Numerical Algorithms
https://doi.org/10.1007/s11075-021-01097-7

ORIGINAL PAPER

Block GPBi-CG method for solving nonsymmetric linear systems with multiple right-hand sides and its convergence analysis

A. Taherian[1] · **F. Toutounian**[1,2]

Received: 23 September 2019 / Accepted: 8 March 2021 / Published online: 18 May 2021

Abstract
In this paper, the block generalized product-type bi-conjugate gradient (GPBi-CG) method for solving large, sparse nonsymmetric linear systems of equations with multiple right-hand sides is proposed. The new algorithm is based on the block BiCG process. We analyze the convergence behavior of this method and present a bound for the residual norm of block GPBi-CG according to the residual norm of Bl-GMRES method. In addition, we prove that convergence is guaranteed when A is positive real. The numerical experiments show the efficiency of the new method and confirm the theoretical results.

Keywords Multiple right-hand sides · Block Krylov subspace · Block BiCG · Block GPBi-CG · Convergence analysis · Block GMRES

Mathematics Subject Classification (2010) 65F10

1 Introduction

We consider the solution of large linear systems with multiple right-hand sides of the form

$$AX = B, \tag{1.1}$$

✉ F. Toutounian
toutouni@math.um.ac.ir

A. Taherian
a_taherian@mail.um.ac.ir

[1] Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad, Mashhad, Iran

[2] The Center of Excellence on Modeling and Control Systems, Ferdowsi University of Mashhad, Mashhad, Iran

where A is an $n \times n$ nonsingular matrix, and $B = [b^{(1)}, \ldots, b^{(s)}], X = [x^{(1)}, \ldots, x^{(s)}]$ are rectangular $n \times s$ matrices. In practice, s is of moderate size with $s \ll n$. Many applications such as electromagnetic scattering problem and structural mechanics problems require the solution of linear systems with multiple right-hand sides (1.1).

To solve (1.1), the block Krylov subspace methods often show better convergence behavior than the corresponding Krylov subspace methods for systems with a single right-hand side. For symmetric and positive definite problems, the block conjugate gradient (Bl-CG) algorithm [15] and its variants [14] are useful for solving the linear system (1.1). For nonsymmetric problems, the block BCG method [15] and a stabilized version of BiCG [17], the block GMRES [16], the block BiCGSTAB algorithm [9], the block Lanczos method [8], the block quasi minimum residual (Bl-QMR) algorithm [5], the block LSQR method [11], the block LSMR method [22], and the block CMRH method [1, 2] have been developed. Frommer et al. [6, 7] have contributed several results to the theory of block Krylov subspace methods for linear systems and for matrix functions. These results hold for general block inner products, and thus in particular for the classical block methods and the so-called global methods. In [13], the authors extended well-known GMRES/Arnoldi convergence results to the block case using the framework of [6, 7].

Considering the structure of residual polynomials in the CGS [19] and BiCGSTAB [23] methods, Zhang [24] derived generalized product-type method based on BiCG (GPBi-CG) without enlarging both computation and storage cost. In this paper, we present a block GPBi-CG (Bl-GPBiCG) algorithm for solving non-symmetric linear systems with multiple right-hand sides. As the block BiCGSTAB algorithm [9], the use of matrix-valued orthogonal polynomials will allow us to derive the Bl-GPBiCG algorithm. In addition, we derive the bounds for the residual norm of Bl-GPBiCG iterations.

The paper is organized as follows. In Section 2, we shortly review the Bl-BiCG method and show the connection between this method and the matrix-valued polynomials. In Section 3, we first briefly review the GPBiCG method. Then, we present the Bl-GPBiCG algorithm and compare the memory requirements and computational costs of Bl-GPBiCG with those of Bl-BiCGSTAB and Bl-BiCG. In Section 4, an upper bound for residual norm of Bl-GPBiCG algorithm according to the residual norm of Bl-GMRES is presented. Also, we obtained some results about convergence behavior of Bl-GPBiCG residual norm when matrix A is positive real. In Section 5, some numerical examples are presented to show the efficiency of the method and to confirm the theoretical results. Finally, we make some concluding remarks in Section 6.

Throughout this paper, we use the following notations. For two $n \times s$ matrices X and Y, we define the following inner product: $\langle X, Y \rangle_F = \mathrm{tr}(X^T Y)$, where $\mathrm{tr}(Z)$ denotes the trace of the square matrix Z. The associated norm is the Frobenius norm denoted by $\| \ . \ \|_F$. We will use the notation $\langle ., . \rangle_2$ for the usual inner product in $\mathbb{R}^n$ and the related norm will be denoted by $\| \ . \ \|_2$. For a matrix $V \in \mathbb{R}^{n \times s}$, the block Krylov subspace $\mathcal{K}_k(A, V)$ is defined by $\mathcal{K}_k(A, V) =$

span$\{V, AV, A^2V, \ldots, A^{k-1}V\}$. Finally, 0_s, I_s, and $0_{l\times s}$ will denote the zero, the identity, and zero matrices in $\mathbb{R}^{s\times s}$, $\mathbb{R}^{s\times s}$, and $\mathbb{R}^{l\times s}$, respectively.

2 Matrix-valued polynomials and block BiCG algorithm

2.1 Block BiCG method

In this section, we remind the block BiCG (Bl-BiCG) algorithm which was first proposed by O'Leary [15] for solving the problem (1.1). Let $X_0 \in \mathbb{R}^{n\times s}$ be an initial matrix for the solution of system (1.1) with the residual $R_0 = B - AX_0$ and let $\tilde{R}_0$ be an arbitrary $n \times s$ matrix. The Bl-BiCG algorithm constructs two block bases $P_0, \ldots, P_k$ and $\tilde{P}_0, \ldots, \tilde{P}_k$ of the block Krylov subspaces $\mathcal{K}_{k+1}(A, R_0)$ and $\mathcal{K}_{k+1}(A^T, \tilde{R}_0)$,respectively. The algorithm can be summarized as follows [15].

Algorithm 1 Bl-BiCG.

1. Compute $R_0 = B - AX_0$ for a given X_0
2. $\tilde{R}_0$ is an arbitrary $n \times s$ matrix
3. Set $P_0 = R_0$ and $\tilde{P}_0 = \tilde{R}_0$
4. **for** $k = 1, 2, \ldots$ until convergence do
5. $\quad \alpha_k = (\tilde{P}_k^T A P_k)^{-1} \tilde{R}_k^T R_k$
6. $\quad X_{k+1} = X_k + P_k \alpha_k$
7. $\quad R_{k+1} = R_k - A P_k \alpha_k$
8. $\quad \tilde{\alpha}_k = (P_k^T A^T \tilde{P}_k)^{-1} R_k^T \tilde{R}_k$
9. $\quad \tilde{R}_{k+1} = \tilde{R}_k - A^T \tilde{P}_k \tilde{\alpha}_k$
10. $\quad \beta_k = (\tilde{R}_k^T R_k)^{-1} \tilde{R}_{k+1}^T R_{k+1}$
11. $\quad \tilde{\beta}_k = (R_k^T \tilde{R}_k)^{-1} R_{k+1}^T \tilde{R}_{k+1}$
12. $\quad P_{k+1} = R_{k+1} + P_k \beta_k$
13. $\quad \tilde{P}_{k+1} = \tilde{R}_{k+1} + \tilde{p}_k \beta_k$
14. **end**

Algorithm 1 breaks down if the matrices $\tilde{P}_k^T A P_k$ or $\tilde{R}_k^T R_k$ are singular. The matrix residuals and matrix directions generated by Algorithm 1 satisfy the following properties.

Proposition 1 [15] *If no breakdown occurs, the matrices computed by the Bl-BiCG algorithm satisfy the following relations:*

(1) $\tilde{R}_i^T R_j = 0$ *and* $\tilde{P}_i^T A P_j = 0$ *for* $i < j$.
(2) $span\{P_0, \ldots, P_k\} = span\{R_0, \ldots, A^k R_0\} = \mathcal{K}_{k+1}(A, R_0)$.
(3) $span\{\tilde{P}_0, \ldots, \tilde{P}_k\} = span\{\tilde{R}_0, \ldots, A^{T^k} \tilde{R}_0\} = \mathcal{K}_{k+1}(A^T, \tilde{R}_0)$.
(4) $R_k - R_0 \in \mathcal{K}_k(A, R_0)$ *and the columns of* R_k *are orthogonal to* $\mathcal{K}_k(A^T, \tilde{R}_0)$.

In the sequel, we assume that no breakdown occurs in the Bl-BiCG algorithm. In the next section, we use matrix-valued polynomials to give an expression of the

matrices computed by Bl-BiCG. This will be useful for defining the block GPBi-CG algorithm.

