BIT Numer Math
DOI 10.1007/s10543-016-0631-z

A breakdown-free block conjugate gradient method

Hao Ji[1] · **Yaohang Li**[2]

Received: 6 March 2015 / Accepted: 14 September 2016

Abstract In this paper, we analyze all possible situations of rank deficiency that cause breakdown in block conjugate gradient (BCG) solvers. A simple solution, breakdown-free block conjugate gradient (BFBCG), is designed to address the rank deficiency problem. The rationale of the BFBCG algorithm is to derive new forms of parameter matrices based on the potentially reduced search subspace to handle rank deficiency. Orthogonality properties and convergence of BFBCG in case of rank deficiency are justified accordingly with mathematical rigor. BFBCG yields faster convergence than restarting BCG when breakdown occurs. Numerical examples suffering from rank deficiency are provided to demonstrate the robustness of BFBCG.

Keywords Rank deficiency · Breakdown-free block conjugate gradient method · Block Krylov subspace · Multiple right-hand sides · Near-breakdown problem

Mathematics Subject Classification 65F10 · 65F50 · 68W40

1 Introduction

The Conjugate Gradient (CG) algorithm [19,50] is an effective computational method extensively used to solve a system of linear operator equations

✉ Yaohang Li
yaohang@cs.odu.edu

Hao Ji
hji@cpp.edu

1 Department of Computer Science, California State Polytechnic University, Pomona, CA 91768, USA

2 Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA

Published online: 05 October 2016

$$Ax = b$$

with a large, sparse $n \times n$ Symmetric Positive Definite (SPD) matrix A. The Block Conjugate Gradient (BCG) algorithm [34,38] is a generalization of the CG algorithm for solving a system of equations

$$AX = B,$$

where B is a block matrix containing $s (s \geq 1)$ right-hand sides. Compared to CG, BCG has the advantages of potentially faster convergence and being more suitable for parallel computing architectures [26,30,39], which has been widely used in a variety of applications [4,7,8,33–35]. In today's large-scale linear systems where A is either stored outside of the system memory or the elements of A have to be regenerated at each use, accessing A becomes the main computational bottleneck of a linear solver. BCG and related block algorithms are hence particularly attractive, due to their capability of exploring multiple search directions in a single pass over A.

Despite the above attractive features, a well-known practical issue of BCG is the rank deficiency problem that can lead to BCG breakdown. Theoretically, a necessary and sufficient condition is given by Broyden [5] for absence of breakdown by analyzing the Krylov subspace sequences. In practice, at a certain BCG iteration, the block search direction vectors may become linearly dependent and cannot span an s–dimensional search subspace. Consequently, at least one of the parameter matrices in BCG turns out to be singular, which results in BCG breakdown. Rank deficiency may be caused by many factors, which will be thoroughly analyzed in Sect. 2. Restarting BCG is suggested as a practical remedy to breakdown [38]. Instead of restarting, other approaches, such as normalizing the right-hand side vectors [7], orthogonalizing the residual columns to remove dependent vectors [14], and reducing block size adaptively [37] help reduce the risk of BCG breakdown, but cannot completely guarantee numerical stability.

In this paper, we present a simple solution to address the rank deficiency problem in BCG, which results in a Breakdown-Free Block Conjugate Gradient (BFBCG) algorithm. The fundamental idea of BFBCG is, in case of the rank of a block matrix being reduced, the parameter matrices are calculated in the reduced Krylov subspace to minimize the block nonnegative quadratic function of

$$F(X) = \text{trace}\left(\left(X - X^*\right)^T A \left(X - X^*\right)\right), \tag{1.1}$$

where trace$(\cdot)$ denotes the trace of a matrix and $X^* = A^{-1}B$ is the desired block solution. As a result, BFBCG avoids inverting a potentially non-full rank matrix and thus addresses the breakdown problem caused by rank deficiency.

The rest of the paper is organized as follows. Section 2 analyzes the factors that lead to rank deficiency in BCG. In Sect. 3, we derive the BFBCG algorithm with new forms of parameter matrices. Then, in Sect. 4 we discuss the convergence properties of BFBCG. The numerical results are presented in Sect. 5. Section 6 describes the relation of BFBCG to other work. Finally, Sect. 7 summarizes the paper.

2 Block conjugate gradient and rank deficiency

Algorithm 1: The original BCG algorithm

Input: matrix $A \in \mathbb{R}^{n\times n}$, matrix $B \in \mathbb{R}^{n\times s}$, initial guess $X_0 \in \mathbb{R}^{n\times s}$, preconditioner $M \in \mathbb{R}^{n\times n}$, tolerance $tol \in \mathbb{R}$, and maximum number of iterations $maxit \in \mathbb{R}$.
Output: an approximate solution $X_{sol} \in \mathbb{R}^{n\times s}$.

$R_0 = B - AX_0$
$Z_0 = MR_0$
$P_0 = Z_0\gamma_0$
For $i = 0, \cdots, maxit$
  $\alpha_i = (P_i^T AP_i)^{-1}\gamma_i^T(Z_i^T R_i)$
  $X_{i+1} = X_i + P_i\alpha_i$
  $R_{i+1} = R_i - AP_i\alpha_i$
  If converged within tol, then stop.
  $Z_{i+1} = MR_{i+1}$
  $\beta_i = \gamma_i^{-1}(Z_i^T R_i)^{-1}(Z_{i+1}^T R_{i+1})$
  $P_{i+1} = (Z_{i+1} + P_i\beta_i)\gamma_{i+1}$
End
$X_{sol} = X_{i+1}$

Algorithm 1 is the original BCG algorithm proposed by O'Leary [38] for simultaneously solving a linear system with s right-hand sides in an $n \times s$ block matrix B. X_0 is an initial solution guess and M is an SPD preconditioner. α_i and β_i are $s \times s$ parameter matrices to ensure orthogonality of the residual matrix R_{i+1} and the search matrix P_i as well as conjugacy (A-orthogonality) of $P_0, \ldots, P_{i+1}$, respectively. γ_i is an arbitrary nonsingular $s \times s$ matrix, which in practice is selected, for example, to orthogonalize P_i to decrease round–off errors and enhance numerical stability [38]. As shown in Proposition 2.1, the preconditioned residual matrix Z_i and the search matrix P_i have the same matrix rank as R_i. Therefore, loss of full rank in R_i will lead to rank deficiency of Z_i and P_i during BCG iterations. Consequently, $Z_i^T R_i$ and $P_i^T AP_i$ become singular and thus it is improbable to obtain $(Z_i^T R_i)^{-1}$ and $(P_i^T AP_i)^{-1}$ to evaluate α_i and β_i. As a result, BCG breakdown occurs.

Proposition 2.1 *Suppose R_i is an $n \times s$ residual matrix of rank r_i ($r_i \le s$) at the ith iteration, then*

$$\operatorname{rank}(P_i) = \operatorname{rank}(Z_i) = \operatorname{rank}(R_i) = r_i,$$

where $\operatorname{rank}(\cdot)$ *denotes the rank of a matrix.*

Proof First, we show that $\operatorname{rank}(Z_i) = \operatorname{rank}(R_i) = r_i$. From Algorithm 1, matrix Z_i is defined as $Z_i = MR_i$. Since M is assumed to be SPD, then $\operatorname{rank}(Z_i) = \operatorname{rank}(R_i) = r_i$.

Next we show that $\operatorname{rank}(P_i) = \operatorname{rank}(R_i)$. The search matrix P_i is given by

$$P_i = (Z_i + P_{i-1}\beta_{i-1})\gamma_i. \tag{2.1}$$

Left multiplying (2.1) by $P_i^T A$ on both sides, we get

$$P_i^T AP_i = P_i^T AZ_i\gamma_i + P_i^T AP_{i-1}\beta_{i-1}\gamma_i.$$

Notice that P_i is A-orthogonal to P_{i-1}, i.e., $P_i^T A P_{i-1} = 0$, then

$$P_i^T A P_i = P_i^T A Z_i \gamma_i.$$

Using the basic properties of matrix rank, we can get

$$\begin{aligned} \text{rank}(P_i) &= \text{rank}(P_i^T A P_i) = \text{rank}(P_i^T A Z_i \gamma_i) \\ &\leq \text{rank}(Z_i) = \text{rank}(R_i). \end{aligned} \tag{2.2}$$

On the other hand, since R_i is orthogonal to P_{i-1}, i.e., $R_i^T P_{i-1} = 0$, left multiplying both sides of (2.1) by R_i^T and then eliminating the zero terms, we obtain

$$R_i^T P_i = R_i^T Z_i \gamma_i = R_i^T M R_i \gamma_i.$$

According to the basic properties of matrix rank again, we have

$$\begin{aligned} \text{rank}(P_i) &\geq \text{rank}(R_i^T P_i) = \text{rank}(R_i^T M R_i \gamma_i) \\ &= \text{rank}(R_i). \end{aligned} \tag{2.3}$$

Based on (2.2) and (2.3), $\text{rank}(P_i) = \text{rank}(R_i) = r_i$ is concluded. □

Rank deficiency may be caused by many different reasons in practice, for instance, inappropriate guess of initial vectors, unbalanced convergence speeds of solutions with respect to multiple right-hand sides, and accumulation of round-off errors. The possible situations of rank deficiency in BCG are summarized as follows:

- Two or more vector components in the initial residual matrix R_0 are linearly dependent. For example, if the multiple right-hand sides in matrix B contain linearly dependent vectors and X_0 simply takes zero vectors as the initial guess, then the initial residual matrix R_0 will include linearly dependent vectors. In practice, this breakdown situation can be eliminated by ensuring linear independence of the column vectors in R_0, such as carefully selecting an initial guess X_0. An alternative approach is orthogonalizing R_0 [14] to eliminate the dependent vectors;
- One or more but not all vector components in the residual matrix R_i reach convergence. During BCG iterations, solutions with respect to some right-hand sides may converge faster than the others, which results in near-zero vectors in R_i. This typically happens when the norms of the component vectors in R_0 are significantly different in magnitude. An obvious approach is to normalize the right-hand sides in B so as to keep the norms of the component vectors of R_0 at a similar scale [7] to hopefully balance the number of convergence steps for the multiple right-hand sides. Since convergence has already been achieved in some solutions, removing these solutions and their corresponding residual vectors [38] not only avoids BCG breakdown, but also eliminates unnecessary numerical computations; and
- Two or more vector components in the residual matrix R_i at the ith iteration become linearly dependent. If one is only interested in a single solution with respect to a specific right-hand side, i.e., the multiple right-hand sides block is expanded from a single right-hand side, the variable BCG algorithm [37] can sufficiently address the breakdown problem caused by this factor through constructing an A-orthogonal

projector to reduce the block size. Nevertheless, if solutions to all right-hand sides are of interest, assuming that the right–hand sides of the corresponding linearly dependent vectors have not converged yet and thus none of the vector components in R_i are zero, reducing the block sizes will result in loss of solutions.

