A New Projected Variant of the Deflated Block Conjugate Gradient Method

Article *in* Journal of Scientific Computing · August 2019

DOI: 10.1007/s10915-019-00969-4

CITATIONS
6

READS
459

3 authors:

Yan-Fei Xiang
University of Strasbourg
8 PUBLICATIONS **10** CITATIONS
SEE PROFILE

Yan-Fei Jing
University of Electronic Science and Technology of China
57 PUBLICATIONS **405** CITATIONS
SEE PROFILE

Ting-Zhu Huang
University of Electronic Science and Technology of China
740 PUBLICATIONS **13,020** CITATIONS
SEE PROFILE

Journal of Scientific Computing (2019) 80:1116–1138
https://doi.org/10.1007/s10915-019-00969-4

A New Projected Variant of the Deflated Block Conjugate Gradient Method

Yan-Fei Xiang[1] · **Yan-Fei Jing**[1] · **Ting-Zhu Huang**[1]

Received: 18 August 2018 / Revised: 17 April 2019 / Accepted: 25 April 2019 / Published online: 4 May 2019

Abstract
The deflated block conjugate gradient (D-BCG) method is an attractive approach for the solution of symmetric positive definite linear systems with multiple right-hand sides. However, the orthogonality between the block residual vectors and the deflation subspace is gradually lost along with the process of the underlying algorithm implementation, which usually causes the algorithm to be unstable or possibly have delayed convergence. In order to maintain such orthogonality to keep certain level, full reorthogonalization could be employed as a remedy, but the expense required is quite costly. In this paper, we present a new projected variant of the deflated block conjugate gradient (PD-BCG) method to mitigate the loss of this orthogonality, which is helpful to deal with the delay of convergence and thus further achieve the theoretically faster convergence rate of D-BCG. Meanwhile, the proposed PD-BCG method is shown to scarcely have any extra computational cost, while having the same theoretical properties as D-BCG in exact arithmetic. Additionally, an automated reorthogonalization strategy is introduced as an alternative choice for the PD-BCG method. Numerical experiments demonstrate that PD-BCG is more efficient than its counterparts especially when solving ill-conditioned linear systems or linear systems suffering from rank deficiency.

Keywords Block conjugate gradient · Deflated block conjugate gradient · Deflation · Projection · Rank deficiency · Reorthogonalization

This research is supported by NSFC (61772003) and Science Strength Promotion Programme of UESTC.

✉ Yan-Fei Jing
00jyfvictory@163.com

Yan-Fei Xiang
yfxiang0amber@126.com

Ting-Zhu Huang
tingzhuhuang@126.com

[1] School of Mathematical Sciences/Institute of Computational Science, University of Electronic Science and Technology of China, Chengdu 611731, Sichuan, People's Republic of China

1 Introduction

In this paper, we consider a new projected variant of the deflated block conjugate gradient method for the solution of linear systems with multiple right-hand sides given simultaneously

$$AX = B, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is a large sparse symmetric positive definite (SPD) matrix, and $B = [b^{(1)}, b^{(2)}, \ldots, b^{(s)}] \in \mathbb{R}^{n\times s}$ are the given right-hand sides. This problem may originate from various scientific and industrial applications, such as multi-objective optimization [16], electro-magnetics [14], direct current resistivity problem [11], and the real-life packaging problem [32]. The block conjugate gradient (BCG) method [30] has always been considered to be attractive [10,24,30,41] for solving (1) as a block version of the conjugate gradient (CG) method [23]. However, as observed in [30], the block matrices involved in BCG may suffer from linear dependencies along with the algorithm implementation, which results in the so-called breakdown problem. That is, the algorithm terminates early without finding a satisfactory approximate solution. Refer to [4,6–8] for more comprehensive descriptions of such breakdown issue, which arouses many interests in developing various strategies [10,28,30]. In particular, based on the potential reduced block search subspaces, Ji and Li formulate the parameter matrices in a novel way to handle this breakdown issue gracefully, resulting in the breakdown-free block conjugate gradient (BFBCG) method [24].

Instead of grouping the multiple right-hand sides into blocks, another approach to solve (1) is to apply deflation [13,19] techniques to CG to solve each individual system sequentially, like deflated CG methods [25,29,35] and augmented CG methods [5,17,36]. Actually, the later type of methods can be viewed as a particular case of the deflated CG method. Moreover, deflation can be also used in related methods for solving nonsymmetric and more general cases, like GMRES-DR and its block variants [3,20,26,27,37–40], as well as subspace recycling methods [31,34]. A typical deflation technique is to inject to the Krylov subspace a deflation subspace containing a few approximate eigenvectors usually corresponding to the eigenvalues close to the origin, which are deemed to hamper the convergence.

In practice, when using the deflated CG method to solve the SPD systems with multiple right-hand sides, the approximate subspace is obtained during the course of CG iteration and dynamically updated once a system is solved. And it is reused immediately for deflation to solve the next system sequentially [35]. If the approximate subspace is expensive to construct, another strategy is to run a separate single-vector Lanczos algorithm to compute the desired deflation subspace [2,10]. In fact, deflation can be considered as a method that implicitly modifies the spectrum of the original matrix and reduces its condition number to improve the convergence behaviour [19].

However, as pointed out by Saad et al. [35], the deflated variant for CG method does not always behave so well due to the gradual loss of orthogonality during the algorithm implementation. And the same issue has again been mentioned that the loss of orthogonality between the block residual vectors and the deflation subspace in practise hampers convergence seriously when Chen developed the deflated version of the block conjugate gradient (D-BCG) method [10]. As a matter of fact, this issue could be worse particularly when solving ill-conditioned systems, since the accumulated round-off errors in computation may be magnified greatly [9,22,30], and thus the loss of orthogonality may be so high to ruin convergence. In order to deal with this catastrophe, both Saad et al. [35] and Chen [10] use reorthogonalization to maintain this orthogonality. However, the reorthogonalization process is quite costly and may not be necessary [33].

In order to address the instability problem of deflated BCG variants indicated by Chen [10] and Saad et al. [35], in this paper, by means of transforming the projection procedures [12], we present a projected variant of the deflated block conjugate gradient (PD-BCG) method to mitigate the loss of orthogonality between the block residual vectors and the deflation subspace, hoping to further improve the computational convergence rate of the D-BCG [10] method. Actually, when the loss of orthogonality is mitigated, the algorithm may reach its targeted convergence threshold before the accumulated loss of orthogonality is so high to ruin convergence, which is quite significant in finite precision arithmetic. Briefly speaking, in order to construct the PD-BCG method, we implement orthogonal projection procedures of the deflation subspace against the block search vectors rather than against the (preconditioned) block residual vectors. The latter is usually used by standard deflated variants. No extra computational cost is needed for this new deflated variant compared with the standard one, because it just needs to transform the projection object of the deflation subspace during the block iteration. Moreover, the PD-BCG method also inherites the novel techniques to construct parameter matrices from the BFBCG method [24] to handle the aforementioned breakdown problem of block methods. Subsequently, an additional automated reorthogonalization is developed as an alternative choice to further make the PD-BCG method more robust and effective even with badly ill-conditioned systems and right-hand sides of rank deficiency. It is proved that the PD-BCG method is mathematically equivalent to the classical D-BCG method [10] in exact arithmetic. However, in the context of finite precision arithmetic, PD-BCG outperforms D-BCG remarkably as shown by our numerical experiments.

The paper is organized as follows. In Sect. 2, we first have a brief review of the BCG method and the D-BCG method, and discuss their orthogonality and convergence properties accordingly. In Sect. 3, we develop the PD-BCG method with an analysis of its orthogonality and convergence properties. Subsequently, we present the construction of deflation subspace and the new automated reorthogonalization strategy to be designed for the PD-BCG method. Numerical experiments are presented in Sect. 4 to demonstrate the efficiency of our proposed methods, and conclusions are given in Sect. 5.

The symbol $\|\cdot\|_2$ denotes the Euclidean norm. The superscripts T and $\perp$ respectively denote the transpose and the orthogonal space of a vector or a matrix. For convenience of the algorithm illustration and presentation, some MATLAB notation is used. A subscript k for a scalar or a matrix is used to indicate that the scalar or the matrix is obtained at iteration k. A superscript $^{(i)}$ labels the ith column of a matrix. The symbol "$\sim$" indicates that these matrices may have rank deficiency. The identity matrix is denoted by I, and block-span $\{C_1, C_2, \ldots, C_k\}$ denotes the subspace spanned by these block matrices C_i $(1 \leq i \leq k)$.

2 Deflated Block Conjugate Gradient Method

In this section, we first review the standard form of the block conjugate gradient (BCG) method proposed by O'Leary [30] and a deflated version of the block conjugate gradient (D-BCG) method presented by Chen [10]. Then, we have an analysis on associated orthogonality and convergence properties of BCG and D-BCG, which will be used to illustrate the necessity and importance to keep the orthogonality between the block residual vectors and the deflation subspace.

2.1 Block Conjugate Gradient Algorithm

We recapitulate the main results of the block conjugate gradient method [30] briefly. Given an SPD preconditioner M, full rank matrices γ_j of dimension $s \times s$, $j = 0, 1, \ldots$, an initial guess X_0 with the corresponding initial block residual $R_0 = B - AX_0$, $Z_0 = MR_0$ and initial block search direction vectors $P_0 = Z_0\gamma_0$. The standard BCG algorithm runs the following iterations until convergence:

$$\begin{aligned} X_{j+1} &= X_j + P_j\alpha_j, \\ R_{j+1} &= R_j - AP_j\alpha_j, \\ Z_{j+1} &= MR_{j+1}, \\ P_{j+1} &= (Z_{j+1} + P_j\beta_j)\gamma_{j+1}, \quad j = 0, 1, \ldots \end{aligned}$$

where the α_j and β_j are parameter matrices with various forms (one could refer to [10,24,30] for the details), the $X_j = [x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(s)}]$ are approximate solution vectors, and R_j are the corresponding block residual vectors. The matrices γ_j are arbitrary nonsingular matrices selected to orthogonalize the columns of P_j and thus to improve the numerical stability of BCG.

