K}_{n+1}(A, r_0)$; by definition, $z_{n+1} \in \mathscr{K}_{n+2}(A, r_0)$. Then it follows that $x_{n+2} \in x_0 + \mathscr{K}_{n+2}(A, r_0)$. From Proposition 9, we have $r_{n+2} \perp A^H \mathscr{K}_{n+2}(A^H, r^*_0)$. Therefore, x_{n+2} exactly satisfies the Petrov-Galerkin condition defining the BiCOR iterates. □

Before ending this section, we present a best approximation result for the CSBiCOR method, which uses the same framework as that for the CSBCG method [19]. For discussion about convergence results for Krylov subspaces methods, refer to [2, 6, 32, 33] and the references therein. To the best of our knowledge, this is the first attempt to study the convergence result for the BiCOR method [7].

First let $\mathscr{V}_m = \mathrm{span}(v_1, v_2, \ldots, v_m)$ and $\mathscr{W}_m = \mathrm{span}(w_1, w_2, \ldots, w_m)$ denote the Krylov subspaces generated by the biconjugate A-orthonormalization procedure after running Algorithm 1 m steps. The norms associated with the spaces $\mathscr{V}_N \equiv \mathbb{C}^N$ and $\mathscr{W}_N \equiv \mathbb{C}^N$ are defined as

$$\|v\|^2_r = v^H M_r v, \qquad \|w\|^2_l = w^H M_l w, \tag{36}$$

where, $v, w \in \mathbb{C}^N$ while M_r and M_l are symmetric positive definite matrices. Then we have the following proposition with the initial guess x_0 of the linear system being $\mathbf{0} \in \mathbb{C}^N$. The case with a nonzero initial guess can be treated adaptively.

Proposition 11. *Suppose that for all* $v \in \mathscr{V}_N$ *and for all* $w \in \mathscr{W}_N$, *one has*

$$\left| w^H A^2 v \right| \le \Gamma \|v\|_r \|w\|_l, \tag{37}$$

where, Γ *is a constant independent of* v *and* w. *Furthermore, assuming that for those steps in the composite step Biconjugate A-Orthogonal Residual (CSBiCOR) method in which we compute an approximation* x_m, *we have*

$$\inf_{\substack{v \in \mathscr{V}_m \\ \|v\|_r = 1}} \sup_{\substack{w \in \mathscr{W}_m \\ \|w\|_l \le 1}} w^H A^2 v \ge \gamma_k \ge \gamma > 0. \tag{38}$$

Then,

$$\left\| x - x_m \right\|_r \le \left(1 + \frac{\Gamma}{\gamma}\right) \inf_{v \in \mathscr{V}_m} \|x - v\|_r. \tag{39}$$

Proof. From the Petrov-Galerkin condition (15), we have for $w \in \mathscr{W}_m$,

$$\left(A^H w\right)^H (Ax - Ax_m) = w^H A^2 (x - x_m) = 0, \tag{40}$$

yielding with arbitrary $v \in \mathscr{V}_m$

$$w^H A^2 (x_m - v) = w^H A^2 (x - v). \tag{41}$$

It is noted that $x_m - v \in \mathscr{V}_m$ because of the assumption of the zero initial guess. Then taking the sup of both sides of the above equation for all $\|w\|_l \le 1$ with the inf-sup condition (38) to bound the left-hand side and the continuity assumption (37) to bound the right-hand side results in

$$\gamma_k \left\| x_m - v \right\|_r \le \Gamma \|x - v\|_r \|w\|_l \le \Gamma \|x - v\|_r, \tag{42}$$

which yields

$$\left\| x - x_m \right\|_r \le \|x - v\|_r + \left\| x_m - v \right\|_r \le \left(1 + \frac{\Gamma}{\gamma}\right) \|x - v\|_r \tag{43}$$

with the triangle inequality. The final assertion (39) follows immediately since $v \in \mathscr{V}_m$ is arbitrary. □

6. Implementation Issues

For the purpose of conveniently and simply presenting the CSBiCOR method, we first prove some relations simplifying the solutions of the four 2×2 linear systems represented as in (24), (25), (29), and (30).

Lemma 12. *For the four* 2×2 *linear systems represented as in* (24), (25), (29), *and* (30), *the following relations hold:*

(1) $\langle p^*_n, Ar_n \rangle = \overline{\langle Ap_n, r^*_n \rangle} = \langle r^*_n, Ar_n \rangle = \langle r^*_n, \widehat{r}_n \rangle \equiv \rho_n$ *and* $\langle z^*_{n+1}, Ar_n \rangle = \langle Az_{n+1}, r^*_n \rangle = 0$, $f_n = \overline{f}^*_n$.

(2) $\langle A^H p^*_n, Ar_{n+2} \rangle = \overline{\langle Ap_n, A^H r^*_{n+2} \rangle} = 0$ *and* $\langle A^H z^*_{n+1}, Ar_{n+2} \rangle = \overline{\langle Az_{n+1}, A^H r^*_{n+2} \rangle} = -\rho_{n+2}/f^{(2)}_n$, *where* $\rho_{n+2} \equiv \langle r^*_{n+2}, Ar_{n+2} \rangle$, $g_n = \overline{g}^*_n$.

(3) $\langle A^H p^*_n, Az_{n+1} \rangle = \overline{\langle Ap_n, A^H z^*_{n+1} \rangle} = -\theta_{n+1}/\rho_n$, *where* $\theta_{n+1} \equiv \langle z^*_{n+1}, Az_{n+1} \rangle$.