2.2 Matrix-valued polynomials and Bl-BiCG method

Let $\mathcal{P}$ be a matrix-valued polynomial of degree k defined by

$$\mathcal{P}(t) = \sum_{i=0}^{k} t^i \Omega_i, \quad \Omega_i \in \mathbb{R}^{s\times s}, \quad t \in \mathbb{R}. \tag{2.1}$$

As in [8, 9], we use the notation $\circ$ for the product

$$\mathcal{P}(A) \circ Y = \sum_{i=0}^{k} A^i Y \Omega_i, \tag{2.2}$$

where Y is an $n \times s$ matrix. With these definitions, we have the following propositions.

Proposition 2 [8, 9] *Let $\mathcal{P}$ and $\mathcal{Q}$ be two matrix-valued polynomials and let Y and Θ be two matrices of dimensions $n \times s$ and $s \times s$, respectively. Then we have*

$$\begin{aligned}(\mathcal{P}\mathcal{Q})(A) \circ Y &= \mathcal{Q}(A) \circ (\mathcal{P}(A) \circ Y),\\ (\mathcal{P}(A) \circ Y)\Theta &= (\mathcal{P}\Theta)(A) \circ Y,\\ (\mathcal{P} + \mathcal{Q})(A) \circ Y &= \mathcal{P}(A) \circ Y + \mathcal{Q}(A) \circ Y.\end{aligned}$$

As shown in [9], by using matrix-valued polynomials, the residual $R_k^{Bl-BiCG}$ and matrix direction $P_k^{Bl-BiCG}$ can be expressed as follows:

$$R_k^{Bl-BiCG} = \mathcal{R}_k(A) \circ R_0, \quad P_k^{Bl-BiCG} = \mathcal{P}_k(A) \circ R_0. \tag{2.3}$$

These matrix polynomials are also related by the recurrence formulas:

$$\mathcal{R}_{k+1}(t) = \mathcal{R}_k(t) - t\mathcal{P}_k(t)\alpha_k, \tag{2.4}$$

$$\mathcal{P}_{k+1}(t) = \mathcal{R}_{k+1}(t) + \mathcal{P}_k(t)\beta_k, \tag{2.5}$$

with $\mathcal{R}_0(t) = \mathcal{P}_0(t) = I_s$ for $t \in \mathbb{R}$, α_k and β_k are $s \times s$ matrices defined in Bl-BiCG algorithm.

As in [8, 9], let $\mathcal{C}$ and $\mathcal{C}^{(1)}$ be functionals defined on the set of matrix-valued polynomials with coefficients in $\mathbb{R}^{s\times s}$ and given by

$$\mathcal{C}(\mathcal{P}) = \tilde{R}_0^T(\mathcal{P}(A) \circ R_0), \tag{2.6}$$

$$\mathcal{C}^{(1)}(\mathcal{P}) = \mathcal{C}(t\mathcal{P}), \tag{2.7}$$

where $\mathcal{P}$ is a matrix-valued polynomial. With these definitions, we have the following proposition:

Proposition 3 [8, 9] *The functional $\mathcal{C}$ defined above satisfies the following properties:*

$$\begin{aligned}\mathcal{C}(\mathcal{P} + \mathcal{Q}) &= \mathcal{C}(\mathcal{P}) + \mathcal{C}(\mathcal{Q}),\\ \mathcal{C}(\mathcal{P}\Omega) &= \mathcal{C}(\mathcal{P})\Omega, \quad \Omega \in \mathbb{R}^{s\times s}.\end{aligned}$$

The same relations are also satisfied by $\mathcal{C}^{(1)}$.

The next proposition shows that the matrix-valued polynomials $\mathcal{R}_k$ and $\mathcal{P}_k$ belong to the family of formal orthogonal polynomials with respect to $\mathcal{C}$ and $\mathcal{C}^{(1)}$, respectively.

Proposition 4 [8, 9] *Let* $(\mathcal{R}_k)$ *and* $(\mathcal{P}_k)$, $(k \geq 1)$ *be the sequences of matrix-valued polynomials defined by the relations* (2.4) *and* (2.5)*, respectively. If* $\mathcal{T}_i$ *is an arbitrary matrix-valued polynomial of degree* i, $i = 0, 1, \ldots, k-1$, *then we have the following orthogonality properties:*

$$\mathcal{C}(\mathcal{R}_k \mathcal{T}_i) = 0, \quad \text{for } i < k, \tag{2.8}$$

$$\mathcal{C}^{(1)}(\mathcal{P}_k \mathcal{T}_i) = 0, \quad \text{for } i < k. \tag{2.9}$$

Now, using these matrix-valued polynomials we are ready to explain block GPBi-CG algorithm.

3 The block GPBi-CG algorithm

In this section, we first present a brief of the GPBi-CG method [24] for solving the nonsingular system $Ax = b$. Let x_0 and $r_0 = b - Ax_0$ denote the initial guess and the corresponding initial residual, respectively. Then, the residual vector r_k^{BiCG} generated by BiCG is expressed by $r_k^{BiCG} = R_k(A)r_0$, where $R_k(t)$ is the residual polynomial of BiCG. It is a multiple of the so-called Bi-Lanczos polynomial [20], which satisfies the recurrence relation

$$R_0(t) = 1, \quad R_1(t) = (1 - \alpha_0 t)R_0(t),$$
$$R_{k+1}(t) = (1 + \frac{\beta_{k-1}}{\alpha_{k-1}}\alpha_k - \alpha_k t)R_k(t) - \frac{\beta_{k-1}}{\alpha_{k-1}}\alpha_k R_{k-1}(t), \quad k = 1, 2, \ldots$$

for certain coefficients α_k and β_{k-1}. The residual vector r_k of the GPBi-CG method is expressed as $H_k(A)r_k^{BiCG}$ by combining BiCG with an auxiliary polynomial $H_k(t)$ of degree k. The polynomial $H_k(t)$ is chosen to speed up and stabilize convergence, while satisfying the three-term recurrence relations [24]:

$$H_0(t) := 1, \quad H_1(t) := (1 - \zeta_0 t)H_0(t),$$
$$H_{k+1}(t) := (1 + \eta_k - \zeta_k t)H_k(t) - \eta_k H_{k-1}(t), \quad k = 1, 2, \ldots.$$

Also, the two-term recurrence relation for H_k is defined as

$$H_0(t) := 1, \quad G_0(t) = 1, \tag{3.1}$$

$$H_{k+1}(t) := H_k(t) - \zeta_k t G_k(t), \tag{3.2}$$

$$G_{k+1}(t) := H_{k+1}(t) + \zeta_k \frac{\eta_{k+1}}{\zeta_{k+1}} G_k(t), \quad k = 1, 2, \ldots. \tag{3.3}$$

The recurrence coefficients η_k and ξ_k are computed to minimize the term

$$\min_{\eta_k, \xi_k} \|r_k\|_2 = \min_{\eta_k, \xi_k} \|H_k(A)r_k^{BiCG}\|_2 = \min_{\eta_k, \xi_k} \|H_k(A)R_k(A)r_0\|_2. \tag{3.4}$$

In many applications, the GPBi-CG method is indeed more efficient and robust than the BiCGSTAB method. In addition, GPBi-CG has smoother convergence behavior than BiCGSTAB. However, the total computational time can be higher than that of the BiCGSTAB method due to the solution of the minimization problem (3.4).

According to the two-term recurrence relations (3.1)–(3.3), Zhang [24] could derive the GPBi-CG algorithm which is as follows:

Algorithm 2 GPBi-CG Algorithm.

1. Initialization: select initial guess x_0 and compute $r_0 = b - Ax_0$
2. Select $\tilde{r}_0$ such that $(\tilde{r}_0, r_0) \neq 0$, e.g., $\tilde{r}_0 = r_0$; and set $t_{-1} = w_{-1} = 0, \beta_{-1} = 0$;
3. **for** $i = 0, 1, ...$ until $\| r_n \| \leq \varepsilon \| b \|$ do:
4. $p_i = r_i + \beta_{i-1}(p_{i-1} - u_{i-1})$
5. $\alpha_i = \frac{(\tilde{r}_0, r_i)}{(\tilde{r}_0, Ap_i)}$
6. $y_i = t_{i-1} - r_i - \alpha_i w_{i-1} + \alpha_i Ap_i$
7. $t_i = r_i - \alpha_i Ap_i$
8. $\zeta_i = \frac{(y_i,y_i)(At_i,t_i)-(y_i,t_i)(At_i,y_i)}{(At_i,At_i)(y_i,y_i)-(y_i,At_i)(At_i,y_i)}$
9. $\eta_i = \frac{(At_i,At_i)(y_i,t_i)-(y_i,At_i)(At_i,t_i)}{(At_i,At_i)(y_i,y_i)-(y_i,At_i)(At_i,y_i)}$
10. (if $i = 0$, then $\zeta_i = \frac{(At_i,t_i)}{(At_i,At_i)}$, $\eta_i = 0$)
11 $u_i = \zeta_i Ap_i + \eta_i(t_{i-1} - r_i + \beta_{i-1}u_{i-1})$
12. $z_i = \zeta_i r_i + \eta_i z_{i-1} - \alpha_i u_i$
13. $x_{i+1} = x_i + \alpha_i p_i + z_i$
14. $r_{i+1} = t_i - \eta_i y_i - \zeta_i At_i$
15. $\beta_i = \frac{\alpha_i}{\zeta_i} \cdot \frac{(\tilde{r}_0, r_{i+1})}{(\tilde{r}_0, r_i)}$
16. $w_i = At_i + \beta_i Ap_i$
17. **end**

Now, we define block GPBi-CG (Bl-GPBiCG) method that is a generalization of the single right-hand side GPBi-CG algorithm. This algorithm is smoother than Bl-BiCGSTAB method.