In addition to the original BCG algorithm, alternative BCG algorithms [38] based on different CG forms are also developed. These BCG forms have different ways to calculate the parameter matrices. The Hestenes and Stiefel form BCG [23] uses

$$\alpha_{i-1} = \left(P_{i-1}^T A P_{i-1}\right)^{-1} \gamma_{i-1}^T \left(R_{i-1}^T M R_{i-1}\right)$$

and

$$\beta_{i-1} = \gamma_{i-1}^{-1} \left(R_{i-1}^T M R_{i-1}\right)^{-1} \left(R_i^T M R_i\right).$$

The block minimum residual (B-MR) algorithm [15] employs

$$\alpha_{i-1} = \left(P_{i-1}^T A M A P_{i-1}\right)^{-1} \gamma_{i-1}^T \left(R_{i-1}^T M A M R_{i-1}\right)$$

and

$$\beta_{i-1} = \gamma_{i-1}^{-1} \left(R_{i-1}^T M A M R_{i-1}\right)^{-1} \left(R_i^T M A M R_i\right),$$

instead. The change of variables form BCG [2,40] adopts

$$\alpha_{i-1} = \left(P_{i-1}^T M A M P_{i-1}\right)^{-1} \gamma_{i-1}^T \left(R_{i-1}^T M R_{i-1}\right)$$

and

$$\beta_{i-1} = \gamma_{i-1}^{-1} \left(R_{i-1}^T M R_{i-1}\right)^{-1} \left(R_i^T M R_i\right).$$

The Rutishauser form BCG [42,44] is based on a three–term recurrence relation form of CG, which updates the solution and the residual matrix by

$$X_{i+1} = X_i + \left(M R_i - \left(X_i - X_{i-1}\right) \eta_{i-1}\right) \omega_{i+1}$$

and

$$R_{i+1} = R_i + \left(\left(R_i - R_{i-1}\right) \eta_{i-1} - A M R_i\right) \omega_{i+1},$$

respectively, where

$$\omega_{i+1}^{-1} = \left(R_i^T M R_i\right)^{-1} R_i^T M A M R_i - \eta_{i-1}$$

and

$$\eta_i = \omega_{i+1}^{-1} \left(R_i^T M R_i \right)^{-1} R_{i+1}^T M R_{i+1}.$$

The deflated BCG [7] updates the search matrix P_i with a deflation matrix W by

$$P_i = \left(Z_i + P_{i-1}\beta_{i-1} - W \left(W^T A W \right)^{-1} W^T A Z_i \right) \gamma_i$$

and uses the same formulas in Algorithm 1 to calculate α_i and β_i. In summary, all of these alternative BCG forms require estimation of the inverse of matrices involving either R_i or P_i. Consequently, breakdown in these BCG forms is unavoidable when loss of full rank in R_i or P_i occurs.

3 Breakdown-free block conjugate gradient algorithm

In this section, we describe a Block Conjugate Gradient algorithm that can avoid breakdown completely. The resulting method is called the Breakdown-Free Block Conjugate Gradient (BFBCG) algorithm. To illustrate the differences in comparison with the original BCG algorithm described in Algorithm 1, the matrix symbols with a "$\sim$" notation are used to indicate that the dimensions of these matrices may reduce in case of rank deficiency in BFBCG (Algorithm 2). The rationale of the BFBCG algorithm is to derive new forms of the parameter matrices $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$ based on the potentially reduced search subspace. In case of linear dependence in the search directions or the residual vectors, $\widetilde{\alpha}_i$ is designed to ensure that the column space of the next residual matrix R_{i+1} is orthogonal to the search space $\mathscr{P}_i$. A new form of $\widetilde{\beta}_i$ is derived so that the new search space $\mathscr{P}_{i+1}$ is conjugate to all previous search spaces $\mathscr{P}_j$ $(j < i+1)$.

Algorithm 2: The breakdown–free BCG algorithm

Input: matrix $A \in \mathbb{R}^{n\times n}$, matrix $B \in \mathbb{R}^{n\times s}$, initial guess $X_0 \in \mathbb{R}^{n\times s}$, preconditioner $M \in \mathbb{R}^{n\times n}$, tolerance $tol \in \mathbb{R}$, and maximum number of iterations $maxit \in \mathbb{R}$.
Output: an approximate solution $X_{sol} \in \mathbb{R}^{n\times s}$.

$R_0 = B - AX_0$
$Z_0 = MR_0$
$\widetilde{P}_0 = orth(Z_0)$
For $i = 0, \cdots, maxit$
  $Q_i = A\widetilde{P}_i$
  $\widetilde{\alpha}_i = (\widetilde{P}_i^T Q_i)^{-1} (\widetilde{P}_i^T R_i)$
  $X_{i+1} = X_i + \widetilde{P}_i \widetilde{\alpha}_i$
  $R_{i+1} = R_i - Q_i \widetilde{\alpha}_i$
  If converged within tol, then stop.
  $Z_{i+1} = MR_{i+1}$
  $\widetilde{\beta}_i = -(\widetilde{P}_i^T Q_i)^{-1} (Q_i^T Z_{i+1})$
  $\widetilde{P}_{i+1} = orth(Z_{i+1} + \widetilde{P}_i \widetilde{\beta}_i)$
End
$X_{sol} = X_{i+1}$

Compared to the original BCG algorithm [38], the BFBCG algorithm has the following major differences:

- Matrix operation $orth(\cdot)$ is employed for extracting an orthogonal basis $\widetilde{P}_i \in \mathbb{R}^{n\times r_i}$ from the search space $\mathscr{P}_i$. $orth(\cdot)$ can be efficiently implemented using QR decomposition with column pivoting. In case of rank deficiency, the dimension of the search space $\mathscr{P}_i$ will be reduced, which avoids the situations of revisiting the subspace already visited in the BCGAdQ algorithm described in [10];
- If rank deficiency occurs at the ith iteration, $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$ will turn into rectangular matrices of size $r_i \times s$, where r_i is the dimension of the search space $\mathscr{P}_i$. In comparison, the parameter matrices are restricted as square matrices in the original BCG algorithm; and
- Matrices γ_i are no longer necessary in the BFBCG algorithm.

In addition to breakdown avoidance, the BFBCG algorithm maintains several favorable features in practice. For example, at each iteration, matrix A is visited only once. Meanwhile, $(\widetilde{P}_i^T Q_i)^{-1}$ calculated in $\widetilde{\alpha}_i$ can be reused for computing $\widetilde{\beta}_i$.

We use Theorems 3.1 and 3.2 to justify the derivations of $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$, respectively. Theorem 3.1 shows, in case of rank deficiency at the ith iteration in BFBCG, the rectangular parameter matrix $\widetilde{\alpha}_i$ ensures that the column space of R_{i+1} is orthogonal to the search space $\mathscr{P}_i$.

Theorem 3.1 *Suppose that R_i loses full rank at the ith iteration. Let $\mathscr{P}_i$ denote the corresponding search space with dimension r_i ($r_i < s$). Given a matrix $\widetilde{\alpha}_i \in \mathbb{R}^{r_i\times s}$ so that*

$$\widetilde{\alpha}_i = \left(\widetilde{P}_i^T Q_i\right)^{-1}\left(\widetilde{P}_i^T R_i\right),$$

where $\widetilde{P}_i \in \mathbb{R}^{n\times r_i}$ consists of the orthonormal basis of $\mathscr{P}_i$ and $Q_i \in \mathbb{R}^{n\times r_i}$ denotes the matrix product $A\widetilde{P}_i$, the column space of R_{i+1} derived from $\widetilde{\alpha}_i$ is orthogonal to the search space $\mathscr{P}_i$.

