Journal of Computational Mathematics
Vol.32, No.3, 2014, 248–265.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1401-CR13

GENERALIZED CONJUGATE A-ORTHOGONAL RESIDUAL SQUARED METHOD FOR COMPLEX NON-HERMITIAN LINEAR SYSTEMS*

Jianhua Zhang
Department of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
Department of Mathematics, Anhui Science and Technology University, Fengyang 233100, China
Email: zhangjhnuaa@gmail.com
Hua Dai
Department of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
Email: hdai@nuaa.edu.cn

Abstract

Recently numerous numerical experiments on realistic calculation have shown that the conjugate A-orthogonal residual squared (CORS) method is often competitive with other popular methods. However, the CORS method, like the CGS method, shows irregular convergence, especially appears large intermediate residual norm, which may lead to worse approximate solutions and slower convergence rate. In this paper, we present a new product-type method for solving complex non-Hermitian linear systems based on the biconjugate A-orthogonal residual (BiCOR) method, where one of the polynomials is a BiCOR polynomial, and the other is a BiCOR polynomial with the same degree corresponding to different initial residual. Numerical examples are given to illustrate the effectiveness of the proposed method.

Mathematics subject classification: 65F10.
Key words: Krylov subspace, BiCOR method, CORS method, Complex non-Hermitian linear systems.

1. Introduction

Some science and engineering applications, for instance in discretizing Helmhlotz and Maxwell equations, require the solution of large linear systems

$$Ax = b, \tag{1.1}$$

where A is an $N \times N$ complex non-Hermitian matrix and N is large.

In recent years, there have been many advances in Krylov subspace methods for solution of complex non-Hermitian linear systems, see, e.g., [1]. If storage requirement is not considered, the generalized minimal residual (GMRES) method [2] and its variant flexible GMRES (FGMRES) [3] are popular options due to their robustness and smooth convergence, see [4]. In terms of cheaper memory demanding, some of the short-recurrence methods based on Bi-Lanczos process are effective and competitive. The archetype of this class is the BiCG [5] method proposed by

* Received September 30, 2013 / Revised version received January 1, 2014 / Accepted January 15, 2014 / Published online May 22, 2014 /

Fletcher. However, the BiCG method requires the transpose of matrix, suffers from breakdown and converges irregularly. In order to avoid the transpose of matrix, Sonneveld developed the CGS [6] method by making use of the "wasted" extra matrix-vector multiplication. To overcome the erratic residual norms of the CGS method, van der Vorst derived the BiCGSTAB [7] method by incorporating linear minimal residual step at each iteration. Gutknecht, Sleijpen and Fokkema generalized this method to the BiCGSTAB2 [8] and BiCGSTAB(l) [9] methods, respectively. Zhang proposed the generalized product-type of the BiCG (GPBiCG) [10] method and provided a way to show the CGS, BiCGSTAB, BiCGSTAB2 and GPBiCG methods fit into a more general framework. ML(k)BiCGSTAB method [11] is also a BiCGSTAB variant based on multiple left Lanczos starting vectors. In [12], Freund considered an alternative approach and devised the TFQMR method by combining the CGS idea with quasi-minimal residual technique proposed in the QMR [13] method. However, they require many more iterations for some realistic problems [14] compared with the GMRES method.

Recently, Jing et al. [15,16] developed the BiCOR, CORS and BiCORSTAB methods for solving complex non-Hermitian linear system based on the biconjugate A-orthonormalization process, which are also considered as Lanczos-type variants of the conjugate A-orthogonal conjugate residual (COCR) method [17]. Note that an implementation of BiCOR-type methods can be constructed from any BiCG-type method by a formal B-inner product $\langle \widetilde{y}, y\rangle_B = \langle \widetilde{y}, By\rangle$ instead of the standard Hermitian inner product $\langle \widetilde{y}, y\rangle$. The choice of $B = A$ can lead to the BiCOR-type methods, see [18] for more details. As observed from different numerical experiments on some practical physical problems, including radar cross section (RCS) calculation from complex structures, acoustics problems, quantum mechanics and so on, these methods show competitive convergence behavior and are often superior to other Krylov subspaces, see [19-21] for details. In order to accelerate the convergence rate, Zhao and Huang [22] proposed the BiCORSTAB2 method. Under an unified generalized framwork, Zhao and Huang et al. deduced the generalized product-type BiCOR (GPBiCOR) method [23]. Numerical experiments from signal deconvolution show that the GPBiCOR method is effective.

The CORS method is more efficient than the restarted GMRES method on most selected examples, especially coming from realistic RCS calculation [16,21]. However, similarly to the CGS method [6], the CORS method often shows irregular convergence behavior and produces large intermediate residual during the iteration process, which badly affects its convergence rate and accuracy of approximate solutions. Inspired by the generalized CGS (GCGS) method [24], we develop a generalized CORS (GCORS) method which is a new product-type method based on the BiCOR method, where polynomial is products of two nearby BiCOR polynomials. Numerical examples show that this approach may lead to faster convergence as well as to more accurate results. We also show that the CORS and BiCORSTAB methods fit into the framework of the GCORS method.

The remainder of the paper is organized as follows. In Section 2, we give a brief description of the BiCOR method and its properties. The generalized CORS (GCORS) method is derived in Section 3. In Section 4, we present an efficient implementation of the GCORS method, which we will call the generalized CORS2 (GCORS2) method. Finally, numerical experiments are given in Section 5.

Throughout this paper, we use the follow notations. Let the overbar "-" denote the conjugate complex of a scalar, vector or matrix, Z^T and Z^H denote the transpose and the conjugate transpose of a vector or matrix Z, respectively. $\mathbb{P}_m$ denotes the set of complex polynomials $p_m(t)$ of degree m with scalar coefficients satisfying $p_m(0) = 1$. The inner product of two complex vectors $u, v \in \mathbb{C}^N$ is defined as

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

The Krylov subspace $\mathcal{K}_m(A, v)$ generated by a matrix $A \in \mathbb{C}^{N\times N}$ and a vector $v \in \mathbb{C}^N$ is defined as $\mathcal{K}_m(A, v) = span(v, Av, \cdots, A^{m-1}v)$.

2. The BiCOR Method

Now we give a brief description of the BiCOR method for solving the complex non-Hermitian systems. Jing et al. [15,16] proposed the BiCOR method based on the biconjugate A-orthonormalization process. Analogously to the BiCG method, we can get two Krylov subspaces $\mathcal{K}_j(A, r_0) = span(r_0, Ar_0, \cdots, A^{j-1}r_0)$ and $\mathcal{K}_j(A^H, r_0^*) = span(r_0^*, A^H r_0^*, \cdots, (A^H)^{j-1}r_0^*)$ in the BiCOR method. The approximate solution x_j is extracted from the affine subspace $x_0 + \mathcal{K}_j(A, r_0)$ such that $r_j \perp A^H\mathcal{K}_j(A^H, r_0^*)$. The following orthogonal relations in the BiCOR method hold [15]

$$r_j \perp A^H\mathcal{K}_j(A^H, r_0^*) \ \ and \ \ Ap_j \perp A^H\mathcal{K}_j(A^H, r_0^*) \ \ with \ \ r_0^* = p(A)r_0. \tag{2.1}$$

The residual and direction vectors generated during the BiCOR iteration steps satisfy the following properties [15, 29].

Proposition 1. *Let* $R_{n+1} = [r_0, r_1, \cdots, r_n]$, $R_{n+1}^* = [r_0^*, r_1^*, \cdots, r_n^*]$, $P_{n+1} = [p_0, p_1, \cdots, p_n]$, $P_{n+1}^* = [p_0^*, p_1^*, \cdots, p_n^*]$. *Then*

1. $Range(R_{n+1}) = Range(P_{n+1}) = \mathcal{K}_{n+1}(A, r_0)$,
2. $Range(R_{n+1}^*) = Range(P_{n+1}^*) = \mathcal{K}_{n+1}(A^H, r_0^*)$,
3. $(R_{n+1}^*)^H AR_{n+1}$ *is diagonal,*
4. $(P_{n+1}^*)^H A^2 P_{n+1}$ *is diagonal.*

Similarly to the BiCG method, there exist two possible kinds of breakdowns in the BiCOR method. Jing et al. [28,29] used the composite step technique [25-27] to handle the breakdowns of the BiCOR method.

