Electronic Transactions on Numerical Analysis.
Volume 16, pp. 129-142, 2003.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

A BLOCK VERSION OF BICGSTAB FOR LINEAR SYSTEMS WITH MULTIPLE RIGHT-HAND SIDES*

A. EL GUENNOUNI†, K. JBILOU ‡, AND H. SADOK †

Abstract. We present a new block method for solving large nonsymmetric linear systems of equations with multiple right-hand sides. We first give the matrix polynomial interpretation of the classical block biconjugate gradient (Bl-BCG) algorithm using formal matrix-valued orthogonal polynomials. This allows us to derive a block version of BiCGSTAB. Numerical examples and comparisons with other block methods are given to illustrate the effectiveness of the proposed method.

Key words. block Krylov subspace, block methods, Lanczos method, multiple right-hand sides, nonsymmetric linear systems.

AMS subject classifications. 65F10.

1. Introduction. Many applications such as in electromagnetic scattering problem and in structural mechanics problems require the solution of several linear systems of equations with the same coefficient matrix and different right-hand sides. This problem can be written as

$$A\,X = B, \tag{1.1}$$

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, where s is of moderate size with $s \ll N$.

When N is small, one can use direct methods to solve the s given linear systems. We can compute the LU decomposition of A at a cost of $O(N^3)$ operations and then solve the system for each right-hand at a cost of $O(N^2)$ operations. However, for large N, direct methods may become very expensive. Instead of solving each of the s linear systems by some iterative method, it is more efficient to use a block version and generate iterates for all the systems simultaneously.

Starting from an initial guess $X_0 \in \mathbb{R}^{N\times s}$, block Krylov subspace methods determine, at step k, an approximation of the form $X_k = X_0 + Z_k$, where Z_k belongs to the block Krylov subspace $\mathcal{K}_k(A, R_0) = span\{R_0, AR_0, \ldots, A^{k-1}R_0\}$ with $R_0 = B - AX_0$. We use a minimization property or an orthogonality relation to determine the correction Z_k.

For symmetric and positive definite problems, the block conjugate gradient (Bl-CG) algorithm [15] and its variants [14] are useful for solving the linear system (1.1).

When the matrix A is nonsymmetric, the block biconjugate gradient (Bl-BCG) algorithm [15] (see [20] for a stabilized version), the block generalized minimal residual (Bl-GMRES) algorithm introduced in [24] and studied in [21], and the block quasi minimum residual (Bl-QMR) algorithm [7] (see also [13]) are the best known block Krylov subspace methods. The purpose of these block methods is to provide the solutions of a multiple right-hand sides system faster than their single right-hand side counterparts. We note that block solvers are

*Received April 12, 2001. Accepted for publication October 28, 2002. Recommended by M. Gutknecht.
†Laboratoire d'Analyse Numérique et d'Optimisation, Université des Sciences et Technologies de Lille, France. Email: elguenn@ano.univ-lille1.fr
‡Université du Littoral, zone universitaire de la Mi-voix, batiment H. Poincaré, 50 rue F. Buisson, BP 699, F-62228 Calais Cedex, France. Email: jbilou@lmpa.univ-littoral.fr and sadok@calais.univ-littoral.fr

ETNA
Kent State University
etna@mcs.kent.edu

effective compared to their single right-hand versions when the matrix A is "relatively dense". They are also attractive when a preconditioner is added to the block solver.

In [9] and [10] we proposed new methods called global GMRES and global Lanczos-based methods. These methods are obtained by projecting the initial block residual onto a matrix Krylov subspace.

Another approach for solving the problem (1.1) developed in the last few years consists in selecting one seed system and the corresponding Krylov subspace and projecting the residuals of the other systems onto this Krylov subspace. The process is repeated with an other seed system until all the systems are solved. This procedure has been used in [23] and [4] for the conjugate gradient method and in [22], when the matrix A is nonsymmetric for the GMRES algorithm [18]. This approach is also effective when the right-hand sides of (1.1) are not available at the same time (see [16], [17] and [26]). Note also that block methods such as the block Lanczos method [8], [5] and the block Arnoldi method [19] are used for solving large eigenvalue problems.

The Bl-BCG algorithm uses a short three-term recurrence formula, but in many situations the algorithm exhibits a very irregular convergence behavior. This problem can be overcome by using a block smoothing technique as defined in [11] or a block QMR procedure [7]. A stabilized version of the Bl-BCG algorithm has been proposed in [20]. This method converges quite well but is in general more expensive than Bl-BCG.

As for single right-hand side Lanczos-based methods, it cannot be excluded that break-downs occur in the block Lanczos-based methods.

We note that some block methods include a deflation procedure in order to detect and delete linearly or almost linearly dependent vectors in the block Krylov subspaces generated during the iterations. This technique has been used in [7] for the Bl-QMR algorithm. See also [14] for a detailed discussion on loss of rank in the iteration matrices for the block conjugate gradient algorithm.

In the present paper, we define a new transpose-free block algorithm which is a generalization of the well-known single right-hand side BiCGSTAB algorithm [25]. This new method is named block BiCGSTAB (Bl-BiCGSTAB). Numerical examples show that the new method is more efficient than the Bl-BCG, Bl-QMR and Bl-GMRES algorithms.

The remainder of the paper is organized as follows. In Section 2, we give a matrix polynomial interpretation of Bl-BCG using right matrix orthogonal polynomials. In Section 3, we define the Bl-BiCGSTAB algorithm for solving (1.1). Finally, we report in Section 4 some numerical examples.