Proof As $\widetilde{P}_i \in \mathbb{R}^{n\times r_i}$ is the orthonormal basis of the search space $\mathscr{P}_i$ and $Q_i = A\widetilde{P}_i$, $\widetilde{P}_i^T Q_i \in \mathbb{R}^{r_i\times r_i}$ is nonsingular. Therefore, there exists a matrix $\widetilde{\alpha}_i \in \mathbb{R}^{r_i\times s}$ such that

$$\widetilde{\alpha}_i = (\widetilde{P}_i^T Q_i)^{-1}(\widetilde{P}_i^T R_i). \tag{3.1}$$

Since R_{i+1} is constructed from

$$R_{i+1} = R_i - Q_i\widetilde{\alpha}_i \tag{3.2}$$

in BFBCG, left multiplying both sides of (3.2) by $\widetilde{P}_i^T$ and then by (3.1), we can get

$$\begin{aligned}\widetilde{P}_i^T R_{i+1} &= \widetilde{P}_i^T R_i - \widetilde{P}_i^T Q_i\widetilde{\alpha}_i\\ &= \widetilde{P}_i^T R_i - \widetilde{P}_i^T Q_i(\widetilde{P}_i^T Q_i)^{-1}\widetilde{P}_i^T R_i\\ &= 0,\end{aligned}$$

which indicates that the column space of R_{i+1} is orthogonal to the search space $\mathscr{P}_i$. □

Based on Theorem 3.1, other orthogonality properties of BFBCG can be easily obtained, which are summarized as the following two corollaries. Corollary 3.1 extends Theorem 3.1 and shows that the column space of R_{i+1} is not only orthogonal to the search space $\mathscr{P}_i$ at the ith iteration, but also to all previous search spaces $\mathscr{P}_j$ ($j < i+1$). Moreover, observing that the search spaces $\mathscr{P}_j$ are derived from the subspace spanned by the preconditioned residual matrix Z_j ($j < i+1$), Corollary 3.2 states that R_{i+1} is M–orthogonal to all previous residual matrices under the preconditioner M (assuming that M is SPD).

Corollary 3.1 $R_{i+1}^T \widetilde{P}_j = 0$, *for all* $j < i+1$.

Corollary 3.2 $R_{i+1}^T M R_j = Z_{i+1}^T R_j = 0$, *for all* $j < i+1$.

At the ith iteration, BFBCG explores the block Krylov subspace [21,38], defined as

$$\begin{aligned} D_i(A, M, R_0) &= \text{block-span}\{MR_0, MAMR_0, \ldots, (MA)^i MR_0\} \\ &= \left\{ \sum_{j=0}^{i} (MA)^j MR_0 \Psi_j;\ \ \Psi_j \in \mathbb{R}^{s\times s} \right\}, \end{aligned}$$

which is the union of the previous subspaces spanned by the matrices MR_j ($j < i+1$). Here Ψ_j's are related to the parameter matrices $\widetilde{\alpha}_k$ and $\widetilde{\beta}_k$ ($k \le j$). According to Corollary 3.2, the column space of R_{i+1} is orthogonal to the block Krylov subspace as well, which implies that X_{i+1} from BFBCG is the minimizer of the block nonnegative quadratic function (1.1) over the block Krylov subspace $X_0 + D_i(A, M, R_0)$ at the ith iteration.

The other parameter matrix $\widetilde{\beta}_i$ in BFBCG is chosen to ensure that the next search space $\mathscr{P}_{i+1}$ is conjugate to the previous search space $\mathscr{P}_j$ ($j < i+1$) in case of rank deficiency, which is shown in Theorem 3.2. The following Lemmas 3.1 and 3.2 will be used for the proof of Theorem 3.2.

Lemma 3.1 *Suppose that* R_i *is an* $n \times s$ *residual matrix of rank* r_i ($r_i \le s$) *at the* i*th iteration, then*

$$\text{rank}(\widetilde{P}_i^T R_i) = r_i.$$

Proof Let $\widetilde{P}_i$ denote an orthonormal basis of the search space $\mathscr{P}_i$, which is the range of $Z_i + \widetilde{P}_{i-1}\widetilde{\beta}_{i-1}$ shown in Algorithm 2, then $Z_i + \widetilde{P}_{i-1}\widetilde{\beta}_{i-1}$ can be expressed as

$$Z_i + \widetilde{P}_{i-1}\widetilde{\beta}_{i-1} = \widetilde{P}_i \delta, \tag{3.3}$$

where δ is an $r_i \times s$ matrix of rank r_i. Left multiplying (3.3) by R_i^T on both sides, we can get

$$R_i^T Z_i + R_i^T \widetilde{P}_{i-1}\widetilde{\beta}_{i-1} = R_i^T \widetilde{P}_i \delta.$$

According to Corollary 3.1, $R_i^T \widetilde{P}_{i-1} = 0$. Then,

$$R_i^T Z_i = R_i^T \widetilde{P}_i \delta.$$

According to Proposition 2.1, we can obtain $\text{rank}((R_i^T \widetilde{P}_i)\delta) = \text{rank}(R_i^T Z_i) = \text{rank}(R_i)$. Again, applying the basic rules of matrix rank, $\text{rank}(\widetilde{P}_i^T R_i) = \text{rank}((R_i^T \widetilde{P}_i)\delta) = r_i$ is derived. □

Lemma 3.1 indicates that the matrix rank of $\widetilde{P}_i^T R_i$ is always equal to that of R_i, which will be used in the proof of Lemma 3.2. We can also learn from Lemma 3.1 that the parameter matrix $\widetilde{\alpha}_i$ has rank r_i which is consistent with the rank of R_i. In other words, $\widetilde{\alpha}_i$ will not be a zero matrix unless R_i is a zero matrix. This fundamentally prevents BFBCG from suffering the potential stagnation problem occurred in many Krylov subspace methods [29,57], where the solution matrices in two (and further) consecutive iterations will not be updated due to a zero parameter matrix while convergence has not been reached yet.

Lemma 3.2 *Z_{i+1} is conjugate to the search spaces $\mathscr{P}_j$ where $j < i$.*

Proof Since R_{j+1} is generated by

$$R_{j+1} = R_j - A\widetilde{P}_j\widetilde{\alpha}_j,$$

left multiplying by Z_{i+1}^T on both sides and we have

$$Z_{i+1}^T R_{j+1} = Z_{i+1}^T R_j - Z_{i+1}^T A\widetilde{P}_j\widetilde{\alpha}_j.$$

When $j < i$, according to Corollary 3.2, $Z_{i+1}^T R_j = 0$ and $Z_{i+1}^T R_{j+1} = 0$ hold. Thus, we can get

$$Z_{i+1}^T A\widetilde{P}_j\widetilde{\alpha}_j = 0$$

for all $j < i$.

Based on Theorem 3.1, $\widetilde{\alpha}_j = (\widetilde{P}_j^T A\widetilde{P}_j)^{-1}\widetilde{P}_j^T R_j$, we have

$$Z_{i+1}^T A\widetilde{P}_j(\widetilde{P}_j^T A\widetilde{P}_j)^{-1}\widetilde{P}_j^T R_j = 0.$$

Due to the fact that $\widetilde{P}_j^T A\widetilde{P}_j$ is an $r_j \times r_j$ matrix with full rank, $\widetilde{P}_j^T R_j$ is an $r_j \times s$ matrix with rank r_j by Lemma 3.1, and $r_j \leq s$, it is easy to obtain $Z_{i+1}^T A\widetilde{P}_j = 0$ $(j < i)$. □

Lemma 3.2 indicates that Z_{i+1} from BFBCG is conjugate to all previous search spaces $\mathscr{P}_j$ $(j < i)$ except $\mathscr{P}_i$. This inspires us to derive a parameter matrix $\widetilde{\beta}_i$ to construct a new search space $\mathscr{P}_{i+1}$ from Z_{i+1} by removing the components not conjugate to $\mathscr{P}_i$. Theorem 3.2 shows that, in case of rank deficiency occurring at the ith iteration, the rectangular parameter matrix $\widetilde{\beta}_i$ ensures that the new search space $\mathscr{P}_{i+1}$ is conjugate to all previous search spaces $\mathscr{P}_j$ $(j < i + 1)$.

Theorem 3.2 *Suppose that R_i loses full rank at the ith iteration. Let $\mathscr{P}_i$ denote the corresponding search space with dimension r_i ($r_i < s$). Given a matrix $\widetilde{\beta}_i \in \mathbb{R}^{r_i \times s}$ so that*

$$\widetilde{\beta}_i = -(\widetilde{P}_i^T Q_i)^{-1} Q_i^T Z_{i+1},$$

where $\widetilde{P}_i \in \mathbb{R}^{n \times r_i}$ consists of the orthonormal basis of $\mathscr{P}_i$ and $Q_i \in \mathbb{R}^{n \times r_i}$ denotes the matrix product $A\widetilde{P}_i$, then the new search space $\mathscr{P}_{i+1}$ obtained from $\widetilde{\beta}_i$ is conjugate to all previous search spaces $\mathscr{P}_j$ where $j < i + 1$.

