Columbia International Publishing
American Journal of Algorithms and Computing
(2016) Vol. 3 No. 1 pp. 1-13
doi:10.7726/ajac.2016.1001
Research Article

A Note on the Block and Seed BiCGSTAB Algorithms for Nonsymmetric Multiple Linear Systems

K. Jbilou[1] *

Received 1 July 2015; Published online 20 February 2016

Abstract

In this paper we first give a new derivation of the Bl-BiCGSTAB algorithm for solving block linear systems. This derivation is based only on linear algebra and didn't needs matrix orthogonal polynomials. We also propose a new method for solving linear systems with the same coefficient matrix and different right-hand sides. The proposed method is based on the BiCGSTAB algorithm and is especially efficient when the right-hand sides vector-columns are closely related. Some numerical examples are reported to illustrate the effectiveness of the proposed methods.

Keywords: Block Krylov subspace; Lanczos method; Multiple right-hand sides; Nonsymmetric linear systems; Seed systems

2000 Mathematics Subject Classification: Primary: 65F10

1. Introduction

Many applications require the solution of several linear systems of equations with the same coefficient matrix and different right-hand sides. The problem can be written as

$$A\,X = B, \tag{1}$$

where A is an $n \times n$ nonsingular and nonsymmetric real matrix, B and X are $n \times s$ rectangular matrices whose columns are $b^{(1)}, b^{(2)}, \ldots, b^{(s)}$ and $x^{(1)}, x^{(2)}, \ldots, x^{(s)}$, respectively and s is of moderate size $s \ll n$.

When n is small, we can use the LU factorization; we first decompose the matrix A at a cost of $O(n^3)$ operations and then solve for each right-hand side at a cost of $O(n^2)$. So, for direct methods, the main cost is the decomposition of A and each right-hand side is solved efficiently by making use of this decomposition. However, for large n, direct methods become very expensive.

*Corresponding email: jbilou@lmpa.univ-littoral.fr.
1 Université du Littoral Côte d'Opale, zone universitaire de la Mi-voix, batiment H. Poincaré, 50 rue F. Buisson, BP 699, F-62228 Calais Cedex, France.

For large problems, we can use solvers such as the GMRES method with a preconditioner to each linear system. However, as the matrix of these linear systems is the same, it is more efficient to use a method for all systems simultaneously. Thus, several generalizations of the classical Krylov subspace methods have been developed in the last years.

The first class of these methods contains the block solvers such as the block Bi-Conjugate Gradient algorithm (Bl-BCG) [13] (see [17] for a stabilized version), the block generalized minimal residual (Bl-GMRES) algorithm[18] and the block quasi minimal residual (Bl-QMR) algorithm [6, 10]) and recently the block BiCGSTAB algorithm (Bl-BiCGSTAB) [4]. The first three methods construct, at step k, an approximation of the form $X_k = X_0 + Z_k$ where Z_k belongs to the block Krylov subspace $\mathcal{K}_k(A, R_0) = \text{Range}(\{R_0, AR_0, \ldots, A^{k-1}R_0\})$ with $R_0 = B - AX_0$ and $X_0 \in \mathbb{R}^{n\times s}$ is an initial guess. The correction Z_k is determined by using a (semi)-minimization or an orthogonality property. The purpose of these block methods is to converge faster than their single right-hand side counterparts.

Another approach consists in pro jecting the initial residual matrix globally onto a matrix Krylov subspace. This second class contains the global GMRES algorithm [7] and global Lanczos-based methods [8]; other global approaches could be found in [11].

A third class of methods uses a single linear system, named the seed system and then consider the corresponding Krylov subspace. The initial residuals of the other linear systems are projected onto this Krylov subspace. The whole process is repeated with an other seed system until all the systems are solved. This procedure has been used in [1, 14, 15, 20, 22] for the conjugate gradient method and in [19] when the matrix A is nonsymmetric using the GMRES algorithm. The seed approach is very effective when the right-hand sides are closely related [1].

In this paper we first give a new derivation of the block BiCGSTAB algorithm without using matrix orthogonal polynomials as we did in [4]. This technique could be used to cure possible breakdowns in the method. We also propose a new seed BiCGSTAB algorithm which is effective, in particular, when solving multiple linear systems with closely related right-hand sides.

The remainder of the paper is organized as follows. In Section 2, we show how to use only basic linear algebra tools to define the block BiCGSTAB algorithm. In Section 3 we define the new seed method without any use of orthogoal polynomials. Section 4 reports on some numerical examples and comparisons with other well known methods.