Throughout this paper, we use the following notations. For two $N \times s$ matrices X and Y, we consider the following inner product: $\langle X, Y\rangle_F = tr(X^T Y)$, where $tr(Z)$ denotes the trace of the square matrix Z. The associated norm is the Frobenius norm denoted by $\|\ .\ \|_F$. We will use the notation $\langle\ ,\ \rangle_2$ for the usual inner product in $\mathbb{R}^N$. For a matrix $V \in \mathbb{R}^{N\times s}$, the block Krylov subspace $\mathcal{K}_k(A, V)$ is the subspace generated by the columns of the matrices $V, AV, \ldots, A^{k-1}V$. Finally, O_s and I_s will denote the zero and the identity matrices in $\mathbb{R}^{s\times s}$.

2. Matrix polynomial interpretation of the block BCG algorithm.

2.1. The block BCG algorithm. The block biconjugate gradient (Bl-BCG) algorithm was first proposed by O'Leary [15] for solving the problem (1.1). This algorithm is a generalization of the well-known BCG algorithm [6]. Bl-BCG computes two sets of direction matrices $\{P_0, \ldots, P_k\}$ and $\{\tilde{P}_0, \ldots, \tilde{P}_k\}$ that span the block Krylov subspaces $\mathcal{K}_{k+1}(A, R_0)$ and $\mathcal{K}_{k+1}(A^T, \tilde{R}_0)$, where $R_0 = B - AX_0$ and $\tilde{R}_0$ is an arbitrary $N \times s$ matrix. The algorithm can be summarized as follows [15]:

ALGORITHM 1: **Block BCG**

X_0 is an $N \times s$ initial guess, $R_0 = B - AX_0$.
$\tilde{R}_0$ is an arbitrary $N \times s$ matrix.
$P_0 = R_0$, $\tilde{P}_0 = \tilde{R}_0$.
For $k = 0, 1, \ldots$ compute
$\alpha_k = (\tilde{P}_k^T\, AP_k)^{-1}\tilde{R}_k^T\, R_k$
$X_{k+1} = X_k + P_k\alpha_k$
$R_{k+1} = R_k - AP_k\alpha_k$
$\tilde{\alpha_k} = (P_k^T\, A^T\tilde{P}_k)^{-1}R_k^T\, \tilde{R}_k$
$\tilde{R}_{k+1} = \tilde{R}_k - A^T\tilde{P}_k\tilde{\alpha}_k$
$\beta_k = (\tilde{R}_k^T R_k)^{-1}\tilde{R}_{k+1}^T R_{k+1}$
$\tilde{\beta}_k = (R_k^T\tilde{R}_k)^{-1}R_{k+1}^T\tilde{R}_{k+1}$
$P_{k+1} = R_{k+1} + P_k\beta_k$
$\tilde{P}_{k+1} = \tilde{R}_{k+1} + \tilde{P}_k\tilde{\beta}_k$
end.

The algorithm breaks down if the matrices $\tilde{P}_k^T AP_k$ or $\tilde{R}_k^T R_k$ are singular. Note also that the coefficient matrices computed in the algorithm are solutions 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.

Bl-BCG can also exhibits very irregular behavior of the residual norms. To remedy this problem one can use a block smoothing technique [11] or a block quasi-minimization procedure (see [7] and [20]).

The matrix residuals and the matrix directions generated by ALGORITHM 1 satisfy the following properties.

PROPOSITION 1. *[15] If no breakdown occurs, the matrices computed by the Bl-BCG algorithm satisfy the following relations*

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

From now on, we assume that no breakdown occurs in the Bl-BCG algorithm.
In the following subsection, we use matrix-valued polynomials to give a new expression of the iterates computed by Bl-BCG. This will allow us to define the block BiCGSTAB algorithm.

2.2. Connection with matrix-valued polynomials. In what follows, a matrix-valued polynomial $\mathcal{P}$ of degree k is a polynomial of the form

$$\mathcal{P}(t) = \sum_{i=0}^{k} t^i\,\Omega_i, \tag{2.1}$$

where $\Omega_i \in \mathbb{R}^{s\times s}$ and $t \in \mathbb{R}$.

We use the notation $\circ$ (used in [21]) for the product

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

where Y is an $N \times s$ matrix, and we define, for any $s \times s$ matrix Θ, the matrix polynomial $\mathcal{P}\,\Theta$ by

$$(\mathcal{P}\Theta)(t) = \sum_{i=0}^{k} t^i\, \Omega_i \Theta. \tag{2.3}$$

With these definitions, we have the following properties:

PROPOSITION 2. *Let $\mathcal{P}$ and $\mathcal{Q}$ be two matrix-valued polynomials and let Y and Θ be two matrices of dimensions $N \times s$ and $s \times s$, respectively. Then we have*

(1) $(\mathcal{P}(A) \circ Y)\Theta = (\mathcal{P}\Theta)(A) \circ Y$,

(2) $(\mathcal{P} + \mathcal{Q})(A) \circ Y = \mathcal{P}(A) \circ Y + \mathcal{Q}(A) \circ Y$.

Proof. (1) Let $\mathcal{P}$ be the matrix polynomial of degree k defined by (2.1). Then using (2.2) and (2.3) it follows that

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

The relation (2) is easy to verify. □

When solving a single right-hand side linear systems $Ax = b$, it is well-known that the residual r_k and the direction p_k computed by the BCG algorithm can be expressed as $r_k = \phi_k(A) r_0$ and $p_k = \psi_k(A) r_0$. These polynomials are related by some recurrence formulas [1].