The orthogonality properties, such as the residual vectors R_{j+1} being orthogonal to the previous search directions P_i (i.e., $R_{j+1}^T P_i = 0$ for all $i < j + 1$), the A-orthogonality of the search directions P_j 's and the M-orthogonality of the residual vectors R_j 's (i.e., $P_i^T AP_j = 0$ and $R_i^T MR_j = 0$ for all $i \neq j$), have all been deduced by O'Leary in [30].

As for the convergence rate of BCG, define the error matrix E_j as

$$E_j = \left[e_j^{(1)}, \ldots, e_j^{(s)}\right] = X_* - X_j$$

at the jth iteration, where $e_j^{(i)}$ is the ith column of E_j and $X_* = A^{-1}B$ is the exact block solution of Eq. (1). Then according to [30, Section 4], the block residual R_j at iteration j ($j = 1, 2, 3, \ldots$) of the BCG algorithm is orthogonal to block-span$\{MR_0, (MA)MR_0, \ldots, (MA)^{j-1}MR_0\}$ denoted as $\mathscr{K}_j^M(A, R_0)$, and thus X_j is the minimizer of the error trace $\left[(X_* - X)^T A(X_* - X)\right]$ over all $X \in X_0 + \mathscr{K}_j^M$ and

$$\text{trace}\left[(X_* - X_j)^T A(X_* - X_j)\right] = \sum_{i=1}^{s} \|e_j^{(i)}\|_A^2.$$

To determine the convergence rate of BCG, the initial residual matrix R_0 plays an important role in bounding the errors at each iteration step. Under the assumption that R_0 has full column rank, O'Leary [30] showed that the minimum error in component i ($1 \leq i \leq s$) is bounded as

$$\|e_j^{(i)}\|_A^2 \leq c^{(i)} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^{2j}, \tag{2}$$

where $c^{(i)}$ is a constant only related to $e_0^{(i)}$, and $\kappa = \lambda_n/\lambda_s$. Here, for all j, the λ_j's are the jth eigenvalues of the preconditioned system $M^{1/2}AM^{1/2^T}$ (or equivalently written as MA) with eigenvalues in increasing order, and s is the column number of the right-hand sides.

2.2 Deflated Block Conjugate Gradient Algorithm

The deflated BCG (D-BCG) algorithm discussed in [10] uses an $n \times t$ ($t < n$) nonsingular matrix W for deflation, whose columns correspond to the targeted eigenvalues which are deemed to hamper the convergence. Within the framework of D-BCG, the initial vectors X_0 are determined by an orthogonal property: $R_0 = B - AX_0 \perp W$, and the relevant initial block iterates are computed as $Z_0 = MR_0$ and $P_0 = \text{orth}(Z_0 - W(W^TAW)^{-1}W^TAZ_0)$, where orth$(\cdot)$ is a matrix operation in MATLAB used to generate an orthonormal basis for the range of the underlying matrix for the consideration of numerical stability. We directly recall the D-BCG method with parameter matrices developed in [24] in Algorithm 1.

Algorithm 1 Deflated block conjugate gradient algorithm [10]

Require: a matrix $A \in \mathbb{R}^{n\times n}$, a matrix $B \in \mathbb{R}^{n\times s}$, an initial guess $X_{-1} \in \mathbb{R}^{n\times s}$, a deflation matrix $W \in \mathbb{R}^{n\times t}$, an SPD preconditioner $M \in \mathbb{R}^{n\times n}$, the targeted backward error $\varepsilon > 0$, and maximum number of iterations $maxit \in N^+$.
Ensure: an approximate solution $X_{j+1} \in \mathbb{R}^{n\times s}$.
1: Choose t linearly independent vectors $\omega_1, \ldots, \omega_t$. Define $W = [\omega_1, \ldots, \omega_t]$
2: Choose X_{-1}
3: $R_{-1} = B - AX_{-1}$
4: $X_0 = X_{-1} + W(W^TAW)^{-1}W^TR_{-1}$
5: $R_0 = B - AX_0$
6: $Z_0 = MR_0$
7: solve $(W^TAW)\mu_0 = W^TAZ_0$ for μ_0
8: $P_0 = \text{orth}(-W\mu_0 + Z_0)$
9: **for** $j = 0, 1, 2, \ldots, maxit$ **do**
10: $Q_j = AP_j$
11: $\alpha_j = (P_j^TQ_j)^{-1}(P_j^TR_j)$
12: $X_{j+1} = X_j + P_j\alpha_j$
13: $R_{j+1} = R_j - Q_j\alpha_j$
14: If converged with respect to ε, then stop.
15: $Z_{j+1} = MR_{j+1}$
16: $\beta_j = -(P_j^TQ_j)^{-1}(Q_j^TZ_{j+1})$
17: solve $(W^TAW)\mu_{j+1} = W^TAZ_{j+1}$ for μ_{j+1}
18: ${Z_{j+1}}' = -W\mu_{j+1} + Z_{j+1}$
19: $P_{j+1} = \text{orth}({Z_{j+1}}' + P_j\beta_j)$
20: **end for**

Compared with the standard BCG algorithm, the iteration in Algorithm 1 uses different search directions P_j for updating approximate solutions. Moreover, it is clear that in addition to the matrices A and M, another four block matrices are required to be stored: W, AW, W^TAW and μ. Define two auxiliary matrices as

$$H = I - W(W^TAW)^{-1}(AW)^T, \tag{3}$$

to be the A-orthogonal projector onto $W^{\perp_A}$ and

$$H^T = I - AW(W^TAW)^{-1}W^T, \tag{4}$$

to be the A^{-1}-orthogonal projector onto $W^{\perp}$. From lines 4 and 5 of Algorithm 1 and Eq. (4), the initial block residual vector is orthogonal to the deflation matrix. Moreover, the projector H satisfies the following relations based on Eqs. (3) and (4)

$$AH = H^TA = H^TAH. \tag{5}$$

By a simple induction and formula (3), it is easy to show the A-orthogonality of the P_j 's and the M-orthogonality of the residual vectors R_j 's, which inherit from the standard BCG algorithm. Besides, the residual vectors R_j and block vectors AP_j are both orthogonal to the deflation matrix W. That is, $W^T R_j = 0$ and $W^T A P_j = 0$ in step j.

As for the convergence rate of D-BCG, Chen [10, Corollary 3.3] showed the minimum error in component i $(1 \leq i \leq s)$ is bounded as

$$\|e_j^{(i)}\|_A^2 \leq c^{(i)} \left(\frac{\sqrt{\kappa_D} - 1}{\sqrt{\kappa_D} + 1} \right)^{2j}, \tag{6}$$

where $\kappa_D = \lambda_n / \lambda_{s+t}$ with t being the dimension of the deflation matrix. Compared to κ presented in (2), it is obvious that $\kappa_D < \kappa$ and the error bound obtained by (6) is smaller than that obtained by (2) in step j. According to O'Leary [30, Section 4], the prominent factor that determines the convergence rate of the BCG-like algorithms is the distribution of eigenvalues of MA (if the same initial guess X_0 is chosen). For fast convergence, M can be chosen so that the preconditioned matrix MA has a narrow cluster of $n - s + 1$ eigenvalues, which becomes a more clustered spectrum with $n - s - t + 1$ eigenvalues after deflating the targeted t eigenvalues to zero [10]. Theoretically, the convergence rate of D-BCG cannot be slower than the standard BCG method, especially when the deflation matrix W is sufficiently accurate to approximate the targeted eigenvectors which hamper the convergence behaviour. From this perspective, exploiting deflation can be considered to use an extra implicit preconditioner for the preconditioned system to improve the convergence rate.

However, due to the gradual loss of orthogonality between the block residual vectors R_j and the deflation matrix W in the finite precision arithmetic, the convergence of D-BCG seems to be unstable and we thus cannot theoretically realize the faster convergence as shown in (6). Moreover, what is worse is that the loss of this orthogonality may be so serious that it even ruins convergence [10,35]. Instead of using the costly reorthogonalization process, in the next section, we present a new projected variant of the deflated block conjugate gradient algorithm to deal with this instability of D-BCG [10] with scarcely any extra computational cost.

3 A New Projected Variant of the Deflated BCG Method

In this section, we first develop the projected variant of the deflated block conjugate gradient (PD-BCG) method, which additionally exploits the technique of the breakdown-free block conjugate gradient (BFBCG) method [24] to address the (near) breakdown problem [24,30] that may occur as the PD-BCG algorithm progresses. And then we theoretically analyze the orthogonal properties and convergence rate of the PD-BCG method. The associated construction of the deflation matrix W and the new automated reorthogonalization strategy to be taken in this paper are discussed afterwards.

3.1 Derivation of PD-BCG

For the simplicity of expression and easy cross-reading, we use most of the notation from the D-BCG method [10]. But the copperplate $\mathscr{P}_j$ is used to denote the matrix whose columns span the search space at iteration j in the remaining parts. We still ensure the initial residual

vectors to satisfy $R_0 = B - AX_0 \perp W$, and define $Z_0 = MR_0$. And then with left-multiplying the projector H expressed in (3), we start with

$$\mathscr{P}_0 = HZ_0 = (I - W(W^TAW)^{-1}W^T)Z_0,$$

which are the descent directions and orthogonal to the deflation matrix W.

In fact, once the deflation matrix W has been constructed, considering the construction of the residual vectors $R_{j+1} = R_j - A\mathscr{P}_j\alpha_j$ and the condition that $W^TR_0 = 0$ is rigorously established in D-BCG, the loss of orthogonality between the residual vectors and the deflation matrix W mainly originates from the accumulated round-off errors in finite precision computation and the computation of the search directions. Although the accumulated round-off error is inevitable, we could adjust the construction of the search space matrix $\mathscr{P}_j$ $(j = 1, 2, \ldots)$ to make sure the subspaces span$\{A\mathscr{P}_j\}$ belong to the orthogonal space of span$\{W\}$. Thus, we introduce auxiliary block search directions $\mathscr{P}_j{}'$ constructed as those of the D-BCG method (i.e., $\mathscr{P}_j{}' = HZ_j + \mathscr{P}_{j-1}\beta_{j-1}$). And then we project the auxiliary directions $\mathscr{P}_j{}'$ onto the subspace span$\{W^{\perp_A}\}$ by (3) to update the block search directions as

$$\mathscr{P}_j = H\mathscr{P}_j{}' = HZ_j + \mathscr{P}_{j-1}\beta_{j-1} - W(W^TAW)^{-1}W^TA\mathscr{P}_{j-1}\beta_{j-1}, \tag{7}$$

which indicates that the PD-BCG algorithm can be viewed as a projection method. That is, we successively ensure the A-orthogonality between the search space matrix $\mathscr{P}_j$ and the deflation matrix W with the help of projector H. Moreover, observe that if the third term in the right-hand side expression of (7) vanishes when the search vectors are orthogonal to the deflation matrix, then PD-BCG reduces to the classical D-BCG algorithm [10].