Proof. (1) If a 1×1 step is taken, then $p^*_n = r^*_n + \overline{\beta}_n p^*_{n-1}$, giving that $\langle p^*_n, Ar_n \rangle = \langle r^*_n + \overline{\beta}_n p^*_{n-1}, Ar_n \rangle = \langle r^*_n, Ar_n \rangle + \beta_n \langle p^*_{n-1}, Ar_n \rangle = \langle r^*_n, Ar_n \rangle$ because of the last-term

vanishment due to Proposition 9. Analogously, the iteration $p_n = r_n + \beta_n p_{n-1}$ similarly gives $\langle Ap_n, r_n^* \rangle = \langle Ar_n + \beta_n Ap_{n-1}, r_n^* \rangle = \langle Ar_n, r_n^* \rangle + \overline{\beta}_n \langle Ap_{n-1}, r_n^* \rangle = \langle Ar_n, r_n^* \rangle = \overline{\langle r_n^*, Ar_n \rangle}$ from Proposition 9 and the Hermitian property of the standard Hermitian inner product. If a 2×2 step is taken, then by construction as shown in (27) and Proposition 9, we have $\langle p_n^*, Ar_n \rangle = \langle r_n^* + \overline{g_{n-2}^{*(1)}} p_{n-2}^* + \overline{g_{n-2}^{*(2)}} z_{n-1}^*, Ar_n \rangle = \langle r_n^*, Ar_n \rangle + \overline{g_{n-2}^{*(1)}} \langle p_{n-2}^*, Ar_n \rangle + \overline{g_{n-2}^{*(2)}} \langle z_{n-1}^*, Ar_n \rangle = \langle r_n^*, Ar_n \rangle$. Similarly, with Proposition 9 and the Hermitian property of the standard Hermitian inner product, it can be shown that $\langle Ap_n, r_n^* \rangle = \langle A(r_n + g_{n-2}^{(1)} p_{n-2} + g_{n-2}^{(2)} z_{n-1}), r_n^* \rangle = \langle Ar_n, r_n^* \rangle + g_{n-2}^{(1)} \langle Ap_{n-2}, r_n^* \rangle + g_{n-2}^{(2)} \langle Az_{n-1}, r_n^* \rangle = \langle Ar_n, r_n^* \rangle = \overline{\langle r_n^*, Ar_n \rangle}$. The equalities $\langle z_{n+1}^*, Ar_n \rangle = \langle Az_{n+1}, r_n^* \rangle = 0$ directly follow from Proposition 9. Finally, because the coefficient matrices of the symmetric systems in (24) and (25) respectively governing f_n and f_n^* are Hermitian to each other as shown in Lemma 7 while the corresponding right-hand sides are conjugate to each other as shown here, we can deduce that $f_n = \overline{f}_n^*$.

(2) From Proposition 9, it can directly verify that $\langle A^H p_n^*, Ar_{n+2} \rangle = \langle Ap_n, A^H r_{n+2}^* \rangle = 0$ with the fact that $Ap_n \in \mathcal{K}_{n+2}(A, r_0)$ and $A^H p_n^* \in \mathcal{K}_{n+2}(A^H, r_0^*)$. Next, it can be noted that if $\sigma_n = 0$, then $f_n^{(2)} \neq 0$. Otherwise, the first equation in the linear system (24) cannot hold for $\langle A^H p_n^*, Az_{n+1} \rangle \neq 0$ as proved in Lemma 8 and $\langle p_n^*, Ar_n \rangle = \rho_n \neq 0$ as shown in the above (1). Then by construction of (22) and Proposition 9, we have $\langle A^H z_{n+1}^*, Ar_{n+2} \rangle = \langle (1/f_n^{*(2)})(r_n^* - r_{n+2}^* - \overline{f_n^{*(1)}} A^H p_n^*), Ar_{n+2} \rangle = (1/\overline{f_n^{*(2)}})(\langle r_n^*, Ar_{n+2} \rangle - \langle r_{n+2}^*, A\overline{r_{n+2}} \rangle - \overline{f_n^{*(1)}} \langle A^H p_n^*, Ar_{n+2} \rangle) = (-1/\overline{f_n^{*(2)}}) \langle r_{n+2}^*, Ar_{n+2} \rangle = (-1/\overline{f_n^{*(2)}}) \rho_{n+2} = (-1/f_n^{(2)}) \rho_{n+2}$ with the fact that $f_n = \overline{f}_n^*$ proved in the above (1). By construction of (21) and Proposition 9, we can also show that $\langle Az_{n+1}, A^H r_{n+2}^* \rangle = \overline{-\rho_{n+2}/f_n^{(2)}}$. Similarly to the relationships between f_n and f_n^*, we can deduce that $g_n = \overline{g}_n^*$, because the coefficient matrices of the symmetric systems in (29) and (30) respectively governing g_n and g_n^* are Hermitian to each other as shown in Lemma 7 while the corresponding right-hand sides are conjugate to each other as shown here.

(3) By construction of (19) and Proposition 9, it follows that $\langle A^H p_n^*, Az_{n+1} \rangle = \langle (1/\overline{\rho}_n)(\overline{\sigma}_n r_n^* - z_{n+1}^*), Az_{n+1} \rangle = (1/\rho_n)(\sigma_n \langle r_n^*, Az_{n+1} \rangle - \langle z_{n+1}^*, Az_{n+1} \rangle) = (-1/\rho_n) \langle z_{n+1}^*, Az_{n+1} \rangle = -\theta_{n+1}/\rho_n$, where $\theta_{n+1} \equiv \langle z_{n+1}^*, Az_{n+1} \rangle$. Because it is already known that $\langle A^H z_{n+1}^*, Ap_n \rangle = \langle A^H p_n^*, Az_{n+1} \rangle$ as proved in Lemma 7, we have $\langle Ap_n, A^H z_{n+1}^* \rangle = \overline{\langle A^H z_{n+1}^*, Ap_n \rangle} = \overline{\langle A^H p_n^*, Az_{n+1} \rangle} = \overline{-\theta_{n+1}/\rho_n}$. The proof is completed. □

As a result of Lemmas 6 and 12, when a 2×2 step is taken, the solutions for f_n and g_n involved in the four linear systems represented as in (24), (25), (29), and (30) can be simplified only by solving the following two linear systems:

$$\begin{bmatrix} \sigma_n & -\dfrac{\theta_{n+1}}{\rho_n} \\ -\dfrac{\theta_{n+1}}{\rho_n} & \zeta_{n+1} \end{bmatrix} \begin{bmatrix} f_n^{(1)} \\ f_n^{(2)} \end{bmatrix} = \begin{bmatrix} \rho_n \\ 0 \end{bmatrix},$$

$$\begin{bmatrix} \sigma_n & -\dfrac{\theta_{n+1}}{\rho_n} \\ -\dfrac{\theta_{n+1}}{\rho_n} & \zeta_{n+1} \end{bmatrix} \begin{bmatrix} g_n^{(1)} \\ g_n^{(2)} \end{bmatrix} = \begin{bmatrix} 0 \\ \dfrac{\rho_{n+2}}{f_n^{(2)}} \end{bmatrix}, \tag{44}$$

where $\zeta_{n+1} = \langle A^H z_{n+1}^*, Az_{n+1} \rangle$.