Using matrix-valued polynomials we give in the following proposition new expressions for the residuals and the matrix directions generated by Bl-BCG.

PROPOSITION 3. *Let (R_k) and (P_k) be the sequences of matrices generated by the Bl-BCG algorithm. Then there exist two families of matrix-valued polynomials $(\mathcal{R}_k)$ and $(\mathcal{P}_k)$ of degree at most k such that*

$$R_k = \mathcal{R}_k(A) \circ R_0, \tag{2.4}$$

and

$$P_k = \mathcal{P}_k(A) \circ R_0. \tag{2.5}$$

These matrix polynomials are also related by the recurrence formulas

$$\mathcal{R}_{k+1}(t) = \mathcal{R}_k(t) - t\mathcal{P}_k(t)\alpha_k, \tag{2.6}$$

and

$$\mathcal{P}_{k+1} = \mathcal{R}_{k+1}(t) + \mathcal{P}_k(t)\beta_k, \tag{2.7}$$

with $\mathcal{P}_0(t) = \mathcal{R}_0(t) = I_s$ for $t \in \mathbb{R}$.

Proof. The case $k = 0$ is obvious. Assume that the relations hold for k. The residual R_{k+1} computed by the Bl-BCG algorithm is given by

$$R_{k+1} = R_k - AP_k\alpha_k.$$

Hence by the induction hypothesis, we get

$$R_{k+1} = \mathcal{R}_k(A) \circ R_0 - A(\mathcal{P}_k(A) \circ R_0)\alpha_k.$$

ETNA
Kent State University
etna@mcs.kent.edu

Then using Proposition 2, we obtain

$$\begin{aligned} R_{k+1} &= \mathcal{R}_k(A) \circ R_0 - (t\mathcal{P}_k\alpha_k)(A) \circ R_0 \\ &= [\mathcal{R}_k - t\mathcal{P}_k\alpha_k](A) \circ R_0 \\ &= \mathcal{R}_{k+1}(A) \circ R_0, \end{aligned}$$

where $\mathcal{R}_{k+1}(t) = \mathcal{R}_k(t) - t\mathcal{P}_k(t)\alpha_k$. This proves the first part of the proposition.

The matrix direction P_{k+1} computed by Bl-BCG is given as

$$P_{k+1} = R_{k+1} + P_k\beta_k,$$

hence using (2.4) and the induction hypothesis, it follows that

$$\begin{aligned} P_{k+1} &= [\mathcal{R}_{k+1}(A) + (\mathcal{P}_k\beta_k)(A)] \circ R_0 \\ &= \mathcal{P}_{k+1}(A) \circ R_0, \end{aligned}$$

where the matrix-valued polynomial $\mathcal{P}_{k+1}$ is defined by

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

□

Let $\mathcal{C}$ be the functional defined on the set of matrix-valued polynomials with coefficients in $\mathbb{R}^{s\times s}$ and given by

$$\mathcal{C}(\mathcal{P}) = \tilde{R}_0^T\,(\mathcal{P}(A) \circ R_0), \tag{2.8}$$

where $\mathcal{P}$ is a matrix-valued polynomial.

We also define the functional $\mathcal{C}^{(1)}$ by

$$\mathcal{C}^{(1)}(\mathcal{P}) = \mathcal{C}(t\mathcal{P}). \tag{2.9}$$

With these definitions, it is easy to prove the following relations.

PROPOSITION 4. *The functional $\mathcal{C}$ defined above satisfies the following properties:*

(1) $\mathcal{C}(\mathcal{P} + \mathcal{Q}) = \mathcal{C}(\mathcal{P}) + \mathcal{C}(\mathcal{Q})$.
(2) $\mathcal{C}(\mathcal{P}\,\Omega) = \mathcal{C}(\mathcal{P})\,\Omega$ *if* $\Omega \in \mathbb{R}^{s\times s}$.

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

This result shows that the functionals $\mathcal{C}$ and $\mathcal{C}^{(1)}$ are linear. The matrix polynomials $\mathcal{R}_k$ and $\mathcal{P}_k$ associated by (2.4) and (2.5) with the residual and the direction polynomials generated by BCG belong to families of formal orthogonal polynomials with respect to $\mathcal{C}$ and $\mathcal{C}^{(1)}$ (see [1]). These results are generalized in the next proposition.

PROPOSITION 5. *Let $(\mathcal{R}_k)$ and $(\mathcal{P}_k)$, $k \geq 1$, be the sequences of matrix-valued polynomials defined in Proposition 3. If $\mathcal{T}_i$ is an arbitrary matrix-valued polynomial of degree i, $i = 0, 1, \ldots, k-1$, then we have the following orthogonality properties*

$$\mathcal{C}(\mathcal{R}_k\mathcal{T}_i) = O \quad \text{for} \quad i < k, \tag{2.10}$$

and

$$\mathcal{C}^{(1)}(\mathcal{P}_k\mathcal{T}_i) = O \quad \text{for} \quad i < k. \tag{2.11}$$

ETNA
Kent State University
etna@mcs.kent.edu

Proof. We have seen (Proposition 1) that at step k, the columns of the residual R_k produced by Bl-BCG are orthogonal to the block Krylov subspace $\mathcal{K}_k(A^T, \tilde{R}_0)$. This can be expressed as

$$\tilde{R}_0^T A^i R_k = O_s, \quad \text{for} \quad i = 0, \ldots, k-1. \tag{2.12}$$