On the other hand, the breakdown problem caused by rank deficiency, which is originated from the linear dependence of the residual vectors R_j or the search space matrix $\mathscr{P}_j$, is a well-known practical difficulty during the implementation of the BCG variants. When breakdown occurs, at least one of the parameter matrices involved in the BCG variants is unavailable, leading these algorithms to terminate early without finding a satisfactory approximation. It is remarkably noticed that Ji and Li [24] recently developed a breakdown-free BCG method with novel forms of parameter matrices and the matrix operation orth$(\cdot)$, used to select an orthogonal basis, to avoid the breakdown problem cased by rank deficiency. Specifically, according to the orthogonal properties of the BCG method, the parameter matrix α_j is derived by imposing the orthogonal condition that the next residual matrix R_{j+1} be orthogonal to the current search space matrix $\mathscr{P}_j$, while the parameter matrix β_j is defined by the conjugacy condition that the new search direction matrix $\mathscr{P}_{j+1}$ is conjugate to all the previous search direction matrices $\mathscr{P}_k$ $(k < j + 1)$. Moreover, the matrix operation orth$(\cdot)$ is used to extract an orthogonal basis as the columns of $\tilde{P}_j$ from the search space matrix $\mathscr{P}_j$, which guarantees that the matrix $\tilde{P}_j^TA\tilde{P}_j$ is nonsingular and thus meaningful to obtain its inverse to compute the parameter matrices α_j and β_j. When rank deficiency associated with the search space matrix $\mathscr{P}_j$ occurs, the parameter matrices may turn into rectangular matrices of size $r_j \times s$, where r_j is the rank dimension of the current search space matrix $\mathscr{P}_j$ and is possibly less than s with s being the original number of the right-hand sides in (1). And then, there is $\mathscr{P}_j = \tilde{P}_j\sigma_j$, where σ_j is an $r_j \times s$ matrix of rank $r_j (r_j \le s)$. We here extend the main idea of addressing the breakdown issue in [24] to the PD-BCG method to handle rank deficiency.

As a summary, we present the PD-BCG method as in Algorithm 2, in which the $\tilde{P}_j$ obtained from $\mathscr{P}_j$ is used in iterations. Compared with lines 17 and 18 of Algorithm 1, where the D-BCG method implements orthogonal projection procedures of the deflation subspace against the preconditioned block residual vectors, this new projection-based deflated variant transforms the projection object of the deflation subspace into the block search vectors

alternatively, which is illustrated in lines 9 and 10 of Algorithm 2. Moreover, according to line 17 of Algorithm 1 and line 9 of Algorithm 2, if the column number of $\tilde{P}_j{}'$ equals to that of Z_{j+1}, then the computational complexity of Algorithm 2 is the same as that of Algorithm 1. Since the adoption of orth$(\cdot)$ and the possible rank deficiency of $\mathscr{P}_j{}'$ existing in actual computation could reduce the number of multiplications with A (i.e., the $A\tilde{P}_j{}'$ in line 9 of Algorithm 2), the computational complexity of Algorithm 2 would be less than that of Algorithm 1.

Algorithm 2 Projected variant of the deflated BCG algorithm

Require: a matrix $A \in \mathbb{R}^{n\times n}$, a matrix $B \in \mathbb{R}^{n\times s}$, an initial guess $X_{-1} \in \mathbb{R}^{n\times s}$, a deflation matrix $W \in \mathbb{R}^{n\times t}$, an SPD preconditioner $M \in \mathbb{R}^{n\times n}$, the targeted backward error $\varepsilon > 0$, and maximum number of iterations $maxit \in N^+$.

Ensure: an approximate solution $X_{j+1} \in \mathbb{R}^{n\times s}$.

1: Choose t linearly independent vectors $\omega_1, \ldots, \omega_t$. Define $W = [\omega_1, \ldots, \omega_t]$
2: Choose X_{-1}
3: $R_{-1} = B - AX_{-1}$
4: $X_0 = X_{-1} + W(W^TAW)^{-1}W^TR_{-1}$
5: $R_0 = B - AX_0$
6: $Z_0 = MR_0$
7: $\tilde{P}_0{}' = \text{orth}(Z_0)$
8: **for** $j = 0, 1, 2, \ldots, maxit$ **do**
9: solve $(W^TAW)\mu_j = W^TA\tilde{P}_j{}'$ for μ_j
10: $\tilde{P}_j = \tilde{P}_j{}' - W\mu_j$
11: $Q_j = A\tilde{P}_j$
12: $\alpha_j = (\tilde{P}_j^TQ_j)^{-1}(\tilde{P}_j^TR_j)$
13: $X_{j+1} = X_j + \tilde{P}_j\alpha_j$
14: $R_{j+1} = R_j - Q_j\alpha_j$
15: **If** converged with respect to ε, **then** stop.
16: $Z_{j+1} = MR_j$
17: $\beta_j = -(\tilde{P}_j^TQ_j)^{-1}(Q_j^TZ_{j+1})$
18: **end if**
19: $\tilde{P}_{j+1}{}' = \text{orth}(Z_{j+1} + \tilde{P}_j\beta_j)$
20: **end for**

3.2 Properties of PD-BCG

In this section, based on these properties of the projector H presented in Eqs. (3), (4) and (5), we first present orthogonal properties of the PD-BCG method in Theorem 1, subsequently deduce the Petrov-Galerkin condition and convergence rate of PD-BCG in Theorem 2 and Sect. 3.3, respectively.

Theorem 1 *Letting M be an SPD matrix, in Algorithm 2, we have*

(a) *R_j and $A\tilde{P}_j$ are both orthogonal to W, that is, $W^TR_j = 0$ and $W^TA\tilde{P}_j = 0$.*
(b) *R_j's are M-orthogonal and the $\tilde{P}_j$'s are A-orthogonal, that is, $R_i^TMR_j = 0$ and $\tilde{P}_i^TA\tilde{P}_j = 0$ for all $i \neq j$.*

Proof We first prove $W^TR_j = 0$ and $W^TA\tilde{P}_j = 0$ by induction. It is easy to deduce this is true for the initial iteration $j = 0$ by the given condition $R_0 \perp W$ and the definition of H

shown in Eq. (3). Assume it holds at iteration j, then at iteration $j+1$, according to line 14 of Algorithm 2, we obtain

$$W^T R_{j+1} = W^T R_j - W^T A\tilde{P}_j \alpha_j = 0$$

by induction. And according to lines 9 and 10 of Algorithm 2, we have $\tilde{P}_{j+1} = H\tilde{P}_{j+1}{}'$, then

$$W^T A\tilde{P}_{j+1} = W^T AH\tilde{P}_{j+1}{}' = 0$$

by the fact that H is the A-orthogonal projector onto $W^{\perp_A}$, which is stated in (3).

Next, from $Z_j = MR_j$, we prove $Z_j^T R_{m+1} = 0$ and $\tilde{P}_j^T A\tilde{P}_{m+1} = 0$ for all $j \le m$ by induction similarly. This is true at the initial iteration $m = 0$. Assume it is true at iteration m, then at iteration $m+1$, according to line 14 of Algorithm 2, we obtain

$$Z_j^T R_{m+1} = Z_j^T R_m - Z_j^T A\tilde{P}_m \alpha_m.$$

We first check the cases $j \le m-1$. We can get $Z_j^T R_m = 0$ by induction. According to line 19 of Algorithm 2, $Z_j + \tilde{P}_{j-1}\beta_{j-1}$ can be expressed as

$$Z_j + \tilde{P}_{j-1}\beta_{j-1} = \tilde{P}_j{}'\sigma_j,$$

where σ_j ia an $r_j \times s$ matrix of rank r_j and r_j is the rank dimension of $\tilde{P}_j{}'$. Moreover, from $\tilde{P}_m = H\tilde{P}_m{}'$, we have

$$Z_j^T A\tilde{P}_m = (\tilde{P}_j{}'\sigma_j - \tilde{P}_{j-1}\beta_{j-1})^T A\tilde{P}_m = \sigma_j^T(\tilde{P}_j{}')^T AH\tilde{P}_m{}' - \beta_{j-1}^T \tilde{P}_{j-1}^T A\tilde{P}_m = 0$$

by the property $AH = H^T AH$ shown in (5) and by induction. Then for $j = m$, we have

$$\begin{aligned} Z_m^T R_{m+1} &= (\tilde{P}_m{}'\sigma_m - \tilde{P}_{m-1}\beta_{m-1})^T (R_m - A\tilde{P}_m\alpha_m), \\ &= \sigma_m^T(\tilde{P}_m{}')^T(R_m - A\tilde{P}_m\alpha_m) = \sigma_m^T(\tilde{P}_m^T R_m - \tilde{P}_m^T A\tilde{P}_m\alpha_m) \end{aligned}$$

by induction and $R_m = H^T R_m$ obtained from $W^T R_j = 0$ and the definition of H^T shown in Eq. (4). Furthermore, we have $\tilde{P}_m^T R_m - \tilde{P}_m^T A\tilde{P}_m\alpha_m = 0$ by the definition of α_m presented in line 12 of Algorithm 2. Thus, we obtain $Z_m^T R_{m+1} = 0$.