It should be emphasized that these above two systems have the same representations for the CSBCG method [19] but differ in detailed computational formulae for the corresponding quantities because of the different underlying procedures mentioned in Section 1. As a result, f_n and g_n can be explicitly represented as

$$f_n = \frac{(\zeta_{n+1}\rho_n, \theta_{n+1})\, \rho_n^2}{\eta_n}, \qquad g_n = \left(\frac{\rho_{n+2}}{\rho_n}, \frac{\sigma_n \rho_{n+2}}{\theta_{n+1}} \right), \tag{45}$$

where $\eta_n \equiv \sigma_n \zeta_{n+1} \rho_n^2 - \theta_{n+1}^2$.

Combining all the recurrences discussed above for either a 1×1 step or a 2×2 step and taking the strategy of reducing the number of matrix-vector multiplications by introducing an auxiliary vector recurrence and changing variables adopted for the BiCOR method [7] as well as for the BiCR method [31], together lead to the CSBiCOR method. The pseudocode for the preconditioned CSBiCOR with a left preconditioner B can be represented by Algorithm 3, where s_{n+1} and z_{n+1} are used to respectively denote the unpreconditioned and the preconditioned residuals. It is obvious to note that the number of matrix-vector multiplications in this algorithm remains the same per step as in the BiCOR algorithm [7]. Therefore, the cost of the use of 2×2 composite steps is negligible. That is, 2×2 composite steps cost approximately twice as much as 1×1 steps just as stated for the CSBCG method [18]. Thus, from the point of view of choosing 1×1 or 2×2 steps, there is no significant difference with respect to algorithm cost. However, the presented algorithm is with the motivation of obtaining smoother and, hopefully, faster convergence behavior in comparison with the BiCOR method besides eliminating pivot breakdowns.

For the issue of deciding between 1×1 and 2×2 updates, the heuristic based on the magnitudes of the residuals developed for the CSBCG method [18] is borrowed for the CSBiCOR method in order to choose the step size which maximizes numerical stability as well as to avoid overflow. The principle is to avoid "spikes" in the convergence history of the norm of the residuals. A 2×2 update will be chosen if the following circumstance satisfies

$$\|r_{n+1}\| > \max\{\|r_n\|, \|r_{n+2}\|\}. \tag{46}$$

For completeness of algorithm implementation, we recall the test procedure as follows. Define $\pi_{n+2} \equiv \eta_n r_{n+2} = \eta_n r_n - \zeta_{n+1} \rho_n^3 Ap_n - \theta_{n+1} \rho_n^2 Az_{n+1}$. Then with the scaled versions of the corresponding quantities, the tests $\|r_{n+1}\| \leq \|r_n\|$ and

(1) Compute $r_0 = b - Ax_0$ for some initial guess x_0.
(2) Choose $r_0^* = P(A)\, r_0$ such that $\langle r_0^*, Ar_0 \rangle \neq 0$, where $P(t)$ is a polynomial in t.
(e.g., $r_0^* = Ar_0$). Set $p_0 = r_0$, $\widetilde{p}_0 = \widetilde{r}_0$, $q_0 = Ap_0$, $\widetilde{q}_0 = A^H \widetilde{p}_0$.
(3) Compute $\rho_0 = \langle \widetilde{r}_0, Ar_0 \rangle$.
(4) Begin LOOP ($n = 0, 1, 2, \ldots$)
(5) $\sigma_n = \langle \widetilde{q}_n, q_n \rangle$
(6) $s_{n+1} = \sigma_n r_n - \rho_n q_n$
(7) $\widetilde{s}_{n+1} = \overline{\sigma}_n \widetilde{r}_n - \overline{\rho}_n \widetilde{q}_n$
(8) $y_{n+1} = As_{n+1}$
(9) $\widetilde{y}_{n+1} = A^H \widetilde{s}_{n+1}$
(10) $\theta_{n+1} = \langle \widetilde{s}_{n+1}, y_{n+1} \rangle$
(11) $\zeta_{n+1} = \langle \widetilde{y}_{n+1}, y_{n+1} \rangle$
(12) **if** 1×1 step **then**
(13) $\alpha_n = \rho_n / \sigma_n$
(14) $\rho_{n+1} = \theta_{n+1} / \sigma_n^2$
(15) $\beta_{n+1} = \rho_{n+1} / \rho_n$
(16) $x_{n+1} = x_n + \alpha_n p_n$
(17) $r_{n+1} = r_n - \alpha_n q_n$
(18) $\widetilde{r}_{n+1} = \widetilde{r}_n - \overline{\alpha}_n \widetilde{q}_n$
(19) $p_{n+1} = s_{n+1} / \sigma_n + \beta_{n+1} p_n$
(20) $q_{n+1} = y_{n+1} / \sigma_n + \beta_{n+1} q_n$
(21) $\widetilde{q}_{n+1} = \widetilde{y}_{n+1} / \overline{\sigma}_n + \overline{\beta}_{n+1} \widetilde{q}_n$
(22) $n \leftarrow n + 1$
(23) **else**
(24) $\delta_n = \sigma_n \zeta_{n+1} \rho_n^2 - \theta_{n+1}^2$
(25) $\alpha_n = \zeta_{n+1} \rho_n^3 / \delta_n$
(26) $\alpha_{n+1} = \theta_{n+1} \rho_n^2 / \delta_n$
(27) $x_{n+2} = x_n + \alpha_n p_n + \alpha_{n+1} s_{n+1}$
(28) $r_{n+2} = r_n - \alpha_n q_n - \alpha_{n+1} y_{n+1}$
(29) $\widetilde{r}_{n+2} = \widetilde{r}_n - \overline{\alpha}_n \widetilde{q}_n - \overline{\alpha}_{n+1} \widetilde{y}_{n+1}$
(30) **solve** $Bz_{n+2} = r_{n+2}$
(31) **solve** $B^H \widetilde{z}_{n+2} = \widetilde{r}_{n+2}$
(32) $\widetilde{z}_{n+2} = Az_{n+2}$
(33) $\widehat{\widetilde{z}}_{n+2} = A^H \widetilde{z}_{n+2}$
(34) $\rho_{n+2} = \langle \widehat{\widetilde{z}}_{n+2}, r_{n+2} \rangle$
(35) $\beta_{n+1} = \rho_{n+2} / \rho_n$
(36) $\beta_{n+2} = \rho_{n+2} \sigma_n / \theta_{n+1}$
(37) $p_{n+2} = z_{n+2} + \beta_{n+1} p_n + \beta_{n+2} s_{n+1}$
(38) $q_{n+2} = \widetilde{z}_{n+2} + \overline{\beta}_{n+1} q_n + \overline{\beta}_{n+2} y_{n+1}$
(39) $\widetilde{q}_{n+2} = \widehat{\widetilde{z}}_{n+2} + \overline{\beta}_{n+1} \widetilde{q}_n + \overline{\beta}_{n+2} \widetilde{y}_{n+1}$
(40) $n \leftarrow n + 2$
(41) **end if**
(42) Check convergence; continue if necessary
(43) End LOOP