Throughout this paper, we use the following notations. For X and Y two $n \times s$ matrices, we consider the matrix inner product defined by $\langle X, Y\rangle_F = tr(X^T Y)$ where $tr(Z)$ denotes the trace of the square matrix Z and X^T is the transpose of the matrix X. The associated norm is the Frobenius norm denoted by $\| \ . \ \|_F$. The notation $\langle \ , \ \rangle_2$ is for the usual Euclidean inner product in $\mathbb{R}^n$.

2. A New Derivation of the Block BiCGSTAB Algorithm

2.1 *The block BCG method*

The block Bi-Conjugate Gradient (Bl-BCG) algorithm was first proposed by O'Leary [13] for solving the problem (1.1). This algorithm is a generalization of the one right-hand side well known BCG algorithm [5]. Bl-BCG computes two sets of direction matrices $\{P_0^b, \ldots, P_k^b\}$ and $\{\tilde{P}_0^b, \ldots, \tilde{P}_k^b\}$ whose columns form a basis of the block Krylov subspaces $\mathcal{K}_{k+1}(A, R_0^b)$ and $\mathcal{K}_{k+1}(A^T, \tilde{R}_0^b)$, respectively

$$\mathcal{K}_{k+1}(A, R_0) = \text{Range}(\{\tilde{P}_0^b, \ldots, \tilde{P}_k^b\}), \text{ and } \mathcal{K}_{k+1}(A^T, \tilde{R}_0^b) = (\text{Range}\{\tilde{P}_0^b, \ldots, \tilde{P}_k^b\}),$$

where $R_0^b = B - AX_0^b$ is the initial residual for a given starting block X_0^b and $\tilde{R}_0^b$ is an arbitrary $n \times s$ matrix. The following algorithm, computes at each iteration k a new approximate solution X_k^b of the solution of the problem (1.1) and the corresponding residual $R_k^b = B - AX_k^b$. The algorithm is summarized as follows [13]

Algorithm 1 The Block BCG algorithm

1. X_0 is an $n \times s$ initial guess, $R_0^b = B - AX_0^b$; $\tilde{R}_0^b$ is an arbitrary $n \times s$ matrix. $P_0^b = R_0^b$, $\tilde{P}_0^b = \tilde{R}_0^b$.
2. For $k = 1, 2, \ldots$ compute
   - $\alpha_k = ((\tilde{P}_{k-1}^b)^T AP_{k-1}^b)^{-1}(\tilde{R}_{k-1}^b)^T R_{k-1}^b$
   - $X_k^b = X_{k-1}^b + P_{k-1}^b \alpha_k$
   - $R_k^b = R_{k-1}^b - AP_{k-1}^b \alpha_k$
   - $\tilde{\alpha_k} = ((P_{k-1}^b)^T A^T \tilde{P}_{k-1}^b)^{-1}(R_{k-1}^b)^T \tilde{R}_{k-1}^b$
   - $\tilde{R}_k^b = \tilde{R}_{k-1}^b - A^T \tilde{P}_{k-1}^b \tilde{\alpha}_k$
   - $\beta_k = ((\tilde{R}_{k-1}^b)^T R_{k-1}^b)^{-1}(\tilde{R}_k^b)^T R_k^b$
   - $\tilde{\beta}_k = ((R_{k-1}^b)^T \tilde{R}_{k-1}^b)^{-1}(R_k^b)^T \tilde{R}_k^b$
   - $P_k^b = R_k^b + P_{k-1}^b \beta_k$
   - $\tilde{P}_k^b = \tilde{R}_k^b + \tilde{P}_{k-1}^b \tilde{\beta}_k$
3. end.

We have a breakdown in the preceding algorithm if one of the matrices $(\tilde{P}_{k-1}^b)^T AP_{k-1}^b$ or $(\tilde{R}_{k-1}^b)^T R_{k-1}^b$ are singular. Note also that the coefficient matrices computed in the algorithm are solution of $s \times s$ linear systems. The matrices of these systems could be very ill-conditioned and this would affect the iterates computed by Bl-BCG.

The Bl-BCG algorithm uses a short three-term recurrence but in many situations, the algorithm typically exhibits very irregular convergence behavior. This problem can be overcome by using a block smoothing technique as defined in [9] or a block QMR procedure [6]. A stabilized version of the Bl-BCG algorithm has been proposed in [17]; this method converges quite well but is generally more expensive than Bl-BCG.

The matrix residuals and the matrix directions generated by Algorithm 1 verify the following properties.

Proposition 2.1 *[13] If no break-down occur, the matrices computed by the Bl-BCG algorithm satisfy the following relations:*

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

2.2 *The block BiCGSTAB algorithm*

In [3], we defined the block BiCGSTAB algorithm by using right matrix orthogonal polynomials ; see [4] for some connections between block Lanczos-based methods and matrix orthogonal polynomials.