In the following, as an alternative, we prove $\tilde{P}_j^T A\tilde{P}_{m+1}\sigma_{m+1} = 0$ to obtain $\tilde{P}_j^T A\tilde{P}_{m+1} = 0$, where σ_{m+1} is an $r_{m+1} \times s$ matrix with rank $r_{m+1}(r_{m+1} \le s)$. We first check the cases $j \le m-1$. From the expression of H in (3) and the full row rank property of σ_{m+1}, we have

$$\begin{aligned} \tilde{P}_j^T A\tilde{P}_{m+1}\sigma_{m+1} &= \tilde{P}_j^T AH(Z_{m+1} + \tilde{P}_m\beta_m)\Psi_{m+1}, \\ &= \tilde{P}_j^T AZ_{m+1}\Psi_{m+1} - \tilde{P}_j^T AW\Theta^* + \tilde{P}_j^T A\tilde{P}_m\beta_m\Psi_{m+1}, \end{aligned}$$

where Ψ_{m+1} is a $s \times s$ nonsingular matrix and $\Theta^* = (W^T AW)^{-1}(AW)^T(Z_{m+1} + \tilde{P}_m\beta_m)\Psi_{m+1}$ is a $t \times s$ matrix. By imposing a transpose on the first item of the right side, we obtain

$$(\tilde{P}_j^T AZ_{m+1}\Psi_{m+1})^T = \Psi_{m+1}^T Z_{m+1}^T A\tilde{P}_j = \Psi_{m+1}^T R_{m+1}^T M(R_j - R_{j+1})/\alpha_j = 0$$

by line 14 of Algorithm 2 and the M-orthogonality between $R_j{}'s$ just proved above. Moreover, it is clear that $\tilde{P}_j^T A\tilde{P}_m = 0$ by induction, and $\tilde{P}_j^T AW = 0$ by the proved item (a), so that $\tilde{P}_j^T A\tilde{P}_{m+1}\sigma_{m+1} = 0$. Thus, we have $\tilde{P}_j^T A\tilde{P}_{m+1} = 0$ by the full row rank property of σ_{m+1}. As for $j = m$, based on $\tilde{P}_j^T AW = 0$, we have

$$\tilde{P}_m^T A\tilde{P}_{m+1}\sigma_{m+1} = (\tilde{P}_m^T AZ_{m+1} + \tilde{P}_m^T A\tilde{P}_m\beta_m)\Psi_{m+1} = 0$$

by the definition of β_m corresponding to line 17 of Algorithm 2. □

Theorem 1 indicates that the properties $W^T A\tilde{P}_j = 0$ are theoretically well established for all $j = 0, 1, \ldots$ in the PD-BCG iterations. And so does $W^T A\mathscr{P}_j = 0$ by $\mathscr{P}_j = \tilde{P}_j\sigma_j$. Thus, the PD-BCG method is mathematically equivalent to the D-BCG method in exact arithmetic. However, although these two block deflated variants are theoretically equivalent, they perform differently in the finite precision arithmetic as shown in the numerical experiments. In fact, the PD-BCG method successively projects the subspace span$\{A\tilde{P}_j\}$ (or span$\{A\mathscr{P}_j\}$) onto the subspace $W^\perp$ by the projector H at the initial steps of each loop (i.e., $W^T A\tilde{P}_j = 0$ is guaranteed for all $j = 0, 1, \ldots$) and this characteristic could be also helpful for projecting the residual vectors R_{j+1} onto the subspace $W^\perp$ indirectly. These facts contribute to the mitigation of the gradual loss of orthogonality between the residual vectors and the deflation matrix during the implementation of D-BCG [10] so as to realize the theoretically fast convergence as far as possible.

Next, we deduce the relations between the block search space matrix $\mathscr{P}_j$ and the block residual vectors R_j. And then we derive the solution space condition of PD-BCG.

According to lines 9, 10, 16 and 19 of Algorithm 2 and the property $W^T A\tilde{P}_j = 0$ in Theorem 1, the block search space matrix $\mathscr{P}_j$ at iteration j can be written as

$$\begin{aligned}\mathscr{P}_j &= \tilde{P}_j\sigma_j = H\tilde{P}_j{}'\sigma_j = HMR_j\Psi_j + \tilde{P}_{j-1}\beta_{j-1}\Psi_j,\\ &= \cdots\\ &= HMR_j\Psi_j + HMR_{j-1}\Psi_{j-1}\Psi_j + \cdots + HMR_0\Psi_0\Psi_1\ldots\Psi_{j-1},\end{aligned}$$

by induction and in which these $\Psi_k (0 \le k \le j-1)$ are $s \times s$ nonsingular matrices. For simplicity, we set

$$\text{block-span}\,\{\mathscr{P}_0, \mathscr{P}_1, \ldots, \mathscr{P}_{j-1}\} = \text{block-span}\,\{\mathbb{P}_j\} \subset \mathbb{R}^{n\times s}, \tag{8}$$

where "block-span" means a linear combination of all the $n \times s$ columns in $\{\mathscr{P}_0, \mathscr{P}_1, \ldots, \mathscr{P}_{j-1}\}$, which is equivalent to the definition of "block span" for block Krylov space methods mentioned by Gutknecht [20, Section 6]. To further clarify this point, for $\forall C \in$ block-span $\{\mathscr{P}_0, \mathscr{P}_1, \ldots, \mathscr{P}_{j-1}\}$, we give the following definition

$$C = \sum_{k=0}^{j-1} \mathscr{P}_k\delta_k;\ \exists\delta_k \in \mathbb{R}^{s\times s}, (k = 0, \ldots, j-1) \in \mathbb{R}^{n\times s}.$$

Thus it is clear that

$$\text{block-span}\,\{\mathbb{P}_j\} = \text{block-span}\,\{HMR_0, HMR_1, \ldots, HMR_{j-1}\}. \tag{9}$$

According to line 14 of Algorithm 2, we have

$$R_{j-1} \in \text{block-span}\,\{R_{j-2}, A\tilde{P}_{j-2}\alpha_{j-2}\} \subset \text{block-span}\,\{R_{j-2}, A\mathscr{P}_{j-2}\},$$

which gives rise to

$$R_{j-1} \in \text{block-span}\,\{R_0, A\mathscr{P}_0, A\mathscr{P}_1, \ldots, A\mathscr{P}_{j-2}\}. \tag{10}$$

From (9) and (10), we conclude that

$$R_{j-1} \in \text{block-span}\,\{R_0, AHMR_0, (AHM)^2R_0, \ldots, (AHM)^{j-1}R_0\}. \tag{11}$$

Then from (5) and (11), we obtain

$$\text{block-span}\,\{R_0, R_1, \ldots, R_{j-1}\} = \mathscr{K}_j(AHM, R_0) = \mathscr{K}_j(H^T AHM, R_0).$$

Therefore, just as Erhel and Guyomarch [17] have pointed out that the deflated version of the BCG method can be viewed as BCG preconditioned by both HH^T and M.

On the other hand, from (9), it is easy to get

$$\mathscr{P}_0 \in \text{block-span}\,\{HMR_0\}.$$

Furthermore, from (9) and (11), we obtain

$$\begin{aligned}\text{block-span}\,\{\mathbb{P}_j\} &= \text{block-span}\,\{HMR_0, HM(AHM)R_0, \ldots, HM(AHM)^{j-1}R_0\},\\ &= H\cdot\text{block-span}\,\{MR_0, (MAH)MR_0, \ldots, (MAH)^{j-1}MR_0\},\\ &= \text{block-span}\,\{HMR_0, (HMA)HMR_0, \ldots, (HMA)^{j-1}HMR_0\},\end{aligned}$$

by the formula $HM(AHM)^j = H(MAH)^jM = (HMA)^jHM$. Hence, we have

$$\text{block-span}\,\{\mathbb{P}_j\} = H\cdot\mathscr{K}_j(MAH, MR_0) = \mathscr{K}_j(HMA, HMR_0). \tag{12}$$

Similarly, it is easy to conclude

$$\text{block-span}\,\{Z_0, Z_1, \ldots, Z_{j-1}\} = \mathscr{K}_j(MAH, MR_0), \tag{13}$$

where $Z_j = MR_j$. Hence, the vectors HZ_j and $\mathscr{P}_j$ span the same Krylov subspace, which corresponds to lines 17, 18 and 19 of Algorithm 1 or lines 9, 10 and 19 of Algorithm 2. It is noted that in the unpreconditioned case, that is $M = I$, we have

$$\text{block-span}\,\{R_0, R_1, \ldots, R_{j-1}\} = \mathscr{K}_j(AH, R_0) = \text{block-span}\,\{Z_0, Z_1, \ldots, Z_{j-1}\}.$$

For simplicity, from (13) and these orthogonal properties stated in Theorem 1, we define a new Krylov subspace as

$$\mathscr{K}_{t,j}(A, W, R_0) = \text{span}\,(W) \oplus \mathscr{K}_j(MAH, MR_0), \tag{14}$$

where t is the dimension of the deflation matrix W and the direct sum system $\mathscr{K}_{t,j}(A, W, R_0)$ obtained by "$\oplus$" is the union of two disjointed systems span (W) and $\mathscr{K}_j(MAH, MR_0)$.

Then, from (14), the first two terms of (12) and the definition of H in (3), we obtain the following relation

$$\text{block-span}\,\{\mathbb{P}_j\} = \mathscr{K}_{t,j}(A, W, R_0). \tag{15}$$

Next, we develop some more properties of Algorithm 2 in Theorem 2, which are used to exploit the convergence property of PD-BCG in the next subsection.

Theorem 2 *Letting M be an SPD matrix, in Algorithm 2, we have*

(a) *the solution space is*

$$X_j \in X_0 + \mathscr{K}_{t,j}(A, W, R_0).$$

(b) *The block Petrov–Galerkin condition is*

$$V^T R_j = 0 \text{ and } V^T A\mathscr{P}_j = 0 \text{ for all } V \in \mathscr{K}_{t,j}(A, W, R_0).$$

Proof According to line 13 of Algorithm 2, it can be shown that

$$\begin{aligned} X_j &= X_{j-1} + \tilde{P}_{j-1}\alpha_{j-1}, \\ &= \cdots \\ &= X_0 + \tilde{P}_0\alpha_0 + \ldots + \tilde{P}_{j-1}\alpha_{j-1}, \\ &= X_0 + \mathscr{P}_0{\Psi_0}' + \ldots + \mathscr{P}_{j-1}{\Psi_{j-1}}', \end{aligned}$$

where these ${\Psi_k}'(0 \le k \le j-1)$ are $s \times s$ coefficient matrices. Hence from Eqs. (8) and (15), we obtain

$$X_j \in X_0 + \mathscr{K}_{t,j}(A, W, R_0).$$

Then, in order to prove the property (b) of Theorem 2, from (13), (14) and (15), we could first prove $W^T R_j = 0$ and $Z_i^T R_j = R_i^T M R_j = 0$ to verify $V^T R_j = 0$. And then, from $\mathscr{P}_j = \tilde{P}_j \sigma_j$ as well as $V^T A \tilde{P}_j = 0$ or $\tilde{P}_i^T A \tilde{P}_j = 0$ for all $i \ne j$, which have all been proved in Theorem 1, we have $V^T A \mathscr{P}_j = 0$. Therefore, the Petrov-Galerkin condition still holds in Algorithm 2. □

3.3 Convergence Analysis

From Theorem 2, R_j is orthogonal to $\mathscr{K}_{t,j}(A, W, R_0)$ at iteration j. Using this property, we further obtain that the jth approximate solution X_j minimizes the solution error trace $\left[(X_* - X)^T A (X_* - X)\right]$ over the subspace $X_0 + \mathscr{K}_{t,j}(A, W, R_0)$. Noting that $X_j \in X_0 + \mathscr{K}_{t,j}(A, W, R_0)$, we can write $x_j^{(i)}$ for each ith column of X_j as a polynomial form, which has been used to analyze similar properties of block methods in [1, Section 5].