Proof Based on the Gram–Schmidt conjugation process, the new search directions P_{i+1} at the ith iteration can be generated by

$$P_{i+1} = Z_{i+1} + \sum_{j=0}^{i} \widetilde{P}_j \beta_{i+1,j},$$

where $\widetilde{P}_j$ is the orthonormal basis of $\mathscr{P}_j$ and $\beta_{i+1,j}$ is the associated weight matrix of $\widetilde{P}_j$. As $\widetilde{P}_i^T Q_i \in \mathbb{R}^{r_i \times r_i}$ is nonsingular, by selecting $\beta_{i+1,j} = 0$ for all $j < i$ and $\widetilde{\beta}_i = \beta_{i+1,i} = -(\widetilde{P}_i^T Q_i)^{-1} Q_i^T Z_{i+1}$, it is easy to show the following:

1. for any $\widetilde{P}_j$ where $j < i$, according to Lemma 3.2,

$$\begin{aligned} \widetilde{P}_j^T A P_{i+1} &= \widetilde{P}_j^T A Z_{i+1} + \widetilde{P}_j^T A \sum_{k=0}^{j} \widetilde{P}_k \beta_{i+1,k} \\ &= \widetilde{P}_j^T A Z_{i+1} = 0; \text{ and} \end{aligned}$$

2. for $\widetilde{P}_j$ where $j = i$,

$$\begin{aligned} \widetilde{P}_i^T A P_{i+1} &= \widetilde{P}_i^T A Z_{i+1} + \widetilde{P}_i^T A \sum_{k=0}^{i} \widetilde{P}_k \beta_{i+1,k} \\ &= \widetilde{P}_i^T A Z_{i+1} + \widetilde{P}_i^T A \widetilde{P}_i \beta_{i+1,i} \\ &= \widetilde{P}_i^T A Z_{i+1} - \widetilde{P}_i^T A \widetilde{P}_i (\widetilde{P}_i^T A \widetilde{P}_i)^{-1} \widetilde{P}_i^T A Z_{i+1} \\ &= \widetilde{P}_i^T A Z_{i+1} - \widetilde{P}_i^T A Z_{i+1} = 0. \end{aligned}$$

Let the range of P_{i+1} be the new search space $\mathscr{P}_{i+1}$ and then the new search space $\mathscr{P}_{i+1}$ is conjugate to all previous search spaces $\mathscr{P}_j$ ($j < i + 1$). □

In fact, $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$ defined in BFBCG are generalized forms of the parameter matrices α_i and β_i in the BCG algorithms to avoid breakdown during BFBCG iterations. When R_i's have full column rank, BFBCG is equivalent to the original BCG algorithm. In particular, $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$ are square matrices with full rank that coincide with α_i and β_i from the original BCG algorithm where γ_i in Algorithm 1 is replaced by the inverse of the upper triangular matrix by QR decomposition, while a simplified form of $\widetilde{\beta}_i$ is chosen in BFBCG to avoid the augmented condition number of $Z_i^T R_i$. On the other

hand, if R_i loses full rank during BFBCG iterations, the rectangular parameter matrices $\widetilde{\alpha}_i$ and $\widetilde{\beta}_i$ are employed to maintain the orthogonality properties in the block Krylov subspace and avoid breakdown due to rank deficiency.

4 Convergence analysis

The convergence properties of the BFBCG algorithm are investigated in this section. By taking the rank deficiency problem into consideration, we start from studying the theoretical number of iterations needed for convergence of BFBCG. Then, the convergence rate of BFBCG is further analyzed.

4.1 Number of iterations

To solve a linear system with s right-hand sides using BCG, the block Krylov subspace

$$\begin{aligned} D_i(A, M, R_0) &= \text{block-span}\{MR_0, MAMR_0, \ldots, (MA)^i MR_0\} \\ &= \text{block-span}\{MR_0, MR_1, \ldots, MR_i\} \end{aligned}$$

is constructed to find an approximate solution block X_{i+1} at the ith iteration, where M is an SPD preconditioner. As pointed out in [38], if the impact of round-off errors can be ignored, the BCG algorithm is able to find the exact solutions within at most $\lceil n/s \rceil$ iterations, where s is the number of the right-hand sides.

As a generalized form of BCG, BFBCG shares the same convergence property if the residual matrices remain full rank s during all iterations. When rank deficiency occurs, BFBCG continues to explore the block Krylov subspace from the reduced search spaces. Proposition 4.1 shows that once a residual matrix loses full rank, rank deficiency will be inherited in the subsequent residual matrices.

Proposition 4.1 *If R_i loses full column rank at the ith iteration, the subsequent residual matrices R_j $(j > i)$ are also rank deficient.*

Proof Since

$$\begin{aligned} R_{i+1} &= R_i - A\widetilde{P}_i\widetilde{\alpha}_i \\ &= R_i - A\widetilde{P}_i(\widetilde{P}_i^T A\widetilde{P}_i)^{-1}\widetilde{P}_i^T R_i \\ &= (I - A\widetilde{P}_i(\widetilde{P}_i^T A\widetilde{P}_i)^{-1}\widetilde{P}_i^T)R_i, \end{aligned}$$

then $\text{rank}(R_i) \geq \text{rank}(R_{i+1})$ can be obtained based on the properties of matrix rank. For $j > i$, $\text{rank}(R_i) \geq \text{rank}(R_j)$ can be derived in a similar way. □

In the case that rank deficiency occurs, the block Krylov subspace can no longer be expanded by s dimensions in future iterations. Instead, the dimension of the corresponding block Krylov subspace increases by the rank of the residual matrix, which is less than s, at each subsequent iteration step. Consequently, in general, more than $\lceil n/s \rceil$ iterations are needed in BFBCG to find the solutions when rank deficiency occurs.

4.2 Convergence rate

Defining the error matrix E_{i+1} as

$$E_{i+1} = \left[e_{i+1}^{(0)}, e_{i+1}^{(1)}, \ldots, e_{i+1}^{(s-1)}\right] = X_{i+1} - X^*$$

at the ith iteration, where $e_{i+1}^{(k)}$ is the kth column of E_{i+1} and $X^* = A^{-1}B$ is the desired block solution, the block nonnegative quadratic function can be represented as

$$\text{trace}\left(\left(X_{i+1} - X^*\right)^T A(X_{i+1} - X^*)\right) = \sum_{k=0}^{s-1} \|e_{i+1}^{(k)}\|_A^2.$$

To determine the convergence rate of BCG, the initial residual matrix $R_0 = B - AX_0$ plays an important role in bounding the errors at each iteration step. Under the assumption that R_0 has full column rank, O'Leary [38] showed that the minimum error square norm $\|e_{i+1}^{(k)}\|_A^2$ $(0 \le k \le s-1)$ is bounded as

$$\|e_{i+1}^{(k)}\|_A^2 \le c^{(k)} \left(\frac{1 - \sqrt{\kappa^{-1}}}{1 + \sqrt{\kappa^{-1}}}\right)^{2(i+1)}$$

at each iteration. Here $\kappa = \lambda_n/\lambda_s$ where λ_n and λ_s are the nth and sth eigenvalues of MA, respectively, and $c^{(k)}$ is a constant only related to $e_0^{(k)}$. Nevertheless, if R_0 does not have full rank, the above error bound does not hold. Instead, assuming that R_0 has rank r_0, Theorem 4.1 shows that the convergence rate of the BFBCG algorithm is bounded by $\left(\frac{1-\sqrt{\kappa'^{-1}}}{1+\sqrt{\kappa'^{-1}}}\right)^2$, where $\kappa' = \lambda_n/\lambda_{r_0}$.

Theorem 4.1 *Suppose R_0 is rank deficient with rank r_0 $(r_0 < s)$. The minimum error square norm $\|e_{i+1}^{(k)}\|_A^2$ is bounded as*

$$\|e_{i+1}^{(k)}\|_A^2 \le c \left(\frac{1 - \sqrt{\kappa'^{-1}}}{1 + \sqrt{\kappa'^{-1}}}\right)^{2(i+1)},$$

where c is a constant only related to E_0 and $\kappa' = \lambda_n/\lambda_{r_0}$.

Proof Assuming that the $n \times s$ residual matrix R_0 has rank r_0, which is potentially rank deficient, 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. Since $E_0 = A^{-1}R_0$ and $E_{i+1} = \Phi_i(MA)E_0$, where $\Phi_i(MA)$ is a polynomial of degree i, we have $E_{i+1} = (E_{i+1}', 0)\delta$ and each column in E_{i+1} can be expressed as

$$e_{i+1}^{(k)} = \sum_{j=0}^{r_0-1} \delta_{jk} e'^{(j)}_{i+1},$$

where $e'^{(j)}_{i+1}$ is the jth column of E'_{i+1}.

Hence, the error bound of the square norm $\|e_{i+1}^{(k)}\|_A^2$ becomes

$$\begin{aligned}\|e_{i+1}^{(k)}\|_A^2 &= \|\sum_{j=0}^{r_0-1} \delta_{jk} e'^{(j)}_{i+1}\|_A^2 \le \sum_{j=0}^{r_0-1} \delta_{jk}^2 \|e'^{(j)}_{i+1}\|_A^2 \\ &\le \sum_{j=0}^{r_0-1} \delta_{jk}^2 c^{(j)} \left(\frac{1-\sqrt{\kappa'^{-1}}}{1+\sqrt{\kappa'^{-1}}}\right)^{2(i+1)} \le c\left(\frac{1-\sqrt{\kappa'^{-1}}}{1+\sqrt{\kappa'^{-1}}}\right)^{2(i+1)},\end{aligned}$$

where $c = \sum_{j=0}^{r_0-1} \delta_{jk}^2 c^{(j)}$ and $\kappa' = \lambda_n/\lambda_{r_0}$. □

In case of rank deficiency, i.e., R_i loses full rank to r_i, BCG has to restart with a reduced block size. Restarting is unfavorable in parallel computing, where reinitiating processes and redistributing workload are necessary. More importantly, the restarting BCG uses the range of R_i as the initial search space and abandons all search spaces explored before. As a result, the restarted BCG has a lower convergence rate of $\left(\frac{1-\sqrt{\kappa''^{-1}}}{1+\sqrt{\kappa''^{-1}}}\right)^2$, where $\kappa'' = \lambda_n/\lambda_{r_i}$. In contrast, without restarting, BFBCG yields faster convergence than restarting BCG, because BFBCG still takes advantage of the search space information constructed previously. Hence, the overall convergence rate of BFBCG lies between $\left(\frac{1-\sqrt{\kappa''^{-1}}}{1+\sqrt{\kappa''^{-1}}}\right)^2$ and $\left(\frac{1-\sqrt{\kappa^{-1}}}{1+\sqrt{\kappa^{-1}}}\right)^2$, where $\kappa'' = \lambda_n/\lambda_{r_i}$ and $\kappa = \lambda_n/\lambda_s$, respectively.