In this subsection, we show how to derive Bl-BiCGSTAB from some algebraic orthogonalities. This new derivation allows one to introduce a near breakdown procedure in order to stabilize the convergence of the Bl-BiCGSTAB algorithm. In the present paper, we assume that we have no breakdown or near breakdown problems. Let R_k^b and P_k^b denote the k-th residual and the k-th direction produced by Bl-BCG.

We define the k-th residual of Bl-BiCGSTAB by

$$R_k = (I - \omega_k A)\, S_k, \ \text{ for } \ k \geq 1 \tag{2.1}$$

where

$$S_k = (I - \omega_{k-1} A) \ldots (I - \omega_1 A)\, R_k^b \text{ for } k \geq 2 \text{ and } \ S_1 = R_1^b. \tag{2.2}$$

The parameter ω_k is chosen to minimize the F-norm of R_k so we get

$$\omega_k = \frac{\langle S_k, AS_k \rangle_F}{\langle AS_k, AS_k \rangle_F}.$$

The Bl-BCG residual R_k^b verifies the relation

$$R_k^b = R_{k-1}^b - AP_{k-1}^b\, \alpha_k \tag{2.3}$$

therefore, S_k can be written as

$$S_k = R_{k-1} - AP_{k-1}\, \alpha_k \tag{2.4}$$

where

$$P_{k-1} = (I - \omega_{k-1} A) \ldots (I - \omega_0 A)\, P_{k-1}^b \text{ for } k \geq 2, \tag{2.5}$$

and

$$P_0 = R_0 = R_0^b.$$

Now since R_k^b is orthogonal to the block Krylov subspace $\mathcal{K}_k(A^T, \tilde{R}_0)$, it follows from (2.2) that

$$\tilde{R}_0^T\, S_k = 0. \tag{2.6}$$

Using this orthogonality in (2.4), we get

$$\alpha_k = (\tilde{R}_0^T\, AP_{k-1})^{-1}\, \tilde{R}_0^T\, R_{k-1}.$$

On the other hand the Bl-BCG direction P_k^b satisfies

$$P_k^b = R_k^b + P_{k-1}^b\, \beta_k. \tag{2.7}$$

From (2.5) and (2.7) the Bl-BiCGSTAB direction P_k is given by

$$P_k = (I - \omega_k A) \ldots (I - \omega_1 A)\, [R_k^b + P_{k-1}^b\, \beta_k]$$

which is also written as

$$P_k = (I - \omega_k A)(S_k + P_{k-1}\, \beta_k). \tag{2.8}$$

Therefore

$$P_k = (I - \omega_k A)\, Q_k$$

where

$$\begin{aligned} Q_k &= S_k + P_{k-1}\,\beta_k \\ &= (I-\omega_k A)\ldots(I-\omega_1 A)\,P_k^b \end{aligned}$$

Now we have to compute the matrix coefficient β_k by using the fact that P_{k+1}^b is orthogonal to the block Krylov subspace $A^T\mathcal{K}_{k+1}(A^T,\tilde{R}_0)$. It follows that

$$\tilde{R}_0^T\,AQ_k = 0.$$

Then we obtain

$$\beta_k = -(\tilde{R}_0^T\,AP_k)^{-1}\,(\tilde{R}_0^T\,AS_k).$$

The Bl-BiCGSTAB algorithm is given as follows [4]

Algorithm 2 The Bl-BiCGSTAB algorithm

1. X_0 and $\tilde{R}_0$ arbitrary ; $P_0 = R_0 = B - A\,X_0$.
2. For $k = 1, 2, \ldots$
   - $V_{k-1} = AP_{k-1}$;
   - $\alpha_k = (\tilde{R}_0^T\,V_{k-1})^{-1}(\tilde{R}_0^T\,R_{k-1})$;
   - $S_k = R_{k-1} - V_{k-1}\alpha_k$;
   - $T_k = AS_k$;
   - $\omega_k = \langle T_k, S_k\rangle_F / \langle T_k, T_k\rangle_F$;
   - $X_k = X_{k-1} + P_{k-1}\alpha_k + \omega_k S_k$;
   - $R_k = S_k - \omega_k T_k$;
   - $\beta_k = -(\tilde{R}_0^T\,V_{k-1})^{-1}\,(\tilde{R}_0^T\,T_k)$;
   - $P_k = R_k + (P_{k-1} - \omega_k V_{k-1})\,\beta_k$;
3. end.