Specifically, from (15), the first and third items of (12), we can write $x_j^{(i)}$ as

$$x_j^{(i)} = x_0^{(i)} + \sum_{i=1}^{s} \mathcal{P}_j(HMA) HMA r_0^{(i)},$$

where $\mathcal{P}_j(HMA)$ is a polynomial of degree less than or equal to $j-1$. Then, under the assumption that R_0 has full column rank, O'Leary [30, Section 4] showed that the minimum error square norm $\|e_j^{(i)}\|_A^2$ $(1 \le i \le s)$ is bounded as

$$\|e_j^{(i)}\|_A^2 \le c_1 \left(\frac{\sqrt{\kappa}_1 - 1}{\sqrt{\kappa}_1 + 1} \right)^{2j}, \tag{16}$$

where $\kappa_1 = \lambda_n / \lambda_s$, λ_n and λ_s are respectively the nth and sth (s is the block size of the right-hand sides B) eigenvalues of HMA, and c_1 is a constant only related to $\|E_0\|$.

Analogously, from (15) and the first and second items of (12), we can also write $x_j^{(i)}$ in an alternative form

$$x_j^{(i)} = x_0^{(i)} + H \sum_{i=1}^{s} \mathcal{J}_j(MAH) MA r_0^{(i)},$$

where $\mathcal{J}_j(MAH)$ is a polynomial of degree less than or equal to $j-1$. Then, according to Chen [10, Section 3.4], the minimum error square norm $\|e_j^{(i)}\|_A^2$ $(1 \le i \le s)$ is bounded as

$$\|e_j^{(i)}\|_A^2 \le c_2 \left(\frac{\sqrt{\kappa}_2 - 1}{\sqrt{\kappa}_2 + 1} \right)^{2j}, \tag{17}$$

where $\kappa_2 = \lambda_n/\lambda_{s+t}$, λ_n and λ_{s+t} are respectively the nth and $(s+t)$th eigenvalues of MA, c_2 is a constant only related to $\|E_0\|$, and t is the dimension of the deflation matrix W.

In fact, with an effective preconditioner, the preconditioned system MA has almost all the eigenvalues clustered within a narrow range, except for a few extreme eigenvalues deviating from the clustered part obviously and thus hampering the convergence. In such circumstances, deflation is particularly favorable since these extreme eigenvalues associated with the eigenvectors that are the columns of W can be deflated to be zero (since $MAHW = 0$ by Eq. (3)), which contributes to further narrow the range of spectrum and thus possibly to further speed up the computational convergence rate. Therefore, the spread of the spectrum of MAH is never wider than that of MA. In the ideal case that the t columns of the deflation matrix W correspond to the eigenvectors of MA associated with the t smallest eigenvalues in magnitude, then $\kappa_1 = \kappa_2$, and the boundaries presented in (16) and (17) are mathematically equal under the same choice of the initial guess X_0. Furthermore, under the framework of exact arithmetic and such ideal case, it is easy to observe that the convergence rate of PD-BCG described in (16) or (17) is the same as that of D-BCG presented in (6). However, the derivation of PD-BCG shows its feature of mitigating the loss of orthogonality. Thus, the actual computational convergence rate of PD-BCG is possibly faster than that of D-BCG in finite precision arithmetic, which will be substantiated by the numerical results to be presented in Sect. 4. As for realizing the theoretical supporting evidence of the faster convergence rate of PD-BCG compared with D-BCG, deducing these algorithms (i.e., D-BCG and PD-BCG) in finite precision and analyzing their growth of local roundoff errors are indispensable, which could be considered as an open problem for us at the moment.

When there is rank deficiency occurring in the initial residual matrix R_0, that is R_0 has rank $r_0 < s$, the convergence rate of PD-BCG is discussed in the following theorem.

Theorem 3 *Suppose R_0 is rank deficient with rank r_0 ($r_0 < s$). The minimum error square norm $\|\mathrm{e}_j^{(i)}\|_A^2$ $(1 \le i \le s)$ is bounded as*

$$\|\mathrm{e}_j^{(i)}\|_A^2 \le \mathrm{c}_3 \left(\frac{\sqrt{\kappa_3} - 1}{\sqrt{\kappa_3} + 1} \right)^{2j},$$

where c_3 is a constant only related to $\|\mathrm{E}_0\|$, and $\kappa_3 = \lambda_n/\lambda_{r_0}$, with λ_n and λ_{r_0} being respectively the nth and r_0th eigenvalues of HMA.

Proof Assume that the $n \times s$ residual matrix R_0 has rank r_0, then there exists a nonsingular $s \times s$ matrix δ such that

$$R_0 = (R_0{}', 0)\delta,$$

where $R_0{}'$ is an $n \times r_0$ matrix with full column rank.

It is easy to deduce $E_0 = A^{-1}R_0$. By the definition of E_j and line 13 of Algorithm 2, we have

$$\begin{aligned} E_j &= X_* - (X_{j-1} + \tilde{P}_{j-1}\alpha_{j-1}), \\ &= \cdots \\ &= X_* - (X_0 + \mathscr{P}_0\Psi_0{}' + \ldots + \mathscr{P}_{j-1}\Psi_{j-1}{}'), \\ &= E_0 - \mathscr{P}_0\Psi_0{}' - \cdots - \mathscr{P}_{j-1}\Psi_{j-1}{}'. \end{aligned}$$

From $\mathscr{P}_{j-1} \in$ block-span$\{HMR_0, HM(AHM)R_0, \ldots, HM(AHM)^{j-1}R_0\}$, we obtain

$$E_j = E_0 + HMR_0{\Psi_0}'' + HM(AHM)R_0{\Psi_1}'' + \cdots + HM(AHM)^{j-1}R_0{\Psi_{j-1}}'',$$
$$= E_0 + HMAE_0{\Psi_0}'' + (HMA)^2E_0{\Psi_1}'' + \cdots + (HMA)^jE_0{\Psi_{j-1}}'', \quad (18)$$

where ${\Psi_k}''$ $(k = 0, \ldots, j-1)$ are the $s \times s$ coefficient matrices.

Therefore, by defining a polynomial of degree j as $\Phi_j(HMA)$ and assume $\Phi_0 = I$, and using (18), we have

$$E_j = \Phi_j(HMA)E_0 = \Phi_j(HMA)A^{-1}R_0,$$
$$= \Phi_j(HMA)A^{-1}({R_0}', 0)\delta,$$
$$= (\Phi_j(HMA)A^{-1}{R_0}', 0)\delta = ({E_j}', 0)\delta.$$

Hence, each column in E_j can be expressed as

$$e_j^{(i)} = \sum_{k=1}^{r_0} \delta_{k,i} e_j^{'(k)},$$

where the $e_j^{'(k)}$ is the kth $(k = 1, 2, \ldots, r_0)$ column of ${E_j}'$. Then by taking norms on both sides and according to (16), we obtain an upper bound on $\|e_j^{(i)}\|_A^2$ as

$$\|e_j^{(i)}\|_A^2 = \left\| \sum_{k=1}^{r_0} \delta_{k,i} e_j^{'(k)} \right\|_A^2,$$
$$\leq \sum_{k=1}^{r_0} \delta_{k,i}^2 \cdot \|e_j^{'(k)}\|_A^2,$$
$$\leq c_3 \left(\frac{\sqrt{\kappa_3} - 1}{\sqrt{\kappa_3} + 1} \right)^{2j},$$

where $c_3 = \sum_{k=1}^{r_0} \delta_{k,i}^2 \cdot c_1$ is only related to $\|E_0\|$, and $\kappa_3 = \lambda_n/\lambda_{r_0}$, with λ_n and λ_{r_0} being respectively the nth and r_0th eigenvalue of HMA. □

From Theorem 3 and $r_0 < s$, we have $\frac{\sqrt{\kappa_1}-1}{\sqrt{\kappa_1}+1} < \frac{\sqrt{\kappa_3}-1}{\sqrt{\kappa_3}+1}$ since $\kappa_1 < \kappa_3$.

3.4 Deflation Matrix *W*

Generally, an approximation of the extreme eigenvalues is often obtained by a separate Lanczos [10,35] or Arnoldi process [3,26]. Under the framework of the block algorithms, Chen has made a comprehensive analysis of the construction of the deflation matrix W in [10, Section 3.5], which aims at using the corresponding extreme eigenvectors of the preconditioned system MA as the columns of deflation matrix W. Based on Chen's analysis [10, Section 3.5], in this paper, a separate single-vector Lanczos algorithm presented in [10, Algorithm 2] is used to compute the desired deflation matrix W for these two deflated BCG variants (i.e., D-BCG and PD-BCG). In general, the number of columns in W is problem-dependent and is usually set to be small as compared to the dimension of A. This makes the storage and time cost of this procedure be reasonable and these extra expenses could be easily amortized by the reduced iterations stemming from the accelerated convergence rate.

For general problem, however, it could be more difficult to estimate the eigenvectors than to solve the system of equations. Besides, how accurate the approximated deflation

subspace should be is also an important problem to utilize the deflation techniques [18]. It is because the computational performance of the deflated methods may be inferior to the theoretical result stated in (16) or (17) when a badly low-accurate eigen-subspace is used as the deflation subspace, which will be supported by experiments presented in Sect. 4. In such poorly inaccurate case, full reorthogonalization related to such deflation matrix W should also be avoided because these repeated full reorthogonalization processes are likely to amplify the original errors existed in the construction of W. In other words, projecting the generated block residual matrix onto the orthogonal space of such deflation subspace step by step seems unreasonable. Obviously, the construction of W used for deflation is crucial for the deflation technique. And we refer readers to [6,10,17,21,35] for more detailed developments of this area.