5 Numerical results

In this section, we first use a simple linear system with a 6×6 coefficient matrix and two right-hand side columns as an example to demonstrate how BFBCG handles each individual rank deficiency situation. Then, BFBCG is used to solve a more "realistic" linear system with a 7102×7102 coefficient matrix and 200 right-hand side columns and address combined rank deficiency situations. Moreover, we compare the performance between BCG with restarting and BFBCG in handling rank deficiency on a set of SPD matrices. In these examples, the solutions of all right–hand sides are concerned. The tolerance of convergence is set to 10^{-7}.

5.1 A simple 6 × 6 example

Considering a linear system with two right-hand sides, where the SPD coefficient matrix A is given as follows

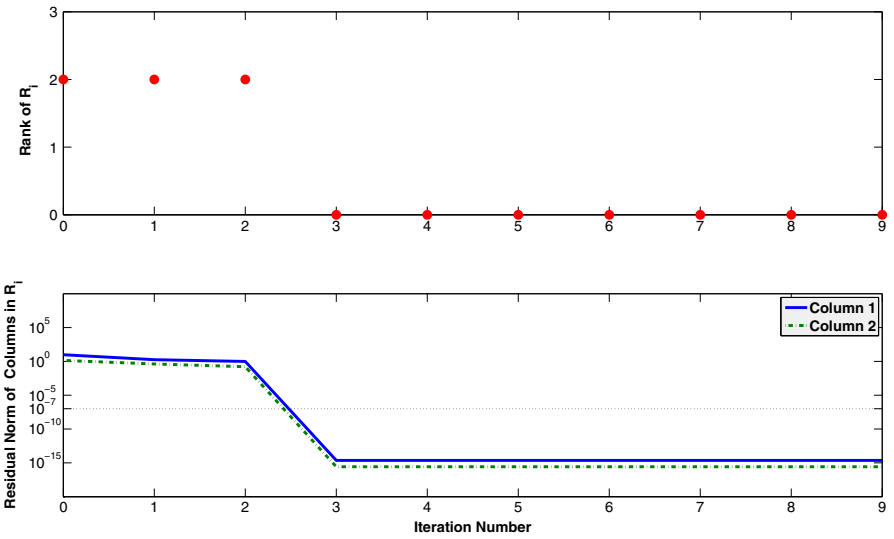


Fig. 1 Matrix rank and the two column residual norms of R_i in Case 1 (columns in each block residual R_i are linearly independent) along BFBCG iterations

$$A = \begin{bmatrix} 15 & 5 & 4 & 3 & 2 & 1 \\ 5 & 35 & 9 & 8 & 7 & 6 \\ 4 & 9 & 46 & 12 & 11 & 10 \\ 3 & 8 & 12 & 50 & 14 & 13 \\ 2 & 7 & 11 & 14 & 19 & 15 \\ 1 & 6 & 10 & 13 & 15 & 45 \end{bmatrix}.$$

Four initial residual matrices are used as test cases to study the behavior of BFBCG. For simplicity, the preconditioner matrix M is set to identity.

Case 1 (the residual matrix R_i without rank deficiency)

In this case, the initial block residual matrix R_0 is selected as

$$R_0 = \begin{bmatrix} 1 & 0.537266261211281 \\ 2 & 0.043775211060964 \\ 3 & 0.964458562037146 \\ 4 & 0.622317517840541 \\ 5 & 0.552735938776748 \\ 6 & 0.023323943544997 \end{bmatrix},$$

where the elements in the second column are random values uniformly generated over interval (0, 1) and thus the second column is linearly independent with the first one. Figure 1 shows the matrix rank of residual matrix R_i (upper) and the norm of columns in R_i (lower) at each iteration step. One can find that eventually BFBCG, which is equivalent to BCG in this case, converges within three iterations and each residual matrix R_i has full rank 2 before convergence.

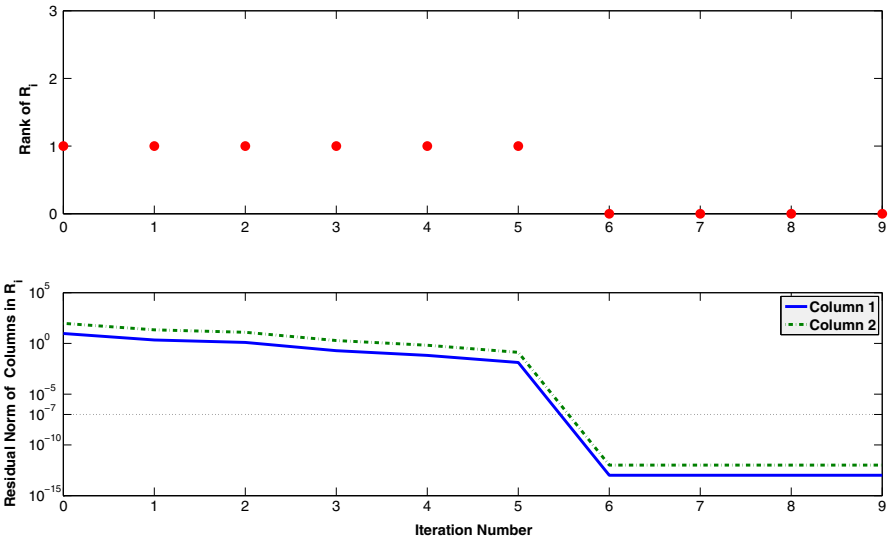


Fig. 2 Matrix rank and the two column residual norms of R_i in Case 2 (columns in the initial block residual R_0 are linearly dependent) along BFBCG iterations

Case 2 (columns in the initial block residual R_0 are linearly dependent)
In this case, the initial block residual matrix R_0 is set to

$$R_0 = \begin{bmatrix} 1 & 10 \\ 2 & 20 \\ 3 & 30 \\ 4 & 40 \\ 5 & 50 \\ 6 & 60 \end{bmatrix},$$

where the second column of R_0 is a scalar multiple of the first one and thus the rank of R_0 is one. As shown in Fig. 2, the behavior of BFBCG is the same as that of running two individual parallel CGs, where the matrix rank of R_i is maintained at one until BFBCG converges at step 6.

Case 3 (convergence of one or more but not all columns in the residual matrix R_i)
The initial block residual matrix R_0 is

$$R_0 = \begin{bmatrix} 1 & 0.027212780358615 \\ 2 & 0.117544343373396 \\ 3 & 0.140184539179715 \\ 4 & 0.605659566833592 \\ 5 & 0.323269030695212 \\ 6 & 0.590821508384101 \end{bmatrix}.$$

Clearly, the two columns in R_0 are linearly independent and R_0 has full column rank. As the second system converges first at the 1st iteration shown in Fig. 3, which leads to

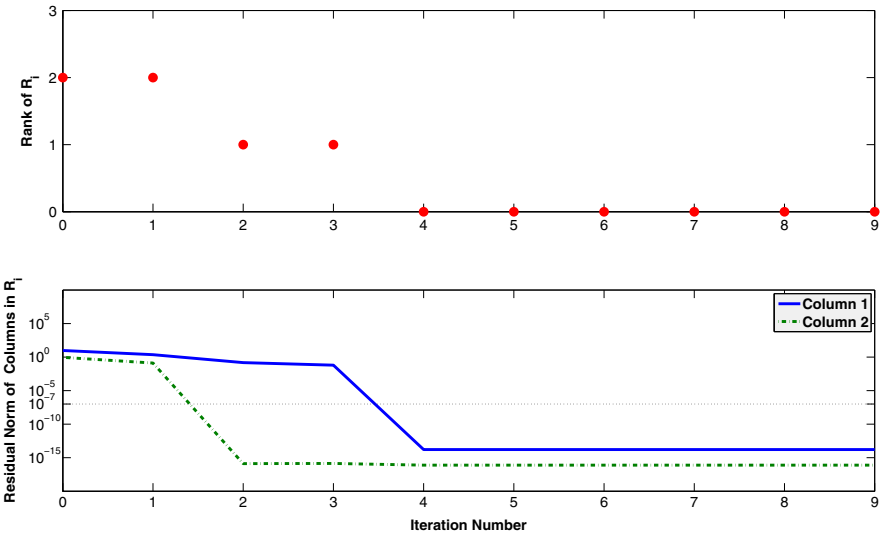


Fig. 3 Matrix rank and the two column residual norms of R_i in Case 3 (convergence of one or more columns in the residual matrix R_i) along BFBCG iterations

the rank deficiency in R_2. However, since the first system has not reached convergence yet, BFBCG continues to update the first column in the consequent residual matrices. The overall system converges at the 4th iteration, which is faster than Case 2.

Case 4 (columns in the residual matrix R_i become linearly dependent during BFBCG iterations)

In Case 4, we adopt the following the initial block residual matrix R_0

$$R_0 = \begin{bmatrix} 1 & -8.888614458250306 \\ 2 & -10.999025290685955 \\ 3 & -19.339674247091921 \\ 4 & -10.289152668326622 \\ 5 & 18.107579559267656 \\ 6 & -8.930794511222629 \end{bmatrix},$$

where the two columns are linearly independent and R_0 has rank of two. The key difference from Case 3, as illustrated in Fig. 4, is that the two column vectors in R_2 become identical. As a result, the rank of R_2 is reduced to 1 but none of the systems have converged yet. In the subsequent iterations, BFBCG updates both columns of the residual matrices and converges at the 4th iteration.