Using (2.4) of Proposition 3, it follows that

$$\tilde{R}_0^T (A^i \mathcal{R}_k(A)) \circ R_0 = O_s, \quad \text{for} \quad i = 0, \ldots, k-1.$$

This shows that

$$\mathcal{C}(t^i \mathcal{R}_k) = 0, \;\; i = 0, \ldots, k-1.$$

By the linearity of $\mathcal{C}$ (expressed in Proposition 4) we can conclude that if $\mathcal{T}_i$ is a matrix-valued polynomial of degree i ($i = 0, \ldots, k-1$), we have

$$\mathcal{C}(\mathcal{R}_k \mathcal{T}_i) = O_s, \quad \text{for} \quad i = 0, \ldots, k-1.$$

To prove (2.11), we use the relation

$$AP_k = (R_k - R_{k+1})\alpha_k^{-1}$$

to get

$$\tilde{R}_0^T A^{i+1} P_k = \tilde{R}_0^T A^i (R_k - R_{k+1})\alpha_k^{-1},$$

and using (2.12), we obtain

$$\tilde{R}_0^T A^{i+1} P_k = O_s \quad \text{for} \quad i = 0, \ldots, k-1.$$

Since $P_k = \mathcal{P}_k(A) \circ R_0$, it follows that

$$\tilde{R}_0^T A^{i+1} (\mathcal{P}_k(A) \circ R_0) = O_s, \;\; i = 0, \ldots, k-1,$$

therefore

$$\mathcal{C}^{(1)}(t^i \mathcal{P}_k) = 0, \;\; i = 0, \ldots, k-1,$$

and thus

$$\mathcal{C}^{(1)}(\mathcal{P}_k \mathcal{T}_i) = O_s \quad \text{for} \quad i = 0, \ldots, k-1.$$

□

The results of Proposition 5 show that $\mathcal{R}_k$ and $\mathcal{P}_k$ are the matrix-valued polynomials of degree at most k belonging to the families of matrix-valued orthogonal polynomials with respect to $\mathcal{C}$ and $\mathcal{C}^{(1)}$ respectively and normalized by the conditions $\mathcal{R}_k(0) = I_s$ and $\mathcal{P}_k(0) = I_s$. Note that if U_i is a scalar polynomial we also have $\mathcal{C}(U_i \mathcal{P}_k) = O_s$ and $\mathcal{C}^{(1)}(U_i \mathcal{P}_k) = O_s$ for $i = 0, \ldots, k-1$. In general, this is not true in the block case.

Using these matrix-valued polynomials, we will see in the next subsection how to define the block BiCGSTAB algorithm.

3. The block BiCGSTAB algorithm. The single right-hand side BiCGSTAB algorithm [25] is a transpose-free and a smoother variant of the BCG algorithm. The residuals produced by BiCGSTAB are defined by

$$r_k = Q_k(A)\phi_k(A)r_0,$$

where ϕ_k is the scalar residual polynomial associated with the BCG algorithm and Q_k is another polynomial of degree k updated from step to step by multiplication with a new linear factor with the goal of stabilizing the convergence behavior of the BCG algorithm:

$$Q_{k+1}(t) = (1-\omega_k t)Q_k(t).$$

The selected parameter ω_{k+1} is determined by a local residual minimization condition.

The search direction p_k is defined by

$$p_k = Q_k(A)\psi_k(A)r_0,$$

where ψ_k is the search direction polynomial associated with the BCG algorithm.

In detail the BiCGSTAB algorithm for solving a single right-hand side linear system $Ax = b$ is defined as follows [25]:

ALGORITHM 2: **BiCGSTAB**

x_0 an initial guess; $r_0 = b - Ax_0$; $p_0 = r_0$;
$\tilde{r}_0$ is an arbitrary vector satisfying $\langle \tilde{r}_0, r_0\rangle_2 \neq 0$;
for $k = 0, 1, 2, \ldots$
  $v_k = Ap_k$;
  $\alpha_k = \langle \tilde{r}_0, r_k\rangle_2 / \langle \tilde{r}_0, v_k\rangle_2$;
  $s_k = r_k - \alpha_k v_k$;
  $t_k = As_k$;
  $\omega_k = \langle t_k, s_k\rangle_2 / \langle t_k, t_k\rangle_2$;
  $x_{k+1} = x_k + \alpha_k p_k + \omega_k s_k$;
  $r_{k+1} = s_k - \omega_k t_k$;
  $\beta_k = \dfrac{\langle \tilde{r}_0, r_{k+1}\rangle_2}{\langle \tilde{r}_0, r_k\rangle_2}\,\dfrac{\alpha_k}{\omega_k}$;
  $p_{k+1} = r_{k+1} + \beta_k(p_k - \omega_k v_k)$;
end.

We will see later that the coefficient β_k used in ALGORITHM 2 can be computed by a simpler formula. Techniques for curing breakdowns in BiCGSTAB are given in [3].

We define now a new block method for solving (1.1), which is a transpose-free and a smoother converging variant of the Bl-BCG algorithm. The new method is named block BiCGSTAB (Bl-BiCGSTAB) since it is a generalization of the single right-hand side BiCGSTAB algorithm.

We have seen that the residual R_k^{Bl-BCG} and the matrix direction P_k^{Bl-BCG} computed by the Bl-BCG algorithm are such that

$$R_k^{Bl-BCG} = \mathcal{R}_k(A) \circ R_0,$$

ETNA
Kent State University
etna@mcs.kent.edu

and

$$P_k^{Bl-BCG} = \mathcal{P}_k(A) \circ R_0,$$

where $\mathcal{R}_k$ and $\mathcal{P}_k$ are the matrix-valued polynomials satisfying the relations (2.6) and (2.7).