3.5 Automated Reorthogonalization Strategy

Although the PD-BCG method can improve the D-BCG [10] method by mitigating the loss of orthogonality between the block residual vectors and the deflation matrix, the loss of this orthogonality is still inevitable in practical implementation. Considering the high cost and possible inapplicability (caused by the above-mentioned possibly low-accurate deflation matrix) of full reorthogonalization, we alternatively construct an additional automated reorthogonalization strategy here as a choice for the PD-BCG method, which is especially meaningful under the situation that merely mitigating the loss of orthogonality is not enough to ensure the convergence.

We first define a function at iteration j ($j = 0, 1, 2, \ldots$) as

$$\mathrm{othor}(j) = \min_{1\le l\le s}\min_{1\le i\le t}\left(\frac{|\ W(:,i)^T R_j(:,l)\ |}{\|W(:,i)\|_2\|R_j(:,l)\|_2}\right) \tag{19}$$

to monitor the level of the loss of orthogonality between the block residual vectors R_j and the deflation matrix W as these deflated block algorithms progress. It is pointed out that if a single right-hand side for (1) needs to be solved, Eq. (19) would reduce to the function othor(j) defined in [35, Section 7] for the deflated CG method, which shows the orthogonality between the single residual vector r_j and the columns of W.

In order to establish the automated reorthogonalization strategy, an adaptive criterion for this reorthogonalization should be available. In general, we assume the orthogonality between the block residual vectors and the deflation matrix maintains well during the first several iterations. Therefore, we develop an adaptive reorthogonalization criterion as

$$\mathrm{othor}(j) \ge c\cdot\sqrt{\mathrm{othor}(l)}, \qquad (j > l), \tag{20}$$

in which c is a given scale parameter used to control the level of this loss of orthogonality and l is the first iteration number satisfying $\mathrm{othor}(l) \ne 0$ (generally $l = 1$ or 2), to perform the reorthogonalization process between R_j and W automatically. Then we add the adaptive criterion (20) after line 14 of Algorithm 2. If the criterion (20) is satisfied, we implement the following operation

$$R_{j+1} = R_{j+1} - W(W^T W)^{-1} W^T R_{j+1}$$

to reorthogonalize R_{j+1} against W adaptively. For simplicity, the PD-BCG method with this automated reorthogonalization is denoted as PD-BCG-ARO.

Table 1 Information of the four sets of test problems

Ex.	*Name and group*	*ID*	*Row*	*Column*	*Nonzero*	*Cond.*
1	1138_bus/HB	1	1138	1138	4054	8.573e+06
2	cbuckle/TKK	1912	13,681	13,681	676,515	3.299e+07
3	s3rmt3m3/Cylshell	1611	5357	5357	207,123	2.401e+10
	s3rmq4m1/Cylshell	1607	5489	5489	262,943	1.766e+10

Example and condition number are respectively simplified as Ex. and *Cond.*

4 Numerical Experiments

4.1 Experimental Setting

Numerical experiments are carried out on a set of four matrices chosen from the University of Florida Sparse Matrix Collection [15]. The information of these four test matrices is described in Table 1. In order to illustrate the numerical benefits of our proposed PD-BCG method, the performance of PD-BCG is evaluated in comparison with the BCG method [30] and the D-BCG method [10] in aspects of number of matrix-vector products (referred to as *Mvps*), iteration number (referred to as *Iters*) and CPU computing time in seconds (referred to as *CPU*). Since the error is generally unavailable in practice, we compare the convergence histories of the BCG, D-BCG and PD-BCG methods in terms of the smallest and largest norms of the residuals at each iteration. Furthermore, we also plot the function othor(j) along with the iteration number to monitor the loss of orthogonality between R_j and W during the implementations of these two deflated BCG methods (i.e., D-BCG and PD-BCG). All the experiments are carried out in MATLAB 2015a with machine precision 10^{-16}.

We set the initial guess X_0 to be $0 \in \mathbb{R}^{n\times s}$ and the maximum number of matrix-vector products (i.e., the value of $maxit \times s$) to be 25, 000. We use $\|b^{(i)} - Ax_j^{(i)}\|_2/\|b^{(i)}\|_2 < \varepsilon$ $(1 \le i \le s)$ with a targeted accuracy threshold $\varepsilon = 10^{-8}$ as the stopping criterion. The dimension of W is set by trial and error. And the symbol "-" is used to indicate that the method did not meet the targeted accuracy threshold ε before *maxit* or did not converge at all.

4.2 Example 1: Experiment on Convergence Acceleration Using Deflation Under the Condition of Different Numbers of Right-Hand Sides

It is known that the block size of right-hand sides can affect the behavior of the block methods [3,6]. Thus, we first consider solving a linear system with the coefficient matrix named as *1138_bus* with different numbers of right-hand sides to compare the behaviour of the BCG, D-BCG and PD-BCG methods in order to show the improved convergence by using our projection-based deflated variant. No preconditioner is involved here. The number of the right-hand sides B (i.e., the numerical value of s) is set to be $s = 3, 6, 12, 18$, and the elements of the right-hand sides B are chosen as randomly generated numbers with standard normal distribution. The deflation matrix W used in both D-BCG and PD-BCG is constructed by implementing 33 steps of a separate single-vector Lanczos algorithm [10, Algorithm 2]. That is, $\dim(W) = 33$ or $t = 33$. Because the convergence curves of these block methods (i.e., BCG, D-BCG and PD-BCG) with such variable right-hand sides are similar, we only display

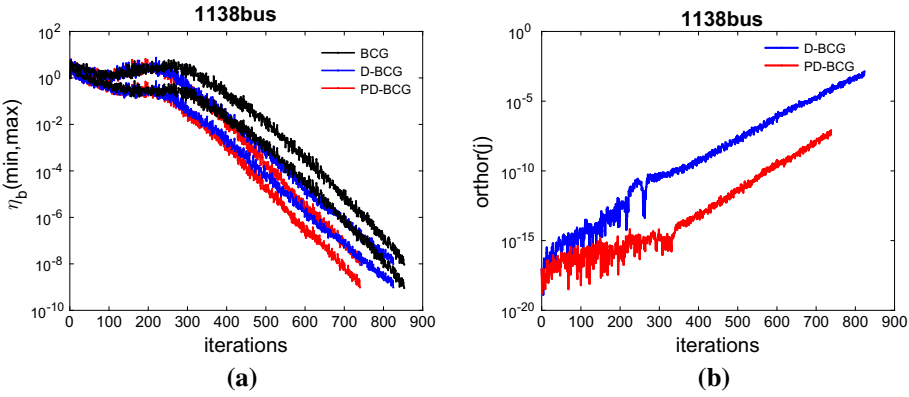

Fig. 1 **a** Convergence histories of Example 1 with $s = 6$ and $\varepsilon = 10^{-8}$; **b** numerical values of othor(j) along with iteration numbers

Table 2 Numerical results of BCG, D-BCG and PD-BCG for different numbers of right-hand sides in Example 1 with $\varepsilon = 10^{-8}$

Matrix: 1138_bus					
Num RHSs	*Method*	*t*	*Mvps*	*Iters*	*CPU*
$s = 3$	BCG		4893	1631	8
	D-BCG	33	4518	1484	8.2656
	PD-BCG	33	**4245**	**1393**	**7.6563**
$s = 6$	BCG		5112	852	5.5469
	D-BCG	33	5004	823	5.875
	PD-BCG	33	**4494**	**738**	**5.1094**
$s = 12$	BCG		4800	400	**2.8281**
	D-BCG	33	5286	435	3.375
	PD-BCG	33	**4436**	**364**	2.9219
$s = 18$	BCG		3888	216	**1.8281**
	D-BCG	33	4278	234	2.3125
	PD-BCG	33	**3450**	**188**	1.9375

Number of the right-hand sides simplified as *Num RHSs* (i.e., the numerical value of s)
Bold values indicate the minimum respectively for *Mvps, Iters* and *CPU* among the involved methods

the convergence behaviour of the case $s = 6$ in the left plot of Fig. 1. And the corresponding loss of orthogonality between R_j and W for D-BCG and PD-BCG is displayed in the right plot of Fig. 1.

From the left plot of Fig. 1, we can see that both of the convergence behaviors of D-BCG and PD-BCG are better than that of BCG, which verifies the expression (17) that the deflated variants of BCG own a faster convergence rate compared with BCG. Moreover, the convergence rate of PD-BCG is obviously faster than that of D-BCG under the same deflation matrix W. It could be explained that PD-BCG mitigates the loss of orthogonality between the block residual vectors R_j and the deflation matrix W as shown in the right plot of Fig. 1 so as to realize the theoretically faster convergence rate of the deflated variants of BCG in finite precision arithmetic.

The required *Mvps*, *Iters* and *CPU* of BCG, D-BCG and PD-BCG with such different numbers of right-hand sides are reported in Table 2. It is observed that these block methods

become more efficient in terms of *Iters* and *CPU* along with the increasing block size. In the case of $s = 6$, it is obviously noticed that these two deflated variants of BCG (i.e., D-BCG and PD-BCG) yield different performance in comparison with BCG. The overall number of *Mvps* is reduced by 12.1% for PD-BCG, while that is solely reduced by 2.1% for D-BCG. Moreover, PD-BCG performs the best in terms of *CPU* since the reduced *Mvps* of PD-BCG is pretty considerable and enough to compensate the cost of constructing the deflation matrix W. Similar numerical results are also discovered in the case of $s = 3$. When it comes to $s = 12, 18$, however, D-BCG cannot realize the supposed faster convergence rate compared to BCG because of the impact of the loss of orthogonality. But PD-BCG can (at least in terms of *Iters*). Thus our PD-BCG is demonstrated to be more favorable and stable in comparison with D-BCG in the finite precision computation.

Considering the observed fact that the block size of right-hand sides can affect the behaviour of deflated BCG methods and this issue may need another special study, we only discuss the fixed block size in the following two experiments.