5.2 Example with a bigger coefficient matrix with combined rank deficiency situations

We use a matrix "Kuu" from the University of Florida sparse matrix collection [9] as the coefficient matrix of a block linear system with 200 right-hand sides to demonstrate

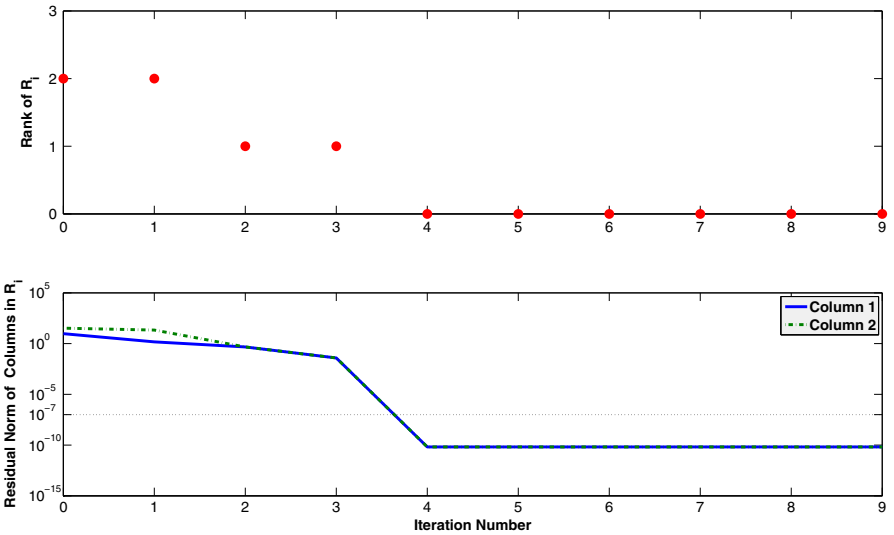
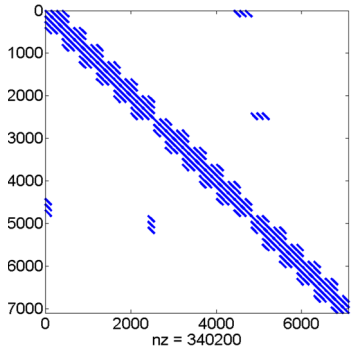


Fig. 4 Matrix rank and the two residual norms of R_i in Case 4 (columns in the residual matrix R_i become linearly dependent at the 2nd iteration) along BFBCG iterations

Fig. 5 Sparse pattern of matrix "Kuu"



the effectiveness of BFBCG in addressing the breakdown problem with combined rank deficiency situations. "Kuu" is a 7102×7102 SPD matrix with 340, 200 nonzero elements arisen from a structural problem whose sparse pattern is shown in Fig. 5. To construct linearly dependent vector components in the initial residual matrix R_0, we intentionally set the elements in the first 198 columns of the right–hand side matrix B as randomly generated numbers while the last two columns are created as linear combinations of the first 198 columns. The initial guess X_0 is set to be the same as B and a preconditioner M is constructed using the Crout version of ILU factorization [31,46] with 0.01 element drop tolerance.

Figure 6 illustrates the change of the matrix rank of R_i (upper), the condition number of $P_i^T A P_i$ (middle), as well as the maximum and minimum residual norms of the columns in R_i (lower) along the BFBCG iterations. The condition number of $P_i^T A P_i$ is bounded by the condition number of A. One can find that rank deficiency happens at the very beginning because of the linearly dependent vectors in B that we set intentionally.

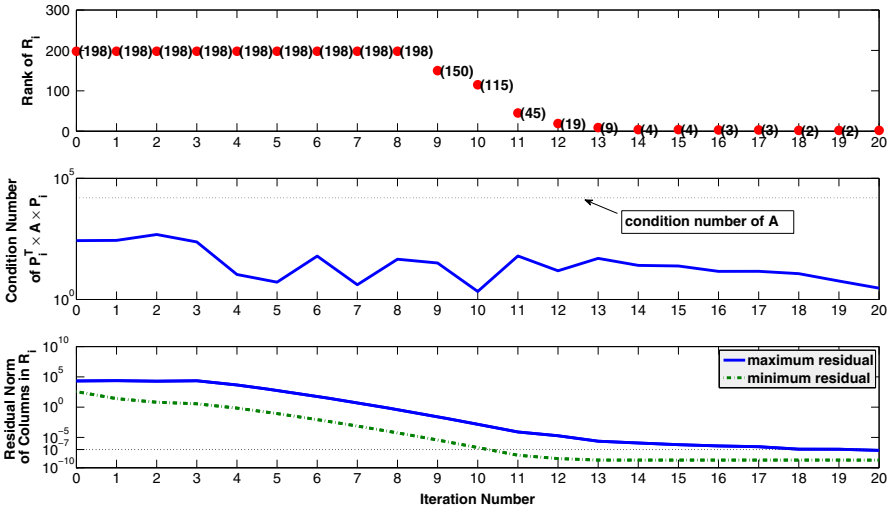


Fig. 6 Matrix rank of R_i, condition number of $P_i^T A P_i$, and the corresponding maximum and minimum residual norm for a block linear system with 200 right–hand sides using "Kuu" as the coefficient matrix along BFBCG iterations

The rank of the residual matrix R_i starts to drop down to 150 at the 9th iteration because further linear dependence occurs during the BFBCG process; however, none of the systems converge to the desired resolution yet. At the 11th iteration, the residual norms of some columns in R_i are smaller than the given error tolerance, indicating that some but not all systems have reached convergence. Correspondingly, the matrix rank of R_i decreases further to 45. After all, BFBCG is able to deal with the combination of various rank deficiency situations and continues to improve the solution accuracy based on the reduced Krylov subspace. Eventually, all systems reach convergence at the 20th iteration.

5.3 Comparison with the restarting scheme

When breakdown actually occurs, the original BCG algorithm has to restart with a reduced block size. Table 1 compares the performance of BCG with restarting and BFBCG on a set of SPD matrices from the structural engineering applications in the Harwell–Boeing sparse matrix collection [11]. We use a right-hand side matrix B consisting of ten random column vectors. Particularly, we scale the elements in the first eight columns of B by the matrix norm of A to amplify the magnitude difference among column vectors so that rank deficiency can easily occur. The Crout version of ILU preconditioners is applied. The computational experiments are carried out on the XSEDE TACC Stampede System [54]. When breakdown happens and restarting causes the loss of all search spaces that have been explored before, BCG typically takes more iteration steps to reach convergence than BFBCG. In contrast, BFBCG is able to continue to update the solution blocks from the reduced search spaces without being interrupted. Moreover, restarting requires additional operations to reinitiate the

Table 1 Performance comparison between BFBCG and BCG with restarting

Name	Rows	Columns	Nonzeros	BCG with restarting			BFBCG	
				# of iterations	# of restarts	Time (s)	# of iterations	Time (s)
BCSSTK14	1806	1806	32,630	9	5	1.54	8	0.68
BCSSTK15	3948	3948	60,882	19	11	6.28	14	3.18
BCSSTK16	4884	4884	147,631	8	4	3.8	8	2.44
BCSSTK17	10,974	10,974	219,812	19	16	40.69	15	17.39
BCSSTK18	11,948	11,948	80,519	14	8	28.49	14	18.44

computational process, which results in significantly more computational time in BCG than that in BFBCG.

6 Relation to other work

6.1 Handling multiple right-hand sides

In the literature, quite a few Krylov subspace methods have been developed to handle linear systems with multiple right-hand sides. These methods can be roughly classified into three categories according to their ways of forming the basis of the underlying Krylov subspace.

The first category is based on the seed methods, where a seed system with a selected single right-hand side is solved by a Krylov subspace method at first and then the Krylov subspace information of the seed system is recycled to accelerate the convergence in the other systems until solutions to all right-hand sides are found. The seed scheme appears to be effective in the cases where the multiple right-hand side vectors are similar to each other. Following this idea, a set of seed methods, including the seed Lanczos methods [41,45], the seed CG methods [6,13,53,56], and the seed GMRES methods [51], have been designed for various applications. As the multiple right-hand sides are solved in a sequential order, the basis of the Krylov subspace for each system is generally formed by a series of matrix-vector multiplications. In particular, the Krylov subspace $\text{span}\{r_0^{(k)}, Ar_0^{(k)}, \ldots, A^i r_0^{(k)}\}$ is constructed in solving the kth system for the kth right-hand side. Here the solution vector $x_{i+1}^{(k)}$ can be expressed as

$$x_{i+1}^{(k)} = x_0^{(k)} + \sum_{j=0}^{i} \psi_j^{(k)} A^j r_0^{(k)},$$

where $\psi_j^{(k)}$'s are the evaluated scalars in the seed methods.

The second category is based on the global methods, which projects the initial residual matrix onto the Krylov subspace in a matrix form, so called the matrix Krylov subspace [24], and allows solutions to the multiple right-hand sides to be evaluated simultaneously. Jbilou et al. [24] introduced the global FOM and global GMRES algorithms, where the F-orthonormal basis of the Krylov subspace is generated using the global Arnoldi or Lanczos process. Variations of the global Krylov subspace

methods have been proposed later, including global CG [48], global BiCG [25], and global BCR [58]. These global methods are built on the matrix Krylov subspace span$\{R_0, AR_0, \ldots, A^i R_0\}$ [24]. As pointed out in [21], the solution block in the global methods is treated as an optimal linear combination of a set of basis matrices of the matrix Krylov subspace, such that at the ith iteration,

$$X_{i+1} = X_0 + \sum_{j=0}^{i} \psi_j A^j R_0,$$

where ψ_j's are scalar parameters.