The Bl-BiCGSTAB algorithm produces iterates whose residual matrices are of the form

$$R_k = (\mathcal{Q}_k \mathcal{R}_k)(A) \circ R_0, \tag{3.1}$$

where $\mathcal{Q}_k$ is still a scalar polynomial defined recursively at each step to stabilize the convergence behavior of the original Bl-BCG algorithm. Specifically, $\mathcal{Q}_k$ is defined by the recurrence formula

$$\mathcal{Q}_{k+1}(t) = (1 - \omega_k t)\mathcal{Q}_k(t). \tag{3.2}$$

As will be seen later, the scalar ω_k is selected by imposing a minimization of the Frobenius residual norm. It is also possible to choose ω_k different for each right-hand side. In all our numerical tests, we obtained similar results for these two cases.

We want now to define a recurrence formula for the new residuals defined by (3.1). From (2.6), we immediately obtain

$$\mathcal{Q}_{k+1}\mathcal{R}_{k+1} = \mathcal{Q}_{k+1}(\mathcal{R}_k - t\mathcal{P}_k\alpha_k).$$

Therefore

$$\mathcal{Q}_{k+1}\mathcal{R}_{k+1} = (\mathcal{Q}_k\mathcal{R}_k - t\mathcal{Q}_k\mathcal{P}_k\alpha_k) - \omega_k t\mathcal{Q}_k\,(\mathcal{R}_k - t\mathcal{P}_k\alpha_k). \tag{3.3}$$

Using (2.7) and (3.2) we get

$$\mathcal{Q}_{k+1}\mathcal{P}_{k+1} = \mathcal{Q}_{k+1}\mathcal{R}_{k+1} + (\mathcal{Q}_k\mathcal{P}_k - \omega_k t\mathcal{Q}_k\mathcal{P}_k)\beta_k. \tag{3.4}$$

We also have

$$\mathcal{Q}_k\mathcal{R}_{k+1} = \mathcal{Q}_k\mathcal{R}_k - t\mathcal{Q}_k\mathcal{P}_k\alpha_k. \tag{3.5}$$

We now define the new matrix direction P_k by

$$P_k = (\mathcal{Q}_k\mathcal{P}_k)(A) \circ R_0, \tag{3.6}$$

and we set

$$S_k = (\mathcal{Q}_k\mathcal{R}_{k+1})(A) \circ R_0. \tag{3.7}$$

Then, using these formulas, the iterates R_k and P_k are computed by the following recurrence formulas

$$R_{k+1} = S_k - \omega_k A S_k, \tag{3.8}$$

and

$$P_{k+1} = R_{k+1} + (P_k - \omega_k A P_k)\beta_k. \tag{3.9}$$

The scalar parameter ω_k is chosen to minimize the Frobenius norm of the $N \times s$ residual $R_{k+1} = S_k - \omega_k A S_k$. This implies that

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

Finally, we have to compute the $s \times s$ matrix coefficients α_k and β_k needed in the recurrence formulas. This is done by using the fact that the polynomials $\mathcal{R}_k$ and $\mathcal{P}_k$ belong to the families of formal matrix orthogonal polynomials with respect to the functionals $\mathcal{C}$ and $\mathcal{C}^{(1)}$, respectively. Hence, using the relations (2.6) and (2.7), Proposition 5 and the fact that $\mathcal{Q}_k$ is a scalar polynomial, we have

$$\mathcal{C}(\mathcal{Q}_k\mathcal{R}_k) = \mathcal{C}^{(1)}(\mathcal{Q}_k\mathcal{P}_k)\alpha_k, \tag{3.10}$$

and

$$\mathcal{C}^{(1)}(\mathcal{Q}_k\mathcal{R}_{k+1}) = -\mathcal{C}^{(1)}(\mathcal{Q}_k\mathcal{P}_k)\beta_k. \tag{3.11}$$

Therefore, by the definitions of the functionals $\mathcal{C}$ and $\mathcal{C}^{(1)}$, the relations (3.10) and (3.11) become

$$(\tilde{R}_0^T A P_k)\alpha_k = \tilde{R}_0^T R_k, \tag{3.12}$$

and

$$(\tilde{R}_0^T A P_k)\beta_k = -\tilde{R}_0^T A S_k. \tag{3.13}$$

So, the $s \times s$ matrices α_k and β_k can be computed by solving two $s \times s$ linear systems with the same coefficient matrix $\tilde{R}_0^T A P_k$. They will be solved by computing the LU factorization of the matrix $(\tilde{R}_0^T A P_k)$.

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

ALGORITHM 3: **Bl-BiCGSTAB**

X_0 an initial guess; $R_0 = B - AX_0$; $P_0 = R_0$;
$\tilde{R}_0$ an arbitrary $N \times s$ matrix;
For $k = 0, 1, 2, \ldots$
  $V_k = AP_k$;
  solve $(\tilde{R}_0^T V_k)\alpha_k = \tilde{R}_0^T R_k$;
  $S_k = R_k - V_k\alpha_k$;
  $T_k = AS_k$;
  $\omega_k = \langle T_k, S_k\rangle_F / \langle T_k, T_k\rangle_F$;
  $X_{k+1} = X_k + P_k\alpha_k + \omega_k S_k$;
  $R_{k+1} = S_k - \omega_k T_k$;
  solve $(\tilde{R}_0^T V_k)\beta_k = -\tilde{R}_0^T T_k$;
  $P_{k+1} = R_{k+1} + (P_k - \omega_k V_k)\beta_k$;
end.