As observed from the BiCOR method, the vectors r_j, p_j, r_j^*, p_j^* can be expressed as follows

$$r_j = \phi_j(A)r_0, p_j = \psi_j(A)r_0, \tag{2.2}$$
$$r_j^* = \phi_j(A^H)r_0^*, p_j^* = \psi_j(A^H)r_0^*, \tag{2.3}$$

where $\phi_j(t), \psi_j(t) \in \mathbb{P}_j$. These scalar polynomials are related by the following recurrence formulas

$$\begin{cases} \phi_{j+1}(t) = \phi_j(t) - \alpha_j t\psi_j(t), \\ \psi_{j+1}(t) = \phi_{j+1}(t) + \beta_{j+1}\psi_j(t). \end{cases} \tag{2.4}$$

Using the corresponding polynomial representations of residual and direction vectors, the iteration coefficients α_j and β_{j+1} can be computed as follows

$$\alpha_j = \frac{\langle r_j^*, Ar_j\rangle}{\langle A^H p_j^*, Ap_j\rangle} = \frac{\langle \phi_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}{\langle A^H\psi_j(A^H)r_0^*, A\psi_j(A)r_0\rangle} = \frac{\langle r_0^*, A\phi_j^2(A)r_0\rangle}{\langle r_0^*, A^2\psi_j^2(A)r_0\rangle} \tag{2.5}$$

and

$$\beta_{j+1} = \frac{\langle r_{j+1}^*, Ar_{j+1}\rangle}{\langle r_j^*, Ar_j\rangle} = \frac{\langle \phi_{j+1}(A^H)r_0^*, A\phi_{j+1}(A)r_0\rangle}{\langle \phi_j(A^H)r_0^*, A\phi_j(A)r_0\rangle} = \frac{\langle r_0^*, A\phi_{j+1}^2(A)r_0\rangle}{\langle r_0^*, A\phi_j^2(A)r_0\rangle}, \tag{2.6}$$

which lead to the CORS method. See [15,16] for more details.

3. The Generalized CORS Method

In this section, we construct a new product-type method based on the BiCOR method. Its residual satisfies $r_j = \widetilde{\phi}_j(A)\phi_j(A)r_0$, where ϕ_j is the BiCOR polynomial and $\widetilde{\phi}_j$ is a polynomial defined by the following coupled two-term recurrence

$$\widetilde{\phi}_0(0) = \widetilde{\psi}_0(0) = 1, \tag{3.1a}$$
$$\widetilde{\phi}_{j+1}(t) = \widetilde{\phi}_j(t) - \widetilde{\alpha}_j t\widetilde{\psi}_j(t), \tag{3.1b}$$
$$\widetilde{\psi}_{j+1}(t) = \widetilde{\phi}_{j+1}(t) + \widetilde{\beta}_{j+1}\widetilde{\psi}_j(t). \tag{3.1c}$$

For convenience, the vectors p_j, p_j^*, r_j, and r_j^* generated by the BiCOR method are replaced by p_j^{BiCOR}, $\widetilde{p}_j^{BiCOR}$, r_j^{BiCOR}, and $\widetilde{r}_j^{BiCOR}$, respectively. $\widetilde{\phi}_j(A)$ and $\widetilde{\psi}_j(A)$ are abbreviated as $\widetilde{\phi}_j$ and $\widetilde{\psi}_j$, respectively.

Now, we attempt to compute $r_{j+1} = \widetilde{\phi}_{j+1}r_{j+1}^{BiCOR}$. We will assume that iteration coefficients $\widetilde{\alpha}_j$ and $\widetilde{\beta}_{j+1}$ are explicitly given, and the vectors $\widetilde{\psi}_j p_j^{BiCOR}$, $\widetilde{\psi}_j r_{j+1}^{BiCOR}$, and $\widetilde{\phi}_{j+1}p_j^{BiCOR}$ have been known at the j-th step. Using the recurrence relations (3.1), it follows from

$$r_{j+1}^{BiCOR} = r_j^{BiCOR} - \alpha_j Ap_j^{BiCOR}, \quad p_{j+1}^{BiCOR} = r_{j+1}^{BiCOR} + \beta_{j+1}p_j^{BiCOR},$$

that the following recurrence relations hold

$$\widetilde{\phi}_{j+1}p_{j+1}^{BiCOR} = \widetilde{\phi}_{j+1}r_{j+1}^{BiCOR} + \beta_{j+1}\widetilde{\phi}_{j+1}p_j^{BiCOR}, \tag{3.2a}$$
$$\widetilde{\psi}_{j+1}p_{j+1}^{BiCOR} = \widetilde{\psi}_{j+1}r_{j+1}^{BiCOR} + \beta_{j+1}\widetilde{\psi}_{j+1}p_j^{BiCOR}, \tag{3.2b}$$
$$\widetilde{\psi}_{j+1}r_{j+1}^{BiCOR} = \widetilde{\phi}_{j+1}r_{j+1}^{BiCOR} + \widetilde{\beta}_{j+1}\widetilde{\psi}_j r_{j+1}^{BiCOR}, \tag{3.2c}$$
$$\widetilde{\psi}_{j+1}p_j^{BiCOR} = \widetilde{\phi}_{j+1}p_j^{BiCOR} + \widetilde{\beta}_{j+1}\widetilde{\psi}_j p_j^{BiCOR}, \tag{3.2d}$$
$$\widetilde{\psi}_j r_{j+1}^{BiCOR} = \widetilde{\psi}_j r_j^{BiCOR} - \alpha_j A\widetilde{\psi}_j p_j^{BiCOR}, \tag{3.2e}$$
$$\widetilde{\phi}_{j+1}p_j^{BiCOR} = \widetilde{\phi}_j p_j^{BiCOR} - \widetilde{\alpha}_j A\widetilde{\psi}_j p_j^{BiCOR}, \tag{3.2f}$$
$$\widetilde{\phi}_j r_{j+1}^{BiCOR} = \widetilde{\phi}_j r_j^{BiCOR} - \alpha_j A\widetilde{\phi}_j p_j^{BiCOR}, \tag{3.2g}$$
$$\widetilde{\phi}_{j+1}r_{j+1}^{BiCOR} = \widetilde{\phi}_j r_{j+1}^{BiCOR} - \widetilde{\alpha}_j A\widetilde{\psi}_j r_{j+1}^{BiCOR}. \tag{3.2h}$$

Combining (3.2g) and (3.2h), we obtain

$$\widetilde{\phi}_{j+1}r_{j+1}^{BiCOR} = \widetilde{\phi}_j r_j^{BiCOR} - A(\alpha_j\widetilde{\phi}_j p_j^{BiCOR} + \widetilde{\alpha}_j\widetilde{\psi}_j r_{j+1}^{BiCOR}). \tag{3.3}$$

Let

$$\begin{aligned} &h_j = \widetilde{\phi}_{j+1}p_j^{BiCOR}, && u_{j+1} = \widetilde{\phi}_{j+1}p_{j+1}^{BiCOR}, && p_{j+1} = \widetilde{\psi}_{j+1}p_{j+1}^{BiCOR}, \\ &r_{j+1} = \widetilde{\phi}_{j+1}r_{j+1}^{BiCOR}, && s_j = \widetilde{\psi}_j r_{j+1}^{BiCOR}, && t_j = \widetilde{\psi}_j r_j^{BiCOR}. \end{aligned} \tag{3.4}$$

From the recurrence relations (3.2a)-(3.2f), (3.3) and above auxiliary iterates, we can get the following iterate sequences

$$
\begin{aligned}
s_j &= t_j - \alpha_j A p_j, && \text{(3.5a)}\\
h_j &= u_j - \widetilde{\alpha}_j A p_j, && \text{(3.5b)}\\
r_{j+1} &= r_j - A(\alpha_j u_j + \widetilde{\alpha}_j s_j), && \text{(3.5c)}\\
t_{j+1} &= r_{j+1} + \widetilde{\beta}_{j+1} s_j, && \text{(3.5d)}\\
u_{j+1} &= r_{j+1} + \beta_{j+1} h_j, && \text{(3.5e)}\\
p_{j+1} &= t_{j+1} + \beta_{j+1}(h_j + \widetilde{\beta}_{j+1} p_j). && \text{(3.5f)}
\end{aligned}
$$

Using $r_{j+1} = \widetilde{\phi}_{j+1}(A)\phi_{j+1}(A)r_0 = b - Ax_{j+1}$ and (3.5c), we have

$$x_{j+1} = x_j + \alpha_j u_j + \widetilde{\alpha}_j s_j. \tag{3.6}$$

Now, we consider how to compute the iteration coefficients α_j and β_{j+1}. It follows from (2.4) and (3.1b) that degrees of the polynomials $\psi_j - \phi_j$ and $\widetilde{\psi}_j - \widetilde{\phi}_j$ are less than j, respectively. Since Ap_j^{BiCOR} is orthogonal to the Krylov subspace $A^H\mathcal{K}_j(A^H, r_0^*)$, then we have

$$
\begin{aligned}
\langle A^H(\psi_j(A^H) - \phi_j(A^H))r_0^*, A\psi_j(A)r_0\rangle &= 0, && \text{(3.7a)}\\
\langle A^H(\widetilde{\psi}_j(A^H) - \widetilde{\phi}_j(A^H))r_0^*, A\psi_j(A)r_0\rangle &= 0. && \text{(3.7b)}
\end{aligned}
$$