Remark 1

- *At step k, a breakdown will occur in Algorithm 2 if the matrix $(\tilde{R}_0^T\,AP_{k-1})$ is singular.*
- *If $s = 1$, Algorithm 2 reduces to the classical BiCGSTAB algorithm [21] defined by using recurrences of orthogonal polynomials. Note that the expression of the coefficient β_k computed by Algorithm 2 with $s = 1$ is more simple than the one given in [21].*
- *For Bl-BiCGSTAB, we have the following orthogonalities*

$$(\tilde{R}_0^T\,S_k) = 0 \ \textit{and}\ (\tilde{R}_0^T\,AQ_k) = 0.$$

3. The Seed BiCGSTAB Method

In this section, we present a new method for solving linear systems with multiple right-hand sides. This method consists in selecting one of the systems, called the *seed system*, to solve it by the BiCGSTAB algorithm and then construct the residuals of the others by using the informations obtained from the seed systems trary.

We note also that in practice, for example in wave scattering problems, in time marching methods for PDE's or in structural mechanic problems, the right-hand sides are not arbitrary; see for example [20]. They are usually discrete values of some function $b(t)$. In other words,

$$b^{(i)} = b(t_i), \quad i = 1, \ldots, s.$$

In these cases, the solutions $x^{(i)}$ are discrete values of some function $x(t)$. So if the $b^{(i)}$'s are close, we may also expect the $x^{(i)}$'s to be close [1]. In theses situations, 'near' linear dependence may arise among the right-hand sides. This makes the block methods ineffective and they may even suffer from a near breakdown problem. Special treatment is needed to handle this situation. However, the seed methods are very effective when the right-hand sides b^i's are close, it usually only takes few restarts to solve all the systems. Theoretical analysis has been given in [1] to explain this phenomena for the seed Conjugate Gradient method.

In the sequel, we note by l the index of the seed system. To solve it, we use the BiCGSTAB algorithm with another choice of the parameter ω. So, given $\tilde{r}_0 \in \mathbb{R}^N$, the seed iterates are defined by

$$r_k^{(l)} = s_k^{(l)} - \omega_k A s_k^{(l)},$$

where

$$s_k^{(l)} = r_{k-1}^{(l)} - \alpha_k^{(l)} A p_{k-1}^{(l)},$$

and

$$p_k^{(l)} = r_k^{(l)} + \beta_k^{(l)} (p_{k-1}^{(l)} - \omega_k A p_{k-1}^{(l)}).$$

Note that the residual $r_k^{(l)}$ and the direction $p_k^{(l)}$ of the seed system are given by

$$r_k^{(l)} = (I - \omega_k A) \ldots (I - \omega_1 A) r_k^{(l),b}, \tag{3.1}$$

and

$$p_k^{(l)} = (I - \omega_k A) \ldots (I - \omega_1 A) p_k^{(l),b}, \tag{3.2}$$

where $r_k^{(l),b}$ and $p_k^{(l),b}$ are the k-th residual and the the k-th direction produced by the one right hand side version of the BCG algorithm. Thus

$$r_k^{(l),b} \perp \mathcal{K}_k(A^T, \tilde{r}_0), \tag{3.3}$$

and

$$p_k^{(l),b} \perp A^T \mathcal{K}_k(A^T, \tilde{r}_0). \tag{3.4}$$

This leads to

$$\begin{aligned} \alpha_k^{(l)} &= \langle \tilde{r}_0, r_{k-1}^{(l)} \rangle_2 / \langle \tilde{r}_0, A p_{k-1}^{(l)} \rangle_2, \\ \beta_k^{(l)} &= -\langle \tilde{r}_0, A s_k^{(l)} \rangle_2 / \langle \tilde{r}_0, A p_{k-1}^{(l)} \rangle_2. \end{aligned}$$

The nonseed residual matrix will be constructed by

$$R_k = (I - \omega_k A) S_k, \tag{3.5}$$

where

$$S_k = (I - \omega_{k-1}A)\dots(I - \omega_1 A)\, R_k^b. \tag{3.6}$$

and R_k^b is orthogonal to the Krylov subspace $\mathcal{K}_k(A^T, \tilde{r}_0)$ i.e

$$R_k^b \perp \mathcal{K}_k(A^T, \tilde{r}_0). \tag{3.7}$$

A simple argument induction shows that R_k^b can be expressed as

$$R_k^b = R_{k-1}^b - Ap_{k-1}^{(l),b}\,\bar{\alpha}_k, \tag{3.8}$$

where $p_{k-1}^{(l),b}$ is the BCG direction of the seed system.
$\bar{\alpha}_k$ is determined by using the orthogonality (3.7), which leads to

$$\tilde{r}_0^T S_k = 0,$$

therefore

$$\bar{\alpha}_k = \frac{\tilde{r}_0^T\, R_{k-1}}{\langle \tilde{r}_0, Ap_{k-1}^{(l)}\rangle_2}.$$