Note that when a single linear system is solved, ALGORITHM 3 reduces to BiCGSTAB. However, the coefficient β_k computed by ALGORITHM 3 with $s = 1$ is given by $\beta_k = -\langle \tilde{R}_0, T_k\rangle_2 / \langle \tilde{R}_0, V_k\rangle_2$. This expression for β_k is simpler than the one given in ALGORITHM 2 but requires an extra inner product.

ETNA
Kent State University
etna@mcs.kent.edu

The Bl-BiCGSTAB algorithm will also suffer from a breakdown when $\tilde{R}_0^T V_k$ is singular. In situations where the matrix $\tilde{R}_0^T V_k$ is nonsingular but is very ill-conditioned the computation of the iterates will also be affected. To overcome these problems, one can restart with a different $\tilde{R}_0$. Look-ahead strategies could also be defined, but this will not be treated in this paper.

Due to the local Frobenius norm minimization steps, Bl-BiCGSTAB has smoother convergence behavior than Bl-BCG. However, the norms of the residuals produced by Bl-BiCGSTAB may oscillate for some problems. In this case, we can use the block smoothing procedure defined in [11] to get nonincreasing Frobenius norms of the residuals.

For solving the linear system (1.1), Bl-BiCGSTAB requires per iteration the evaluation of $2s$ matrix-vector products with A and a total of $6Ns^2+4Ns+O(s^3)$ multiplications. This is to be compared to s matrix-vector product with A, s matrix-vector product with A^T and a total of $8Ns^2 + O(s^3)$ multiplications needed at each iteration for Bl-BCG.

Storage requirements (excluding those of A, X and B) and the major computational costs (multiplications) per iteration for the Bl-BCG and the Bl-BiCGSTAB algorithms are listed in Table 1.

Table 1
Memory requirements and computational costs (multiplications) for Bl-BCG and Bl-BiCGSTAB

$Costs$	Bl-BCG	Bl-BiCGSTAB
Mat-Vec with A	s	$2s$
Mat-Vec with A^T	s	-
Multiplications	$8Ns^2$+$O(s^3)$	$6Ns^2$+$4Ns$+$O(s^3)$
Memory locations	$5Ns + O(s^2)$	$4Ns + O(s^2)$

The costs of Bl-BCG and Bl-BiCGSTAB rapidly increase with the number s of right-hand sides. However, these block solvers converge in fewer iterations than their single right-hand side counterparts since the dimensions of the search spaces $\mathcal{K}_k(A, R_0)$ and $\mathcal{K}_{2k}(A, R_0)$ increase by s and $2s$, respectively, in each step.

4. Numerical examples. In this section we give some experimental results. Our examples have been coded in Matlab and have been executed on a SUN SPARC workstation.

We compare the performance of Bl-BiCGSTAB, Bl-QMR, Bl-GMRES and Bl-BCG for solving the multiple linear system (1.1). For the experiments with Bl-QMR, we used the algorithm that generates a band matrix with a deflation procedure to eliminate nearly linearly dependent columns of the search space [7]. In all but the last experiment, the initial guess was $X_0 = 0$ and $B = rand(N, s)$, where function the $rand$ creates an $N \times s$ random matrix with coefficients uniformly distributed in [0,1] except for the last experiment.

Example 1. In this example, we use two sets of experiments. The tests were stopped as soon as

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

Experiment 1: The first matrix test A_1 represents the five-point discretization of the operator

$$L(u) = -u_{xx} - u_{yy} + \gamma u_x \tag{4.1}$$

on the unit square [0,1] $\times$ [0,1] with Dirichlet boundary conditions. The discretization was performed using a grid size of $h = 1/51$ which yields a matrix of dimension $N = 2500$. The matrix A_1 is nonsymmetric and has a positive definite symmetric part. We choose $\gamma = 10$. The number of second right-hand sides was $s = 10$. We set $\tilde{R}_0 = R_0$ in the Bl-BiCGSTAB and the Bl-BCG algorithms. Figure 1 reports on convergence history for Bl-BiCGSTAB, Bl-QMR, Bl-BCG and Bl-GMRES(10). In this figure, we plotted the maximum of the s residual norms (on a logarithmic scale) versus the flops (number of arithmetic operations).

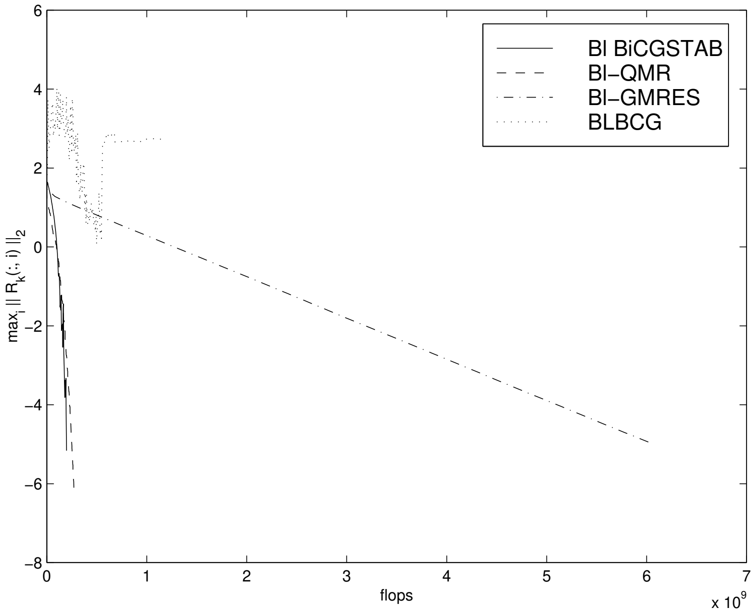

Figure 1: $A = A_1$; $s = 10$

In Figure 2, we plotted for the same example only the results obtained with the Bl-BiCGSTAB and the Bl-QMR algorithms. As observed from Figure 1 and Figure 2, Bl-BiCGSTAB returns the best results.