Therefore we obtain

$$
\begin{aligned}
\langle A^H\psi_j(A^H)r_0^*, A\psi_j(A)r_0\rangle &= \langle A^H\phi_j(A^H)r_0^*, A\psi_j(A)r_0\rangle, && \text{(3.8a)}\\
\langle A^H\widetilde{\psi}_j(A^H)r_0^*, A\psi_j(A)r_0\rangle &= \langle A^H\widetilde{\phi}_j(A^H)r_0^*, A\psi_j(A)r_0\rangle. && \text{(3.8b)}
\end{aligned}
$$

If a_j and b_j denote the leading coefficients of polynomials $\widetilde{\phi}_j$ and ϕ_j, respectively, then it is easy to deduce $a_{j+1} = -\widetilde{\alpha}_j a_j$ and $b_{j+1} = -\alpha_j b_j$ from (2.4) and (3.1b). Based on the obove results, we have

$$
\begin{aligned}
\alpha_j &= \frac{\langle \widetilde{r}_j^{BiCOR}, Ar_j^{BiCOR}\rangle}{\langle A^H\widetilde{p}_j^{BiCOR}, Ap_j^{BiCOR}\rangle} = \frac{\langle \phi_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}{\langle A^H\psi_j(A^H)r_0^*, A\psi_j(A)r_0\rangle}\\
&= \frac{\langle \phi_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}{\langle A^H\phi_j(A^H)r_0^*, A\psi_j(A)r_0\rangle} = \frac{\langle \widetilde{\phi}_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}{\langle A^H\widetilde{\phi}_j(A^H)r_0^*, A\psi_j(A)r_0\rangle}\\
&= \frac{\langle \widetilde{\phi}_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}{\langle A^H\widetilde{\psi}_j(A^H)r_0^*, A\psi_j(A)r_0\rangle} = \frac{\langle r_0^*, A\widetilde{\phi}_j(A)\phi_j(A)r_0\rangle}{\langle r_0^*, A^2\widetilde{\psi}_j(A)\psi_j(A)r_0\rangle} = \frac{\langle r_0^*, Ar_j\rangle}{\langle r_0^*, A^2p_j\rangle}, \qquad \text{(3.9)}
\end{aligned}
$$

and

$$
\begin{aligned}
\beta_{j+1} &= \frac{\langle \widetilde{r}_{j+1}^{BiCOR}, Ar_{j+1}^{BiCOR}\rangle}{\langle \widetilde{r}_j^{BiCOR}, Ar_j^{BiCOR}\rangle} = \frac{\langle \phi_{j+1}(A^H)r_0^*, A\phi_{j+1}(A)r_0\rangle}{\langle \phi_j(A^H)r_0^*, A\phi_j(A)r_0\rangle}\\
&= \frac{\frac{b_{j+1}}{a_{j+1}}\langle \widetilde{\phi}_{j+1}(A^H)r_0^*, A\phi_{j+1}(A)r_0\rangle}{\frac{b_j}{a_j}\langle \widetilde{\phi}_j(A^H)r_0^*, A\phi_j(A)r_0\rangle} = \frac{\alpha_j}{\widetilde{\alpha}_j}\frac{\langle r_0^*, Ar_{j+1}\rangle}{\langle r_0^*, Ar_j\rangle}. \qquad \text{(3.10)}
\end{aligned}
$$

Thus, the generalized CORS (GCORS) method is described as follows.

Algorithm 1. Generalized CORS (GCORS) method

```
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 $t$ (for
   example, $r_0^* = Ar_0$),
3. set $u_0 = t_0 = r_0$, $\widehat{r}_0 = Ar_0$, $q_0 = \widehat{u}_0 = \widehat{t}_0 = \widehat{r}_0$, $\widehat{q}_0 = Aq_0$, $\rho_0 = \langle r_0^*, \widehat{r}_0\rangle$,
4. for $j = 0, 1, \cdots$, do
5. $\sigma_j = \langle r_0^*, \widehat{q}_j\rangle$,
6. $\alpha_j = \rho_j/\sigma_j$,
7. choose $\widetilde{\alpha}_j$,
8. $s_j = t_j - \alpha_j q_j$,
9. $\widehat{s}_j = \widehat{t}_j - \alpha_j \widehat{q}_j$,
10. $h_j = u_j - \widetilde{\alpha}_j q_j$,
11. $\widehat{h}_j = \widehat{u}_j - \widetilde{\alpha}_j \widehat{q}_j$,
12. $x_{j+1} = x_j + \alpha_j u_j + \widetilde{\alpha}_j s_j$,
13. $r_{j+1} = r_j - \alpha_j \widehat{u}_j - \widetilde{\alpha}_j \widehat{s}_j$,
14. $\widehat{r}_{j+1} = Ar_{j+1}$,
15. $\rho_{j+1} = \langle r_0^*, \widehat{r}_{j+1}\rangle$,
16. $\beta_{j+1} = \frac{\rho_{j+1}}{\rho_j} \times \frac{\alpha_j}{\widetilde{\alpha}_j}$,
17. choose $\widetilde{\beta}_{j+1}$,
18. $t_{j+1} = r_{j+1} + \widetilde{\beta}_{j+1} s_j$,
19. $\widehat{t}_{j+1} = \widehat{r}_{j+1} + \widetilde{\beta}_{j+1} \widehat{s}_j$,
20. $u_{j+1} = r_{j+1} + \beta_{j+1} h_j$,
21. $\widehat{u}_{j+1} = \widehat{r}_{j+1} + \beta_{j+1} \widehat{h}_j$,
22. $q_{j+1} = \widehat{t}_{j+1} + \beta_{j+1}(\widehat{h}_j + \widetilde{\beta}_{j+1} q_j)$,
23. $\widehat{q}_{j+1} = Aq_{j+1}$,
24. end do.
```

Using the same way in [15,16,22], the variables characterized with symbol $\widehat{}$ are introduced to eliminate matrix-vector multiplications.

The CORS and BiCORSTAB methods may fit in the frame of the GCORS method. In fact, if we choose

$$\widetilde{\alpha}_j = \alpha_j, \quad \widetilde{\beta}_j = \beta_j, \tag{3.11}$$

then the vector s_j and h_j, $\widehat{s}_j$ and $\widehat{h}_j$, t_j and u_j as well as $\widehat{t}_j$ and $\widehat{u}_j$ are identical, and the GCORS method reduces to the CORS method [15,16]. If we choose

$$\widetilde{\beta}_{j+1} = 0, \quad \widetilde{\alpha}_j = \arg\min_{\widetilde{\alpha}_j} \|\widetilde{\phi}_j r_{j+1}^{BiCOR} - \widetilde{\alpha}_j A\widetilde{\phi}_j r_{j+1}^{BiCOR}\|_2, \tag{3.12}$$

then the GCORS method reduces to the BiCORSTAB method [15].

Let M_k be any symmetric positive definite (SPD) matrix, and the norm is defined as $|||v|||_k^2 \equiv v^H M_k v, v \in \mathbb{C}^N$. we have the following lemma.

Lemma 1. *Let $A \in \mathbb{C}^{N\times N}, v \in \mathbb{C}^N, \widetilde{A}_k = M_k^{\frac{1}{2}} A M_k^{-\frac{1}{2}}$, then*

$$|||p(A)v|||_k \leq \|p(\widetilde{A}_k)\|_2 |||v|||_k \tag{3.13}$$

holds, where $p(t)$ is a polynomial.

