Electronic Transactions on Numerical Analysis.
Volume 20, pp. 119-138, 2005.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

OBLIQUE PROJECTION METHODS FOR LINEAR SYSTEMS WITH MULTIPLE RIGHT-HAND SIDES*

K. JBILOU†, H. SADOK‡, AND A. TINZEFTE§

Abstract. In the present paper, we describe new Lanczos-based methods for solving nonsymmetric linear systems of equations with multiple right-hand sides. These methods are based on global oblique projections of the initial residual onto a matrix Krylov subspace. We first derive the global Lanczos process to construct biorthonormal bases and we give some of its properties. Then we introduce new methods such as the global BCG and the global BiCGSTAB algorithms. Look-ahead versions of these algorithms are also given. Finally numerical examples will be given.

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

AMS subject classifications. 65F10, 65F25

1. Introduction.

In many applications, we have to solve a few linear systems of equations with the same coefficient matrix and different right-hand sides. This is the case, for example, in numerical simulation of wave propagation. When all the right hand sides are available simultaneously, the problem we are concerned with can be expressed as

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

where A is an $N \times N$ real nonsymmetric matrix, $B = [b_1, b_2, \ldots, b_s]$ and $X = [x_1, x_2, \ldots, x_s]$ are rectangular matrices of order $N \times s$ with $s \ll N$.

For nonsymmetric problems, several block Krylov subspace methods have been developed during the last years. The most popular methods are the block-biconjugate gradient (Bl-BCG) method [11, 17], the block-generalized minimum residual (BGMRES) algorithm [14, 20], the block-quasi-minimal residual (Bl-QMR) algorithm [8, 10] and the block-biconjugate gradient stabilized (Bl-BiCGSTAB) method [6]. We note that block-methods require a deflation procedure to detect and delete linearly or almost linearly dependent vectors in the block Krylov subspaces generated during the iterations; see [8] for details.

The matrix equation (1.1) can also be solved by applying a method for a single-vector right-hand side to one of the columns, say b_q, $q \in \{1, \ldots, s\}$, of B and solving the linear system

$$Ax_q = b_q.$$

The preceding linear system is refereed as a seed system. The residuals of the other systems with a single right-hand side are then projected onto the Krylov subspace associated with the seed system. This procedure has been used in [4, 18, 19]. This technique is especially attractive when the right-hand sides b_j, $j = 1, \ldots, s$ of (1.1) are not available at the same time; see for example [12, 21].

In the present paper, we use a third approach for solving the problem (1.1). This approach, which we previously used to define the global GMRES algorithm [9], is based on

*Received October 19, 2004. Accepted for publication January 11, 2005. Recommended by L. Reichel.

†Université du Littoral, zone universitaire de la Mi-voix, batiment H. Poincarré, 50 rue F. Buisson, BP 699, F-62228 Calais Cedex, France. (jbilou@lmpa.univ-littoral.fr).

‡Université du Littoral, zone universitaire de la Mi-voix, batiment H. Poincarré, 50 rue F. Buisson, BP 699, F-62228 Calais Cedex, France. (sadok@lmpa.univ-littoral.fr).

§Laboratoire d'analyse numérique et d'Optimisation, UFR IIEA-M3, Université des sciences et technologies de Lille, F-59655 Villeneuve d'Ascq, France. (tinzefte@math.univ-lille1.fr).

oblique projections onto a matrix Krylov subspace and allows us to define the global Lanczos procedure that will be used to obtain the global Lanczos algorithm. We introduce global Lanczos-based algorithms, such as the global biconjugate gradient (Gl-BCG) algorithm, the global BiCGSTAB algorithm and look-ahead versions of these algorithms.

The paper is organized as follows. In Section 2, we introduce the global Lanczos process with some properties. In Section 3, we describe the global BCG algorithm and give some techniques for curing breakdowns. Section 4 is devoted to the global BiCGSTAB method with a look-ahead version. Finally, in Section 5 we give some numerical examples.

In the last section, we give some numerical examples to show the effectiveness of these new methods.

Throughout this paper, we use the following notations. For two matrices X and Y in $\mathbb{R}^{N\times s}$, we define the inner product $< X, Y >_F = tr(X^T\,Y)$, where $tr(Z)$ denotes the trace of the square matrix Z and X^T the transpose of the matrix X. The associated norm is the Frobenius norm which we denote by $\|\ .\ \|_F$. For a matrix $V \in \mathbb{R}^{N\times s}$, the matrix Krylov subspace $\mathcal{K}_k(A, V)$ is the subspace of $\mathbb{R}^{N\times s}$ generated by the matrices $V, A\,V, \ldots, A^{k-1}\,V$. A system of matrices of $\mathbb{R}^{N\times s}$ is said to be F-orthogonal if it is orthogonal with respect to the inner product $< .,. >_F$.

2. The global Lanczos process. Let V be an $N\times s$ real matrix and denote by $\mathcal{K}_k(A, V)$ the matrix Krylov subspace of $\mathbb{R}^{N\times s}$ spanned by $V, A\,V, \ldots, A^{k-1}V$. Note that

$$Z \in \mathcal{K}_k(A, V) \Longleftrightarrow Z = \sum_{i=1}^{k} \alpha_i A^{i-1} V; \;\; \alpha_i \in \mathbb{R}; \;\; i = 1, \ldots, k.$$

In other words $\mathcal{K}_k(A, V)$ is the subspace of $\mathbb{R}^{N\times s}$ of all $N\times s$ matrices which can be written as $Z = P(A)V$, where P is a polynomial of degree not exceeding $k-1$. This means that each column of Z is associated with one Krylov subspace.

Remark also that the matrix Krylov subspace $\mathcal{K}_k(A, V)$ is quite different from the block Krylov subspace $\tilde{\mathcal{K}}_k(A, V)$ used in block methods. In fact

$$Z \in \tilde{\mathcal{K}}_k(A, V) \Longleftrightarrow Z = \sum_{i=1}^{k} A^{i-1} V \Omega_i; \;\; \Omega_i \in \mathbb{R}^{s\times s}, \; i = 1, \ldots, k.$$

In this case, each column of Z is associated with a sum of s Krylov subspaces.

The minimal polynomial of A for V is the nonzero monic polynomial of lowest degree such that $P(A)V = 0$. The degree m of this polynomial does not exceed N. We have the following result which is easy to prove.

PROPOSITION 2.1. *Let m be the degree of the minimal polynomial of A for V. Then we have*

(1) $\mathcal{K}_m(A, V)$ is invariant under A.
(2) $dim(\mathcal{K}_k(A, V)) = min(k, m)$.

Let V_1 and W_1 be two $N\times s$ matrices and denote by $\mathcal{K}_k(A, V_1)$ and $\mathcal{K}_k(A^T, W_1)$ the matrix Krylov subspaces generated by $\{V_1, AV_1, \ldots, A^{(k-1)}V_1\}$ and $\{W_1, A^T W_1, \ldots, A^{T(k-1)} W_1\}$, respectively.