From (3.5), (3.6) and (3.8) we obtain

$$\begin{aligned} R_k &= S_k - \omega_k AS_k, \\ S_k &= R_{k-1} - Ap_{k-1}^{(l)}\bar{\alpha}_k. \end{aligned}$$

Finally to minimize the F-norm of the seed and nonseed residuals $r_k^{(l)}$ and R_k, we choose ω_k as follows

$$\omega_k = \langle [s_k^{(l)}\ S_k], [As_k^{(l)}\ AS_k]\rangle_F / \langle [As_k^{(l)}\ AS_k], [As_k^{(l)}\ AS_k]\rangle_F.$$

Remark 2 *The expression of R_k^b in (3.8) can be replaced by*

$$R_k^b = R_{k-1}^b - Ap_{k-1}\,\tilde{\alpha}_k,$$

for any direction which verifies

$$p_{k-1} \perp A^T\mathcal{K}_k(A^T, \tilde{r}_0),$$

where $\tilde{\alpha}_k$ is determined from the orthogonality (3.7).

The result stated in Remark 2 is very important. It shows that, at any iteration, we can change the seed system by another among the nonseed ones. More precisely, suppose that we want to change the seed system at the m-th iteration and let denote by l' the index of the new seed system. We define the direction corresponding to the new seed system by

$$p_m^{(l')} = r_m^{(l')} + \beta_m^{(l')}(p_{m-1}^{(l)} - \omega_m Ap_{m-1}^{(l)}),$$

where

$$\beta_m^{(l')} = -\frac{\langle \tilde{r}_0, As_m^{(l')}\rangle_2}{\langle \tilde{r}_0, Ap_{m-1}^{(l)}\rangle_2}. \tag{3.9}$$

In fact, by construction, we have

$$r_m^{(l')} = (I - \omega_m A)\dots(I - \omega_1 A) r_m^{(l'),b},$$

and

$$p_{m-1}^{(l)} = (I - \omega_{m-1}A)\ldots(I - \omega_1 A)p_{m-1}^{(l),b}$$

such that

$$\begin{aligned} r_m^{(l')} &\perp \mathcal{K}_m(A^T, \tilde{r}_0), \\ p_{m-1}^{(l)} &\perp A^T\mathcal{K}_{m-1}(A^T, \tilde{r}_0). \end{aligned}$$

Thus we can express $p_m^{(l')}$ as follows

$$p_m^{(l')} = (I - \omega_m A)\ldots(I - \omega_1 A)p_m^{(l'),b},$$

where

$$p_m^{(l'),b} = r_m^{(l'),b} + \beta_m^{(l')}p_{m-1}^{(l),b}.$$

By remarking that the choice of $\beta_m^{(l')}$ in (3.9) ensures the orthogonality of $p_m^{(l'),b}$ and the Krylov subspace $\mathcal{K}_m(A^T, \tilde{r}_0)$ and using the result of Remark 2 we see that we can change the seed system (the seed direction also) and construct the residual matrix corresponding to the seed and non-seed systems while keeping the orthogonality conditions. Therefore, the relation (3.7) ensures the convergence in at most N iteration.

In practice, we change the seed system only when breakdown occurs or when the seed system has converged. The seed BiCGSTAB algorithm is summarized as follows

Remark 3

- *We notice that, in our numerical tests, the seed systems were chosen arbitrary. Remark also that the seed BiCGSTAB algorithm requires extra matrix-vector products which is not the case for other well known seed methods. But since the orthogonolaties of the non-seed systems are conserved at each restart, we may expect faster convergence in fewer iterations.*
- *It is also possible to combine the block BiCGSTAB and the seed approach to define a new seed-block BiCGSRTAB. In this case, instead of using one seed vector we can take a block part of the right hand side B as a seed block.*
- *A convergence analysis such as the one given in [1] for the seed CG is more difficult. This is due to the fact that we haven't theoretical results on the convergence of the one right-hand side BiCGSTAB algorithm. This convergence analysis is still under investigation.*

4. Numerical Experiments

This section reports on some experimental results obtained by applying the block and seed BiCGSTAB algorithms to linear systems with different right-hand sides. All the experiments were performed on a 1.3GHz Intel Core i5 laptop with 8Gb of RAM and coded with Matlab R2010a.