Algorithm 3: Left preconditioned CSBiCOR method.

$\|r_{n+2}\| \leq \|r_{n+1}\|$ can be, respectively, replaced by $\|z_{n+1}\| \leq |\sigma_n| \|r_n\|$ and $|\sigma_n| \|\pi_{n+2}\| \leq |\eta_n| \|z_{n+1}\|$. Consequently, the test can be implemented with the code fragment shown in Algorithm 4.

Analogously to the effect obtained by the CSBCG method [19] in circumstances where (46) is satisfied, the use of 2×2 updates for the CSBiCOR method is also able to cut off spikes in the convergence history of the norm of the residuals possibly caused by taking two 1×1 steps in such circumstances. The resulting CSBiCOR algorithm inherits the promising properties of the CSBCG method [18, 19]. That is, the composite strategy employed can eliminate near pivot breakdowns while can provide some smoothing of the convergence history of the norm of the residuals without involving any arbitrary machine-dependent or user-supplied tolerances. However, it should be emphasized that the CSBiCOR method could only eliminate those spikes due to small pivots in the upper left principal submatrices T_k ($1 \leq k \leq m$) of T_m formed in Proposition 1 but not all spikes. So that the residual norm of the CSBiCOR method does not decrease monotonically. By the way, learning from the mathematically equivalent but numerically different variants

If $||z_{n+1}|| \leq |\sigma_n|||r_n||$, Then
  1×1 Step
Else
  $||\pi_{n+2}|| = ||\eta_n r_n - \zeta_{n+1}\rho_n^3 q_n - \theta_{n+1}\rho_n^2 y_{n+1}||$
  If $|\sigma_n|||\pi_{n+2}|| \leq |\eta_n|||z_{n+1}||$, Then
    2×2 Step
  Else
    1×1 Step
  End If
End If

Algorithm 4

Table 1: Structures of the three sets of test problems.

Ex.	Group and name	id	# rows	# cols	Nonzeros	Sym
1	Dehghani/light_in_tissue	1873	29,282	29,282	406,084	0%
2	Kim/kim1	862	38,415	38,415	933,195	0%
3	HB/young1c	278	841	841	4,089	85%

of the CSBCG method [18], we could also develop such kinds of variants of the CSBiCOR method, which will be taken into further account.

7. Examples and Numerical Experiments

This section mainly focuses on the analysis of different numerical effects of the composite step strategy employed into the BiCOR method with, far from being exhaustive, a few typical circumstances of test problems as arising from electromagnetics, discretizations of 2D/3D physical domains, and acoustics, which are described in Table 1. All of them are borrowed in the MATLAB format from the University of Florida Sparse Matrix Collection provided by Davis [34], in which the meanings of the column headers of Table 1 can be found. The effect analysis part may provide some suggestions that when to make use of the composite step strategy should make significant progress towards an exact solution according to the convergence history of the residual norms. It is with the hope that the stabilizing effect of the CSBiCOR method could make the residual norm plot become smoother and, hopefully, faster decreasing since far outlying iterates and residuals are avoided in the smoothed sequences.

In a realistic setting, one would use a preconditioner. With appropriate preconditioning techniques, such as those existing well-established preconditioning methodologies [35–37] based on approximate inverses, all the following involved methods for numerical comparison are very attractive for solving relevant classes of non-Hermitian linear systems. But this is not the point we pursue here. All the experiments are performed *without preconditioning techniques* in default if without other clarification. That is, the preconditioner B in Algorithm 3 will be taken as the identity matrix except otherwise clarified. Refer to the survey by Benzi [38] and the book by Saad [6] on preconditioning techniques for improving the performance and reliability of Krylov subspace methods.

For all these test problems below, the BiCOR method will be implemented using the same code as for the CSBiCOR method with the pivot test being modified to always choose 1-step update steps for the purpose of conveniently showing the stabilizing effect of the composite step strategy on the BiCOR method. So does for the BCG method as implementated in [18, 19].