The global Lanczos process constructs a pair of two global biorthogonal bases $\{V_1, V_2, \ldots, V_k\}$ and $\{W_1, W_2, \ldots, W_k\}$ of the matrix Krylov subspaces $\mathcal{K}_k(A, V_1)$ and $\mathcal{K}_k(A^T, W_1)$, respectively, such that

$$< V_i, W_j >_F = tr(V_i^T\,W_j) = \delta_{ij}; \; i, j = 1, \ldots, k.$$

The algorithm is defined as follows:

ALGORITHM 1 **The global Lanczos process**

1. Choose two $N \times s$ matrices V_1 and W_1 such that $< V_1, W_1 >_F = 1$,
2. set $\beta_1 = \delta_1 = 0$ and $W_0 = V_0 = 0$,
3. for $j = 1, 2, \ldots, k$
   $\alpha_j = tr(W_j^T AV_j)$,
   $\tilde{V}_{j+1} = AV_j - \alpha_j V_j - \beta_j V_{j-1}$,
   $\tilde{W}_{j+1} = A^T W_j - \alpha_j W_j - \delta_j W_{j-1}$,
   $\delta_{j+1} = | \, tr(\tilde{V}_{j+1}^T \tilde{W}_{j+1}) \, |^{1/2}$,
   $\beta_{j+1} = tr(\tilde{V}_{j+1}^T \tilde{W}_{j+1})/\delta_{j+1}$,
   $V_{j+1} = \tilde{V}_{j+1}/\delta_{j+1}$,
   $W_{j+1} = \tilde{W}_{j+1}/\beta_{j+1}$,
   end.

Note that a breakdown occurs in the algorithm if, for some j, $tr(\tilde{V}_{j+1}^T \tilde{W}_{j+1}) = 0$. If $\tilde{V}_{j+1} = 0$ for some j, then the matrix Krylov subspace $\mathcal{K}_j(A, V_1)$ is invariant under A and then $j \geq m$ the degree of the minimal polynomial of A for V_1. We will see that in this case we obtain the exact solution of the problem (1.1).

Below we will give a look-ahead Lanczos-type algorithm that avoids the breakdown. Note that, since Algorithm 1 does not involve matrix inversions, the problem of linear dependence of vectors in the sequences $V_1, AV_1, \ldots$ and $W_1, A^T W_1, \ldots$ is not an issue. Therefore no deflation procedure to delete linearly or almost linearly dependent vectors is required. From now on, we set $\mathcal{V}_k = [V_1, \ldots, V_k]$ and $\mathcal{W}_k = [W_1, \ldots, W_k]$, two matrices of dimension $N \times ks$. Let T_k be the tridiagonal matrix of dimension $k \times k$ defined as:

$$T_k = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \delta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_k \\ & & \delta_k & \alpha_k \end{bmatrix},$$

where α_i, β_i and δ_i are the scalars defined in the Algorithm 1.

Note that, for the Block Lanczos algorithm, the corresponding matrix is a block-tridiagonal matrix of dimension $ks \times ks$.
Define the matrix

$$\tilde{T}_k = \begin{bmatrix} T_k \\ \delta_{k+1} e_k^T \end{bmatrix},$$

where $e_k = (0, \ldots, 0, 1)^T \in \mathbb{R}^k$.

We use the notation $*$ defined in [9] for the following product

$$\mathcal{V}_k * y = \sum_{i=1}^{k} y^i V_i = \mathcal{V}_k (y \otimes I_s), \tag{2.1}$$

where $y = (y^1, y^2, \ldots, y^k)^T$ is a vector of $\mathbb{R}^k$, and analogously

$$\mathcal{V}_k * T_k = [\mathcal{V}_k * T_{.,1}, \mathcal{V}_k * T_{.,2}, \ldots, \mathcal{V}_k * T_{.,k}], \tag{2.2}$$

where $T_{.,i}$ denotes the i-th column of the matrix T_k.

Using these notations, we have the following result.

PROPOSITION 2.2. *Assume that the global Lanczos algorithm does not break down before k steps. Then $\{V_1, \ldots, V_k\}$ and $\{W_1, \ldots, W_k\}$ form bases of the matrix Krylov subspaces $\mathcal{K}_k(A, V_1)$ and $\mathcal{K}_k(A^T, W_1)$, respectively, and we have the following relations :*

$$A\,\mathcal{V}_k = \mathcal{V}_k * T_k + \delta_{k+1}[0, \ldots, 0, V_{k+1}], \tag{2.3}$$

$$A\,\mathcal{V}_k = \mathcal{V}_{k+1} * \tilde{T}_k. \tag{2.4}$$

Proof. From the definition of the product * and the structure of the matrix T_k, we have for $j = 1, \ldots, k-1$,

$$\begin{aligned} \mathcal{V}_k * T_{.,j} &= \alpha_j V_j + \delta_{j+1} V_{j+1} + \beta_j V_{j-1} \\ &= AV_j, \end{aligned}$$

and

$$\begin{aligned} \mathcal{V}_k * T_{.,k} &= \beta_k V_{k-1} + \alpha_k V_k \\ &= AV_k + \delta_{k+1} V_{k+1}. \end{aligned}$$

Then we obtain

$$A\,\mathcal{V}_k = \mathcal{V}_k * T_k + \delta_{k+1}[0, \ldots, 0, V_{k+1}].$$

For the relation (2.4), we have

$$\begin{aligned} \mathcal{V}_{k+1} * \tilde{T}_k &= [\mathcal{V}_k, V_{k+1}] * \tilde{T}_k \\ &= \mathcal{V}_k * T_k + \delta_{k+1} e_k^T V_{k+1} \\ &= \mathcal{V}_k * T_k + \delta_{k+1}[0, \ldots, 0, V_{k+1}]. \end{aligned}$$

Hence, using the relation (2.3), the result of (2.4) holds. □

Consider now the block linear system (1.1), let X_0 be an initial guess and let $R_0 = B - AX_0$ be the corresponding residual. The global Lanczos method for solving (1.1) generates, at step k, the iterate X_k such that

$$X_k - X_0 = Z_k \in \mathcal{K}_k(A, R_0) \tag{2.5}$$

and

$$R_k = B - AX_k \perp_F \mathcal{K}_k(A^T, \tilde{R}_0), \tag{2.6}$$

where $\tilde{R}_0$ is a given $N \times s$ matrix provided that, $< R_0, \tilde{R}_0 >_F \neq 0$. Let $\mathcal{S}_k$ denote the oblique projector onto $A\mathcal{K}_k(A, R_0)$ and orthogonal to $\mathcal{K}_k(A^T, \tilde{R}_0)$. Then it follows from the relations (2.5) and (2.6) that

$$R_k = R_0 - \mathcal{S}_k\, R_0. \tag{2.7}$$

Let $\{V_1, \ldots, V_k\}$ and $\{W_1, \ldots, W_k\}$ be the sets of matrices constructed by Algorithm 1, generating the matrix Krylov subspaces $\mathcal{K}_k(A, R_0)$ and $\mathcal{K}_k(A^T, \tilde{R}_0)$, respectively, with the