The initial guess was $X_0 = 0$. In all the experiments, the tests were stopped as soon as

$$\frac{\max_{i=1:s}\|R_k(:,i)\|_2}{\max_{i=1:s}\|R_0(:,i)\|_2} \leq 10^{-6}.$$

The matrix tests that we used are from the Harwell-Boeing collection and are listed in the following table

Example 1 . In this example, we compared the performance of Bl-BiCGSTAB and Bl-QMR. Figure 1 reports on convergence history for the three methods with the matrix test $A = A_1$. The $n \times s$ right hand side matrix B

Algorithm 3 The seed BiCGSTAB algorithm

1. Choose the seed system of index l; $x_0^{(l)}$ and $\tilde{r}_0$ arbitrary.
2. Set $p_0 = r_0 = b^{(l)} - Ax_0^{(l)}$.
3. Construct the nonseed matrix residual R_0 from an arbitrary X_0.
4. For $k = 0, 1, \ldots$
   - $\alpha_k^{(l)} = \langle \tilde{r}_0, r_{k-1}^{(l)} \rangle_2 / \langle \tilde{r}_0, Ap_{k-1}^{(l)} \rangle_2$;
   - $\bar{\alpha}_k = \tilde{r}_0^T R_{k-1} / \langle \tilde{r}_0, Ap_{k-1}^{(l)} \rangle_2$;
   - $s_k^{(l)} = r_{k-1}^{(l)} - \alpha_k^{(l)} Ap_{k-1}^{(l)}$;
   - $S_k = R_{k-1} - Ap_{k-1}^{(l)} \bar{\alpha}_k$;l
   - $\omega_k = \langle [s_k^{(l)}\ S_k], [As_k^{(l)}\ AS_k] \rangle_F / \langle [As_k^{(l)}\ AS_k], [As_k^{(l)}\ AS_k] \rangle_F$;
   - $r_k^{(l)} = s_k^{(l)} - \omega_k As_k^{(l)}$;
   - $R_k = S_k - \omega_k AS_k$;
   - If convergence of the seed system, take another seed system of index l';
   - $\beta_k^{(l')} = -\langle \tilde{r}_0, As_k^{(l')} \rangle_2 / l \langle \tilde{r}_0, Ap_{k-1}^{(l)} \rangle_2$;
   - $p_k^{(l')} = r_k^{(l')} + \beta_k^{(l')} (p_{k-1}^{(l)} - \omega_k Ap_{k-1}^{(l)})$;
   - $l = l'$;
   - Else
   - $\beta_k^{(l)} = -\langle \tilde{r}_0, As_k^{(l)} \rangle_2 / \langle \tilde{r}_0, Ap_{k-1}^{(l)} \rangle_2$;
   - $p_k^{(l)} = r_k^{(l)} + \beta_k^{(l)} (p_{k-1}^{(l)} - \omega_k Ap_{k-1}^{(l)})$.
5. End.

Table 1: The matrix tests

Matrix	size	number of nonzero elements
A_1=`Sherman1`	$n = 1000$	$nnz(A_1) = 3750$
A_2=`Saylr4`	$n = 3564$	$nnz(A_2) = 22316$
A_3=`add32`	$n = 4960$	$nnz(A_3) = 19848$
A_4=`Sherman5`	$n = 3312$	$nnz(A_4) = 20793$
A_5= `jpwh991`	$n = 991$	$nnz(A_5) = 6027$

was a random matrix with uniformly distributed entries. In Figure 1, we plotted the maximum residual norms versus the flops (number of arithmetic operations). As observed from Figure 1, Bl-BiCGSTAB returns the best

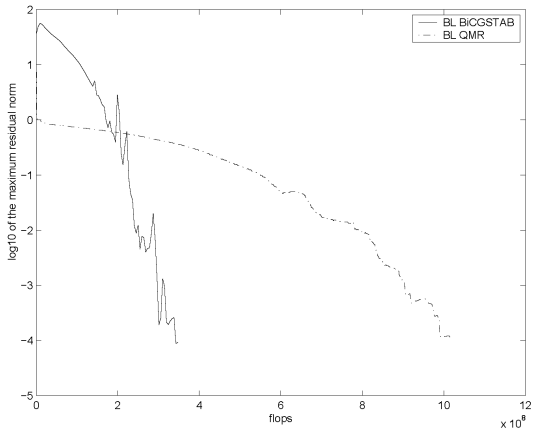

Figure 1: $A = A_1$; $s = 10$

results.