The experiments have been carried out with machine precision 10^{-16} in double precision floating point arithmetic in MATLAB 7.0.4 with a PC-Pentium (R) D CPU 3.00 GHz, 1 GB of RAM. We make comparisons in four aspects: number of iterations (referred to as Iters), CPU consuming time in seconds (referred to as CPU), $\log_{10}$ of the updated and final true relative residual 2-norms defined, respectively, as $\log_{10} \|r_n\|_2/\|r_0\|_2$ and $\log_{10} \|b - Ax_n\|_2/\|r_0\|_2$ (referred to as *Relres* and TRR). Iters takes the form of "$*/*$" recording the total number of iteration steps and the number of 2×2 iteration steps involved in the corresponding composite step methods. Numerical results in terms of Iters, CPU and TRR are reported by means of tables while convergence histories involved are shown in figures with Iters (on the horizontal axis) versus *Relres* (on the vertical axis). The stopping criterion used here is that the 2-norm of the residual be reduced by a factor (referred to as TOL) of the 2-norm of the initial residual, that is, $\|r_n\|_2/\|r_0\|_2 < \text{TOL}$, or when Iters exceeded the maximal iteration number (referred to as *MAXIT*). Here, we take $MAXIT = 500$. All these tests are started with an initial guess equal to $\mathbf{0} \in \mathbb{C}^n$. Whenever the considered problem contains no right-hand side to the original linear system $Ax = b$, let $b = Ae$, where e is the $n \times 1$ vector whose elements are all equal to unity, such that $x = (1, 1, \ldots, 1)^T$ is the exact solution. A symbol "$*$" is used to indicate that the method did not meet the required TOL before *MAXIT* or did not converge at all.

Table 2: Comparison results of Example 1 with TOL = 10^{-6}.

Method	BCG	BiCOR	CSBCG	CSBiCOR
Iters	330	318	327/3	318/0
CPU	40.7807	39.0256	40.1300	39.2116
TRR	−6.0318	−6.0150	−6.0318	−6.0150

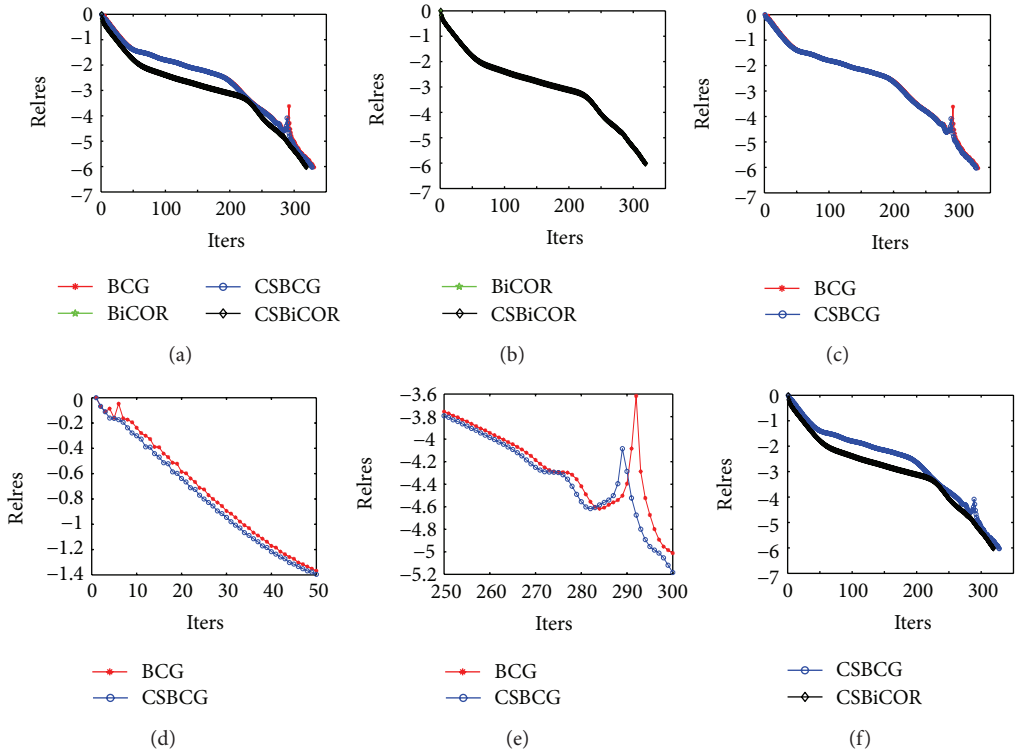

Figure 1: Convergence histories of Example 1 with TOL = 10^{-6}.

7.1. Example 1: Dehghani/Light_in_Tissue. The first example is one in which the BCG method and the BiCOR method have almost quite smooth convergence behaviors and as few as near breakdowns. The CSBiCOR method has exactly the same convergence behavior as the BiCOR method by investigating Table 2 and Figure 1(b). Under such circumstances when the convergence history of the norm of the residuals is smooth enough so that the composite step strategy does not gain a lot for the CSBiCOR method, the BiCOR method is much preferred and there is no need to employ the composite step strategy because the MATLAB implementation may be probably a little penalizing for the CSBiCOR code with respect to CPU. By the way, from Figure 1(c), the CSBCG method works as predicted in [18, 19], smoothing a little the convergence history of the BCG method. In particular, the CSBCG method computes a subset of the BCG iterates by clipping three "spikes" in the BCG history by means of the composite 2×2 steps from the particular investigations given in the bottom first two plots of Figures 1(d), 1(e) and Table 2.

Another interesting observation is that the good properties of the BiCOR method in terms of fast convergence speed and smooth convergence behavior in comparison with the BCG method are instinctually inherited by the CSBiCOR method so that the CSBiCOR method outperforms the CSBCG method in aspects of Iters and CPU as well as smooth effect, as shown in Table 2 and Figure 1(f).

7.2. Example 2: Kim/Kim1. This is a good test example to vividly demonstrate the efficacy of the composite step strategy as depicted in Figure 2. The BCG method, as observed in Figure 2(a), takes on many local "spikes" in the convergence curve. In such a case, the CSBCG method does a wonderful attempt to smooth the convergence history of the norm of the residuals as shown in Figure 2(c), although not leading to a monotonic decreasing convergence behavior. So does the CSBiCOR method to the BiCOR method by observation of Figure 2(d). Moreover, the CSBiCOR method seems to take less CPU than the BiCOR method while keeping approximately the same *TRR* when the BiCOR method is implemented using the same code as for the CSBiCOR method and the pivot test is modified to always choose 1-step update steps. Finally, the CSBiCOR method is again shown to be competitive to the CSBCG method as seen in Table 3 and Figure 2(b).