The Bl-GPBiCG algorithm generates the iterates whose residual matrices are as follows:

$$\begin{aligned} R_k &= (\mathcal{H}_k\mathcal{R}_k)(A) \circ R_0 \\ &= (\mathcal{H}_k(A)\mathcal{R}_k(A)) \circ R_0, \end{aligned} \tag{3.5}$$

where $\mathcal{H}_k$ is an accelerating scalar polynomial which is computed as the following recurrence:

$$\begin{aligned} \mathcal{H}_0(t) &= 1, \quad \mathcal{G}_0(t) = \zeta_0, \\ \mathcal{H}_k(t) &= \mathcal{H}_{k-1}(t) - t\mathcal{G}_{k-1}(t), \\ \mathcal{G}_k(t) &= \zeta_k\mathcal{H}_k(t) + \eta_k\mathcal{G}_{k-1}(t), \quad k = 1, 2, ... \end{aligned} \tag{3.6}$$

The scalar parameters ζ_k and η_k will be determined by a local residual minimization condition. Now, we attempt to compute $R_{k+1} = (\mathcal{H}_{k+1}\mathcal{R}_{k+1})(A) \circ R_0$. Using the

same derivation technique in the GPBi-CG [24], we can get the following recurrence relations:

$$\begin{aligned}
\mathcal{H}_{k+1}\mathcal{R}_{k+1} &= \mathcal{H}_k\mathcal{R}_{k+1} - \eta_k t\mathcal{G}_{k-1}\mathcal{R}_{k+1} - \zeta_k t\mathcal{H}_k\mathcal{R}_{k+1} \\
&= \mathcal{H}_k\mathcal{R}_k - t\mathcal{H}_k\mathcal{P}_k\alpha_k - t\mathcal{G}_k\mathcal{R}_{k+1}, && (3.7)\\
\mathcal{H}_k\mathcal{R}_{k+1} &= \mathcal{H}_k\mathcal{R}_k - t\mathcal{H}_k\mathcal{P}_k\alpha_k, && (3.8)\\
t\mathcal{G}_k\mathcal{R}_{k+2} &= \mathcal{H}_k\mathcal{R}_{k+1} - \mathcal{H}_{k+1}\mathcal{R}_{k+1} - t\mathcal{H}_k\mathcal{P}_{k+1}\alpha_{k+1} \\
&\quad + t\mathcal{H}_{k+1}\mathcal{P}_{k+1}\alpha_{k+1}, && (3.9)\\
\mathcal{H}_{k+1}\mathcal{P}_{k+1} &= \mathcal{H}_{k+1}\mathcal{R}_{k+1} + \mathcal{H}_k\mathcal{P}_k\beta_k - t\mathcal{G}_k\mathcal{P}_k\beta_k, && (3.10)\\
t\mathcal{H}_k\mathcal{P}_{k+1} &= t\mathcal{H}_k\mathcal{R}_{k+1} + t\mathcal{H}_k\mathcal{P}_k\beta_k, && (3.11)\\
t\mathcal{G}_k\mathcal{P}_k &= \zeta_k t\mathcal{H}_k\mathcal{P}_k + \eta_k(\mathcal{H}_{k-1}\mathcal{R}_k - \mathcal{H}_k\mathcal{R}_k + t\mathcal{G}_{k-1}\mathcal{P}_{k-1}\beta_{k-1}), && (3.12)\\
\mathcal{G}_k\mathcal{R}_{k+1} &= \zeta_k\mathcal{H}_k\mathcal{R}_k + \eta_k\mathcal{G}_{k-1}\mathcal{R}_k - t\mathcal{G}_k\mathcal{P}_k\alpha_k. && (3.13)
\end{aligned}$$

By using the auxiliary iterates

$$\begin{aligned}
T_k &= (\mathcal{H}_k\mathcal{R}_{k+1})(A)\circ R_0, \quad Y_k = A(\mathcal{G}_{k-1}\mathcal{R}_{k+1})(A)\circ R_0, \\
P_k &= (\mathcal{H}_k\mathcal{P}_k)(A)\circ R_0, && (3.14)\\
W_k &= A(\mathcal{H}_k\mathcal{P}_{k+1})(A)\circ R_0, \quad U_k = A(\mathcal{G}_k\mathcal{P}_k)(A)\circ R_0, \\
Z_k &= (\mathcal{G}_k\mathcal{R}_{k+1})(A)\circ R_0, && (3.15)
\end{aligned}$$

and the recurrence relations (3.7)–(3.13), we obtain iterative sequences of the Bl-GPBiCG as follows:

$$\begin{aligned}
R_{k+1} &= T_k - \eta_k Y_k - \zeta_k A T_k \\
&= R_k - A P_k\alpha_k - A Z_k, && (3.16)\\
T_k &= R_k - A P_k\alpha_k, && (3.17)\\
Y_{k+1} &= T_k - R_{k+1} - W_k\alpha_{k+1} + A P_{k+1}\alpha_{k+1}, && (3.18)\\
P_{k+1} &= R_{k+1} + (P_k - U_k)\beta_k, && (3.19)\\
W_k &= A T_k + A P_k\beta_k, && (3.20)\\
U_k &= \zeta_k A P_k + \eta_k(T_{k-1} - R_k + U_{k-1}\beta_{k-1}), && (3.21)\\
Z_k &= \zeta_k R_k + \eta_k Z_{k-1} - U_k\alpha_k. && (3.22)
\end{aligned}$$

From (3.16), we have the formula to update the approximate solution X_{k+1}:

$$X_{k+1} = X_k + P_k\alpha_k + Z_k. \tag{3.23}$$

Now, we consider the computation of the $s \times s$ matrix coefficients α_k and β_k which are needed in recurrence. Using Propositions 3, 4 and the relations (3.8), (3.11), and the fact that $\mathcal{H}_k$ is a scalar polynomial of degree k, we have

$$\mathcal{C}(\mathcal{H}_k\mathcal{R}_k) = \mathcal{C}^{(1)}(\mathcal{H}_k\mathcal{P}_k)\alpha_k, \tag{3.24}$$

and

$$\mathcal{C}^{(1)}(\mathcal{H}_k\mathcal{R}_{k+1}) = -\mathcal{C}^{(1)}(\mathcal{H}_k\mathcal{P}_k)\beta_k. \tag{3.25}$$

Using the definitions of the functionals $\mathcal{C}$ and $\mathcal{C}^{(1)}$, these relations can be rewritten as follows:

$$(\tilde{R}_0^T A P_k)\alpha_k = \tilde{R}_0^T R_k, \tag{3.26}$$

and

$$(\tilde{R}_0^T A P_k)\beta_k = -\tilde{R}_0^T A T_k. \tag{3.27}$$

Therefore, the $s \times s$ matrix coefficients α_k and β_k can be computed by solving two $s \times s$ linear systems with the same coefficient matrix $(\tilde{R}_0^T A P_k)$. In addition, we can easily show that the matrix coefficient β_k can be computed by

$$\beta_k = -\alpha_k (\tilde{R}_0^T R_k)^{-1} (\tilde{R}_0^T R_{k+1})/\zeta_k.$$

Finally, we compute the scalar parameters η_k and ζ_k by minimizing the following two-dimension problem:

$$f(\zeta, \eta) = \| R_{k+1} \|_F = \| T_k - \eta_k Y_k - \zeta_k A T_k \|_F . \tag{3.28}$$

Putting all these relations together, the Bl-GPBiCG algorithm can be summarized as follows:

Algorithm 3 Bl-GPBiCG algorithm.