Proof. The proof is similar to that of Lemma 1 in [27], so it is omitted. □

Let x_0 be an initial guess for the solution x of the linear system $Ax = b$, then $e_0 = x - x_0$ is the initial error. Let x_m^{CORS} denote the approximate solution generated by the CORS method at m-th step. It is easy to derive respectively the following errors for the BiCOR, CORS and GCORS methods

$$e_m^{BiCOR} = x - x_m^{BiCOR} = \phi_m(A)e_0, \tag{3.14a}$$
$$e_m^{CORS} = x - x_m^{CORS} = \phi_m^2(A)e_0, \tag{3.14b}$$
$$e_m^{GCORS} = x - x_m^{GCORS} = \widetilde{\phi}_m(A)\phi_m(A)e_0. \tag{3.14c}$$

Using Lemma 1, we can obtain the following error bounds for the BiCOR, CORS and GCORS methods

$$|||e_m^{BiCOR}|||_k \leq \inf_{\phi_m \in \mathbb{P}_m} \|\phi_m(\widetilde{A}_k)\|_2 |||e_0|||_k, \tag{3.15a}$$
$$|||e_m^{CORS}|||_k \leq \inf_{\phi_m \in \mathbb{P}_m} \|\phi_m(\widetilde{A}_k)\|_2^2 |||e_0|||_k, \tag{3.15b}$$
$$|||e_m^{GCORS}|||_k \leq \inf_{\widetilde{\phi}_m, \phi_m \in \mathbb{P}_m} \|\widetilde{\phi}_m(\widetilde{A}_k)\phi_m(\widetilde{A}_k)\|_2 |||e_0|||_k. \tag{3.15c}$$

The CORS method, like the CGS method, is based on squaring the residual polynomial. In case of irregular convergence, this may lead to substantial build-up of rounding errors and worse approximate solutions. In general, the operator $\widetilde{\phi}_n(A)\phi_n(A)$ reducing r_0 will not lead to a bad amplification as the operator $\phi_n^2(A)$ since $|\widetilde{\phi}_n(\lambda)\phi_n(\lambda)|$ is smaller than $max(|\phi_n(\lambda)|^2, |\widetilde{\phi}_n(\lambda)|^2)$. As observed from (3.15c), this situation can lead to faster and smoother convergence properties.

4. The Generalized CORS2 Method

Although two different BiCOR processes are implicitly used in the GCORS method, the iteration steps do not need matrix-vector multiplications in addition to the ones in the GCORS iteration process. In order to efficiently perform the GCORS iteration, we need to select the appropriate parameters $\widetilde{\alpha}_j$ and $\widetilde{\beta}_{j+1}$. Using the similar technique in the CGS2 method [24], we seek coefficients $\widetilde{\alpha}_j$ and $\widetilde{\beta}_{j+1}$ such that

$$\widetilde{\phi}_j r_0, A\widetilde{\psi}_j r_0 \perp A^H \mathcal{K}_j(A^H, s_0^*),$$

for an nonzero vector s_0^* which may be chosen randomly. Similarly to the computation of iteration coefficients α_j and β_{j+1}, it is not difficult to compute

$$\widetilde{\alpha}_j = \frac{\langle \widetilde{\phi}_j(A^H)s_0^*, A\widetilde{\phi}_j(A)r_0\rangle}{\langle A^H\widetilde{\psi}_j(A^H)s_0^*, A\widetilde{\psi}_j(A)r_0\rangle} = \frac{\langle \phi_j(A^H)s_0^*, A\widetilde{\phi}_j(A)r_0\rangle}{\langle A^H\psi_j(A^H)s_0^*, A\widetilde{\psi}_j(A)r_0\rangle}$$

$$= \frac{\langle s_0^*, A\widetilde{\phi}_j(A)\phi_j(A)r_0\rangle}{\langle s_0^*, A^2\widetilde{\psi}_j(A)\psi_j(A)r_0\rangle} = \frac{\langle s_0^*, Ar_j\rangle}{\langle s_0^*, A^2p_j\rangle}, \tag{4.1}$$

and

$$\widetilde{\beta}_{j+1} = \frac{\langle \widetilde{\phi}_{j+1}(A^H)s_0^*, A\widetilde{\phi}_{j+1}(A)r_0\rangle}{\langle \widetilde{\phi}_j(A^H)s_0^*, A\widetilde{\phi}_j(A)r_0\rangle}$$

$$= \frac{\frac{a_{j+1}}{b_{j+1}}\langle \phi_{j+1}(A^H)s_0^*, A\widetilde{\phi}_{j+1}(A)r_0\rangle}{\frac{a_j}{b_j}\langle \phi_j(A^H)s_0^*, A\widetilde{\phi}_j(A)r_0\rangle} = \frac{\widetilde{\alpha}_j}{\alpha_j}\frac{\langle s_0^*, Ar_{j+1}\rangle}{\langle s_0^*, Ar_j\rangle}. \tag{4.2}$$

Using the above results, the GCORS2 method can be described as follows.

Algorithm 2. Generalized CORS2 (GCORS2) method

```
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 $t$ (for
   example, $p(t) = t$),
3. set $u_0 = t_0 = r_0$, $\widehat{r}_0 = Ar_0$, $q_0 = \widehat{u}_0 = \widehat{t}_0 = \widehat{r}_0$, $\widehat{q}_0 = Aq_0$, $\rho_0 = \langle r_0^*, \widehat{r}_0\rangle$,$\widehat{\rho}_0 = \langle s_0^*, \widehat{r}_0\rangle$,
4. for $j = 0, 1, \cdots$, do
5. $\sigma_j = \langle r_0^*, \widehat{q}_j\rangle$,
6. $\widehat{\sigma}_j = \langle s_0^*, \widehat{q}_j\rangle$,
7. $\alpha_j = \rho_j/\sigma_j$,
8. $\widehat{\alpha}_j = \widehat{\rho}_j/\widehat{\sigma}_j$,
9. $s_j = t_j - \alpha_j q_j$,
10. $\widehat{s}_j = \widehat{t}_j - \alpha_j \widehat{q}_j$,
11. $h_j = u_j - \widetilde{\alpha}_j q_j$,
12. $\widehat{h}_j = \widehat{u}_j - \widetilde{\alpha}_j \widehat{q}_j$,
13. $x_{j+1} = x_j + \alpha_j u_j + \widetilde{\alpha}_j s_j$,
14. $r_{j+1} = r_j - \alpha_j \widehat{u}_j - \widetilde{\alpha}_j \widehat{s}_j$,
15. $\widehat{r}_{j+1} = Ar_{j+1}$,
16. $\rho_{j+1} = \langle r_0^*, \widehat{r}_{j+1}\rangle$,
17. $\widehat{\rho}_{j+1} = \langle s_0^*, \widehat{r}_{j+1}\rangle$,
18. $\beta_{j+1} = \frac{\rho_{j+1}}{\rho_j} \times \frac{\alpha_j}{\widetilde{\alpha}_j}$,
19. $\widetilde{\beta}_{j+1} = \frac{\widehat{\rho}_{j+1}}{\widehat{\rho}_j} \times \frac{\widetilde{\alpha}_j}{\alpha_j}$,
20. $t_{j+1} = r_{j+1} + \widetilde{\beta}_{j+1} s_j$,
21. $\widehat{t}_{j+1} = \widehat{r}_{j+1} + \widetilde{\beta}_{j+1} \widehat{s}_j$,
22. $u_{j+1} = r_{j+1} + \beta_{j+1} h_j$,
23. $\widehat{u}_{j+1} = \widehat{r}_{j+1} + \beta_{j+1} \widehat{h}_j$,
24. $q_{j+1} = \widehat{t}_{j+1} + \beta_{j+1}(\widehat{h}_j + \widetilde{\beta}_{j+1} q_j)$,
25. $\widehat{q}_{j+1} = Aq_{j+1}$,
26. end do.
```

Compared with the CORS method, the GCORS2 method needs two matrix-vector multiplications, four more vector updates and two more inner products per iteration. Similarly to the breakdowns of the BiCOR method, there exist Lanczos breakdown and pivot breakdown for the GCORS2 method. These two kinds of breakdowns can be cured using look-ahead techniques [30-33] and composite step techniques [25-29], respectively. Here, we will not further discuss this problem.

To improve the convergence of the GCORS2 method for solving the problem (1.1), it is very effective to use the preconditioning techniques. Preconditioning can easily be incorporated by replacing the Eq. (1.1) with

$$(M_1^{-1}AM_2^{-1})(M_2x) = M_1^{-1}b,$$

and we apply the algorithm to the preconditioned system $\widetilde{A}\widetilde{x} = \widetilde{b}$ with $\widetilde{A} = M_1^{-1}AM_2^{-1}$,$\widetilde{x} = M_2x$ and $\widetilde{b} = M_1^{-1}b$. In this paper we only consider the right preconditioner (i.e. $M_1 = I, M_2 = M$), the detailed derivation is similar to [7] and the right preconditioned GCORS2 method can be described as follows.