ETNA
Kent State University
etna@mcs.kent.edu

initializations $V_1 = R_0 / \parallel R_0 \parallel_F$ and W_1 such that, $< V_1, W_1 >_F = 1$. Now, from the relation (2.5) it follows that,

$$X_k = X_0 + \mathcal{V}_k * y_k \tag{2.8}$$

where y_k is the vector of $\mathbb{R}^k$ obtained from

$$< R_0 - A\mathcal{V}_k * y_k, W_i >_F = 0,$$

which is equivalent to

$$< R_0, W_i >_F = < A\mathcal{V}_k * y_k, W_i >_F ; \; i = 1, \ldots, k. \tag{2.9}$$

Hence, (2.9) can be written as

$$\sum_{j=1}^{k} y_k^j tr(W_1^T A V_j) = \parallel R_0 \parallel_F$$

and

$$\sum_{j=1}^{k} y_k^j tr(W_i^T A V_j) = 0; \; i = 2, \ldots, k.$$

Finally, the preceding linear system can be expressed as

$$T_k \, y_k = \parallel R_0 \parallel_F \, e_1^{(k)}, \tag{2.10}$$

where $e_1^{(k)}$ is the first vector of the canonical basis of $\mathbb{R}^k$. If the tridiagonal matrix T_k is nonsingular, the iterate X_k obtained by the global Lanczos method is then given as

$$X_k = X_0 + \parallel R_0 \parallel_F \; \mathcal{V}_k * T_k^{-1} e_1^{(k)}. \tag{2.11}$$

Let us see now how to compute the norm of the residual R_k without actually having to compute the approximation X_k. This will be useful for determining whether convergence is achieved without explicitly using X_k. The residual R_k is given as

$$R_k = R_0 - A\mathcal{V}_k * y_k.$$

From the relation (2.3) and the fact that, $R_0 = \parallel R_0 \parallel_F V_1$, it follows that,

$$R_k = \parallel R_0 \parallel_F V_1 - \mathcal{V}_k * T_k y_k + \delta_{k+1}[0, 0, \ldots, V_{k+1}] * y_k.$$

On the other hand, since $V_1 = \mathcal{V}_k * e_1^{(k)}$, we obtain

$$R_k = \mathcal{V}_k * (\parallel R_0 \parallel_F e_1^{(k)} - T_k y_k) + \delta_{k+1}[0, 0, \ldots, V_{k+1}] * y_k.$$

Finally, using (2.11) in the preceding equation, we get

$$\parallel R_k \parallel_F = \mid \delta_{k+1} y_k^k \mid \parallel V_{k+1} \parallel_F, \tag{2.12}$$

where y_k^k is the last component of the vector y_k. If m is the degree of the minimal polynomial of A for R_0, then $\mathcal{K}_m(A, R_0)$ is invariant and $X_m = X$ is the exact solution of (1.1). As $m \leq N$, the algorithm converges in at most N iterations.

In what follows, we describe some global Lanczos-based methods for solving the multiple linear system (1.1).

ETNA
Kent State University
etna@mcs.kent.edu

3. The Global Biconjugate Gradient method.

3.1. The Global BCG algorithm. The Global Biconjugate Gradient (Gl-BCG) algorithm can be derived from Algorithm 1 in the same way as the classical BCG has been obtained in [7]. At step k, the residual R_k generated by this algorithm is such that, $R_k - R_0$ lies in the right matrix Krylov subspace $\mathcal{K}_k(A, AR_0) = span\{AR_0, A^2R_0, \ldots, A^kR_0\}$ and R_k is F-orthognal to the left matrix Krylov subspace $\mathcal{K}_k(A^T, \tilde{R}_0) = span\{\tilde{R}_0, A^T\tilde{R}_0, \ldots, {A^T}^{k-1}$ $\tilde{R}_0\}$, where $\tilde{R}_0$ is a given $N \times s$ matrix.

The algorithm is defined as follows

ALGORITHM 2 **The Global Biconjugate Gradient (Gl-BCG) algorithm**

1. Compute $R_0 = B - AX_0$ for a given X_0, and choose $\tilde{R}_0$ such that, $< R_0, \tilde{R}_0 >_F \neq 0$,
2. set $P_0 = R_0$ and $\tilde{P}_0 = \tilde{R}_0$,
3. for $j = 0, 1, \ldots$ compute

   $a.$ $X_{j+1} = X_j + \alpha_j P_j$, where $\alpha_j = \dfrac{< R_j, \tilde{R}_j >_F}{< AP_j, \tilde{P}_j >_F}$,

   $b.$ $R_{j+1} = R_j - \alpha_j AP_j$,

   $c.$ $\tilde{R}_{j+1} = \tilde{R}_j - \alpha_j A^T\tilde{P}_j$,

   $d.$ $P_{j+1} = R_{j+1} + \beta_j P_j$, where $\beta_j = \dfrac{< R_{j+1}, \tilde{R}_{j+1} >_F}{< R_j, \tilde{R}_j >_F}$,

   $e.$ $\tilde{P}_{j+1} = \tilde{R}_{j+1} + \beta_j \tilde{P}_j$.

It is not difficult to prove the next results.

PROPOSITION 3.1. *The matrices produced by the Gl-BCG algorithm satisfy the following relations:*

(1) $< R_k, \tilde{R}_l >_F = 0$ and $< AP_k, \tilde{P}_l >_F = 0; \;\; k \neq l.$

(2) $span\{P_0, \ldots, P_k\} = span\{R_0, \ldots, A^kR_0\}.$

(3) $span\{\tilde{P}_0, \ldots, \tilde{P}_k\} = span\{\tilde{R}_0, \ldots, {A^T}^k\tilde{R}_0\}.$

(4) $R_k - R_0 \in \mathcal{K}_k(A, R_0)$ *and* R_k *is orthogonal to* $\mathcal{K}_k(A^T, \tilde{R}_0)$.

The residual R_k produced by the Gl-BCG algorithm can also be expressed as

$$R_k = \mathcal{P}_k(A)R_0,$$

where $\mathcal{P}_k$ is a polynomial of degree k with scalar coefficients satisfying $\mathcal{P}_k(0) = 1$. The matrix direction P_k can also be written as

$$P_k = \phi_k(A)R_0.$$

Here ϕ_k is a polynomial with scalar coefficients. Note that $\tilde{R}_k$ and $\tilde{P}_k$ also can be expressed as

$$\tilde{R}_k = \mathcal{P}_k(A^T)\tilde{R}_0 \;\text{ and }\; \tilde{P}_k = \phi_k(A^T)\tilde{R}_0.$$

One disadvantage of the Gl-BCG algorithm is the fact that breakdowns may occur in the algorithm. In the following subsection we give a look-ahead Lanczos-type algorithm that avoids this problem.