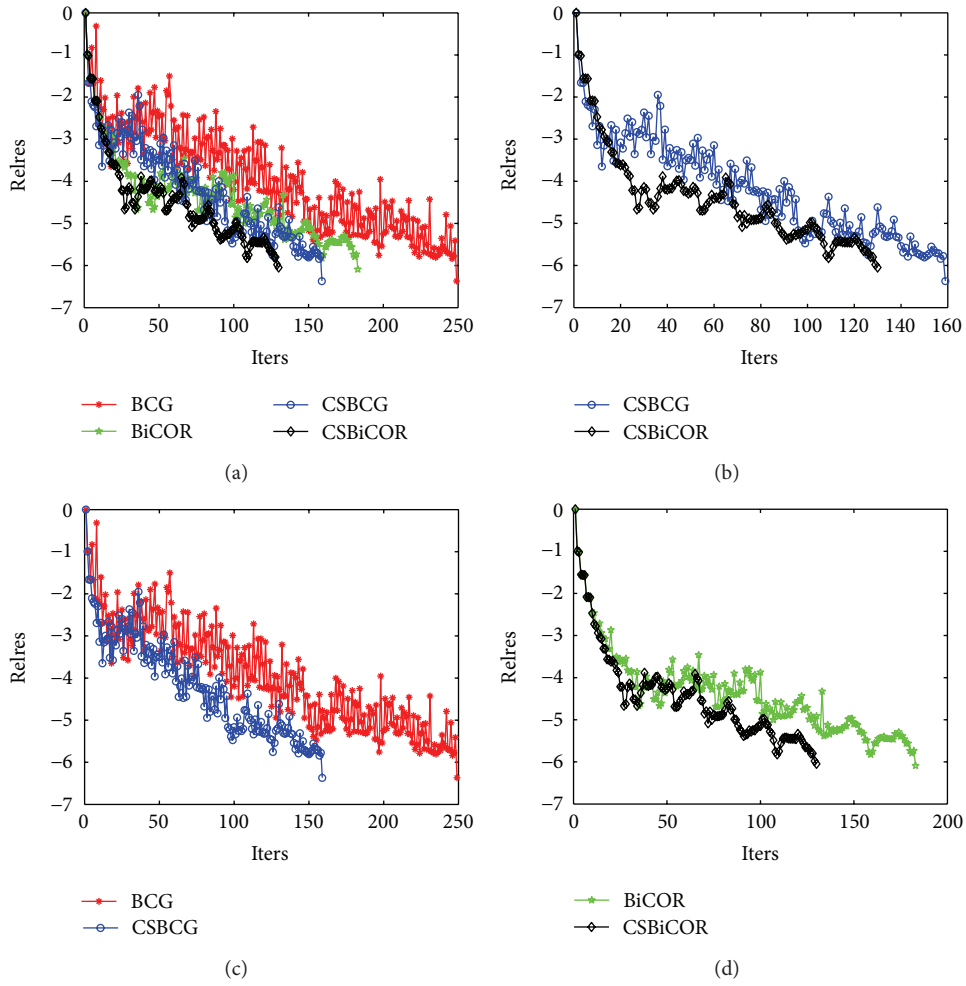

Figure 2: Convergence histories of Example 2 with TOL = 10^{-6}.

Table 3: Comparison results of Example 2 with TOL = 10^{-6}.

Method	BCG	BiCOR	CSBCG	CSBiCOR
Iters	248	182	158/90	129/56
CPU	53.8496	39.2517	43.3087	33.8528
TRR	−6.3680	−6.0909	−6.3680	−6.0483

7.3. Example 3: HB/Young1c. In the third example, the BCG and BiCOR methods have small irregularities and do not converge superlinearly, as reflected in Figure 3(a). The two methods above seem to have almost the same "asymptotical" speed of convergence while the BiCOR method seems a little smoother than the BCG method. From Figures 3(c) and 3(d), the composite step strategy seems to play a surprisingly good stabilizing effect to the BCG and BiCOR method. Furthermore, the CSBCG and CSBiCOR methods take less CPU, respectively, than the BCG and BiCOR methods as reported in Table 4. It is stressed that although the CSBiCOR method consumes a little more Iters and CPU than the CSBCG method, it presents a little more smooth convergence behavior than the latter method as displayed in Figure 3(b). This is still thanks to the promising advantages of the empirically observed stability and fast convergence rate of the BiCOR method over the BCG method.

8. Concluding Remarks

We have presented a new interesting variant of the BiCOR method for solving non-Hermitian systems of linear equations. Our approach is naturally based on and inspired by the composite step strategy taken for the CSBCG method [18, 19]. It is noted that exact pivot breakdowns are rare in practice; however, near breakdowns could cause severe numerical instability. The resulting CSBiCOR method is both theoretically and numerically demonstrated to avoid near pivot breakdowns and compute all the well-defined BiCOR iterates