Algorithm 3. Right preconditioned generalized CORS2 (PGCORS2) method

```
1. compute $r_0 = b - Ax_0$ for some initial guess $x_0$,
2. choose $r_0^* = p(AM^{-1})r_0$ such that $\langle r_0^*, AM^{-1}r_0\rangle \neq 0$, where $p(t)$ is a polynomial in $t$
   (for example, $p(t) = t$),
3. set $Mu_0 = Mt_0 = M^{-1}r_0$, $\widehat{r}_0 = AM^{-1}r_0$, $q_0 = \widehat{u}_0 = \widehat{t}_0 = \widehat{r}_0$, $\widehat{q}_0 = AM^{-1}q_0$,
   $\rho_0 = \langle r_0^*, \widehat{r}_0\rangle$,$\widehat{\rho}_0 = \langle s_0^*, \widehat{r}_0\rangle$,
4. for $j = 0, 1, \cdots$, do
5. $\sigma_j = \langle r_0^*, \widehat{q}_j\rangle$,
6. $\widehat{\sigma}_j = \langle s_0^*, \widehat{q}_j\rangle$,
7. $\alpha_j = \rho_j/\sigma_j$,
8. $\widehat{\alpha}_j = \widehat{\rho}_j/\widehat{\sigma}_j$,
9. $Ms_j = Mt_j - \alpha_j M^{-1}q_j$,
10. $\widehat{s}_j = \widehat{t}_j - \alpha_j\widehat{q}_j$,
11. $Mh_j = Mu_j - \widetilde{\alpha}_j M^{-1}q_j$,
12. $\widehat{h}_j = \widehat{u}_j - \widetilde{\alpha}_j\widehat{q}_j$,
13. $x_{j+1} = x_j + \alpha_j Mu_j + \widetilde{\alpha}_j Ms_j$,
14. $r_{j+1} = r_j - \alpha_j\widehat{u}_j - \widetilde{\alpha}_j\widehat{s}_j$,
15. $\widehat{r}_{j+1} = AM^{-1}r_{j+1}$,
16. $\rho_{j+1} = \langle r_0^*, \widehat{r}_{j+1}\rangle$,
17. $\widehat{\rho}_{j+1} = \langle s_0^*, \widehat{r}_{j+1}\rangle$,
18. $\beta_{j+1} = \frac{\rho_{j+1}}{\rho_j} \times \frac{\alpha_j}{\widetilde{\alpha}_j}$,
19. $\widetilde{\beta}_{j+1} = \frac{\widehat{\rho}_{j+1}}{\widehat{\rho}_j} \times \frac{\widetilde{\alpha}_j}{\alpha_j}$,
20. $Mt_{j+1} = M^{-1}r_{j+1} + \widetilde{\beta}_{j+1}Ms_j$,
21. $\widehat{t}_{j+1} = \widehat{r}_{j+1} + \widetilde{\beta}_{j+1}\widehat{s}_j$,
22. $Mu_{j+1} = M^{-1}r_{j+1} + \beta_{j+1}Mh_j$,
23. $\widehat{u}_{j+1} = \widehat{r}_{j+1} + \beta_{j+1}\widehat{h}_j$,
24. $q_{j+1} = \widehat{t}_{j+1} + \beta_{j+1}(\widehat{h}_j + \widetilde{\beta}_{j+1}q_j)$,
25. $\widehat{q}_{j+1} = AM^{-1}q_{j+1}$,
26. end do.
```

5. Numerical Examples

In this section, we present some numerical examples to empirically study the effectiveness of Algorithm 2 and Algorithm 3 for complex non-Hermitian linear systems. Appropriate preconditioning techniques can make the presented method very attractive for some kinds of complex non-Hermitian linear systems, we only consider the right preconditioner. The test matrices are taken from the University of Florida Sparse Matrix Collection [34]. All computations are carried out using double precision floating point arithmetic in MATLAB 7.11 with a PC-Intel (R) Core (TM)2 Duo CPU T6570 2.10 GHz, and 2GB RAM. The iteration is started with the initial guess $x_0 = 0$ in all cases. Let $r_0^* = Ar_0$ and $s_0^* = A * rand(N,1)$. Its, Mvs, CPU and Res denote numbers of iterations, numbers of matrix-vector multiplications, CPU times in seconds for computing approximate solution and final true relative residual 2-norms defined as $\log_{10}(\frac{\|b-Ax\|_2}{\|b\|_2})$, respectively. GMRES denotes the unrestarted complex GMRES method.

Example 1. We consider a complex Toeplitz matrix A [8,10] of order 1000

$$A = \begin{bmatrix} 4 & 0 & 1 & 0.7 & & \\ \gamma i & 4 & 0 & 1 & 0.7 & \\ & \gamma i & 4 & 0 & 1 & \ddots \\ & & \gamma i & 4 & 0 & \ddots \\ & & & \ddots & \ddots & \ddots \end{bmatrix}.$$

with the symbol $f(z) = \gamma i z^{-1} + 4 + z^2 + 0.7z^3$. By varying parameter γ, we can obtain the resulting spectra and pseudospectra to be associated with the symbol $f(z)$. Thus the parameter γ can have an effect on the convergence of the BiCOR, CORS, BiCORSTAB and GCORS methods. This is confirmed by numerical results (see Table 5.1).

Table 5.1: Numerical results of the BiCOR, CORS, BiCORSTAB and GCORS2 methods.

γ	*BiCOR*			*CORS*			*BiCORSTAB*			*GCORS*2		
	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res
2.0	49	0.01	-10.055	23	0.01	-10.324	26	0.01	-10.210	23	0.01	-10.243
2.5	100	0.03	-10.544	50	0.03	-10.069	38	0.01	-10.044	34	0.02	-10.163
2.7	126	0.03	-10.487	500	0.20	-8.193	47	0.02	-10.209	48	0.02	-10.467
3.0	180	0.05	-10.053	500	0.18	4.538	64	0.02	-10.014	69	0.03	-10.082
3.2	500	0.12	-4.006	500	0.19	-0.208	91	0.03	-10.155	90	0.04	-10.262
3.5	500	0.12	-2.638	500	2.72	NaN	253	0.08	-10.007	171	0.07	-10.246
3.6	500	0.12	0.215	500	3.88	NaN	460	0.15	-10.021	258	0.11	-10.014

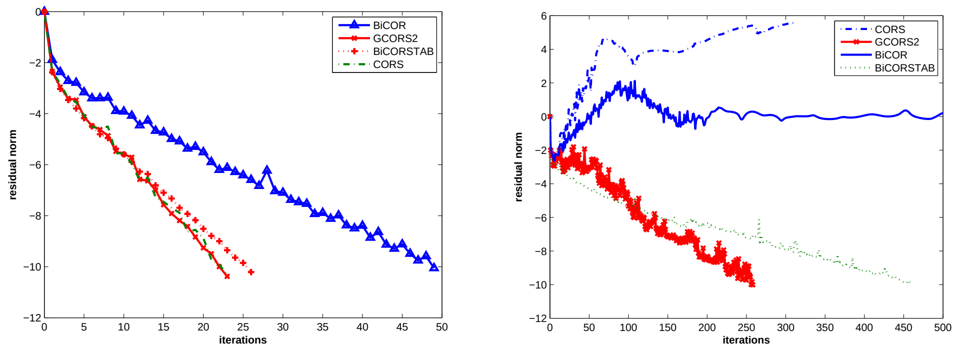

Fig. 5.1. The convergence history for $\gamma = 2.0$(on the left) and $\gamma = 3.6$(on the right).

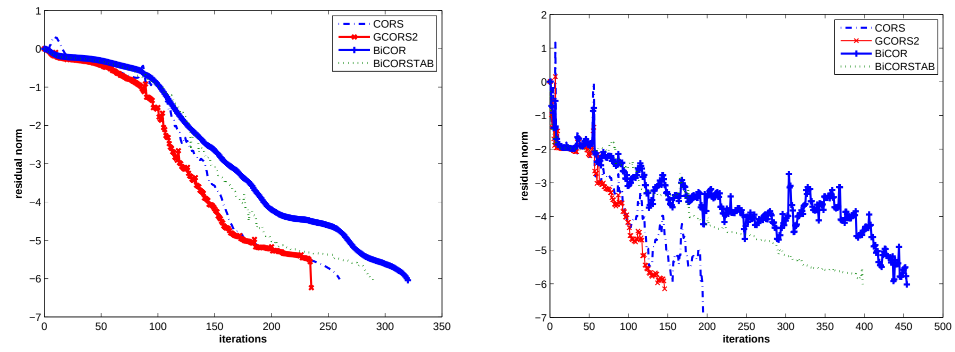

Fig. 5.2. The convergence history for light_in_tissue(on the left) and kim1(on the right).