4.3 Example 2: Experiment on the Residual Vectors R_j with Rank Deficiency

In this example, we take *cbuckle* as the coefficient matrix. Linearly dependent right-hand sides are used to construct the initial residual vectors with rank deficiency to show the relative performance of these block solvers, and especially to verify the benefits of PD-BCG for effectively handling convergence stagnation in comparison with D-BCG. The incomplete Cholesky factorization $ic(0)$ is employed as the preconditioner in this example. The right-hand sides B are consisted of 7 columns, where the elements of the first 5 columns are chosen as randomly generated numbers with standard normal distribution, and the last two columns are linear combinations of the first 5 columns. The dimension of the deflation matrix W is set to 10 here. The convergence histories are displayed in the left plot of Fig. 2, and the loss of orthogonality between R_j and W of D-BCG and PD-BCG is displayed in the right plot of Fig. 2.

The left plot of Fig. 2 shows that D-BCG exhibits a long plateau along iterations after attaining the accuracy of 10^{-6}, for which is accounted by the accumulated loss of orthogonality of D-BCG as observed in the right plot of Fig. 2. The loss of orthogonality of D-BCG as shown in the upper curve of the right plot in Fig. 2 is quite high and does not exhibit any improvement during the subsequent iterations, causing D-BCG to ruin its convergence. In fact, both Saad et al. [35] and Chen [10] have pointed out that the loss of orthogonality between the block residual vectors and the deflation subspace in practise hampers convergence rate of deflated methods seriously, and even completely ruins convergence if such loss of orthogonality is high enough, even though D-BCG owns theoretically faster convergence rate compared with BCG in exact arithmetic. However, numerical results in this example demonstrate PD-BCG still converges well to the targeted accuracy by strengthening the orthogonality as displayed in the red curve in the right plot of Fig. 2. The numerical digits in Table 3 report the superiority of PD-BCG to BCG and D-BCG in both terms of *Mvps* and *CPU*.

4.4 Example 3: Experiment on Handling Ill-Conditioned Coefficient Matrices with Rank Deficient Right-Hand Sides by Deflation

Loss of orthogonality can be explained by the accumulation of round-off errors in finite precision arithmetic, and this phenomenon may be worse when addressing ill-conditioned

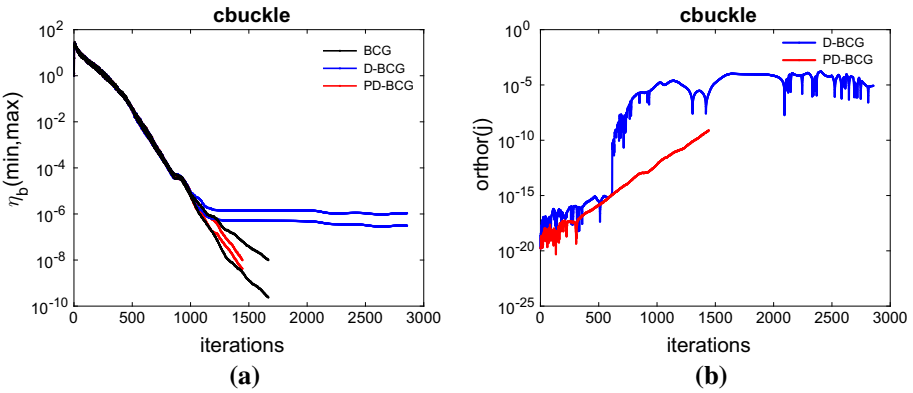

Fig. 2 **a** Convergence histories of Example 2 with $\varepsilon = 10^{-8}$; **b** numerical values of othor(j) along with iteration numbers

Table 3 Numerical results of BCG, D-BCG and PD-BCG in Example 2 with $\varepsilon = 10^{-8}$

Matrix	*Method*	*t*	*Mvps*	*Iters*	*CPU*
cbuckle	BCG		11,662	1666	32.7344
	D-BCG	10	–	–	–
	PD-BCG	10	**10,121**	**1443**	**29.4219**

Bold values indicate the minimum respectively for *Mvps, Iters* and *CPU* among the involved methods

problems [9,22] or rank deficiency problems [6]. In this section, solving linear systems with two ill-conditioned matrices with rank deficient right-hand sides is considered to demonstrate the efficacy of PD-BCG. Information of these two test matrices from structural problem [15] is listed in the last row of Table 1 (i.e., the *s3rmt3m3* and *s3rmq4m1*). The shifted incomplete Cholesky factorization $ict(1e-2)$ with diagcomp option $alpha = 0.6$ is taken as the preconditioner M in this example. And the right-hand sides B contain 5 columns, where the elements of the first 4 columns are chosen to be randomly generated numbers with standard normal distribution, and the last column is constructed as a linear combination of the first 4 columns. The deflation matrix W used for these two ill-conditioned systems is constructed by implementing 11 steps of a separate single-vector Lanczos algorithm [10, Algorithm 2].

Moreover, the PD-BCG method with our automated reorthogonalization process (referred as PD-BCG-ARO) is also considered. The frequency number of implementing this automated reorthogonalization is recorded as N_r. For the test matrices *s3rmt3m3* and *s3rmq4m1*, we set the scale parameter c expressed in (20) as $c = 1$ and $c = 10^{-3}$, respectively. The convergence histories are displayed in the left plots of Fig. 3, and the numerical values of othor(j) of D-BCG, PD-BCG and PD-BCG-ARO are displayed in the right plots of Fig. 3. In this example, since the dimension of W is small, which brings low accuracy by the columns of W of approximation to the eigenvectors corresponding to extreme eigenvalues of this preconditioned matrix, we do not compare PD-BCG-ARO and D-BCG by using full reorthogonalization since the D-BCG method with full reorthogonalization between the block residuals and such W does not converge at all from our numerical experiments. In addition, the construction of W and its impact on algorithms with deflation need further study.

It is observed from the left two plots of Fig. 3 and Table 4 that both our PD-BCG and PD-BCG-ARO methods help to improve the convergence behaviour of BCG to a large extent, even though the classical D-BCG does not converge at all under such ill-conditioned and rank

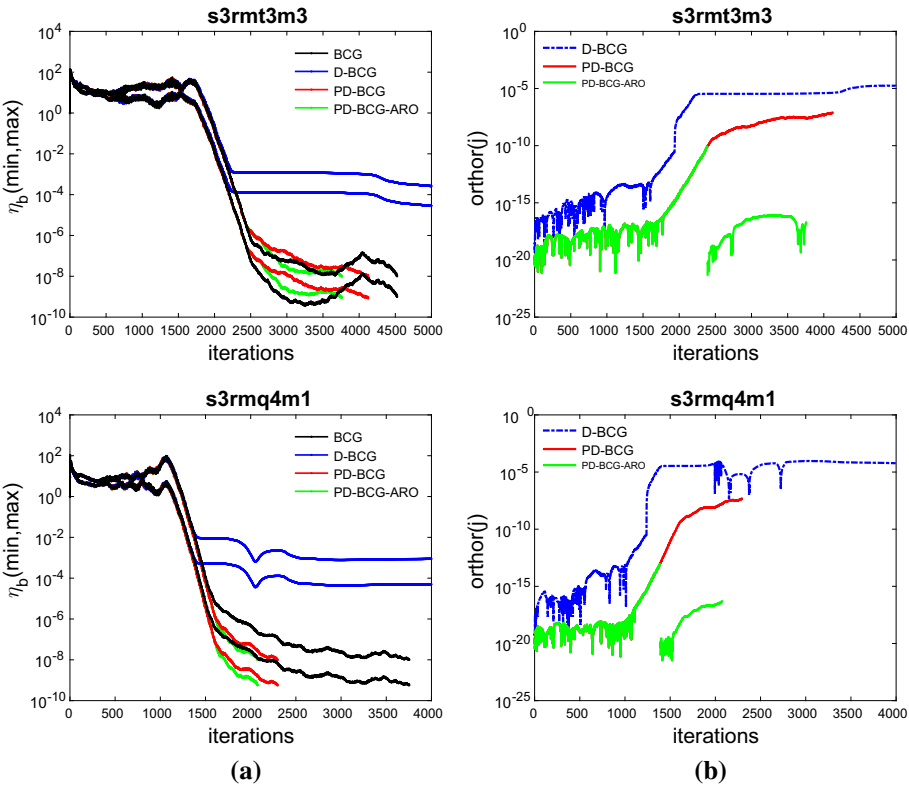

Fig. 3 **a** Convergence histories of Example 3 with $\varepsilon = 10^{-8}$; **b** numerical values of othor(j) along with iteration numbers

deficiency circumstances. Moreover, from the right two plots of Fig. 3, it is convinced that not only can our PD-BCG-ARO method mitigate the loss of orthogonality between R_j and W as the algorithm progresses, but such orthogonality can be maintained to an appropriate level by adjusting the value of the scalar c.

In addition, the effort to perform the automated reorthogonalization process between R_j and W in PD-BCG-ARO accounts for the extra cost in terms of *CPU*. But it is observed that PD-BCG-ARO is the best in terms of *Mvps*. Moreover, the fact of $N_r = 1$ shown in the last column of Table 4 indicates that just one reorthogonalization step is well enough to mitigate the loss of orthogonality by our automated reorthogonalization strategy, which again indicates that the full reorthogonalization is unnecessary [33] when loss of orthogonality occurs.

5 Discussion and Conclusions

The gradual loss of orthogonality between the block residual vectors and the deflation matrix of the deflated block CG-like methods hampers the convergence seriously, which has been fully discussed in [10,35]. To improve convergence, in this paper, a new projected variant of the deflated block conjugate gradient method (PD-BCG) is developed to mitigate the loss of such orthogonality. The main idea behind our new projected deflation variant is to introduce

Table 4 Numerical results of BCG, D-BCG, PD-BCG and PD-BCG-ARO in Example 3 with $\varepsilon = 10^{-8}$

Martix	*Method*	t	c	*Mvps*	*Iters*	*CPU*	N_r
s3rmt3m3	BCG			22,620	4524	235.875	
	D-BCG	11		–	–	–	
	PD-BCG	11		20,637	4123	**211.6094**	
	PD-BCG-ARO	11	1	**18,802**	**3756**	222.625	1
s3rmq4m1	BCG			18,765	3753	225.1406	
	D-BCG	11		–	–	–	
	PD-BCG	11		11,497	2295	**138.8125**	
	PD-BCG-ARO	11	10^{-3}	**10,397**	**2075**	153.5625	1

Bold values indicate the minimum respectively for *Mvps, Iters* and *CPU* among the involved methods

auxiliary block search vectors $\tilde{P}_j{}'$ and to exploit the A-orthogonal projector H expressed in (3) such that the block search vectors are obtained as $\tilde{P}_j = H\tilde{P}_j{}'$ ($\tilde{P}_j$ consists of the orthonormal basis vectors of the block search space matrix $\mathscr{P}_j$), which ensures the A-orthogonality between $\mathscr{P}_j$ and W at each iteration. Furthermore, a new automated reorthogonalization strategy is also proposed, which is an alternative option for PD-BCG. The convergence rate of PD-BCG is discussed under circumstances both with and without the rank deficiency problem.