ETNA
Kent State University
etna@mcs.kent.edu

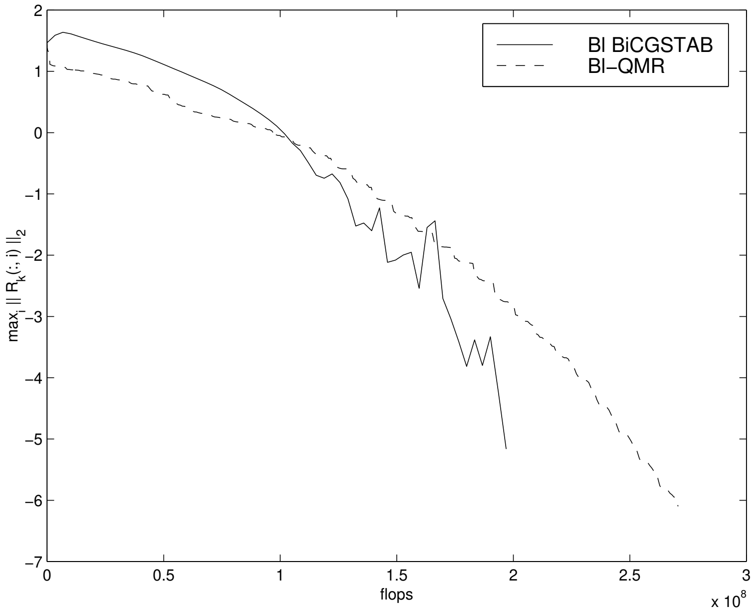

Figure 2: $A = A_2$; $s = 10$

Experiment 2: For the second experiment, we used the matrix test A_3=pde900 from Harwell-Boweing collection (the size of the nonsymmetric matrix A_3 is $N = 900$ and the number of nonzeros entries is $nnz(A_3) = 4380$). Figure 3 reports on results obtained for Bl-BiCGSTAB, Bl-QMR, Bl-BCG and Bl-GMRES(10). For this experiment the number of second right-hand sides was $s = 6$.

As can be seen from Figure 3, Bl-BiCGSTAB requires lower flops for convergence when compared to the other three block methods. The Bl-BCG algorithm failed to converge.

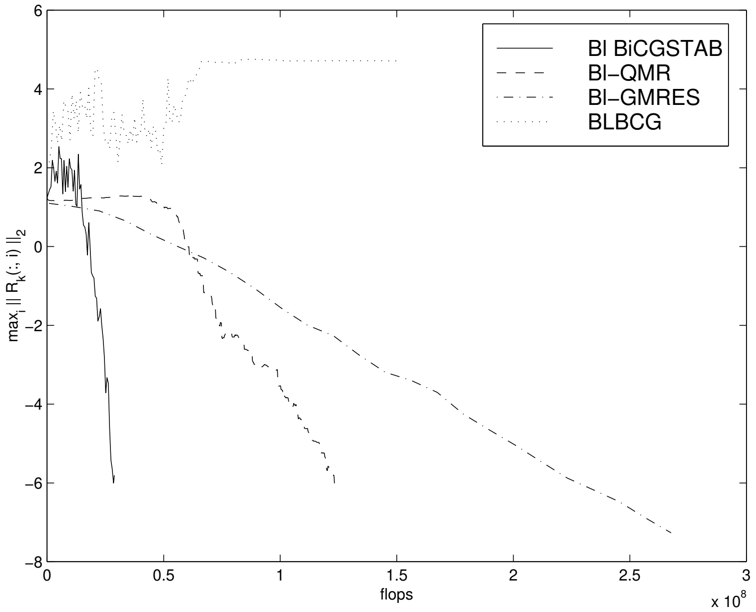

Figure 3: A_3 = pde900; $s = 6$

Example 2. For the second set of experiments, we used matrices from the Harwell-Boeing

ETNA
Kent State University
etna@mcs.kent.edu

collection using ILUT as a preconditioner with a drop tolerance $\tau = 10^{-4}$. We compared the performance (in term of flops) of the Bl-BiCGSTAB algorithm, the Bl-GMRES(10) algorithm and the BiCGSTAB algorithm applied to each single right-hand side. For all the tests the matrix B was an $N \times s$ random matrix. Two different numbers of right-hand sides were used: $s = 5$ and $s = 10$. The iterations were stopped when

$$\max_{i=1:s} \|R_k(:,i)\|_2 \leq 10^{-9}.$$

In Table 2 we list the effectiveness of Bl-BiCGSTAB measured by the ratios $f_1(s)$ and $f_2(s)$ where

- $f_1(s)$=flops(Bl-BiCGSTAB)/(flops(BiCGSTAB)),
- $f_2(s)$=flops(Bl-BiCGSTAB)/flops(Bl-GMRES(10)).

We note that flops(Bi-CGSTAB) corresponds to the flops required for solving the s linear systems by applying the Bi-CGSTAB algorithm s times.