1. Choose an initial guess X_0 and some $\tilde{R}_0$, and compute $R_0 = B - AX_0$
2. Set $T_{-1} = W_{-1} = 0_{N\times s}$, $\beta_{-1} = 0$
3. **for** $k = 0, 1, 2, \ldots$until convergence
4. $\quad P_k = R_k + (P_{k-1} - U_{k-1})\beta_{k-1}$
5. $\quad$ Solve $(\tilde{R}_0^T A P_k)\alpha_k = \tilde{R}_0^T R_k$
6. $\quad Y_k = T_{k-1} - R_k - W_{k-1}\alpha_k + A P_k \alpha_k$
7. $\quad T_k = R_k - A P_k \alpha_k$
8. $\quad \eta_k = \frac{(AT_k, AT_k)_F (Y_k, T_k)_F - (Y_k, AT_k)_F (AT_k, T_k)_F}{(AT_k, AT_k)_F (Y_k, Y_k)_F - (Y_k, AT_k)_F (AT_k, Y_k)_F}$
9. $\quad \zeta_k = \frac{(Y_k, Y_k)_F (AT_k, T_k)_F - (Y_k, T_k)_F (AT_k, Y_k)_F}{(AT_k, AT_k)_F (Y_k, Y_k)_F - (Y_k, AT_k)_F (AT_k, Y_k)_F}$
$\qquad$ (if $k = 0$, then $\eta_k = 0$, $\zeta_k = \frac{(AT_k, T_k)_F}{(AT_k, AT_k)_F}$)
10. $\quad U_k = \zeta_k A P_k + \eta_k (T_{k-1} - R_k + U_{k-1}\beta_{k-1})$
11. $\quad Z_k = \zeta_k R_k + \eta_k Z_{k-1} - U_k \alpha_k$
12. $\quad X_{k+1} = X_k + P_k \alpha_k + Z_k$
13. $\quad R_{k+1} = T_k - \eta_k Y_k - \zeta_k A T_k$
14. $\quad$ Solve $(\tilde{R}_0^T A P_k)\beta_k = -\tilde{R}_0^T A T_k$
15. $\quad W_k = A T_k + A P_k \beta_k$
16. **end**

We end this section by giving an analysis of the computational cost and memory requirement of the Bl-GPBiCG algorithm. For solving the linear system (1.1), the Bl-GPBiCG algorithm requires per iteration the evaluation of $2s$ matrix-vector products with A and a total of $10ns^2 + 11ns + O(s^3)$ multiplications. The storage space required (excluding those of A, X, and B) by Bl-GPBiCG is $9ns + O(s^2)$. In order to compare these results with those of [9], we collected in Table 1 the major computational costs (multiplications) per iteration for the Bl-BiCG, the Bl-BiCGSTAB, and Bl-GPBiCG algorithms. As we observe, in each iteration, the bl-GPBiCG algorithm is more expensive than the block BiCG and the block BiCGSTAB algorithms. On the other hand, the parameters ζ and η, with respect to the residual polynomial of Bl-GPBiCG, are chosen to minimize the residual Frobenius norm per iteration;

Table 1 Memory requirements and computational costs (multiplications)

Costs	Bl-BiCG	Bl-BiCGSTAB	Bl-GPBiCG
Mat-Vec with A	s	$2s$	$2s$
Mat-Vec with A^T	s	–	–
Multiplications	$8ns^2 + O(s^3)$	$6ns^2 + 4ns + O(s^3)$	$10ns^2 + 11ns + O(s^3)$
Memory locations	$5ns + O(s^2)$	$4ns + O(s^2)$	$9ns + O(s^2)$

hence, we expect the Bl-GPBiCG algorithm converges slightly faster than Bl-BiCGSTAB. In Section 5, the numerical experiments show that Bl-GPBiCG is better than Bl-BiCGSTAB in terms of the number of iterations required for convergence.

4 Convergence analysis of Bl-GPBiCG

In this section, we find an upper bound for the residual norm of Bl-GPBiCG. To obtain this bound, we define the parameters $\bar{\zeta}_{j+1} = \zeta_j$, $\bar{\eta}_{j+1} = \eta_j$, and matrices

$$H_{j+1} = \mathcal{H}_j(A)R^{Bl-BiCG}, \qquad G_{j+1} = \frac{1}{\zeta_j}\mathcal{G}_j(A)R^{Bl-BiCG},$$

for $j = 0, \ldots, k$. Using the relations (3.6), the iterates H_j and G_j can be generated by the following recurrence formulas:

$$\begin{aligned} H_1 &= R_k^{Bl-BiCG}, \quad G_1 = R_k^{Bl-BiCG}, \\ H_{j+1} &= H_j - \bar{\zeta}_j A G_j, \\ G_{j+1} &= H_{j+1} + \frac{\bar{\eta}_{j+1}}{\bar{\zeta}_{j+1}}\bar{\zeta}_j G_j, \quad j = 1, 2, ..., k. \end{aligned} \tag{4.1}$$

By assuming that $\bar{\zeta}_j \neq 0, j = 1, ..., k+1$, we have $H_{j+1}, G_{j+1} \in \mathbb{K}_{j+1}(A, R_k^{Bl-BiCG})$. In addition, we get

$$H_{k+1} = R_k^{Bl-GPBiCG}. \tag{4.2}$$

We note that $Z \in \mathbb{K}_j(A, R_k^{Bl-BiCG})$ means that

$$Z = \sum_{i=0}^{j-1} \xi_i A^i R_k^{Bl-BiCG},$$

where $\xi_i \in \mathbb{R}$, for $i = 0, \ldots, j-1$. We recall that the minimal polynomial of $V \in \mathbb{R}^{n\times s}$ with respect to A is the nonzero monic polynomial of lowest degree such that $p(A)V = 0_{n\times s}$. The degree of this polynomial is called the grade of V.

Under the assumption that all the generated coefficients ζ_j are not zero and the grade μ of $R_k^{Bl-BiCG}$ with respect to A is not less than k, the recurrence formulas (4.1) determine the matrices

$$\bar{H}_k = [H_1, H_2, \ldots, H_k] \qquad \text{and} \qquad \bar{G}_k = [G_1, G_2, \ldots, G_k],$$

whose "matrix columns" H_j and G_j are linear independent, respectively. By defining $\gamma_{j+1} = \bar{\zeta}_j \bar{\eta}_{j+1} \bar{\zeta}_{j+1}^{-1}$, for $j = 1, 2, ..., k$, the recurrence formulas (4.1) can be written as follows:

$$A\bar{G}_k = \bar{H}_k \bar{L}_k \bar{\Lambda}_k^{-1} - \bar{\zeta}_k^{-1} H_{k+1} \bar{E}_k^T, \qquad \bar{H}_k = \bar{G}_k \bar{U}_k, \tag{4.3}$$

where $\bar{\Lambda}_k = \text{diag}[\bar{\zeta}_1 I_s, \ldots, \bar{\zeta}_k I_s]$, $\bar{E}_k^T = [0_s, \ldots, 0_s, I_s] \in \mathbb{R}^{s \times ks}$,

$$\bar{L}_k = \begin{pmatrix} I_s & & & \\ -I_s & I_s & & \\ & -I_s & \ddots & \\ & & \ddots & I_s \\ & & & -I_s \ \ I_s \end{pmatrix} \quad \text{and} \quad \bar{U}_k = \begin{pmatrix} I_s & -\gamma_2 I_s & & \\ & I_s & & \\ & & \ddots & \\ & & \ddots & -\gamma_k I_s \\ & & & I_s \end{pmatrix}. \tag{4.4}$$

Combining the two equations in (4.3), we obtain

$$A\bar{H}_k = \bar{H}_k \bar{S}_k - \bar{\zeta}_k^{-1} H_{k+1} \bar{E}_k^T, \tag{4.5}$$

where $\bar{S}_k = \bar{L}_k \bar{\Lambda}_k^{-1} \bar{U}_k$ is an invertible tridiagonal block matrix such that

$$\bar{E}_k^T \bar{S}_k^{-1} \bar{E}_1 = \bar{E}_k^T \bar{U}_k^{-1} \bar{\Lambda}_k \bar{L}_k^{-1} \bar{E}_1 = \bar{\zeta}_k I_s,$$

where $\bar{E}_1^T = [I_s, 0_s, \ldots, 0_s] \in \mathbb{R}^{s \times ks}$. Now, to get the bound for residual $R_k^{Bl-GPBiCG}$, we explain some lemmas.