From the development of PD-BCG, we can see that the mechanism behind Algorithm 2 is quite different from that of the D-BCG algorithm [10, Algorithm 1]. Specifically, the D-BCG method exploits the A-orthogonal projector H as $Z_j = HZ_j{}'$ rather than $\tilde{P}_j = H\tilde{P}_j{}'$ used in the PD-BCG method. Moreover, from the derivation of this automated reorthogonalization strategy, we conclude that this new reorthogonalization process can also be applied to the general D-BCG method. Numerical experiments verify that our methods can mitigate the loss of the orthogonality and conquer the long-term stagnation curve, which is quite meaningful in practical computation with finite precision arithmetic. In fact, PD-BCG would be a more favorable choice than D-BCG if the loss of orthogonality is so high to damage the convergence seriously, which is pretty familiar when handling ill-conditioned linear systems [9,22] or linear systems suffering from ill-conditioned parameter matrices caused by rank deficiency [6]. Further interesting work is to investigate the theoretical convergence rate of these deflated BCG variants including PD-BGC by analyzing the impact of the loss of orthogonality on these methods in finite precision arithmetic. Moreover, it is still necessary to put more efforts to further study the construction of the deflation matrix in order to handle large-scale linear systems.

Acknowledgements The authors would like to gratefully thank the anonymous referees for their insightful comments and valuable suggestions, which enabled us to substantially improve the quality of this paper.

References

1. Arioli, M., Duff, I.S., Ruiz, D., Sadkane, M.: Block Lanczos techniques for accelerating the block Cimmino method. SIAM J. Sci. Comput. **16**(6), 1478–1511 (1995)
2. Abdel-Rehim, A.M., Morgan, R.B., Nicely, D.A., Wilcox, W.: Deflated and restarted symmetric Lanczos methods for eigenvalues and linear equations with multiple right-hand sides. SIAM J. Sci. Comput. **32**(1), 129–149 (2010)
3. Agullo, E., Giraud, L., Jing, Y.-F.: Block GMRES method with inexact breakdowns and deflated restating. SIAM J. Matrix Anal. Appl. **35**(4), 1625–1651 (2014)

4. Aliaga, J.I., Boley, D.L., Freund, R.W., Hernández, V.: A Lanczos-type method for multiple starting vectors. Math. Comput. **69**, 1577–1601 (2000)
5. Bristeau, M.O., Erhel, J.: Augmented conjugate gradient. Application in an iterative process for the solution of scattering problems. Numer. Algorithms **18**, 71–90 (1998)
6. Birk, S.: Deflated shifted block Krylov subspace methods for Hermitian positive definite matrices, Ph.D. thesis. Fakultät für Mathematik und Naturwissenschaften, Bergische Universität Wuppertal (2015)
7. Birk, S., Frommer, A.: A deflated conjugate gradient method for multiple right hand sides and multiple shifts. Numer. Algorithms **67**, 507–529 (2014)
8. Baglama, J.: Dealing with linear dependence during the iterations of the restarted block Lanczos methods. Numer. Algorithms **25**, 23–36 (2000)
9. Björck, A., Palge, C.: Loss and recapture of orthogonality in the modified Gram–Schmidt algorithm. SIAM J. Matrix Anal. Appl. **13**(1), 176–190 (1992)
10. Chen, J.: A deflated version of the block conjugate gradient algorithm with an application to Gaussian process maximum likelihood estimation. Preprint ANL/MCS-P1927-0811, Argonne National Laboratory, Argonne (2011)
11. Cockett, R.: The block conjugate gradient for multiple right hand sides in a direct current resistivity inversion. http://www.row1.ca/s/pdfs/courses/BlockCG.pdf. Accessed 28 Feb 2015
12. Chan, T.-F., Wan, W.-L.: Analysis of projection methods for solving linear systems with multiple right-hand sides. SIAM J. Sci. Comput. **18**(6), 1698–1721 (1997)
13. Chapman, A., Saad, Y.: Deflated and augmented Krylov subspace techniques. Numer. Linear Algebra Appl. **4**(1), 43–66 (1997)
14. Chatfield, D.C., Reeves, M.S., Truhlar, D.G., Duneczky, C., Schwenke, D.W.: Complex generalized minimal residual algorithm for iterative solution of quantum mechanical reactive scattering equations. J. Chem. Phys. **97**(11), 8322–8333 (1992)
15. Davis, T.A., Hu, Y.F.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 249–260 (2011)
16. Deb, K.: Multi-objective Optimization Using Evolutionary Algorithms. Wiley, Chichester, New York (2001)
17. Erhel, J., Guyomarch, F.: An augmented conjugate gradient method for solving consecutive symmetric positive definite linear systems. SIAM J. Matrix Anal. Appl. **21**(4), 1279–1299 (2000)
18. Ekström, S.E., Neytcheva, M.: Enabling the full potential of the deflation techniques? Report. Harbin. In: The 13th International Conference of China Matrix Theory and Its Application (2018)
19. Gaul, A., Gutknecht, M.H., Liesen, J., Nabben, R.: A framework for deflated and augmented Krylov subspace methods. SIAM J. Matrix Anal. Appl. **34**(2), 495–518 (2013)
20. Gutknecht, M.H.: Block Krylov space methods for linear systems with multiple right-hand sides: an introduction. In: Siddiqi, A.H., Duff, I.S., Christensen, O. (eds.) Modern Mathematical Models, Methods and Algorithms for Real World Systems, pp. 420–447. Anamaya, New Delhi (2007)
21. Golub, G.H., Ruiz, D., Touhami, A.: A hybrid approach combining Chebyshev filter and conjugate gradient for solving linear systems with multiple right-hand sides. SIAM J. Matrix Anal. Appl. **29**(3), 774–795 (2007)
22. Giraud, L., Langou, J., Rozloznik, M.: The loss of orthogonality in the Gram–Schmidt orthogonalization process. Comput. Math. Appl. **50**(7), 1069–1075 (2005)
23. Hestenes, M.R., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. **49**(6), 409–436 (1952)
24. Ji, H., Li, Y.-H.: A breakdown-free block conjugate gradient method. BIT Numer. Math. **57**, 379–403 (2017)
25. Kahl, K., Rittich, H.: The deflated conjugate gradient method: convergence, perturbation and accuracy. Linear Algebra Appl. **515**, 111–129 (2017)
26. Morgan, R.B.: A restarted GMRES method augmented with eigenvectors. SIAM J. Matrix Anal. Appl. **16**(4), 1154–1171 (1995)
27. Morgan, R.B.: Restarted block-GMRES with deflation of eigenvalues. Appl. Numer. Math. **54**, 222–236 (2005)
28. Nikishin, A.A., Yeremin, A.Y.: 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**(4), 1135–1153 (1995)
29. Nicolaides, R.A.: Deflation of conjugate gradients with applications to bundary value problems. SIAM J. Numer. Appl. **24**(2), 355–365 (1987)
30. O'Leary, D.P.: The block conjugate gradient algorithm and related methods. Linear Algebra Appl. **29**, 293–322 (1980)

31. Parks, M.L., de Sturler, E., Mackey, G., Johnson, D.D., Maiti, S.: Recycling Krylov subspaces for sequences of linear systems. SIAM J. Sci. Comput. **28**(5), 1651–1674 (2006)
32. Peng, Z., Shao, Y., Lee, J.-F.: Advanced model order reduction technique in real-life IC/package design. In: IEEE Electrical Design of Advanced Package and Systems Symposium (2010)
33. Simon, H.D.: The Lanczos algorithm with partial reorthogonalization. Math. Comput. **42**(165), 115–142 (1984)
34. Soodhalter, K.M.: Block Krylov subspace recycling for shifted systems with unrelated right-hand sides. SIAM J. Sci. Comput. **38**(1), A302–A324 (2016)
35. Saad, Y., Yeung, M., Erhel, J., Guyomarch, F.: A deflated version of the conjugate gradient algorithm. SIAM J. Sci. Comput. **21**(5), 1909–1926 (2000)
36. Saad, Y.: Analysis of augmented Krylov subspace methods. SIAM J. Matrix Anal. Appl. **18**(2), 435–449 (1997)
37. Sun, D.-L., Huang, T.-Z., Carpentieri, B., Jing, Y.-F.: Flexible and deflated variants of the block shifted GMRES method. J. Comput. Appl. Math. **345**, 168–183 (2019)
38. Sun, D.-L., Huang, T.-Z., Jing, Y.-F., Carpentieri, B.: A block GMRES method with deflated restarting for solving linear systems with multiple shifts and multiple right-hand sides. Numer. Linear Algebra Appl. **25**(5), e2148 (2018)
39. Sun, D.-L., Carpentieri, B., Huang, T.-Z., Jing, Y.-F.: A spectrally preconditioned and initially deflated variant of the restarted block GMRES method for solving multiple right-hand sides linear systems. Int. J. Mech. Sci. **144**, 775–787 (2018)
40. Sun, D.-L., Huang, T.-Z., Carpentieri, B., Jing, Y.-F.: A new shifted block GMRES method with inexact breakdowns for solving multi-shifted and multiple right-hand sides linear systems. J. Sci. Comput. **78**(2), 746–769 (2019)
41. Xiang, Y.-F., Jing, Y.-F., Huang, T.-Z., Sun, D.-L.: On adaptive restart procedures for the breakdown-free block conjugate gradient method. Report. Harbin. In: The 13th International Conference of China Matrix Theory and Its Application (2018)