Table 2
Effectiveness of Bl-BiCGSTAB as compared to Bl-GMRES(10) and BiCGSTAB using the ILUT preconditioner. Matrices are from the Harwell-Boeing collection.

Matrix	$s = 5$	$s = 10$	$s = 5$	$s = 10$
Utm 1700a ($N = 1700$) ($nnz(A) = 21313$)	$f_1(5) = 0.49$	$f_1(10) = 0.38$	$f_2(5) = 0.62$	$f_2(10) = 0.36$
SAYLR4($N = 3564$) ($nnz(A) = 22316$)	$f_1(5) = 0.77$	$f_1(10) = 0.89$	$f_2(5) = 0.06$	$f_2(10) = 0.12$
SHERMAN5 ($N = 3312$) ($nnz(A) = 20793$)	$f_1(5) = 0.93$	$f_1(10) = 0.94$	$f_2(5) = 0.26$	$f_2(10) = 0.16$
SHERMAN3 ($N = 5005$) ($nnz(A) = 20033$)	$f_1(5) = 0.82$	$f_1(10) = 1.10$	$f_2(5) = 0.27$	$f_2(10) = 0.25$

$nnz(A)$ denotes the number of nonzero entries in A

As shown in Table 2, Bl-BiCGSTAB is less expensive than Bl-GMRES and than BiCGSTAB applied to each single right-hand side linear system except for the last example with $s = 10$ for which $f_1(10) = 1.1$.

The relative density of the matrices and the use of preconditioners make the multiple right-hand side solvers less expensive and then they are more effective.

5. Conclusion. We have proposed in this paper a new block BiCGSTAB method for nonsymmetric linear systems with multiple right-hand sides. To define this new method, we used formal matrix-valued orthogonal polynomials.

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

REFERENCES

[1] C. BREZINSKI AND H. SADOK, *Lanczos type methods for solving systems of linear equations*, Appl. Numer. Math., 11(1993), pp. 443-473.
[2] C. BREZINSKI, M. REDIVO-ZAGLIA AND H. SADOK, *A breakdown-free Lanczos-type algorithm for solving linear systems*, Numer. Math., 63(1992), pp. 29-38.
[3] C. BREZINSKI AND M. REDIVO-ZAGLIA, *Look-ahead in Bi-CGSTAB and other methods for linear systems*, BIT, 35(1995), pp. 169-201.

[4] T. CHAN AND W. WAN, *Analysis of projection methods for solving linear systems with multiple right-hand sides*, SIAM J. Sci. Comput., 18(1997), pp. 1698-1721.

[5] G. H. GOLUB AND R.R. UNDERWOOD, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large, sparse, real symmetric matrices*, Proc. of 1974 IEEE conf. on Decision avd Control, Phoenix.

[6] R. FLETCHER, *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, 1975.

[7] R. FREUND AND M. MALHOTRA, *A Block-QMR algorithm for non-hermitian linear systems with multiple right-hand sides*, Linear Algebra Appl., 254(1997), pp. 119-157.

[8] G. H. GOLUB AND R. R. UNDERWOOD, *The block Lanczos method for computing eigenvalues*, in Mathematical Software 3, J. R. Rice, ed., Academic Press, New York, 1977, pp. 364-377.

[9] K. JBILOU A. MESSAOUDI AND H. SADOK, *Global FOM and GMRES algorithms for matrix equations*, Appl. Numer. Math., 31(1999), pp. 49-63.

[10] K. JBILOU H. SADOK, *Global Lanczos-based methods with applications*, Technical Report LMA 42, Université du Littoral, Calais, France 1997.

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

[12] C. LANCZOS, *Solution of systems of linear equations by minimized iterations*, J. Research Nat. Bur. Standards, 49(1952) pp. 33-53.

[13] M. MALHOTRA, R. FREUND AND P. M. PINSKY, *Iterative Solution of Multiple Radiation and Scattering Problems in Structural Acoustics Using a Block Quasi-Minimal Residual Algorithm*, Comput. Methods Appl. Mech. Engrg., 146(1997), pp. 173-196.

[14] A. NIKISHIN AND A. YEREMIN, *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(1995), pp. 1135-1153.

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

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

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

[18] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7(1986), pp. 856-869.

[19] M. SADKANE, *Block Arnoldi and Davidson methods for unsymmetric large eigenvalue problems*, Numer. Math., 64(1993), pp. 687-706.

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

[21] V. SIMONCINI AND E. GALLOPOULOS, *Convergence properties of block GMRES and matrix polynomials*, Linear Algebra Appl., 247(1996), pp. 97-119.

[22] V. SIMONCINI AND E. GALLOPOULOS, *An Iterative Method for Nonsymmetric Systems with Multiple Right-hand Sides*, SIAM J. Sci. Comput., 16(1995), pp. 917-933.

[23] C. SMITH, A. PETERSON AND R. MITTRA, *A Conjugate Gradient Algorithm for Tretement of Multiple Incident Electromagnetic Fields*, IEEE Trans. Antennas and Propagation, 37(1989), pp. 1490-1493.

[24] B. VITAL, *Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multiprocesseur*, Ph.D. thesis, Université de Rennes, Rennes, France, 1990.

[25] H. VAN DER VORST, *Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13(1992), pp. 631-644.

[26] H. VAN DER VORST, *An Iterative Solution Method for Solving $f(A) = b$, using Krylov Subspace Information Obtained for the Symmetric Positive Definite Matrix A*, J. Comput. Appl. Math., 18(1987), pp. 249-263.