3.2. A look-ahead global Lanczos-type algorithm. The global Lanczos method constructs a sequence of approximations (X_k), $k = 1, 2, \ldots$ such that,

$$X_k - X_0 \in \mathcal{K}_k(A, R_0)$$

and

$$R_k \perp_F \mathcal{K}_k(A^T, \tilde{R}_0),$$

where $\tilde{R}_0$ and X_0 are chosen $N \times s$ matrices. Hence the residual R_k satisfies

$$R_k = \mathcal{P}_k(A) R_0,$$

where $\mathcal{P}_k$ is a scalar polynomial of degree at most k with $\mathcal{P}_k(0) = 1$. Then the F-orthogonality property gives

$$< A^{T^i} \tilde{R}_0, R_k >_F = 0;\ i = 0, \ldots, k-1.$$

Setting $c_i \ = < \ \tilde{R}_0, A^i R_0 \ >_F$ and defining c to be the linear functional on the space of polynomials by $c(t^i) = c_i$, the orthogonality relation can be written as

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

This shows that $\mathcal{P}_k$ is the polynomial of degree k belonging to the family of orthogonal polynomials with respect to the functional c. The F-orthogonality property shows that the polynomial $\mathcal{P}_k$ exists and is unique if and only if the Hankel determinant

$$\mathcal{H}_k^{(1)} = \begin{vmatrix} c_1 & \ldots & c_k \\ \vdots & & \vdots \\ c_k & \ldots & c_{2k-1} \end{vmatrix} \neq 0.$$

Let $c^{(1)}$ be the linear functional defined on the space of polynomials by $c^{(1)}(t^i) = c(t^{i+1}) = c_{i+1}$ and let $\mathcal{P}_k^{(1)}$ be the monic polynomial of degree k belonging to the family of formal orthogonal polynomials with respect to $c^{(1)}$. $\mathcal{P}_k$ and $\mathcal{P}_k^{(1)}$ exist under the condition that $\mathcal{H}_k^{(1)} \neq 0$.

The recursive computation of $\mathcal{P}_k$ involves the computation of some scalar products which appear as denominators and of the recurrence relationships. Thus, if one of these scalar products vanishes, a breakdown occurs in the algorithm. This can be avoided by jumping over these polynomials and by computing only the existing ones (called regular). This kind of breakdown is called *true breakdown* [1]. There in another possible breakdown in Lanczos-type algorithms (called *ghost breakdown* [1] which is not due to the non-existence of some orthogonal polynomials of the family $\mathcal{P}_k$, but to the recurrence relationship under consideration which cannot be used for computing $\mathcal{P}_k$, for some k. For instance, in Gl-Lanczos/Ortores ($\mathcal{P}_{k+1}$ is computed from $\mathcal{P}_k$ and $\mathcal{P}_{k-1}$) and in Gl-Lanczos/orthomin (Gl-BICG) ($\mathcal{P}_{k+1}$ is computed from $\mathcal{P}_k$ and $\mathcal{P}_k^{(1)}$, and the polynomial $\mathcal{P}_{k+1}^{(1)}$ is computed form $\mathcal{P}_{k+1}$ and $\mathcal{P}_k^{(1)}$). Since we are interested only on the existing polynomials (regular), we still denote by $\mathcal{P}_k$ and $\mathcal{P}_{k+1}$ two successive regular polynomials of degrees n_k and $n_k + m_k$ where m_k is the length of the jump. As shown in [5], m_k is defined such that,

$$c^{(1)}(t^i \mathcal{P}_k^{(1)}) = 0;\ \ i = 0, \ldots, n_k + m_k - 2$$

and

$$c^{(1)}(t^{n_k+m_k-1}\mathcal{P}_k^{(1)}) \neq 0.$$

The polynomials $\mathcal{P}_{k+1}$ et $\mathcal{P}_{k+1}^{(1)}$ are computed by the following recursive relations

$$\begin{aligned}\mathcal{P}_{k+1}(t) &= \mathcal{P}_k(t) - tw_k(t)\mathcal{P}_k^{(1)}(t)\\ \mathcal{P}_{k+1}^{(1)}(t) &= q_k(t)\mathcal{P}_k^{(1)}(t) - C_{k+1}P_{k-1}^{(1)}(t),\end{aligned}$$

where $\mathcal{P}_{-1}^{(1)}(t) = 0$, $C_1 = 0$ and $\mathcal{P}_0^{(1)}(t) = 1$. Here q_k is a monic polynomial of degree m_k and w_k is a polynomial of degree at most $m_k - 1$.
If we set

$$R_k = \mathcal{P}_k(A)R_0 \qquad \text{and} \qquad Z_k = \mathcal{P}_k^{(1)}(A)R_0,$$

then we get the following relations

$$\begin{aligned}R_{k+1} &= R_k - Aw_k(A)Z_k,\\ X_{k+1} &= X_k - w_k(A)Z_k,\\ Z_{k+1} &= q_k(A)Z_k - C_{k+1}Z_{k-1},\end{aligned}$$

with the initializations $Z_0 = R_0$, $Z_{-1} = 0$ and $C_1 = 0$. The scalars C_{k+1} and the coefficients of the polynomials q_k and w_k are computed using the orthogonality relations of the polynomials $\mathcal{P}_{k+1}$ and $\mathcal{P}_{k+1}^{(1)}$. Since $\mathcal{P}_k^{(1)}$ is of degree exactly n_k these orthogonality relations can be expressed as follows

$$\begin{aligned}c(t^i\mathcal{P}_k^{(1)}\mathcal{P}_{k+1}) &= 0;\;\; i = 0,\ldots,m_k-1,\\ c^{(1)}(t^i\mathcal{P}_k^{(1)}\mathcal{P}_{k+1}^{(1)}) &= 0;\;\; i = 0,\ldots,m_k-1,\end{aligned}$$

and

$$\begin{aligned}c^{(1)}(t^i{\mathcal{P}_k^{(1)}}^2) &= 0;\;\; i = 0,\ldots,m_k-2,\\ c^{(1)}(t^{m_k-1}{\mathcal{P}_k^{(1)}}^2) &\neq 0.\end{aligned}$$

Let

$$\tilde{Z}_k = \mathcal{P}_k^{(1)}(A^T)\tilde{R}_0.$$

Then we have the recurrence relation

$$\tilde{Z}_{k+1} = q_k(A^T)\tilde{Z}_k - C_{k+1}\tilde{Z}_{k-1},$$

with $\tilde{Z}_0 = \tilde{R}_0$ et $\tilde{Z}_{-1} = 0$. The length of the jump m_k, is computed by using the following relations

$$< \tilde{Z}_k, A^{i+1}Z_k >_F = 0 \;\text{ for } i = 0,\ldots,m_k-2 \;\text{ and }\; < \tilde{Z}_k, A^{m_k}Z_k >_F \neq 0.$$