We set the right-hand side $b = Ae$, where e is the $N \times 1$ vector whose elements are all equal to unity. Then $x = (1, 1, \cdots, 1)^T$ is the exact solution of $Ax = b$. The stopping criterion is $\frac{\|r_j\|_2}{\|r_0\|_2} \leq 10^{-10}$. The maximum iteration count is set to 500. Note that the NaN returns the IEEE arithmetic representation for not-a-number.

Based on the data in Table 5.1, several observations may be concluded. First, for $\gamma = 2$, the four methods return well convergence properties, and the GCORS2 method converges faster and exhibits much smoother convergence behavior than the other methods as seen from Fig. 5.1 (on the left), and the CORS method gets the most accurate results. Second, for $\gamma > 2.5$, the CORS method begins to diverge, however, the GCORS2 method converges and is competitive with the BiCOR and BiCORSTAB methods. Third, for $\gamma = 3.5, 3.6$, as seen from Fig. 5.1 (on the right), the GCORS2 method only requires respectively about 67.6% and 56.1% in terms of Its of the BiCORSTAB method. In the figure, we plot the relative residual 2-norms (in the logarithmic scale) versus the iterations.

Example 2. In this example, we use the following matrices: young1c, *light_in_tissue* and kim1 which arise from three representative physical problems, see [34] for more details. We set the right-hand side $b = (i, \cdots, i)^T$. The iterations are stopped when $\frac{\|r_j\|_2}{\|r_0\|_2} \leq 10^{-6}$. The maximum iteration count is set to 500. As observed from Table 5.2, the GCORS2 method requires less iteration steps and CUP times than other three methods and reaches the highest precision in terms of Res in most cases. The CORS method obtains the most accurate results

Table 5.2: Numerical results of the BiCOR, CORS, BiCORSTAB and GCORS2 methods.

matrices	*BiCOR*			*CORS*			*BiCORSTAB*			*GCORS2*		
	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res
light_in_tissue	320	4.19	-6.043	260	4.25	-6.017	290	5.11	-6.081	240	5.20	-6.236
kim1	454	10.88	-6.023	195	5.57	-6.956	398	11.27	-6.077	146	4.79	-6.149
young1c	205	0.05	-6.037	500	1.80	0.079	386	0.11	-6.080	198	0.07	-6.176

Table 5.3: Numerical results of the BiCOR, CORS, BiCORSTAB, GCORS2 and GMRES methods.

matrices	*BiCOR*			*CORS*			*BiCORSTAB*			*GCORS2*			*GMRES*		
	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res
young2c	319	0.07	-6.01	500	0.17	5.04	314	0.09	-6.01	192	0.07	-6.03	274	3.01	-6.01
young4c	500	0.11	-4.50	500	0.16	1.89	500	0.15	-4.85	399	0.16	-6.01	413	7.10	-6.01

for the matrix kim1, but the CORS method fails to converge for the matrix young1c, and the GCORS2 method can do well.

As seen from Fig. 5.2, we observe that the GCORS2 method does not amplify the initial residual as much as the CORS method does, the residual norm of the GCORS2 method stays well below that of the CORS method and the convergence behavior of the GCORS2 method is much smoother.

Example 3. In this example, we compare the performances of the GCORS2, CORS, BiCOR, BiCORSTAB and unrestarted complex GMRES methods. We choose the matrices *young*2*c* and *young*4*c* which arise from modeling acoustic scattering phenomena governed by the Helmhlotz equation, see [34] for more details. We set the right-hand side $b = Ae$. The stopping criterion is $\frac{\|r_j\|_2}{\|r_0\|_2} \le 10^{-6}$. The maximum iteration count is set to 500.

Table 5.3 and Fig. 5.3 report the results obtained with the GCORS2 method and other four methods. From these data, we conclude that the GCORS2 method converges faster than other four methods, however, the CORS method fails to converge for the matrices *young*2*c* and *young*4*c*, and both the BiCOR and BiCORSTAB methods fail to converge for the matrix *young*4*c*. Although the GMRES method often converges for the matrices *young*2*c* and *young*4*c*, this method can suffer from high memory requirements. The GMRES method is often recommended due to robustness and smoother convergence, where storage requirement is allowed. The GCORS2 method is competitive with other four methods.

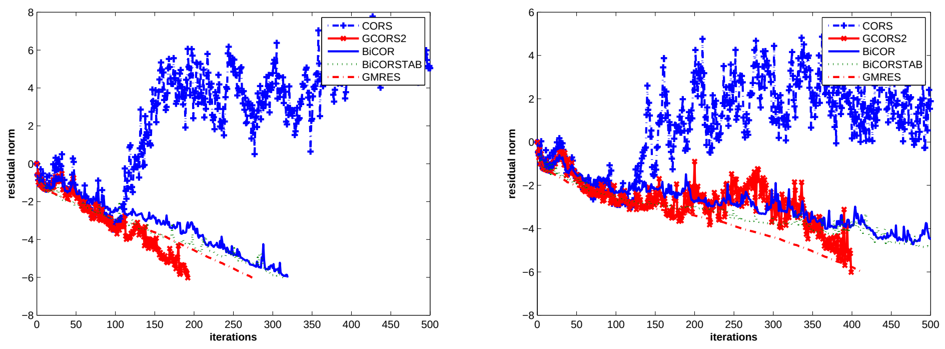

Fig. 5.3. The convergence history for young2c (on the left) and young4c (on the right).

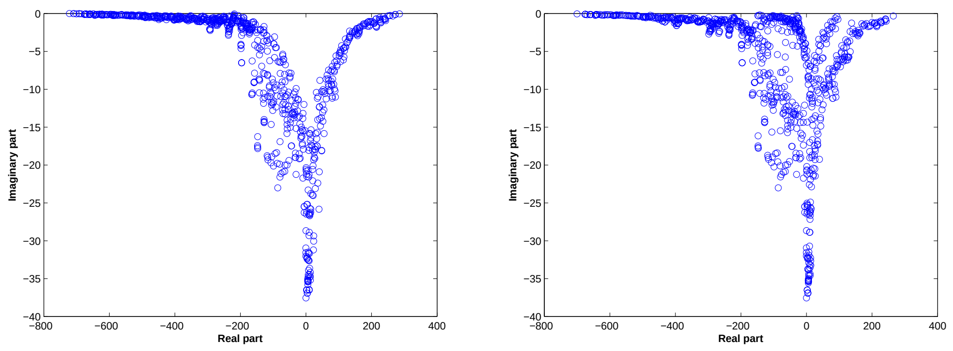

Fig. 5.4. The spectrum of the matrices young2c (on the left) and young4c (on the right).

Fig. 5.4 shows the spectra of the matrices $young2c$ and $young4c$, which have large imaginary components. This figure may also give some reason why the GCORS2 method performs better than other four methods in these two cases. Jing et al. [15] analyzed the same problem. Using the idea of the BiCGSTAB2 method [8], Zhao and Huang [22] proposed the BiCORSTAB2 method that generated better results than the BiCORSTAB method in this situation. Similarly to the BiCGSTAB(l) and GPBiCG methods [9,10], some new methods may be developed to solve this problem.

Example 4. In this example, we compare the performances of the GCORS2 and CGS2 methods [24]. We choose the matrices *ted_AB_unscaled* and $young2c$ [34], and set the right-hand side $b = Ae$. The iterations are stopped when $\frac{\|r_j\|_2}{\|r_0\|_2} \le 10^{-6}$. As seen from Fig. 5.5, the GCORS2 method requires less iteration steps and shows smoother convergence behavior than the CGS2 method, especially for the matrix *ted_AB_unscaled*.