Example 2. For this example, we used the matrix test A_2= `Saylr4`. We also used ILUT as a preconditioner with a dropping tolerance $\tau = 10^{-4}$. The Right-hand side matrix B was the one considered in Example 1. We compared the performance (in term of cpu-times) of the Bl-BiCGSTAB algorithm, the Bl-GMRES(10) algorithm and the BiCGSTAB algorithm applied to each single right-hand side. Two different numbers of right-hand sides were used $s = 5$ and $s = 10$. In Table 2 we listed the following time-ratios:

$$T_1 = \frac{\text{time(Bl} - \text{BiCGSTAB)}}{(\text{time(BiCGSTAB))}}; \quad T_2 = \frac{\text{time(Bl} - \text{BiCGSTAB)}}{\text{time(Bl} - \text{GMRES(10))}}.$$

As shown in Table 2, Bl-BiCGSTAB is less expensive than Bl-GMRES(10) and than BiCGSTAB applied to each

Table 2: Results for the matrix `Saylr4`

	T_1	T_2
$s = 5$	0.77	0.86
$s = 10$	0. 82	0.90

single right-hand side.

Example 3 In this example, we tested the effectiveness of the seed BiCGSTAB algorithm for solving multiple linear systems with closely related right-hand sides.

The matrix test used for this example was $A_3 =$ `add32`. The right-hand side matrix B is the $n \times s$ matrix whose elements are

$$B(i,j) = sin(\frac{2\pi}{N}(i + j - 2)).$$

So the i-th column $b^{(i)}$ of the matrix B is obtained by shifting the components of the column $b^{(i-1)}$ by one position and the first component is replaced by the last one.

Figure 2 reports on the convergence history of the seed BiCGSTAB. In this figure we plotted the residual norms, in the logarithmic scale, for the s linear systems versus the iterations.

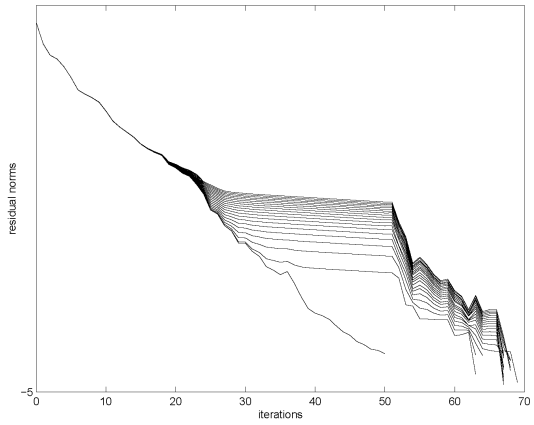

Figure 2: A=add32 ; $s = 20$

We notice that three restarts were necessary to achieve the convergence of the s linear systems. We also notice that no strategy was used to choose the seed linear system.

In Table 3 we listed the number of matrix-vector products required for convergence by the seed BiCGSTAB algorithm and the BiCGSTAB algorithm when applied to each linear system. We used different values of s; $s = 10$, $s = 20$ and $s = 30$.

Table 3: Results for experiment 1 with the matrix add32.

	BiCGSTAB applied at each right-hand side	seed BiCGSTAB
$s = 10$	890	731
$s = 20$	1806	1387
$s = 30$	2684	2299

Example 4 For this example we compared seed BiCGSTAB with BiCGSTAB applied to each linear system using ILUT as a preconditionner (with a dropping tolerance of $\tau = 10^{-5}$). The matrix test was $A_4 =$ Sherman5. Different values of s were used; $s = 10$, $s = 30$ and $s = 50$. The right-hand side matrix B was the same as the one given in Example 3. For this experiment, only one restart was necessary to achieve the convergence for the $s$linear systems. In Table 4, we reported matrix-vector products required for the seed BiCGSTAB and the BiCGSTAB algoritms.

5. Conclusion

In this note, we first showed how to derive the block BiCGSTAB algorithm without using matrix orthogonal polynomials. This technique, allows one to detect and to cure a possible breakdown in this algorithm. In the second part of the paper, we presented a new seed BiCGSRAB method for solving linear systems with multiple right-hand sides and the same coefficient matrix. The new method is especially efficient when the right-hand sides are closesly related and this can be seen from some numerical examples.

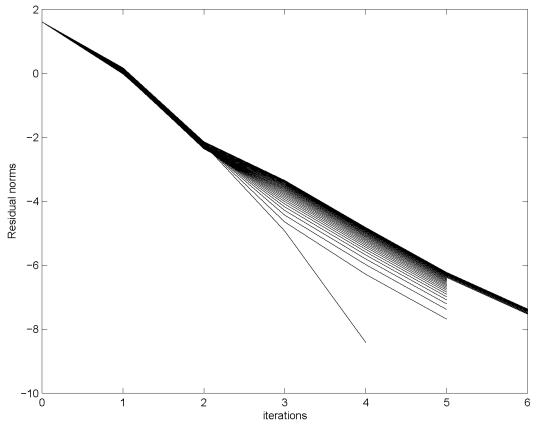

Figure 3: $A = A_4$; $s = 30$

Table 4: Results for experiment 2 with the matrix `Sherman5`.