Let $w_k(t) = \sum_{i=0}^{m_k-1}\beta_i^{(k)}t^i$ and $q_k(t) = \sum_{i=0}^{m_k}\alpha_i^{(k)}t^i$ with $\alpha_{m_k}^{(k)} = 1$, and define

$$\begin{aligned}d_i^{(k)} &= < \tilde{Z}_k, A^iR_k >_F;\;\; i = 0,\ldots,m_k-1,\\ b_i^{(k)} &= < \tilde{Z}_k, A^{m_k+i}Z_k >_F;\;\; i = 0,\ldots,m_k.\end{aligned}$$

The coefficients $\alpha_i^{(k)}$ and $\beta_i^{(k)}$, $i = 0, \dots, m_k - 1$, are given as solutions of the following two triangular linear systems with the same coefficient matrix

$$\begin{bmatrix} b_0^{(k)} & & & & & \\ b_1^{(k)} & b_0^{(k)} & & & 0 & \\ \vdots & b_1^{(k)} & \ddots & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ b_{m_k-2}^{(k)} & b_{m_k-1}^{(k)} & \dots & \ddots & b_0^{(k)} & \\ b_{m_k-1}^{(k)} & b_{m_k-2}^{(k)} & \dots & \dots & b_1^{(k)} & b_0^{(k)} \end{bmatrix} \begin{bmatrix} \beta_{m_k-1}^{(k)} & \alpha_{m_k-1}^{(k)} \\ \beta_{m_k-2}^{(k)} & \alpha_{m_k-2}^{(k)} \\ \vdots & \vdots \\ \vdots & \vdots \\ \beta_1^{(k)} & \alpha_1^{(k)} \\ \beta_0^{(k)} & \alpha_0^{(k)} \end{bmatrix} = \begin{bmatrix} d_0^{(k)} & -b_1^{(k)} \\ d_1^{(k)} & -b_2^{(k)} \\ \vdots & \vdots \\ \vdots & \vdots \\ d_{m_k-2}^{(k)} & -b_{m_k-1}^{(k)} \\ d_{m_k-1}^{(k)} & -b_{m_k}^{(k)} \end{bmatrix}.$$

As $b_0^{(k)} =< \tilde{Z}_k, A^{m_k} Z_k >_F \neq 0$, the matrix of the preceding system is always nonsingular. We also note that, $C_{k+1} = b_0^{(k)}/b_0^{(k-1)}$. Summarizing we get the following algorithm:

ALGORITHM 3 **Global mrz-stab**$(A, B, X_0, \tilde{R}_0)$

1. **initialization**
   $Z_{-1} = 0$, $\tilde{Z}_{-1} = 0$, $R_0 = B - AX_0$, $Z_0 = R_0$, $\tilde{Z}_0 = \tilde{R}_0$
   $n_0 = 0$, $C_1 = 0$, $m_{-1} = 0$ **and** $k = 0$,

2. while $R_k \neq 0$ **do**
   $Z_{k,0} = Z_k$, $\tilde{Z}_{k,0} = \tilde{Z}_k$, $d_0^{(k)} =< \tilde{Z}_{k,0}, R_k >_F$,
   $Z_{k,1} = AZ_{k,0}$, $m_k = 1$, $b_0^{(k)} =< \tilde{Z}_{k,0}, Z_{k,1} >_F$,

3. **while** $b_0^{(k)} = 0$ **do**
      $m_k = m_k + 1$, $Z_{k,m_k} = AZ_{k,m_k-1}$, $b_0^{(k)} =< \tilde{Z}_{k,0}, Z_{k,m_k} >_F$,
   end while

4. $\beta_{m_k-1}^{(k)} = d_0^{(k)}/b_0^{(k)}$
   if $k \neq 0$ **then** $C_{k+1} = b_0^{(k)}/b_0^{(k-1)}$,
   end if

5. **for** $i = 1, \cdots, m_k$ **do**
      $\tilde{Z}_{k,i} = A^T \tilde{Z}_{k,i-1}$
      $b_i^{(k)} =< \tilde{Z}_{k,i}, Z_{k,m_k} >_F$
      if $i \neq m_k$ **then**
         $d_i^{(k)} =< \tilde{Z}_{k,i}, R_k >_F$
         compute $\beta_{m_k-i-1}^{(k)}$
      end if
      compute $\alpha_{m_k-i}^{(k)}$
   end for

6. $X_{k+1} = X_k + \beta_0^{(k)} Z_{k,0} + \beta_1^{(k)} Z_{k,1} + \cdots + \beta_{m_k-1}^{(k)} Z_{k,m_k-1}$
   $R_{k+1} = R_k - [\beta_0^{(k)} Z_{k,1} + \beta_1 Z_{k,2} + \cdots + \beta_{m_k-1}^{(k)} Z_{k,m_k}]$
   $Z_{k+1} = \alpha_0^{(k)} Z_{k,0} + \alpha_1^{(k)} Z_{k,1} + \cdots + \alpha_{m_k-1}^{(k)} Z_{k,m_k-1} + Z_{k,m_k} - C_{k+1} Z_{k-1}$
   $\tilde{Z}_{k+1} = \alpha_0^{(k)} \tilde{Z}_{k,0} + \alpha_1^{(k)} \tilde{Z}_{k,1} + \cdots + \alpha_{m_k-1}^{(k)} \tilde{Z}_{k,m_k-1} + \tilde{Z}_{k,m_k} - C_{k+1} \tilde{Z}_{k-1}$

7. $n_{k+1} = n_k + m_k$
   $k = k + 1$

end while

The computation of the global MRZ-stab algorithm requires the storage of many matrices of dimension $N \times s$. To overcome this problem, we use Horner's algorithm to compute $w_k(A)R_k$ and $q_k(A)Z_k$. Let h_k be the monic polynomial defined by

$$h_k(t) = \gamma_0^{(k)} + \ldots + \gamma_{m_k-1}^{(k)} t^{m_k-1} + t^{m_k}.$$

Then applying Horner's algorithm, the polynomial h_k can be computed recursively as follows

$$\begin{aligned} h_k^{(0)} &= 1, \\ h_k^{(i)} &= t h_k^{(i-1)} + \gamma_{m_k-i}^{(k)} \quad \text{for } i = 1, \ldots, m_k, \\ h_k(t) &= h_k^{(m_k)}(t). \end{aligned}$$