Lemma 1 *Assume that* $A\bar{H}_k = \bar{H}_k \bar{S}_k - \bar{\zeta}_k^{-1} H_{k+1} \bar{E}_k^T$, *with* $\bar{E}_k^T \bar{S}_k^{-1} \bar{E}_1 = \bar{\zeta}_k I_s$. *Then, for any matrix-valued polynomial*

$$\mathcal{P}(t) = \sum_{i=0}^{k} t^i \Omega_i, \quad \Omega_i \in \mathbb{R}^{s \times s}, \quad t \in \mathbb{R}$$

of degree not exceeding k, we have

$$\mathcal{P}(A) \circ H_1 = \bar{H}_k \mathcal{P}(\bar{S}_k) \circ E_1 + H_{k+1} c_k, \tag{4.6}$$

where $c_k = (-1)^k (\bar{\zeta}_1 ... \bar{\zeta}_k \| H_1 \|_F)^{-1} \Omega_k$.

Proof The proof is similar to that of Lemma 3.2 in [21]. □

In the sequel, we assume that the matrix $\bar{H}_{k+1}$ is of full rank and $\bar{H}_{k+1}^+ = (\bar{H}_{k+1}^T \bar{H}_{k+1})^{-1} \bar{H}_{k+1}^T$. From Lemma 1, we have the following relation for H_{k+1}.

Lemma 2 *Assume that* $A\bar{H}_k = \bar{H}_k \bar{S}_k - \bar{\zeta}_k^{-1} H_{k+1} \bar{E}_k^T$, *with* $\bar{E}_k^T \bar{S}_k^{-1} \bar{E}_1 = \bar{\zeta}_k I_s$, *and* $\bar{V} = [I_{ks} \quad 0_{ks \times s}] \bar{H}_{k+1}^+$. *Then for any matrix-valued polynomial* $\mathcal{P}(t)$ *of degree not exceeding k with* $\mathcal{P}(0) = I_s$, *we have*

$$H_{k+1} = (I_n - A\bar{H}_k \bar{S}_k^{-1} \bar{V}) \mathcal{P}(A) \circ H_1. \tag{4.7}$$

Proof The proof is similar to that of Lemma 3.3 in [21]. □

Using this lemma, the relation (4.3), and the definition of H_1, the main result on bounding the residual norm of Bl-GPBiCG can be stated in the following theorem.

Theorem 1 *Suppose* $A\bar{H}_k = \bar{H}_k\bar{S}_k - \bar{\zeta}_k^{-1}H_{k+1}\bar{E}_k^T$ *and* $\bar{V} = [I_{ks} \quad 0_{ks\times s}]\bar{H}_{k+1}^{+}$. *If the matrix* $\bar{H}_{k+1}$ *is of full rank, then we have*

$$\|R_k^{Bl-GPBiCG}\|_F \leq \|\bar{M}_k\|_F \,\|\, \mathcal{P}(A)\circ R_k^{Bl-BiCG}\,\|_F, \tag{4.8}$$

where $\mathcal{P}(t)$ *is a matrix-valued polynomial of degree not exceeding* k *with* $\mathcal{P}(0) = I_s$ *and* $\bar{M}_k = I_n - (\bar{H}_k - H_{k+1}\bar{F}_k^T)\bar{V}$ *with* $\bar{F}_k^T = [I_s, I_s, \ldots, I_s] \in \mathbb{R}^{s\times ks}$.

Proof Let $\bar{M}_k = I_n - A\bar{H}_k\bar{S}_k^{-1}\bar{V}$. By using (4.5) and the fact that

$$\bar{E}_k^T\bar{U}_k^{-1} = \bar{E}_k^T, \qquad \bar{E}_k^T\bar{\Lambda}_k = \bar{\zeta}_k\bar{E}_k^T, \qquad \text{and} \qquad \bar{E}_k^T\bar{L}_k^{-1} = \bar{F}_k^T,$$

we have

$$\begin{aligned}\bar{M}_k &= I_n - A\bar{H}_k\bar{S}_k^{-1}\bar{V}\\ &= I_n - (\bar{H}_k - \bar{\zeta}_k^{-1}H_{k+1}\bar{E}_k^T\bar{S}_k^{-1})\bar{V}\\ &= I_n - (\bar{H}_k - \bar{\zeta}_k^{-1}H_{k+1}\bar{E}_k^T\bar{U}_k^{-1}\bar{\Lambda}_k\bar{L}_k^{-1})\bar{V}\\ &= I_n - (\bar{H}_k - H_{k+1}\bar{F}_k^T)\bar{V}.\end{aligned}$$

This together with (4.7) and (4.2) implies the desired relation (4.8) □

Suppose that $\mathbb{P}_{k,s}$ is the space of matrix-valued polynomials $\mathcal{P}_k$ of degree not greater than k and order s, such that $\mathcal{P}_k(0) = I_s$ (i.e., $\mathcal{P}_k(t) = I_s + \sum_{i=1}^{k} t^i\Omega_i$). The k'th residual of block GMRES (Bl-GMRES) algorithm applied to the system (1.1) can be written as follows:

$$R_k^{Bl-GMRES} = \mathcal{P}_k^{BG}(A)\circ R_0,$$

where the matrix-valued residual polynomial $\mathcal{P}_k^{BG}(t) \in \mathbb{P}_{k,s}$ solves the minimization problem

$$\min_{\mathcal{P}_k\in\mathbb{P}_{k,s}} \|\mathcal{P}_k(A)\circ R_0\|_F$$

(see [18]). By using the matrix-valued residual polynomial of block GMRES algorithm applied to the system (1.1) with $X_0 = X_k^{Bl-BiCG}$, the relation (4.8) can be written as follows:

$$\|R_k^{Bl-GPBiCG}\|_F \leq \|\bar{M}_k\|_F \,\|\, \mathcal{P}_k^{BG}(A)\circ R_k^{Bl-BiCG}\,\|_F\,. \tag{4.9}$$

In addition, by using the residual polynomial of global GMRES [10] algorithm (denoted by $\mathcal{P}_k^{GG}(t)$) applied to the system (1.1) with $X_0 = X_k^{Bl-BiCG}$, we have

$$\|R_k^{Bl-GPBiCG}\|_F \leq \|\bar{M}_k\|_F \,\|\, \mathcal{P}_k^{BG}(A)\circ R_k^{Bl-BiCG}\,\|_F \leq \|\bar{M}_k\|_F \,\|\, \mathcal{P}_k^{GG}(A)\circ R_k^{Bl-BiCG}\,\|_F\,. \tag{4.10}$$

In Section 5, the numerical results are presented and the confirmation of the theoretical results is illustrated.

4.1 Bounds for the positive real matrix

First, by considering the steps 7 and 12 of Algorithm 1, and defining $\bar{R}_{k+1} = [R_0 \quad R_1 \quad ... \quad R_k]$, where R_i, $i = 0, ..., k$, are Bl-BiCG residual matrices, as in Bl-GPBiCG algorithm (the relation 4.5), for Bl-BiCG algorithm, we can get

$$A\bar{R}_k = \bar{R}_k \hat{T}_k - R_{k+1}\alpha_k^{-1}\bar{E}_k^T,$$

with

$$\hat{L}_k = \bar{L}_k, \quad \hat{U}_k = \begin{pmatrix} I_s & -\beta_1 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & I_s & -\beta_{k-1} \\ & & & & I_s \end{pmatrix},$$

$\hat{\Lambda}_k = \text{diag}[\alpha_1, ..., \alpha_k]$ and $\hat{T}_k = \hat{L}_k \hat{\Lambda}_k^{-1} \hat{U}_k$, where α_i and β_i, $i = 1, ..., k$, are $s \times s$ matrices obtained from Bl-BiCG algorithm. In addition, we have $\bar{E}_k^T \hat{T}_k^{-1} \bar{E}_1 = \alpha_k$.

Now, by assuming that $\bar{R}_{k+1}$ is of full rank, and considering $\bar{R}_{k+1}^+ = (\bar{R}_{k+1}^T \bar{R}_{k+1})^{-1} \bar{R}_{k+1}^T$, it is easy to adapt the lemmas 1 and 2 and explain the following theorem which provides a bound for the Bl-BiCG residuals.

Theorem 2 *Suppose* $A\bar{R}_k = \bar{R}_k \hat{T}_k - R_{k+1}\alpha_k^{-1}\bar{E}_k^T$ *and* $\hat{V} = [I_{ks} \quad 0_{ks\times s}]\bar{R}_{k+1}^+$. *If the matrix* $\bar{R}_{k+1}$ *is of full rank, then we have*

$$\| R_k^{Bl-BiCG} \|_F \leq \| \bar{N}_k \|_F \, \| \mathcal{P}(A) \circ R_0 \|_F, \tag{4.11}$$

where $\mathcal{P}(t)$ *is a matrix-valued polynomial of degree not exceeding* k *with* $\mathcal{P}(0) = I_s$ *and* $\bar{N}_k = \| I_n - (\bar{R}_k - R_k \bar{F}_k^T)\hat{V}$ *with* $\bar{F}_k^T = [I_s, I_s, \ldots, I_s] \in \mathbb{R}^{s\times ks}$.