Example 5. We consider the matrices kim1 and A arising in the central difference discretization of the Helmholtz equation [35]

$$\begin{cases} \Delta u + k^2 u = 0, (x,y) \in [0,\pi] \times [0,\pi], \\ u_x(0,y) = i\sqrt{k^2 - 1/4}\cos(y/2), \\ u_x(\pi,y) - i\sqrt{k^2 - 1/4}u = 0, \\ u_y(x,0) = u(x,\pi) = 0. \end{cases} \tag{5.1}$$

The stepsizes along both x and y directions are same, i.e. $h = \pi/m$. Different parameters k

Table 5.4: Numerical results of the CORS, GCORS2, CGS, BiCGSTAB and IDR(s) methods.

methods	matrix A_1			matrix A_2			kim1		
	Mvs	CPU	Res	Mvs	CPU	Res	Mvs	CPU	Res
CORS	993	0.48	-8.13	835	0.62	-8.10	315	4.21	-8.04
GCORS2	929	0.46	-8.21	811	0.66	-8.34	267	4.12	-8.22
CGS	1049	0.50	-8.31	907	0.72	-8.04	391	5.63	-8.50
BiCGSTAB	1601	0.49	-2.36	1601	0.81	-2.45	595	6.69	-8.02
IDR(2)	1407	0.75	-8.41	868	0.73	-8.06	661	10.00	-8.00
IDR(4)	760	0.47	-8.07	547	0.54	-8.06	434	8.66	-8.06
IDR(6)	632	0.48	-8.13	492	0.53	-8.01	434	10.95	-8.01

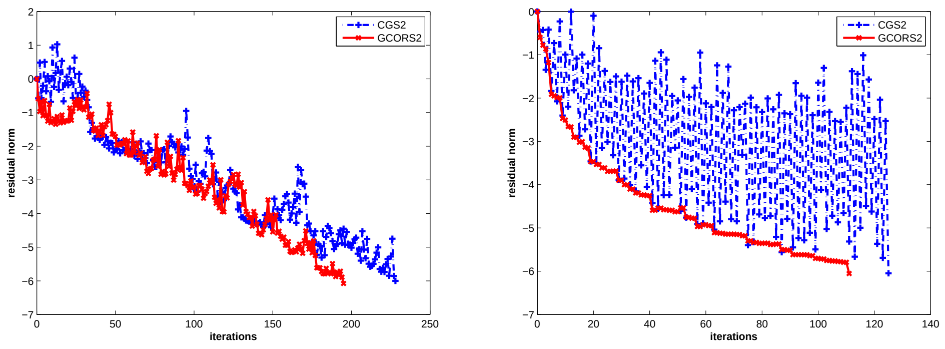

Fig. 5.5. The convergence history for young2c (on the left) and *ted_AB_unscaled* (on the right).

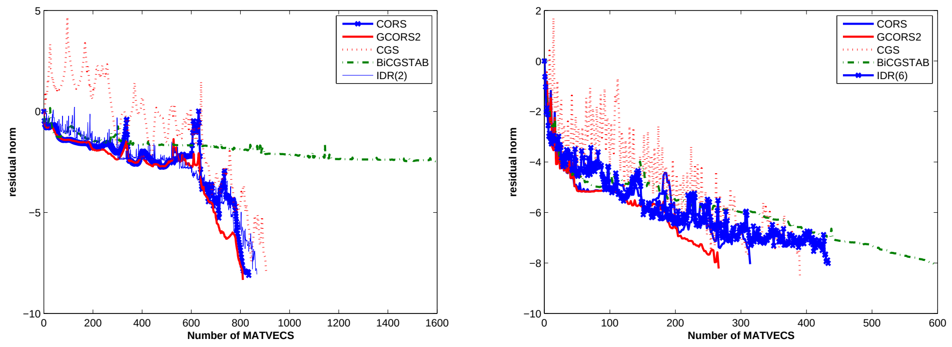

Fig. 5.6. The convergence history for matrix A_2 (on the left) and kim1 (on the right).

lead the linear systems to be highly indefinite. We only test the following two cases: the matrix A_1 for m=50, k=4.16 and the matrix A_2 for m=60, k=2.62. We compare the performances of the CORS, GCORS2, CGS, BiCGSTAB, and IDR(s) methods [36,37], and set the right-hand side $b = Ae$. The iterations are stopped when $\frac{\|r_j\|_2}{\|r_0\|_2} \leq 10^{-8}$.

Computational results are given in Table 5.4 and Fig. 5.6. For the matrix kim1, the GCORS2 method returns the best convergence result in terms of Mvs and CPU, and provides the secondly accuracy in terms of Res. For the matrices A_1 and A_2, the GCORS2 method requires less Mvs than the CORS, CGS, BiCGSTAB and IDR(2) methods, and converges smoother than the CORS and CGS methods, and the BiCGSTAB method fails to converge. The IDR(s) method, which was recently developed by Sonneveld and van Gijzen [36,37], is a family of simple and fast algorithm for solving complex non-Hermitian linear systems and can be competitive with the Krylov subspace methods. As seen from Table 5.4, the IDR(4) and IDR(6) methods demonstrate excellent numerical results for the matrices A_1 and A_2.

Example 6. In this example, we compare the performances of the right preconditioned CORS, GCORS2 and BiCORSTAB methods with the ILU(0) [38] or ILUT (0.04) [39] preconditioners. ILU(0)-PCORS, ILU(0)-PGCORS2 and ILU(0)-PBiCORSTAB denote the right preconditioned CORS, GCORS2 and BiCORSTAB methods with the ILU(0) preconditioner. ILUT-PCORS, ILUT-PGCORS2 and ILUT-PBiCORSTAB denote the right preconditioned CORS, GCORS2 and BiCORSTAB methods with the ILUT (0.04) preconditioner. We choose the matrices *ted_AB*, *chevron*1, *kim*1, *young*1*c*, *young*2*c* and *young*4*c* [34], and set the right-hand side

Table 5.5: Numerical results of the right preconditioned CORS, GCORS2 and BiCORSTAB methods.

methods	chevron1			*ted_AB*			kim1		
	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res
ILU(0)-PCORS	293	9.14	-6.03	640	16.68	-6.63	4	0.37	-7.00
ILU(0)-PGCORS2	258	8.89	-6.05	562	15.23	-6.60	4	0.37	-6.89
ILU(0)-PBiCORSTAB	398	12.53	-6.04	1000	26.19	-4.29	4	0.35	-7.16
CORS	5000	75.78	12.10	5000	56.57	-0.17	100	2.79	-6.33
GCORS2	2165	42.31	-6.04	5000	59.37	-1.38	89	2.71	-6.13
BiCORSTAB	5000	80.43	-1.29	5000	54.55	-0.24	148	3.87	-6.10
methods	young1c			young2c			young4c		
	Its	CPU	Res	Its	CPU	Res	Its	CPU	Res
ILUT-PCORS	24	0.05	-6.30	8	0.04	-6.62	9	0.04	-6.76
ILUT-PGCORS2	24	0.05	-6.49	8	0.03	-6.81	7	0.03	-6.98
ILUT-PBiCORSTAB	26	0.05	-6.05	7	0.03	-6.37	7	0.03	-6.37
CORS	500	0.16	-0.67	500	0.17	5.04	500	0.16	1.89
GCORS2	193	0.07	-6.06	192	0.07	-6.03	399	0.16	-6.01
BiCORSTAB	456	0.14	-6.05	314	0.09	-6.01	500	0.15	-4.85

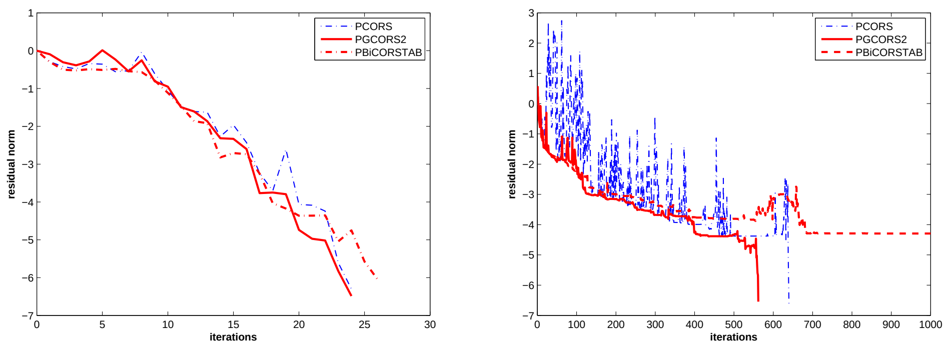

Fig. 5.7. The convergence history for matrix young1c (on the left) and *ted_AB* (on the right).

$b = Ae$. The iterations are stopped when $\frac{\|r_j\|_2}{\|r_0\|_2} \le 10^{-6}$.The maximum iteration count is set to 1000 for the preconditioned methods.

From Table 5.5, we observe that the right preconditioned CORS, GCORS2 and BiCORSTAB methods result in fast convergence and perform better than the unpreconditioned CORS, GCORS2 and BiCORSTAB methods in both iteration steps and CPU times. The preconditioned GCORS2 method with the ILU(0) preconditioner returns the best numerical performances for the matrices chevron1 and *ted_AB*. In other problems, the preconditioned CORS, GCORS2 and BiCORSTAB methods show almost the same performances. As seen from Fig. 5.7, the preconditioned GCORS2 method requires less iteration steps and shows smoother convergence behavior than the preconditioned CORS and BiCORSTAB methods. However, the preconditioned BiCORSTAB method fails to converge for the matrix *ted_AB*.