The coefficients $\gamma_i^{(k)}$, $i = 1, \ldots, m_k$, are computed such that,

$$\begin{bmatrix} b_0^{(k)} & & & & & \\ b_1^{(k)} & b_0^{(k)} & & & 0 & \\ \vdots & b_1^{(k)} & \ddots & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ b_{m_k-2}^{(k)} & b_{m_k-1}^{(k)} & \ldots & \ddots & b_0^{(k)} & \\ b_{m_k-1}^{(k)} & b_{m_k-2}^{(k)} & \ldots & \ldots & b_1^{(k)} & b_0^{(k)} \end{bmatrix} \begin{bmatrix} \gamma_{m_k-1}^{(k)} \\ \gamma_{m_k-2}^{(k)} \\ \vdots \\ \vdots \\ \gamma_1^{(k)} \\ \gamma_0^{(k)} \end{bmatrix} = - \begin{bmatrix} b_1^{(k)} \\ b_2^{(k)} \\ \vdots \\ \vdots \\ b_{m_k-1}^{(k)} \\ b_{m_k}^{(k)} \end{bmatrix}.$$

Then using the expressions of the coefficients $\beta_i^{(k)}$ and $\alpha_i^{(k)}$, one obtains

$$\begin{aligned} w_k(t) &= \frac{1}{b_0^{(k)}} \sum_{j=0}^{m_k-1} d_{m_k-j-1}^{(k)} h_k^{(j)}(t), \\ q_k(t) &= h_k(t) = h_k^{(m_k)}(t). \end{aligned}$$

Note that, with this new approach, we do not have to store all the $b_i^{(k)}$'s but only $b_0^{(k)}$. The new algorithm is called global HMRZ-stab and is summarized as follows.

ALGORITHM 4 **Global HMRZ-stab** $(A, B, X_0, \tilde{R}_0)$

1. **initialization**

   $Z_{-1} = 0, \quad \tilde{Z}_{-1} = 0$
   $R_0 = B - AX_0$
   $\tilde{Z}_0 = \tilde{R}_0$
   $Z_0 = R_0$
   $n_0 = 0$
   $C_1 = 0$
   $b_0^{(-1)} = 0; \quad k = 0$

2. **while** $R_k \neq 0$ **do**

   $d_0^{(k)} = < \tilde{Z}_k, R_k >_F$
   $m_k = 1$
   $Y_k = A^T \tilde{Z}_k$
   $V_k = Y_k$
   $b_0^{(k)} = < Y_k, Z_k >_F$

ETNA
Kent State University
etna@mcs.kent.edu

3. **while** $b_0^{(k)} = 0$ **do**
$m_k = m_k + 1$
$d_{m_k-1}^{(k)} = < Y_k, R_k >_F$
$Y_k = A^T Y_k$
$b_0^{(k)} = < Y_k, Z_k >_F$
end while

4. **if** $k \neq 0$ **then**
$C_{k+1} = b_0^{(k)} / b_0^{(k-1)}$
end if
$T_k = Z_k$
$Z_{k+1} = -C_{k+1} Z_{k-1}$
$\tilde{T}_k = \tilde{Z}_k$
$\tilde{Z}_{k+1} = -C_{k+1} \tilde{Z}_{k-1}$
$X_{k+1} = X_k$
$R_{k+1} = R_k$

5. **for** $i = 1, \cdots, m_k$ **do**
$U_k = AT_k$
$\beta = d_{m_k-i}^{(k)} / b_0^{(k)}$
$X_{k+1} = X_{k+1} + \beta T_k$
$R_{k+1} = R_{k+1} - \beta U_k$
$\gamma = - < Y_k, U_k >_F / b_0^{(k)}$
$T_k = U_k + \gamma Z_k$
if $i \neq 1$ **then**
$V_k = A^T \tilde{T}_k$
end if
$\tilde{T}_k = V_k + \gamma \tilde{Z}_k$
end for

6. $Z_{k+1} = Z_{k+1} + T_k$
$\tilde{Z}_{k+1} = \tilde{Z}_{k+1} + \tilde{T}_k$
$n_{k+1} = n_k + m_k$
$k = k + 1$
end while

4. The global BiCGSTAB algorithm.

4.1. The global BiCGSTAB algorithm. We have seen that at step k the residual R_k^{gb} and the matrix direction P_k^{gb} produced by Gl-BCG satisfy

$$R_k^{gb} = R_{k-1}^{gb} - \alpha_k \, A P_{k-1}^{gb}, \tag{4.1}$$

and

$$P_k^{gb} = R_k^{gb} + \beta_k \, P_{k-1}^{gb}. \tag{4.2}$$

The k-th residual of global BiCGSTAB is defined by

$$R_k = (I - \omega_k A) \, S_k$$

where

$$S_k = (I - \omega_{k-1}A)\dots(I-\omega_1 A)\, R_k^{gb}. \tag{4.3}$$

The parameter ω_k is selected to minimize the F-norm of R_k, so we have

$$\omega_k = \frac{< S_k, AS_k >_F}{< AS_k, AS_k >_F}.$$

Using (4.1) and (4.3) we get

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

where

$$P_{k-1} = (I - \omega_{k-1}A)\dots(I-\omega_1 A)\, P_{k-1}^{gb}.$$

Now since R_k^{gb} $(k \geq 1)$ is F-orthogonal to the matrix Krylov subspace $\mathcal{K}_k(A^T, \tilde{R}_0)$ it follows from (4.3) that,

$$< \tilde{R}_0, S_k >_F = 0; \quad k \geq 1.$$

Using this orthogonality, we get

$$\alpha_k = \frac{< \tilde{R}_0, R_{k-1} >_F}{< \tilde{R}_0, AP_{k-1} >_F}.$$

On the other hand, the global BiCGSTAB direction P_k is given by

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

which can be written as

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

as well as

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

where

$$\begin{aligned} Q_k &= S_k + \beta_k P_{k-1} \\ &= (I - \omega_{k-1}A)\dots(I-\omega_1 A)\, P_k^{gb}. \end{aligned}$$

We now have to compute β_k by using the fact that P_k^{gb} is F-orthogonal to the subspace $\mathcal{K}_k(A^T, \tilde{R}_0)$. It follows that,

$$< \tilde{R}_0, A\, Q_k >_F = 0; \quad k \geq 1.$$

Therefore

$$\beta_k = -\frac{< \tilde{R}_0, AS_k >_F}{< \tilde{R}_0, AP_{k-1} >_F}.$$

The global BiCGSTAB algorithm is given as follows :

ALGORITHM 5 **The Gl-BiCGSTAB algorithm**

Compute $R_0 = B - AX_0$, where X_0 is an initial approximate solution; $P_0 = R_0$ and $\tilde{R}_0$ arbitrary,

for $k = 1, 2, \ldots$

$V_{k-1} = AP_{k-1}$,

$S_k = R_{k-1} - \alpha_k V_{k-1}, \alpha_k = \dfrac{< \tilde{R}_0, R_{k-1} >_F}{< \tilde{R}_0, V_{k-1} >_F}$,

$T_k = AS_k$,

$X_k = X_{k-1} + \alpha_k P_{k-1} + \omega_k S_k, \omega_k = \dfrac{< T_k, S_k >_F}{< T_k, T_k >_F}$,

$R_k = S_k - \omega_k T_k$,

$P_k = R_k + \beta_k (P_{k-1} - \omega_k V_{k-1})$, with $\beta_k = -\dfrac{< \tilde{R}_0, T_k >_F}{< \tilde{R}_0, V_{k-1} >_F}$,

end.