Proof The proof is similar to that of Theorem 1. □

Finally, by using the matrix-valued residual polynomial of block GMRES algorithm applied to the system (1.1) with X_0, the relation (4.11) can be written as follows:

$$\| R_k^{Bl-BiCG} \|_F \leq \| \bar{N}_k \|_F \, \| \mathcal{P}_k^{BG}(A) \circ R_0 \|_F, \tag{4.12}$$

In [7], the authors established the following theorem about the convergence properties of the block GMRES method.

Theorem 3 *Assume that* A *is positive real with respect to the inner product* $\langle ., . \rangle_F$. *Then for* $k = 1, 2, ...$ *the block GMRES residuals* $R_k^{BG} = B - AX_k^{BG}$ *satisfy*

$$\| R_k^{BG} \|_F \leq (1 - \frac{\gamma^2}{\nu_{max}})^{1/2} \| R_{k-1}^{BG} \|_F, \tag{4.13}$$

where

$$\gamma := \min\{\frac{Re(\langle V, AV \rangle_F)}{\langle V, V \rangle_F} : V \in \mathbb{C}^{n\times s}, V \neq 0\},$$

$$\nu_{max} := \max\{\frac{Re(\langle AV, AV \rangle_F)}{\langle V, V \rangle_F} : V \in \mathbb{C}^{n\times s}, V \neq 0\}.$$

Now, based on the above observations, we can state the following theorem about the convergence properties of the block GPB-iCG.

Theorem 4 *Assume that A is positive real with respect to the inner product $\langle ., . \rangle_F$. Then for $k = 1, 2, ...$ the block GP-BiCG residuals $R_k^{Bl-GPBiCG}$ satisfy*

$$\|R_k^{Bl-GPBiCG}\|_F \leq \|\bar{M}_k\|_F \|\bar{N}_k\|_F (1 - \frac{\gamma^2}{\nu_{max}})^k \|R_0\|_F, \tag{4.14}$$

and the convergence is guaranteed.

Proof The result immediately follows from (4.9), (4.12), and (4.13). □

5 Numerical experiments

In this section, numerical examples are presented to illustrate the effectiveness of the block GPBi-CG to solve (1.1). All the numerical experiments were performed in double precision floating point arithmetic in MATLAB R2017b. The machine we have used is a Intel(R) Core(TM) i7, CPU 3.60 GHz, 16.00 GB of RAM. In all the examples, the starting guess was taken to be 0. We consider the right-hand side $B = \text{rand}(n, s)$, where function rand creates an $n \times s$ random matrix with coefficients uniformly distributed in [0, 1]. No preconditioning has been used in any of the test problems. The stopping criterion

$$\frac{\|R_k\|_F}{\|R_0\|_F} \leq 10^{-9}$$

Table 2 Test problems information

	Matrix	Property				Matrix	Property		
		Order	nnz	sym			Order	nnz	sym
1	nos4	100	594	Yes	12	poli4	33,833	73,249	No
2	rdb200l	200	1120	No	13	zhao1	33,861	166,453	No
3	psmigr_ 3	3140	543,160	No	14	cage11	39,082	559,722	No
4	poisson3Da	13,514	352,762	No	15	poisson3Db	85,623	2,374,949	No
5	appu	14,000	1,853,104	No	16	torso2	115,967	1,033,473	No
6	wathen100	30,401	471,601	Yes	17	cage12	130,228	2,032,536	No
7	wathen120	36,441	565,761	Yes	18	FEM_3D_thermal2	147,900	3,489,300	No
8	gr_30_30	900	7744	Yes	19	crashbasis	160,000	1,750,416	No
9	add32	4960	19,848	No	20	torso3	259,156	4,429,042	No
10	cage10	11,397	150,645	No	21	language	399,130	1,216,334	No
11	poli3	16,955	37,849	No	22	cage13	445,315	7,479,343	No

was used. For the tests, a set of 22 problems were taken from the University of Florida Sparse Matrix Collection [4]. These matrices with their generic properties are given in Table 2.

Example 1 In this example, we used the matrices

$$A_1(i, j) = tridiag(1, 4, 1), \qquad A_2(i, j) = \frac{0.5}{n - i - j + 1.5},$$

and the matrices 1−7 of Table 2. In Table 3, we give the ratio $t(s)/t(1)$, for $s = 5, 10, 15, 20, 30$, where $t(s)$ is the CPU time for Bl-GPBiCG algorithm and $t(1)$ is the CPU time obtained when applying GPBi-CG for one right-hand side linear system. Note that the time obtained by GPBi-CG for one right-hand side depends on which right-hand was used. So, in our experiments, $t(1)$ is the average of the times needed for the s right-hand sides using GPBi-CG. We note that Bl-GPBiCG is effective if the indicator $t(s)/t(1)$ is less than s. Table 3 shows that the Bl-GPBiCG algorithm is effective and less expensive than the GPBi-CG algorithm applied to each right-hand side.

Example 2 In the second example, we consider matrix A_1 from Example 1 and matrix A_3 which represents the 5-point discretization of the operator

$$L(u) = -\Delta u + 2\delta_1 u_x + 2\delta_2 u_y - \delta_3 u$$

on the unit square $[0, 1] \times [0, 1]$ with homogeneous Dirichlet boundary conditions [3, 12]. The discretization was performed using a grid size of $h = 1/65$ which yields a matrix of dimension n = 4096; we chose $\delta_1 = 2$, $\delta_2 = 4$ and $\delta_3 = 0$.

Also, we used the matrices 8−14 of Table 2. In this example, we examine the residual bounds given in (4.10). The results are presented in Table 4. In this table, UB^{Bl} and UB^{Gl} denote the upper bounds obtained by applying the block GMRES and global GMRES to the system (1.1) with $X_0 = X_k^{Bl-BiCG}$, respectively. As can

Table 3 Effectiveness of Bl-GPBiCG algorithm measured by $t(s)/t(1)$

Matrix \ s	5	10	15	20	30
A_1, n=1000	2.39	3.56	4.74	5.10	7.45
A_2, n=1000	2.84	3.67	7.62	4.08	5.79
nos4	1.73	1.41	1.75	1.48	1.26
rdb200l	3.47	4.12	3.54	3.99	3.96
psmigr_3	2.69	4.49	6.04	7.43	11.36
poisson3Da	3.45	6.59	9.79	12.41	18.52
appu	3.41	6.48	10.60	13.09	19.47
wathen100	4.70	8.40	11.41	14.72	19.67
wathen120	4.70	8.21	10.94	12.98	18.92

Table 4 The upper bounds of the residual norm for $s = 2, 4, 8$ for Example 2 matrices

Matrix \ s		2	4	8
	UB^{Bl}	4.9757e−07	4.9594e−07	4.8497e−07
A_1, n=1000	UB^{Gl}	5.0615e−07	5.0690e−07	4.9982e−07
	$\| R_k \|_F$	2.5198e−08	3.2259e−08	4.1136e−08
	UB^{Bl}	1.8998e−10	7.8244e−11	1.3526e−10
A_3, n=4096	UB^{Gl}	1.0771e−09	1.7696e−10	9.4841e−10
	$\| R_k \|_F$	1.2146e−10	3.9362e−11	1.3095e−10
	UB^{Bl}	2.0443e−06	7.6744e−06	3.5291e−06
gr_30_30	UB^{Gl}	2.6341e−06	1.0679e−05	6.1416e−06
	$\| R_k \|_F$	1.9448e−07	1.3534e−07	1.6547e−07
	UB^{Bl}	4.4649e−06	2.6993e−06	1.9119e−06
add32	UB^{Gl}	4.6439e−06	3.1868e−06	3.3039e−06
	$\| R_k \|_F$	5.4183e−07	5.1987e−07	8.4386e−07
	UB^{Bl}	5.9681e−06	2.3144e−05	3.2691e−05
cage10	UB^{Gl}	7.7135e−06	3.2498e−05	6.5722e−05
	$\| R_k \|_F$	2.4823e−07	1.0239e−06	1.3101e−06
	UB^{Bl}	1.6640e−06	4.1834e−06	1.0865e−06
poli3	UB^{Gl}	2.9444e−06	1.0545e−05	5.1889e−06
	$\| R_k \|_F$	7.3270e−08	3.5905e−07	8.5699e−08
	UB^{Bl}	4.8318e−07	4.9825e−06	1.2262e−05
poli4	UB^{Gl}	7.6575e−07	3.5743e−05	4.6450e−05
	$\| R_k \|_F$	1.9097e−08	4.9041e−07	4.1436e−07
	UB^{Bl}	9.4143e−06	1.0651e−05	1.1526e−05
zhao1	UB^{Gl}	1.0605e−05	1.4499e−05	2.9009e−05
	$\| R_k \|_F$	2.4241e−07	4.7453e−07	1.8329e−06
	UB^{Bl}	2.8079e−05	3.5318e−05	2.5783e−05
cage11	UB^{Gl}	3.4395e−05	5.1646e−05	6.2362e−05
	$\| R_k \|_F$	1.1850e−06	1.5646e−06	1.4635e−06

be seen, the upper bounds UB^{Bl} and UB^{Gl} are the suitable bounds for the residual norm ($\| R_k \|_F$) of the Bl-GPBiCG method.