	BiCGSTAB applied at each right-hand side	seed BiCGSTAB
$s = 10$	80	54
$s = 30$	240	165
$s = 50$	400	289

References

[1] Chan, T. and Wan, W. (1997). Analysis of Projection Methods for Solving Linear Systems with Multiple Right-hand Sides, SIAM J. Sci. Comput., 18, 1698-1721.

[2] Datta, B. N. and Datta, K. (1986). Theoretical and computational aspects of some linear algebra problems in control theory, in Computational and Combinatorial Methods in Systems Theory, C. I. Byrnes and A. Lindquist, eds Elsevier, Amsterdam, 1986, 201-212.

[3] El Guennouni,A., Jbilou, K. and Sadok, H. (2004). The block Lanczos method for multiple linear systems, Appl. Num. Math., 51, 243-256.

[4] El Guennouni, A., Jbilou, K. and Sadok, H. (2003). Block BICGSTAB for linear systems with multiple right sides, Elect. Trans. Numer. Anal., 16, 129-142.

[5] Fletcher, R. (1975). Conjugate gradient methods for indefinite systems, In G. A. Watson, editor, Proceedings of the Dundee Biennal Conference on Numerical Analysis 1974, pages 73-89. Springer Verlag, New York.

[6] Freund, R. and Malhotra. M. (1997). A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides. Lin. Alg. Appl., 254(1-3)(1997), pp. 119--157.

[7] Jbilou, K.Messaoudi, A. and Sadok, H. (1999). Global FOM and GMRES algorithms for matrix equations, Appl. Numer. Math., 31, 49-63.

[8] Jbilou, K., Sado, H. and Tinzeft, A. (2005) Oblique projection methods for multiple linear systems, Elect. Trans. Num. Anal., 20, 119-138.

[9] Jbilou, K. (1999). Smoothing iterative block methods for linear systems with multiple right-hand sides, J. Comput. Appl. Math., 107, 97-109.

[10] Malhotra, M., Freund, R. and Pinsky, P. M. (1997). Iterative Solution of Multiple Radiation and Scattering Problems in Structural Acoustics Using a Block Quasi-Minimal Residual Algorithm, Comp. Meth. Appl. Mech. Eng., 146, 173--196.

[11] H. Daib, J. Zhaoa and J. Zhanga, A new family of global methods for linear systems with multiple right-hand sides, J. Comput. Appl. Math., 236(6)(2011) pp.1562-1575.

[12] Nikishin, A. and Yeremin, A. (1995). Variable Block CG Algorithms for Solving Large Sparse Symmetric Positive Definite Linear Systems on Parallel Computers, I: General Iterative Scheme, SIAM J. Matrix Anal. Appl., 16, 1135-1153.

[13] O'Leary, D. (1980). The block conjugate gradient algorithm and related methods, Linear Algebra Appl., 29, 293-322.

[14] B. Parlett, *A New Look at the Lanczos Algorithm for Solving Symmetric Systems of Linear Equations*, Linear Algebra Appl., 29(1980), pp. 323-346.

[15] Saad,Y.(1987). On the Lanczos Method for Solving Symmetric Linear Systems with Several Right-Hand Sides, Math. Comp., 48, 651-662.

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

[17] Simoncini, V. (1997). A stabilized QMR version of block BICG, SIAM J. Matrix Anal. Appl., 18(2), 419-434.

[18] Simoncini, V. and Gallopoulos, E. (1996). Convergence properties of block GMRES and matrix polynomials, Linear Algebra Appl., 247, 97-119.

[19] Simoncini, V. and Gallopoulos, E. (1995). *An Iterative Method for Nonsymmetric Systems with Multiple Right-hand Sides*, SIAM J. Sci. Comp., 16, 917-933.

[20] Smith, C., Peterson, A. and Mittra, R. (1989). A Conjugate Gradient Algorithm for Tretement of Multiple Incident Electromagnetic Fields, IEEE Transactions On Antennas and Propagation, 37, 1490-1493.

[21] Van Der Vorst, H. (1992). Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., 13, 631-644.

[22] Van Der Vorst, H. (1987). An Iterative Solution Method for Solving $f(A) = b$, using Krylov Subspace Information Obtained for the Symmetric Positive Definite Matrix A, J. Comp. Appl. Math., 18, 249-263.