6. Conclusion

In this paper, we have developed a new variant of the BiCOR method for solving complex non-Hermitian linear systems. The numerical experiments show that the new method converges faster and smoother than the BiCOR, CORS, CGS and CGS2 methods in most cases, and can compete with the BiCGSTAB, BiCORSTAB, IDR(2) and GMRES methods in some cases. Note that the HSS, BTSS and PSS methods can lead to high-quality preconditioners for Krylov subspace methods, see [40,41,42] for more details. The GCORS method combined with these preconditioners to deal with practical problems is under investigation and will be given in the future.

Acknowledgments. The authors are grateful to the referees for their valuable comments and suggestions which helped to improve the presentation of this paper.

The research is supported by the National Natural Science Foundation of China under grant No.11071118, Natural Science Foundation from Anhui Province Education Department under grant No.KJ2012B058 and AHSTU under grant No.ZRC2013388.

References

[1] V. Simoncini and D.B. Szyld, Recent computatinal developments in krylov subspace methods for linear systems, *Numer. Linear Algebra Appl.*, **14** (2007), 1-59.

[2] Y. Saad and M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.*, **7** (1986), 856-869.

[3] Y. Saad, A flexible inner-outer preconditioned GMRES algorithm, *SIAM J. Sci. Statist. Comput.*, **14** (1993), 461-469.

[4] B. Carpentieri, I. Duff, L. Giraud and G. Sylvand, Combining fast multipole techniques and an approximate inverse preconditioner for large electromagnetism calculations, *SIAM J. Sci. Comput.*, **27** (2005), 774-792.

[5] R. Fletcher, Conjugate gradient methods for indefinite systems, *Lecture Notes in Mathematics*, vol. 506, Springer-Verlag, Berlin, 1976, 73-89.

[6] P. Sonneveld, CGS: a fast Lanczos-type solver for nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.*, **10** (1989), 36-52.

[7] H.A. van der Vorst, BiCGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.*, **13** (1992), 631-644.

[8] M.H. Gutknecht, Variants of BiCGSTAB for matrices with complex spectrum, *SIAM J. Sci. Comput.*, **14** (1993), 1020-1033.

[9] G.L.G. Sleijpen and D.R. Fokkema, BiCGSTAB(l) for linear equations involving unsymmetric matrices with complex spectrum, *Elec. Trans. Numer. Anal.*, **1** (1993), 11-32.

[10] S.-L. Zhang, GPBi-CG: Generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems, *SIAM J. Sci. Comput.*, **18** (1997), 537-551.

[11] M.C. Yeung and T.F. Chan, ML(K)BiCGSTAB: A BiCGSTAB variant based on multiple Lanczos starting vectors, *SIAM J. Sci. Comput.*, **21** (1999), 1263-1290.

[12] R.W. Freund, A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems, *SIAM J. Sci. Comput.*, **14** (1993), 470-482.

[13] R.W. Freund and N.M. Nachtigal, QMR: a quasi-minimal residual method for non- Hermitian linear systems, *Numer. Math.*, **60** (1991), 315-339.

[14] T. Malas and L. Gürel, Incomplete LU preconditioning with multilevel fast multipole algorithm for electromagnetic scattering, *SIAM J. Sci. Comput.*, **29** (2007), 1476-1494.

[15] Y.F. Jing, T.Z. Huang, Y. Zhang, L. Li, G.H. Cheng, Z.G. Ren, Y. Duan, T. Sogabe and B. Carpentieri, Lanczos-type variants of the COCR method for complex nonsymmetric linear systems, *J. Comput. Phys.*, **228** (2009), 6376-6394.

[16] B. Carpentieri, Y.F. Jing and T.Z. Huang, The BiCOR and CORS iterative algorithms for solving nonsymmetric linear systems, *SIAM J. Sci. Comput.*, **33** (2011), 3020-3036.

[17] T. Sogabe and S.-L. Zhang, A COCR method for solving complex symmetric linear systems, *J. Comput. Appl. Math.*, **199** (2007), 297-303.

[18] M.H. Gutknecht, The unsymmetric Lanczos algorithms and their relations to Pade approximation, continued fractions, and the qd algorithm, *Proc. of the Copper Mountain Conference on Iterative Methods*, April 1990, http://www.sam.math.ethz.ch/ mhg/pub/CopperMtn90.ps.gz.

[19] 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.*, **99** (2009), 427-451.

[20] Y.F. Jing, T.Z. Huang, Y. Duan and B. Carpentieri, A comparative study of iterative solutions to linear systems arising in quantum mechanics, *J. Comput. Phys.*, **229** (2010), 8511-8520.

[21] B. Carpentieri, Y.F. Jing, T.Z. Huang, W.C. Pi and X.Q. Sheng, Combining the CORS and BiCORSTAB iterative methods with MLFMA and SAI preconditioning for solving large linear systems in electromagnetics, *Applied Computational Electromagnetics Society (ACES) Journal.*, **27** (2012), 102-111.

[22] L. Zhao and T.Z. Huang, A hybrid variant of the BiCOR method for a nonsymmetric linear system with a complex spectrum, *Applied Mathematics Letters.*, **26** (2013), 457-462.

[23] L. Zhao, T.Z. Huang, Y.F. Jing and L.J. Deng, A generalized product-type BiCOR method and its application in signal deconvolution, *Computers and Mathematics with Applications.*, **66(8)** (2013), 1372-1388.

[24] D.R. Fokkema, G.L.G. Sleijpen and H.A. van der Vorst, Generalized conjugate gradient squared, *J. Comput. Appl. Math.*, **71** (1996), 125-146.

[25] R.E. Bank and T.F. Chan, An analysis of the composite step biconjugate gradient method, *Numer. Math.*, **66** (1993), 295-319.

[26] R.E. Bank and T.F. Chan, A composite step biconjugate gradient algorithm for nonsymmetric linear systems, *Numer. Algo.*, **7** (1994), 1-16.

[27] T.F. Chan and T. Szeto, Composite step product methods for nonsymmetric linear systems, *SIAM J. Sci. Comput.*, **17** (1996), 1491-1508.

[28] Y.F. Jing, T.Z. Huang, B. Carpentieri and Y. Duan, Investigating the composite step biconjugate A-Orthogonal residual method for non-Hermitian dense linear systems in electromagnetics, *Applied Computational Electromagnetics Society (ACES) Journal.*, **27** (2012), 112-122.

[29] Y.F. Jing, T.Z. Huang, B. Carpentieri and Y. Duan, Exploiting the composite step strategy to the biconjugate A-orthogonal residual method for non-Hermitian linear systems, *Journal of Applied Mathematics.*, **vol. 2013** (2013), Article ID 408167, 16 pages. doi:10.1155/2013/408167.

[30] B. Parlett, D. Taylor and Z.S. Liu, A look-ahead Lanczos algorithm for unsymmetric matrices, *Math. Comp.*, **44** (1985), 105-124.

[31] C. Brezinski, M.R. Zaglia and H. Sadok, A breakdown-free Lanczos type algorithm for solving linear systems, *Numer. Math.*, **63** (1992), 29-38.

[32] C. Brezinski, M.R. Zaglia and H. Sadok, New look-ahead Lanczos-type algorithms for linear systems, *Numer. Math.*, **83** (1999), 53-85.

[33] R. Freund, M.H. Gutknecht and N.M. Nachtigal, An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices, *SIAM J. Sci. Stat. Comput.*, **14** (1993), 137-158.

[34] T.A. Davis and Y.F. Hu, The University of Florida Sparse Matrix Collection, *ACM Trans. Math. Softw.*, **Vol. 38, No. 1** (2011), Article 1, 25 pages. doi:10.1145/2049662.2049663.

[35] A. Bayliss, C.I. Glodstein and E. Turkel, An iterative method for the Helmholtz equation, *J. Comput. Phys.*, **49** (1983), 443-457.

[36] P. Sonneveld and M.B. Van Gijzen, IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations, *SIAM J. Sci. Statist. Comput.*, **31** (2008), 1035-1062.

[37] M.B. Van Gijzen and P. Sonneveld, An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties, Report 10-16, Department of Applied Mathematical Analysis, Delft University of Technology, Delft, The Netherlands, 2010.

[38] J.A. Meijerink and H.A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comp.*, **31** (1977), 148-162.

[39] Y. Saad, ILUT: A dual threshold incomplete ILU factorization, *Numer. Linear Algebra Appl.*, **1** (1994), 387-402.

[40] Z.-Z. Bai, G.H. Golub and M.K. Ng, Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems, *SIAM J. Matrix Anal. Appl.*, **24** (2003), 603-626.

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

[42] Z.-Z. Bai, Splitting iteration methods for non-Hermitian positive definite systems of linear equations, *Hokkaido Mathematical Journal.*, **36** (2007), 801-814.