Example 3 For the third set of experiments, we compared the Frobenius norm of the residuals ($\|R_k\|_F$), the number of iterations (Iter),and the CPU time in seconds (CPU) for convergence for the block GPBi-CG, the block BiCGSTAB, and the block BiCG methods. We used the large matrices of Table 2. The results obtained by these algorithms are presented in Table 5. In this table, a dagger (†) indicates that no convergence is achieved after 2500 iterations. As can be seen from Table 5, the methods are similar in Frobenius norm of residuals ($\|R_k\|_F$). In addition, as expected, for all problems the number of iterations of the block GPBi-CG is less than that of the block BiCGSTAB and the block BiGG . The CPU time obtained for the block BiCGSTAB

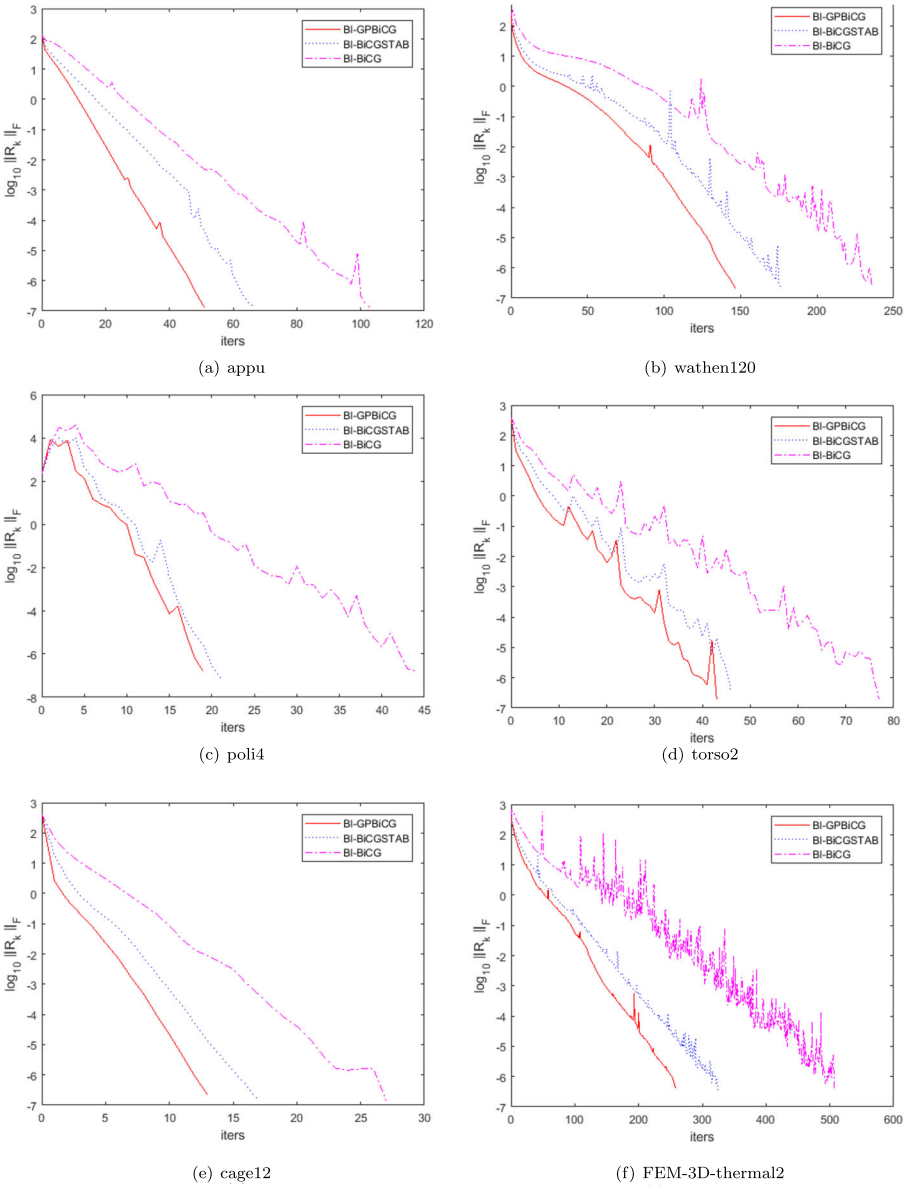
Table 5 Numerical results for Example 3

		Bl-BiCG			Bl-BiCGSTAB			Bl-GPBiCG		
Matrix \ s		2	4	8	2	4	8	2	4	8
poisson3Da	CPU(Iter)	0.35(211)	†	†	0.17(103)	0.27(94)	†	0.21(95)	0.30(81)	0.61(72)
	$\| R_k \|_F$	8.777e−08	0.0315	396.545	8.078e−08	7.343e−08	4.183e+15	9.161e−08	9.982e−08	7.934e−08
appu	CPU(Iter)	0.71(104)	1.37(103)	2.75(107)	0.47(65)	0.90(67)	2.06(76)	0.38(52)	0.74(51)	1.53(49)
	$\| R_k \|_F$	8.247e−08	1.242e−07	1.263e−07	8.732e−08	1.355e−07	5.487e−08	7.202e−08	1.286e−07	1.877e−07
wathen100	CPU(Iter)	0.65(288)	0.95(246)	2.67(201)	0.36(184)	0.56(162)	1.70(155)	0.55(183)	0.84(140)	2.50(121)
	$\| R_k \|_F$	1.331e−07	1.276e−07	2.601e−07	1.418e−07	1.598e−07	2.367e−07	1.363e−07	1.989e−07	2.741e−07
wathen120	CPU(Iter)	0.81(301)	1.87(236)	3.65(219)	0.56(226)	1.16(177)	1.96(140)	0.72(195)	1.74(147)	3.37(129)
	$\| R_k \|_F$	1.149e−07	2.189e−07	2.145e−07	1.284e−07	2.076e−07	1.699e−07	1.486e−07	2.007e−07	2.628e−07
poli3	CPU(Iter)	0.03(29)	0.05(28)	0.16(25)	0.01(17)	0.02(16)	0.06(15)	0.02(14)	0.03(15)	0.13(14)
	$\| R_k \|_F$	1.757e−08	1.222e−07	5.588e−08	1.131e−08	9.783e−09	8.452e−09	7.327e−08	1.085e−08	9.793e−09
poli4	CPU(Iter)	0.06(41)	0.28(44)	0.48(34)	0.03(22)	0.09(21)	0.17(18)	0.05(21)	0.18(19)	0.35(17)
	$\| R_k \|_F$	3.633e−08	1.593e−07	1.482e−07	1.424e−07	7.854e−08	6.178e−08	1.787e−08	1.586e−07	1.025e−07
zhao1	CPU(Iter)	0.08(47)	0.30(47)	0.62(48)	0.11(81)	0.33(71)	1.97(200)	0.06(24)	0.23(24)	0.51(25)
	$\| R_k \|_F$	5.669e−08	1.014e−07	2.797e−07	3.717e−08	1.692e−07	2.738e−07	4.138e−08	7.196e−08	1.312e−07
cage11	CPU(Iter)	0.08(28)	0.28(30)	0.48(26)	0.05(17)	0.12(17)	0.26(18)	0.06(14)	0.19(14)	0.40(14)
	$\| R_k \|_F$	6.340e−08	4.062e−08	3.829e−08	1.144e−07	2.061e−07	1.522e−08	4.591e−08	5.035e−08	5.843e−08
poisson3Db	CPU(Iter)	6.56(335)	†	†	3.68(209)	†	†	4.76(187)	7.58(167)	12.78(141)
	$\| R_k \|_F$	1.480e−07	134.623	5.466e+04	2.100e−07	2.958e+11	9.750e+21	2.391e−07	3.033e−07	4.495e−07

Table 5 (continued)