When $s = 1$ the algorithm reduces to BiCGSTAB of Van der Vorst [22]. We note that global methods do not suffer from dependence of vectors during the iterations until a matrix invariant subspace is obtained (no need for deflation). However a break-down may occur if $< \tilde{R}_0, V_{k-1} >_F = 0$.

4.2. A look-ahead global BiCGSTAB algorithm. The k-th residual produced by the global BiCGSTAB algorithm is expressed as

$$R_k = \mathcal{Q}_k(A)\mathcal{P}_k(A)R_0,$$

where $\mathcal{Q}_k$ is a polynomial satisfying the following recurrence relation

$$\mathcal{Q}_k(t) = (1 - w_k t)\mathcal{Q}_{k-1}(t), \quad \mathcal{Q}_0(t) = 1,$$

and w_k is chosen so that $< R_k, R_k >_F = \|R_k\|_F^2$ is minimized. Let

$$\tilde{\mathcal{P}}_k^{(1)}(t) = (-1)^k \frac{H_k^{(0)}}{H_k^{(1)}} \mathcal{P}_k^{(1)}(t).$$

$\mathcal{P}_k^{(1)}$ is a monic formal orthogonal polynomial with respect to $c^{(1)}$. The polynomials $\mathcal{P}_k$ and $\mathcal{P}_k^{(1)}$ satisfy the recurrence relations

$$\begin{aligned} \mathcal{P}_{k+1}(t) &= \mathcal{P}_k(t) - \alpha_{k+1} t \tilde{\mathcal{P}}_k^{(1)}(t), \\ \tilde{\mathcal{P}}_{k+1}^{(1)}(t) &= \mathcal{P}_{k+1}(t) + \beta_{k+1} \tilde{\mathcal{P}}_k^{(1)}(t), \end{aligned} \tag{4.4}$$

with $\mathcal{P}_0(t) = \tilde{\mathcal{P}}_0(t) = 1$.

If the Hankel determinant $H_k^{(1)}$ vanishes, then the polynomials $\mathcal{P}_k$ and $\tilde{\mathcal{P}}_k^{(1)}$ do not exist and we have a true breakdown. This problem can be cured by jumping over the nonexisting polynomials and by considering only the regular ones. If $H_k^{(1)} \neq 0$ and the two polynomials are not of degree k exactly ($H_k^{(0)} = 0$) then we have a ghost breakdown. This kind of breakdown is not treated in this paper.

In the sequel, we assume that $H_k^{(0)} \neq 0$. The k-th regular polynomials will be denoted by $\mathcal{P}_k$ and $\mathcal{P}_k^{(1)}$ with degree equal to n_k. The next regular polynomials $\mathcal{P}_{k+1}$ and $\tilde{\mathcal{P}}_{k+1}^{(1)}$ have degree $n_{k+1} = n_k + m_k$ where m_k is the jump in the degrees between two successive regular polynomials; see [1] and [3].

The jump m_k is defined by the conditions

$$\begin{aligned} c^{(1)}(t^i\tilde{\mathcal{P}}_k^{(1)}) &= 0 \quad \text{for } i = 0,\ldots,n_k+m_k-2,\\ c^{(1)}(t^{n_k+m_k-1}\tilde{\mathcal{P}}_k^{(1)}) &\neq 0. \end{aligned}$$

Therefore the polynomials $\mathcal{P}_{k+1}$ and $\tilde{\mathcal{P}}_{k+1}^{(1)}$ are computed by the following recurrence relations

$$\begin{aligned} \mathcal{P}_{k+1}(t) &= \mathcal{P}_k(t) - t\eta_k(t)\tilde{\mathcal{P}}_k^{(1)}(t),\\ \tilde{\mathcal{P}}_{k+1}^{(1)}(t) &= \mathcal{P}_{k+1}(t) + \beta_{k+1}\tilde{\mathcal{P}}_k^{(1)}(t), \end{aligned} \tag{4.5}$$

where $\mathcal{P}_0(t) = \tilde{\mathcal{P}}_0(t) = 1$ and η_k is a polynomial of degree $m_k - 1$ at most. The auxiliary polynomial $\mathcal{Q}_k$ is of degree at most n_k and satisfies the recurrence

$$\mathcal{Q}_{k+1}(t) = (1 + w_1^{(k)}t + w_2^{(k)}t^2 + \ldots + w_{m_k}^{(k)}t^{m_k})\mathcal{Q}_k(t). \tag{4.6}$$

The coefficients $({w_i}^{(k)})_{1\le i\le m_k}$ are chosen so that $< R_{k+1}, R_{k+1} >_F$ is minimized.

The jump m_k is determined from the relations

$$\begin{aligned} c^{(1)}(t^i\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)}) &= 0 \quad \text{for } i = 0,\ldots,m_k-2,\\ c^{(1)}(t^{m_k-1}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)}) &\neq 0. \end{aligned} \tag{4.7}$$

Let $\gamma_i^{(k)}, i = 0,\ldots,m_k-1$ be the coefficients of the polynomial η_k:

$$\eta_k(t) = \sum_{i=0}^{m_k-1}\gamma_i^{(k)}t^i$$

and set

$$\begin{aligned} b_i^{(k)} &= c^{(1)}(t^{m_k-1+i}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)}(t)) \qquad &&\text{for } i = 0,\ldots,m_k-1,\\ d_i^{(k)} &= c(t^i\mathcal{Q}_k\mathcal{P}_k(t)) &&\text{for } i = 0,\ldots,m_k-1. \end{aligned}$$

In (4.5), multiplying $\mathcal{P}_{k+1}$ by $[t^i\mathcal{Q}_k]$, $i = 0,\ldots,m_k-1$, applying c and using the orthogonality relations, we get

$$\begin{bmatrix} b_0^{(k)} & & & & & \\ b_1^{(k)} & b_0^{(k)} & & & 0 & \\ \vdots & b_1^{(k)} & \ddots & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ b_{m_k-2}^{(k)} & b_{m_k-1}^{(k)} & \ldots & \ddots & b_0^{(k)} & \\ b_{m_k-1}^{(k)} & b_{m_k-2}^{(k)} & \ldots & \ldots & b_1^{(k)} & b_0^{(k)} \end{bmatrix} \begin{bmatrix} \gamma_{m_k-1}^{(k)} \\ \gamma_{m_k-2}^{(k)} \\ \vdots \\ \vdots \\ \gamma_1^{(k)} \\ \gamma_0^{(k)} \end{bmatrix} = \begin{bmatrix} d_0^{(k)} \\ d_1^{(k)} \\ \vdots \\ \vdots \\ d_{m_k-2}^{(k)} \\ d_{m_k-1}^{(k)} \end{bmatrix}. \tag{4.8}$$