The third category is the block methods, where a block Krylov subspace is explored to evaluate solutions to the multiple right-hand sides simultaneously. The original work of block Krylov subspace linear solvers is presented in [38], where block BiCG, block CG, and block MINRES are first proposed. Since then, a variety of block implementations to the standard Krylov methods, such as block BiCGSTAB [12], block QMR [16,17], and block GMRES [1,52,55], have also been developed. A comprehensive survey on these block methods can be found in [20]. The key difference in block methods compared to the seed methods and the global methods is that the basis vectors are obtained from the block Krylov subspace [21,22,38], i.e., block-span$\{R_0, AR_0, \ldots, A^i R_0\}$. Correspondingly, the approximate solution matrix X_{i+1} at the ith iteration can be expressed as

$$X_{i+1} = X_0 + \sum_{j=0}^{i} A^j R_0 \Psi_j,$$

where Ψ_j's are parameters in matrix form.

The BFBCG algorithm proposed in this paper falls into the category of the block methods. In general, evaluating the parameter matrices Ψ_j requires inverting certain block matrices, which can suffer from breakdown in case of rank deficiency. BFBCG is designed to address the breakdown problem in BCG.

6.2 Addressing the breakdown problem

In block Krylov subspace methods, inverting block matrices is needed to evaluate multiple right-hand sides simultaneously. During the iterations, some of these block matrices may lose rank. Consequently, inverting a block matrix with rank deficiency is one of the roots of the breakdown problem in block Krylov subspace methods. As a result, breakdown becomes a major cause of numerical instability in almost every block Krylov subspace method [5,21,37,38,43]. Although certain work in the literature [7] indicates that it usually happens with a very small probability in practice, breakdown, if it actually occurs, may seriously hurt the computational performance of a block solver. For mission-critical applications, this is particularly unfavorable.

Generally, several strategies have been proposed to address the breakdown problem in block Krylov subspace methods. One strategy is restarting [38], i.e., monitoring the rank of block matrices and restarting block Krylov subspace methods once matrix rank

loss is detected. Restarting requires eliminating the linearly dependent or converged vectors. Moreover, for the block linear systems where all solutions to the multiple right-hand sides are needed, the transformation matrix to the reduced block solution has to be stored each time when rank deficiency occurs. These transformation matrices are later used to recover all block solutions. An alternative strategy is to detect and keep the dependent vectors and then reintroduce them in the subsequent iteration steps [1,28,43], which has been used to handle the inexact breakdown problem in block GMRES. Another strategy is based on reducing the basis of the search space or deflation [16,21], whose rationale is to remove the linearly dependent vectors at the occurrences of rank deficiency and then to continue the remaining iterations with a reduced block size. (Note that the term "deflation" is also used as a convergence acceleration technique in many Krylov subspace methods [7,18,47].)

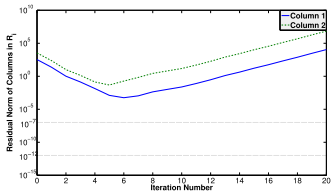
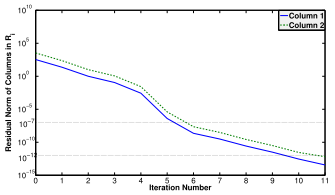
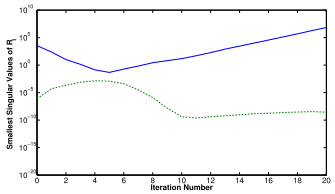
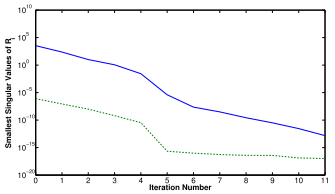
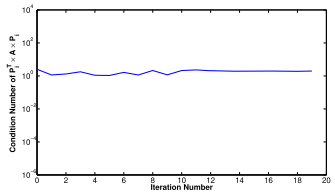
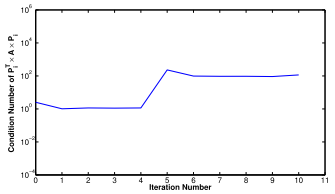
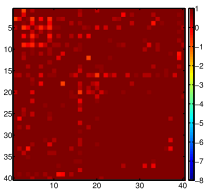
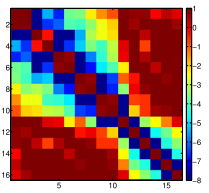
In the context of BCG, both restarting and reducing the basis of the search space have been proposed to remedy the breakdown problem in BCG. As analyzed in Sect. 4, restarting inevitably abandons all search spaces explored before, which may slow down convergence. The variable block BCG algorithm [37] is based on reducing the basis of the search space without restarting, where an A-orthogonal projector is constructed to reduce the size of the block matrices adaptively during BCG iterations; however, this method is only of interest for getting the solution of a single right–hand side and differs from BFBCG in the way of updating search matrices, which requires integrating a series of intermediate matrices. The main contributions of this paper include (1) analyzing all possible rank deficiency situations that can lead to breakdown in BCG; (2) providing a simple solution of BFBCG to avoid breakdown caused by all possible rank deficiency situations in BCG while finding solutions to all right-hand sides; (3) justifying the orthogonality properties and convergence of BFBCG in case of rank deficiency with mathematical rigor; and (4) theoretically analyzing the convergence of BFBCG. The key to avoiding breakdown in BFBCG is to reduce the basis of the search space when rank deficiency occurs. Therefore, alternative formulations of BCG, such as those involving LU or QR decomposition of the search direction block or QR factorization of the residual block [10], can also reduce the basis of the search space and achieve similar breakdown-free implementations.

The breakdown situations in general block Krylov subspace methods are more complicated than BCG. In fact, rank deficiency is not the only factor that causes breakdown in general block Krylov subspace methods. For example, block QMR [16] may encounter division by zero and block GMRES may break down when deflation is applied [18]. It is important to note that these situations do not occur in the BCG methods [3,18,27,36,46]. Another potential problem for numerical instability is stagnation, i.e., the residual norm does not change any more but without reaching convergence [29,57]. Fortunately, our analysis of the parameter matrices indicates that stagnation will not occur in BFBCG, either.

6.3 Handling the near-breakdown problem

Recent studies [20,22,43,49] have showed that the almost linearly dependent vectors in the residual block matrices may cause loss of orthogonality during iterations and

Table 2 Comparison between BCG and BFBCG in case of near-breakdown

	BCG	BFBCG
Convergence		
Smallest singular values of R_i		
Condition number of $P_i^T AP_i$		
Colormap of A–orthogonality between search matrices		

thus slow down or even prevent convergence of block Krylov subspace methods. This is referred to as the near-breakdown problem. We hereby investigate the impacts of the near-breakdown problem on BFBCG in comparison with the original BCG algorithm. To simulate the near-breakdown situations, we use a linear system of a 10×10 random coefficient matrix with a small condition number to eliminate the impact from the matrix itself. We initialize a block residual matrix R_0 with two nearly linearly dependent vectors, where the second column is generated by multiplying the first one by 10 while adding small random perturbations. The coefficient matrix and the right-hand side block matrix have been made available on our website [32].

Table 2 compares BCG and BFBCG in the case when near-breakdown occurs. We monitor the smallest singular value of R_i and a parameter τ is designated as a tolerance threshold of linear dependence among the block residual vectors in BFBCG. Here, τ is set to 10^{-12}. One can find that the nearly linearly dependent vectors in R_i result in a certain loss of A-orthogonality among search matrices during the iterations in both

BCG and BFBCG, which is consistent with the analysis presented in [20,49]. This is due to the fact that constructing the new search matrices is sensitive to the round-off errors when the residual matrices are nearly rank-deficient. Nevertheless, the computation of the parameter matrix β_i in BCG requires evaluation of $\gamma_i^{-1}(Z_i^T R_i)^{-1}$, where the nearly linear dependence of the columns in R_i can lead to large round-off errors. As shown in Table 2, BCG suffers from complete loss of A-orthogonality and fails to converge. In contrast, the computation of $\widetilde{\beta}_i$ in BFBCG relies only on calculating $(\widetilde{P}_i^T A \widetilde{P}_i)^{-1}$ and thus maintains relatively better A-orthogonality. Moreover, BFBCG is designed to enforce A-orthogonality of every two consecutive search matrices. As a result, BFBCG is able to evolve with nearly linear dependence in R_i. When the singularity of R_i falls under threshold τ, the reduced search matrices are generated in such a way that A-orthogonality with the previous search directions is maintained. Consequently, the relatively better A-orthogonality allows BFBCG to reach solutions with desired precision.

7 Conclusions

Considering solving a linear system with multiple right-hand sides, we develop the BFBCG algorithm to address the rank deficiency problem in BCG. All possible situations of rank deficiency causing BCG breakdown are analyzed. New forms of parameter matrices are introduced in BFBCG and are theoretically justified to ensure that BFBCG is able to completely avoid breakdowns due to rank deficiency. Convergence of BFBCG is analyzed accordingly in case of rank deficiency. The effectiveness of BFBCG is shown in several numerical examples suffering breakdown or near breakdown due to rank deficiency.