		Bl-BiCG			Bl-BiCGSTAB			Bl-GPBiCG		
Matrix \ s		2	4	8	2	4	8	2	4	8
torso2	CPU(Iter)	1.13(81)	2.02(77)	†	0.53(47)	1.00(46)	2.04(43)	0.90(42)	1.81(43)	3.49(41)
	$\| R_k \|_F$	2.462e−07	1.877e−07	10.137	1.941e−07	3.528e−07	4.984e−07	2.100e−07	1.919e−07	2.128e−07
cage12	CPU(Iter)	0.46(25)	0.91(27)	1.70(25)	0.26(17)	0.49(17)	0.95(17)	0.36(13)	0.69(13)	1.38(13)
	$\| R_k \|_F$	2.860e−07	1.397e−07	2.890e−07	1.216e−07	1.402e−07	1.690e−07	1.949e−07	2.214e−07	2.514e−07
FEM_3D_thermal2	CPU(Iter)	13.16(572)	20.30(507)	†	6.27(318)	11.41(324)	†	9.68(296)	15.73(258)	30.03(246)
	$\| R_k \|_F$	2.427e−07	3.652e−07	1.210e+03	2.912e−07	3.185e−07	5.054e+11	3.030e−07	4.107e−07	6.047e−07
crashbasis	CPU(Iter)	†	†	†	5.49(327)	14.05(428)	†	8.60(278)	17.17(285)	35.92(304)
	$\| R_k \|_F$	2.455e+05	7.853e+04	5.9293+06	2.510e−07	1.444e−07	2.970e+37	1.380e−07	4.390e−07	4.291e−07
torso3	CPU(Iter)	12.30(319)	†	†	6.71(206)	10.38(176)	18.35(162)	8.17(147)	14.57(140)	25.71(126)
	$\| R_k \|_F$	3.159e−07	604.387	134.184	3.388e−07	5.492e−07	4.597e−07	3.867e−07	4.505e−07	5.477e−07
language	CPU(Iter)	1.94(41)	4.49(45)	8.81(41)	1.08(27)	2.06(27)	3.82(26)	1.77(23)	3.54(24)	7.29(25)
	$\| R_k \|_F$	3.683e−07	4.466e−07	7.366e−07	4.169e−07	1.237e−08	4.591e−09	1.373e−07	2.329e−07	5.788e−08
cage13	CPU(Iter)	1.99(29)	3.39(26)	7.78(27)	1.01(17)	1.93(18)	3.68(18)	1.48(14)	2.70(14)	5.68(15)
	$\| R_k \|_F$	1.803e−07	4.887e−07	2.772e−07	2.755e−07	2.432e−07	4.240e−07	1.823e−07	2.222e−07	1.731e−07

is smaller than the one for the block GPBi-CG (except for examples appu and zhao1 with $s = 2, 4, 8$, poisson3Db with $s = 4, 8$, poisson3Da, FEM_3D_thermal2, and crashbasis with $s = 8$). We also observe that the block GPBi-CG algorithm needs less CPU time than the block BiCG.



(a) appu (b) wathen120

(c) poli4 (d) torso2

(e) cage12 (f) FEM-3D-thermal2

Fig. 1 Convergence history of Bl-GPBiCG, Bl-BiCGSTAB, and Bl-BiCG algorithms for some matrices of Table 2 with s=4. (a) appu (b) wathen120 (c) poli4 (d) torso2 (e) cage12 (f) FEM-3D-thermal2

In Fig. 1, we display the convergence history of Bl-GPBiCG, Bl-BiCGSTAB, and Bl-BiCG algorithms for some matrices of Table 2 with $s = 4$. In this figure, the horizontal axis is the number of iterations (iters) and the vertical axis is the logarithm of the Frobenius norm of residuals ($\log_{10} \parallel R_k \parallel_F$). As can be seen from Fig. 1, the convergence of Bl-GPBiCG method is faster (in terms of the number of iterations required for convergence) than Bl-BiCGSTAB and Bl-BiCG methods and even is smoother.

6 Conclusion

In this paper, we have derived the block GPBi-CG method for nonsymmetric linear systems with multiple right-hand sides. Also, we have proposed the suitable upper bounds for the residual norm of the method in terms of block GMRES residual norm and global residual norm. In order to earn the proposed bounds, we set H_1 and G_1 equal to the k'th residual of Bl-BiCG method and defined H_j and G_j, $j = 2, ..., k+1$ similar to R_j and P_j. Then, by using two lemmas, we obtained the upper bounds. In addition, we proved that convergence is guaranteed when A is positive real. Finally, experimental results showed the effectiveness of Bl-GPBiCG method and confirmed the theoretical results.

Acknowledgements We would like to thank the referees for their valuable remarks and helpful suggestions.

References

1. Addam, M., Heyouni, M., Sadok, H.: The block Hessenberg process for matrix equations. Electron. Trans. Numer. Anal. **46**, 460–473 (2017)
2. Amini, S., Toutounian, F., Gachpazan, M.: The block CMRH method for solving nonsymmetric linear systems with multiple right-hand sides. J. Comput. Appl. Math. **337**, 166–174 (2018)
3. Calvetti, D., Golub, G.H., Reichel, L.: Adaptive Chebyshev iterative methods for nonsymetric linear systems based on modified moments. Numer. Math. **67**, 21–40 (1994)
4. Davis, T., Hu, Y.: The university of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**, 1–25 (2011). Avaiable online at http://www.cise.ufl.edu/research/sparse/matrices/list$_$by$_$id.html
5. Freund, R.W., Malhotra, M.: A Block-QMR algorithm for non-hermitian linear systems with multiple right-hand sides. Linear Algeb. Appl. **254**, 119–157 (1997)
6. Frommer, A., Lund, K., Szyld, D.B.: Block Krylov subspace methods for functions of matrices. Electron. Transact. Numer. Anal. **47**, 100–126 (2017). https://doi.org/10.1553/etna-vol47s100
7. Frommer, A., Lund, K., Szyld, D.B.: Block Krylov subspace methods for functions of matrices II: Modified block FOM, tech. report, MATHICSE cublens. https://doi.org/10.5075/ep-MATHICSE-265508 (2019)
8. El Guennouni, A., Jbilou, K., Sadok, H.: The block Lanczos method for linear systems with multiple right-hand sides. J. App. Numer. Math. **51**, 243–256 (2004)
9. El Guennouni, A., Jbilou, K., Sadok, H.: A block version of BiCGSTAB for linear systems with multiple right-hand sides. Electron. Trans. Numer. Anal. **16**, 129–142 (2003)
10. Jbilou, K., Messaoudi, A., Sadok, H.: Global FOM GMRES algorithms for matrix equations. Appl. Numer. Math. **31**, 49–63 (1999)
11. Karimi, S., Toutounian, F.: The block least squares method for solving nonsymmetric linear systems with multiple right-hand sides. Appl. Math. Comput. **177**, 852–862 (2006)

12. Koulaei, M.H., Toutounian, F.: On computing of block ILU preconditioner for block tridiagonal systems. J. Comput. Appl. Math. **202**, 248–257 (2007)
13. Kubínová, M., Soodhalter, K.M.: Admissible and attainable convergence behavior of block Arnoldi and GMRES. Siam J. Matrix Anal. Appl. **41**, 464–486 (2019)
14. Nikishin, A., Yeremin, A.: Variable block cg algorithms for solving large sparse symmetric positive definite linear systems on parallel computers, I: General iterative scheme. SIAM J. Matrix Anal. Appl. **16**, 1135–1153 (1995)
15. OLeary, D.: The block conjugate gradient algorithm and related methods. Linear Algeb. Appl. **29**, 293–332 (1980)
16. Saad, Y. Iterative Methods for Sparse Linear Systems, 2nd edn. SIAM, Philadelphia (2003)
17. Simoncini, V.: A stabilized QMR version of block BiCG. SIAM J. Matrix Anal. Appl. **18**, 419–434 (1997)
18. Simoncini, V., Gallopoulos, E.: Convergence properties of block GMRES and matrix polynomials. Linear Algeb. Appl. **247**, 97–119 (1996)
19. Sonneveld, P.: CGS, A fast Lanczos-type solver for nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. **10**, 36–52 (1989)
20. Stiefel, E.L.: Kernel polynomial in linear algebra and their numerical applications. In: Further contributions to the determination of eigenvalues, vol. 49, pp. 1–22 (1958)
21. Tong, C.H., Ye, Q.: Analysis Of the finite precision bi-Conjugate gradient algorithm for nonsymmetric linear systems. Math. Comput. **69**, 1559–1575 (1999)
22. Toutounian, F., Mojarrab, M.: The block LSMR method: A novel efficient algorithm for solving non-symmetric linear systems with multiple right-hand sides. Iran J. Sci. Technol. **39**, 69–78 (2015)
23. Van der Vorst, H.A.: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. **13**, 631–644 (1992)
24. Zhang, S.-L.: GPBi-CG: generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems. SIAM J. Sci. Comput. **18**, 537–551 (1997)