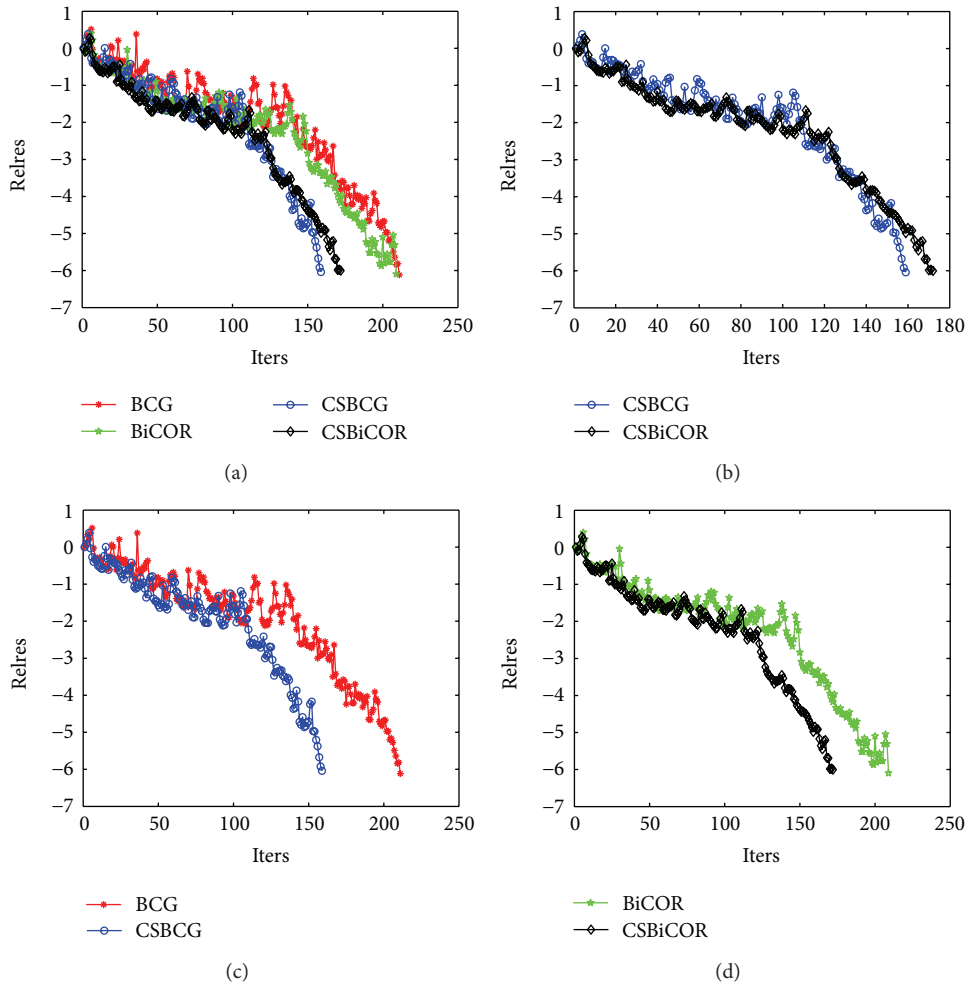

Figure 3: Convergence histories of Example 3 with TOL = 10^{-6}.

Table 4: Comparison results of Example 3 with TOL = 10^{-6}.

Method	BCG	BiCOR	CSBCG	CSBiCOR
Iters	210	208	158/52	171/41
CPU	0.6848	0.5974	0.5372	0.5573
TRR	−6.1155	−6.0984	−6.0389	−6.0017

stably with only minor modifications with the assumption that the underlying biconjugate A-orthonormalization procedure does not break down. Besides reducing the number of spikes in the convergence history of the norm of the residuals to the greatest extent, the CSBiCOR method could provide some further practically desired smoothing behavior towards stabilizing the behavior of the BiCOR method when it has erratic convergence behaviors. Additionally, the CSBiCOR method seems to be superior to the CSBCG method to some extent because of the inherited promising advantages of the empirically observed stability and fast convergence rate of the BiCOR method over the BCG method.

Since the BiCOR method is the most basic variant of the family of Lanczos biconjugate A-orthonormalization methods, its improvement will analogously lead to similar improvements for the CORS and BiCORSTAB methods, which is under investigation.

It is important to note the nonnegligible added complexity when deciding the composite step strategy. That is we have to make some extra computation before deciding to take a 2×2 step. Therefore, when the 2×2 step will not be taken after the decision test, it will lead to extra computation as well as CPU consuming time. It seems critical for us to optimize the code to implement the CSBiCOR method.

Acknowledgments

The authors would like to thank Professor Randolph E. Bank for showing us the "PLTMG" software package and for his insightful and beneficial discussions and suggestions.

Furthermore, the authors wish to acknowledge Professor Zhongxiao Jia and the anonymous referees for their suggestions in the presentation of this article. This research is supported by NSFC (60973015, 61170311, 11126103, 11201055, 11171054), the Specialized Research Fund for the Doctoral Program of Higher Education (20120185120026), Sichuan Province Sci. & Tech. Research Project (2011JY0002), and the Fundamental Research Funds for the Central Universities.

References

[1] Y. Saad and H. A. van der Vorst, "Iterative solution of linear systems in the 20th century," *Journal of Computational and Applied Mathematics*, vol. 43, pp. 1155–1174, 2005.

[2] V. Simoncini and D. B. Szyld, "Recent computational developments in Krylov subspace methods for linear systems," *Numerical Linear Algebra with Applications*, vol. 14, no. 1, pp. 1–59, 2007.

[3] J. Dongarra and F. Sullivan, "Guest editors'introduction to the top 10 algorithms," *Computer Science and Engineering*, vol. 2, pp. 22–23, 2000.

[4] B. Philippe and L. Reichel, "On the generation of Krylov subspace bases," *Applied Numerical Mathematics*, vol. 62, no. 9, pp. 1171–1186, 2012.

[5] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, vol. 17 of *Frontiers in Applied Mathematics*, SIAM, Philadelphia, Pa, USA, 1997.

[6] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, Pa, USA, 2nd edition, 2003.

[7] Y.-F. Jing, T.-Z. Huang, Y. Zhang et al., "Lanczos-type variants of the COCR method for complex nonsymmetric linear systems," *Journal of Computational Physics*, vol. 228, no. 17, pp. 6376–6394, 2009.

[8] Y.-F. Jing, B. Carpentieri, and T.-Z. Huang, "Experiments with Lanczos biconjugate A-orthonormalization methods for MoM discretizations of Maxwell's equations," *Progress in Electromagnetics Research*, vol. 99, pp. 427–451, 2009.

[9] Y.-F. Jing, T.-Z. Huang, Y. Duan, and B. Carpentieri, "A comparative studyof iterative solutions to linear systems arising in quantum mechanics," *Journal of Computational Physics*, vol. 229, no. 22, pp. 8511–8520, 2010.

[10] B. Carpentieri, Y.-F. Jing, and T.-Z. Huang, "The BICOR and CORS iterative algorithms for solving nonsymmetric linear systems," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 3020–3036, 2011.