Acknowledgments Y. Li acknowledges support from National Science Foundation through Grant No. CCF-1066471. H. Ji acknowledges support from Old Dominion University Modeling and Simulation Fellowship. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation Grant No. ACI-1053575. The authors would like to thank Dr. Michiel E. Hochstenbach, Dr. Martin H. Gutkne-cht, and other reviewers for their very helpful comments and suggestions on this work.

References

1. Agullo, E., Giraud, L., Jing, Y.F.: Block GMRES method with inexact breakdowns and deflated restarting. SIAM J. Matrix Anal. Appl. **35**(4), 1625–1651 (2014)
2. Axelsson, O.: On preconditioning and convergence acceleration in sparse matrix problems. CERN 74–10, Genèva, Switzerland (1974)
3. Barrett, R., Berry, M., Chan, T., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., van der Vorst, H.: Templates for the solution of linear systems: building blocks for iterative methods. SIAM, Philadelphia (1994)
4. Birk, S., Frommer, A.: A CG method for multiple right hand sides and multiple shifts in lattice QCD calculations. In: Proceedings of the XXIX International Symposium on Lattice Field Theory, pp. 10–16 (2011)
5. Broyden, C.G.: A breakdown of the block CG method. Optim. Method. Softw. **7**(1), 41–55 (1996)
6. 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)

7. 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, IL (2011)
8. 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
9. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1–25. https://www.cise.ufl.edu/research/sparse/matrices/ (2011). Accessed 28 Feb 2015
10. Dubrulle, A.A.: Retooling the method of block conjugate gradients. Electron. Trans. Numer. Anal. **12**, 216–233 (2001)
11. Duff, I.S., Grimes, R.G., Lewis, J.G.: Users' guide for the Harwell–Boeing sparse matrix collection (Release I). Tech. rep., TR/PA/92/86, CERFACS, Toulouse, France. http://math.nist.gov/MatrixMarket/ (1992). Accessed 28 Feb 2015
12. 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)
13. Erhel, J., Guyomarc'h, F.: An augmented conjugate gradient method for solving consecutive symmetric positive definite linear systems. SIAM J. Matrix Anal. Appl. **21**(4), 1279–1299 (2000)
14. Feng, Y.T., Owen, D.R.J., Perić, D.: A block conjugate gradient method applied to linear systems with multiple right-hand sides. Comput. Methods Appl. Mech. Eng. **127**(1), 203–215 (1995)
15. Fletcher, R.: Conjugate gradient methods for indefinite systems. In: Numerical Analysis of Proceedings of 6th Biennial Dundee Conference, University of Dundee, Dundee, 1975, Lecture notes in mathematics, vol. 506, pp. 73–89. Springer, Berlin (1976)
16. Freund, R.W., Malhotra, M.: A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides. Linear Algebra Appl. **254**(1), 119–157 (1997)
17. Freund, R.W., Nachtigal, N.M.: QMR: a quasi-minimal residual method for non-Hermitian linear systems. Numer. Math. **60**(1), 315–339 (1991)
18. 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)
19. Golub, G.H., van Loan, C.F.: Matrix computations, 4th edn. Johns Hopkins University Press, Baltimore (2012)
20. Gutknecht, M.H.: Block Krylov space solvers: a survey. Tech. rep. http://www.sam.math.ethz.ch/~mhg/talks/bkss.pdf (2005). Accessed 28 Feb 2015
21. 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 Publishers, New Delhi (2007)
22. Gutknecht, M.H., Schmelzer, T.: The block grade of a block Krylov space. Linear Algebra Appl. **430**(1), 174–185 (2009)
23. Hestenes, M.R., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. **49**, 409–436 (1952)
24. Jbilou, K., Messaoudi, A., Sadok, H.: Global FOM and GMRES algorithms for matrix equations. Appl. Numer. Math. **31**(1), 49–63 (1999)
25. Jbilou, K., Sadok, H., Tinzefte, A.: Oblique projection methods for linear systems with multiple right-hand sides. Electron. Trans. Numer. Anal. **20**, 119–138 (2005)
26. Ji, H., Sosonkina, M., Li, Y.: An implementation of block conjugate gradient algorithm on CPU–GPU processors. In: Proceedings of the 1st International Workshop on Hardware–Software Co–Design for High Performance Computing, IEEE Press, pp. 72–77 (2014)
27. Kaasschieter, E.F.: Preconditioned conjugate gradients for solving singular systems. J. Comput. Appl. Math. **24**(1), 265–275 (1988)
28. Langou, J.: Iterative methods for solving linear systems with multiple right-hand sides. Ph.D. thesis, CERFACS, France (2003)
29. Leyk, Z.: Breakdowns and stagnation in iterative methods. BIT Numer. Math. **37**(2), 377–403 (1997)
30. Li, G.: A block variant of the GMRES method on massively parallel processors. Parallel Comput. **23**(8), 1005–1019 (1997)
31. Li, N., Saad, Y., Chow, E.: Crout versions of ILU for general sparse matrices. SIAM J. Sci. Comput. **25**(2), 716–728 (2003)
32. Li, Y.: Matrices for the near-breakdown example. http://www.cs.odu.edu/~yaohang/publications/BFBCGappendix.pdf (2016). Accessed 30 Aug 2015

33. Lim, J.S., Un, C.K.: Block conjugate gradient algorithms for adaptive filtering. Signal Process. **55**(1), 65–77 (1996)
34. McCarthy, J.F.: Block-conjugate-gradient method. Phys. Rev. D. **40**(6), 2149–2152 (1989)
35. Nakamura, Y., Ishikawa, K.I., Kuramashi, Y., Sakurai, T., Tadano, H.: Modified block BiCGSTAB for lattice QCD. Comput. Phys. Comm. **183**(1), 34–37 (2012)
36. Nicolaides, R.A.: Deflation of conjugate gradients with applications to boundary value problems. SIAM J. Numer. Anal. **24**(2), 355–365 (1987)
37. 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)
38. O'Leary, D.P.: The block conjugate gradient algorithm and related methods. Linear Algebra Appl. **29**, 293–322 (1980)
39. O'Leary, D.P.: Parallel implementation of the block conjugate gradient algorithm. Parallel Comput. **5**(1), 127–139 (1987)
40. O'Leary, D.P., Widlund, O.: Capacitance matrix methods for the Helmholtz equation on general three-dimensional regions. Math. Comp. **33**, 849–879 (1979)
41. Parlett, B.N.: A new look at the Lanczos algorithm for solving symmetric systems of linear equations. Linear Algebra Appl. **29**, 323–346 (1980)
42. Reid, J.K.: On the method of conjugate gradients for the solution of large sparse systems of linear equations. In: Reid, J.K. (ed.) Large sparse sets of linear equations, pp. 231–254. Academic Press, New York (1971)
43. Robbé, M., Sadkane, M.: Exact and inexact breakdowns in the block GMRES method. Linear Algebra Appl. **419**(1), 265–285 (2006)
44. Rutishauser, H.: Theory of gradient methods. In: Refined iterative methods for computation of the solution and the eigenvalues of self–adjoint boundary value problems. Basel–Stuttgart, Institute of Applied Mathematics, Birkhäuser Verlag, Zurich, pp. 24–49 (1959)
45. Saad, Y.: On the Lanczos method for solving symmetric linear systems with several right-hand sides. Math. Comput. **48**(178), 651–662 (1987)
46. Saad, Y.: Iterative methods for sparse linear systems, 2nd edn. SIAM, Philadelphia (2003)
47. Saad, Y., Yeung, M., Erhel, J., Guyomarc'h, F.: A deflated version of the conjugate gradient algorithm. SIAM J. Sci. Comput. **21**(5), 1909–1926 (2000)
48. Salkuyeh, D.K.: CG-type algorithms to solve symmetric matrix equations. Appl. Math. Comput. **172**(2), 985–999 (2006)
49. Schmelzer, T.: Block Krylov methods for Hermitian linear systems. Diploma thesis, Department of Mathematics, University of Kaiserslautern, Germany (2004)
50. Shewchuk, J.R.: An introduction to the conjugate gradient method without the agonizing pain. Tech. rep., School of Computer Science, Carnegie Mellon University, Pittsburgh, PA (1994)
51. Simoncini, V., Gallopoulos, E.: An iterative method for nonsymmetric systems with multiple right-hand sides. SIAM J. Sci. Comput. **16**(4), 917–933 (1995)
52. Simoncini, V., Gallopoulos, E.: Convergence properties of block GMRES and matrix polynomials. Linear Algebra Appl. **247**, 97–119 (1996)
53. Smith, C.F., Peterson, A.F., Mittra, R.: A conjugate gradient algorithm for the treatment of multiple incident electromagnetic fields. IEEE Trans. Antennas Propag. **37**(11), 1490–1493 (1989)
54. Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D., Roskies, R., Scott, J.R., Wilkins-Diehr, N.: XSEDE: accelerating scientific discovery. Comput. Sci. Eng. **16**(5), 62–74 (2014)
55. Vital, B.: Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multi-processeur. Ph.D. thesis, Université de Rennes I, Rennes, France (1990)
56. Van der Vorst, H.A.: An iterative solution method for solving $f(A)x = b$, using Krylov subspace information obtained for the symmetric positive definite matrix A. J. Comput. Appl. Math. **18**(2), 249–263 (1987)
57. Zavorin, I., O'Leary, D.P., Elman, H.: Complete stagnation of GMRES. Linear Algebra Appl. **367**, 165–183 (2003)
58. Zhang, J., Dai, H., Zhao, J.: A new family of global methods for linear systems with multiple right-hand sides. J. Comput. Appl. Math. **236**(6), 1562–1575 (2011)