[11] R. Fletcher, "Conjugate gradient methods for indefinite systems," in *Numerical Analysis (Proc 6th Biennial Dundee Conf., Univ. Dundee, Dundee, 1975)*, vol. 506 of *Lecture Notes in Mathematics*, pp. 73–89, Springer, Berlin, Germany, 1976.

[12] W. Joubert, *Generalized conjugate gradient and lanczos methods for the solution of nonsymmetric systems of linear equations [Ph.D. thesis]*, University of Texas, Austin, Tex, USA, 1990.

[13] W. Joubert, "Lanczos methods for the solution of nonsymmetric systems of linear equations," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 3, pp. 926–943, 1992.

[14] M. H. Gutknecht, "A completed theory of the unsymmetric Lanczos process and related algorithms. I," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 2, pp. 594–639, 1992.

[15] M. H. Gutknecht, "A completed theory of the unsymmetric Lanczos process and related algorithms. II," *SIAM Journal on Matrix Analysis and Applications*, vol. 15, no. 1, pp. 15–58, 1994.

[16] M. H. Gutknecht, "Block Krylov space methods for linear systems withmultiple right-hand sides: an introduction," in *Modern Mathematical Models, Methods and Algorithms for Real World Systems*, A. H. Siddiqi, I. S. Duff, and O. Christensen, Eds., Anamaya Publishers, New Delhi, India, 2006.

[17] D. G. Luenberger, "Hyperbolic pairs in the method of conjugate gradients," *SIAM Journal on Applied Mathematics*, vol. 17, pp. 1263–1267, 1969.

[18] R. E. Bank and T. F. Chan, "An analysis of the composite step biconjugate gradient method," *Numerische Mathematik*, vol. 66, no. 3, pp. 295–319, 1993.

[19] R. E. Bank and T. F. Chan, "A composite step bi-conjugate gradient algorithm for nonsymmetric linear systems," *Numerical Algorithms*, vol. 7, no. 1, pp. 1–16, 1994.

[20] R. W. Freund and N. M. Nachtigal, "QMR: a quasi-minimal residual method for non-Hermitian linear systems," *Numerische Mathematik*, vol. 60, no. 3, pp. 315–339, 1991.

[21] M. H. Gutknecht, "The unsymmetric Lanczos algorithms and their relations to Pade approximation, continued fraction and the QD algorithm," in *Proc. Copper Mountain Conference on Iterative Methods*, Book 2, Breckenridge Co., Breckenridge, Colo, USA, 1990.

[22] B. N. Parlett, D. R. Taylor, and Z. A. Liu, "A look-ahead Lánczos algorithm for unsymmetric matrices," *Mathematics of Computation*, vol. 44, no. 169, pp. 105–124, 1985.

[23] B. N. Parlett, "Reduction to tridiagonal form and minimal realizations," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 2, pp. 567–593, 1992.

[24] C. Brezinski, M. Redivo Zaglia, and H. Sadok, "Avoiding breakdown and near-breakdown in Lanczos type algorithms," *Numerical Algorithms*, vol. 1, no. 3, pp. 261–284, 1991.

[25] C. Brezinski, M. Redivo Zaglia, and H. Sadok, "A breakdown-free Lanczos type algorithm for solving linear systems," *Numerische Mathematik*, vol. 63, no. 1, pp. 29–38, 1992.

[26] C. Brezinski, M. Redivo-Zaglia, and H. Sadok, "Breakdowns in the implementation of the Lánczos method for solving linear systems," *Computers & Mathematics with Applications*, vol. 33, no. 1-2, pp. 31–44, 1997.

[27] C. Brezinski, M. R. Zaglia, and H. Sadok, "New look-ahead Lanczos-type algorithms for linear systems," *Numerische Mathematik*, vol. 83, no. 1, pp. 53–85, 1999.

[28] N. M. Nachtigal, *A look-ahead variant of the lanczos algorithm and its application to the quasi-minimal residual method for non-Hermitian linear systems [Ph.D. thesis]*, Massachusettes Institute of Technology, Cambridge, Mass, USA, 1991.

[29] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal, "An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices," *SIAM Journal on Scientific Computing*, vol. 14, no. 1, pp. 137–158, 1993.

[30] M. H. Gutknecht, "Lanczos-type solvers for nonsymmetric linear systems of equations," in *Acta Numerica, 1997*, vol. 6 of *Acta Numerica*, pp. 271–397, Cambridge University Press, Cambridge, UK, 1997.

[31] T. Sogabe, M. Sugihara, and S.-L. Zhang, "An extension of the conjugate residual method to nonsymmetric linear systems," *Journal of Computational and Applied Mathematics*, vol. 226, no. 1, pp. 103–113, 2009.

[32] P. Concus, G. H. Golub, and D. P. O'Leary, "A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations," in *Sparse Matrix Computations (Proc. Sympos., Argonne Nat. Lab., Lemont, Ill., 1975)*, pp. 309–332, Academic Press, New York, NY, USA, 1976.

[33] A. B. J. Kuijlaars, "Convergence analysis of Krylov subspace iterations with methods from potential theory," *SIAM Review*, vol. 48, no. 1, pp. 3–40, 2006.

[34] T. Davis, "The university of Florida sparse matrix collection," *NA Digest*, vol. 97, no. 23, 1997.

[35] T. Huckle, "Approximate sparsity patterns for the inverse of a matrix and preconditioning," *Applied Numerical Mathematics*, vol. 30, no. 2-3, pp. 291–303, 1999.

[36] G. A. Gravvanis, "Explicit approximate inverse preconditioning techniques," *Archives of Computational Methods in Engineering*, vol. 9, no. 4, pp. 371–402, 2002.

[37] B. Carpentieri, I. S. Duff, L. Giraud, and G. Sylvand, "Combining fast multipole techniques and an approximate inverse preconditioner for large electromagnetism calculations," *SIAM Journal on Scientific Computing*, vol. 27, no. 3, pp. 774–792, 2005.

[38] M. Benzi, "Preconditioning techniques for large linear systems: a survey," *Journal of Computational Physics*, vol. 182, no. 2, pp. 418–477, 2002.