Note that, since $c^{(1)}(t^{m_k-1}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)}) = b_0^{(k)} \neq 0$, the matrix of the preceding linear system is nonsingular. Multiplying equation (4.5) by $t^{m_k-1}\mathcal{Q}_k$ and applying $c^{(1)}$, the coefficient β_{k+1}

is given by

$$\begin{aligned}\beta_{k+1} &= -\frac{c(t^{m_k}\mathcal{Q}_k\mathcal{P}_{k+1})}{c^{(1)}(t^{m_k-1}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)})}\\ &= -\frac{c(\mathcal{Q}_{k+1}\mathcal{P}_{k+1})}{w_{m_k}^{(k)}c^{(1)}(t^{m_k-1}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)})}.\end{aligned}$$

It follows from

$$\gamma_{m_k-1} = \frac{c(\mathcal{Q}_k\mathcal{P}_k)}{c^{(1)}(t^{m_k-1}\mathcal{Q}_k\tilde{\mathcal{P}}_k^{(1)})} = \frac{d_0^{(k)}}{b_0^{(k)}},$$

that

$$\beta_{k+1} = -\frac{\gamma_{m_k-1}}{w_{m_k}^{(k)}}\frac{d_0^{(k+1)}}{d_0^{(k)}}. \tag{4.9}$$

Let Z_k and S_k be defined as follows :

$$\begin{aligned}Z_k &= \mathcal{Q}_k(A)\tilde{\mathcal{P}}_k^{(1)}(A)R_0,\\ S_k &= \mathcal{Q}_{k-1}(A)\mathcal{P}_k(A)R_0.\end{aligned}$$

Using the relations (4.5) and (4.6), we obtain

$$\begin{aligned}S_{k+1} &= R_k - A\eta_k(A)Z_k,\\ R_{k+1} &= S_{k+1} + w_1^{(k)}AS_{k+1} + w_2^{(k)}A^2S_{k+1} + \ldots + w_{m_k}^{(k)}A^{m_k}S_{k+1},\\ Z_{k+1} &= R_{k+1} + \beta_{k+1}(Z_k + w_1^{(k)}AZ_k + w_2^{(k)}A^2Z_k + \ldots + w_{m_k}^{(k)}A^{m_k}Z_k).\end{aligned}$$

The approximations X_k are then computed according to

$$X_{k+1} = X_k + \eta_k(A)Z_k - (w_1^{(k)}S_{k+1} + w_2^{(k)}AS_{k+1} + \ldots + w_{m_k}^{(k)}A^{m_k-1}S_{k+1}).$$

Since the coefficients $(w_i^{(k)})_{1\le i\le m_k}$ are chosen so that $< R_{k+1}, R_{k+1} >_F$ is minimized, they are obtained by solving the following $m_k \times m_k$ linear system:

$$\begin{aligned}w_1^{(k)} < A^iS_{k+1}, AS_{k+1} >_F + \ldots + w_{m_k}^{(k)} < A^iS_{k+1}, A^{m_k}S_{k+1} >_F&\\ = - < A^iS_{k+1}, S_{k+1} >_F;\ \ i = 1,\ldots,m_k.&\end{aligned}$$

We note that since $c(t^i) =< \tilde{R}_0, A^iR_0 >_F$, the coefficients $(b_i^{(k)})$ and $(d_i^{(k)})$ are given by

$$\begin{aligned}b_i^{(k)} &= < \tilde{R}_0, A^{i+m_k}Z_k >_F, \quad i = 0,\ldots,m_k-1,\\ d_i^{(k)} &= < \tilde{R}_0, A^iR_k >_F, \quad i = 0,\ldots,m_k-1.\end{aligned}$$

The length of the jump m_k is determined by the following relations

$$< \tilde{R}_0, A^iZ_k >_F = 0 \quad \text{for } i = 1,\ldots,m_k-1, \text{ and } \quad < \tilde{R}_0, A^{m_k}Z_k >_F \ne 0.$$

The look-ahead global BiCGSTAB algorithm is summarized as follows:

ALGORITHM 6 **The Look-ahead global BiCGSTAB** $(A, B, X_0, \tilde{R}_0)$

ETNA
Kent State University
etna@mcs.kent.edu

1. **initialization**
   $Z_{-1} = 0$
   $R_0 = B - AX_0$
   $Z_0 = R_0$
   $d_0^{(0)} =< \tilde{R}_0, R_0 >_F$
   $n_0 = 0$
   $k = 0$

2. **while** $R_k \neq 0$ **do**
   if $d_0^{(0)} = 0$ **then stop.**
   $Z_{k,0} = Z_k$
   $Z_{k,1} = AZ_{k,0}$
   $m_k = 1$
   $b_0^{(k)} =< \tilde{R}_0, Z_{k,1} >_F$

3. **while** $b_0^{(k)} = 0$ **do**
   $m_k = m_k + 1$
   $R_{k,m_k-1} = AR_{k,m_{k-2}}$
   $d_{m_k-1}^{(k)} =< \tilde{R}_0, R_{k,m_k-1} >_F$
   $Z_{k,m_k} = AZ_{k,m_{k-1}}$
   $b_0^{(k)} =< \tilde{R}_0, Z_{k,m_k} >_F$
   end while

4. $\gamma_{m_k-1}^{(k)} = d_0^{(k)}/b_0^{(k)}$ $\quad U_k = Z_{k,m_k}$
   for $i = 1, \cdots, m_k - 1$ **do**
   $U_k = AU_k$
   $b_i^{(k)} =< \tilde{R}_0, U_k >_F$
   compute $\gamma_{m_k-i-1}^{(k)}$
   end for

5. $S_{k+1} = R_k - \gamma_0^{(k)} Z_{k,1} \gamma_1^{(k)} Z_{k,2} - \cdots - \gamma_{m_k-1}^{(k)} Z_{k,m_k}$

6. construct the matrices $M_k = [AS_{k+1}, \ldots, A^{m_k} S_{k+1}]^T [AS_{k+1}, \ldots, A^{m_k} S_{k+1}]$
   and $N_k = [AS_{k+1}, \ldots, A^{m_k} S_{k+1}]^T S_{k+1}$

7. **solve** $M_k w_k = -N_k$ where $w_k = [w_1^{(k)}, w_2^{(k)}, \ldots, w_{m_k}^{(k)}]^T$

8. $X_{k+1} = X_k + \gamma_0^{(k)} Z_{k,0} + \cdots + \gamma_{m_k-1}^{(k)} Z_{k,m_k-1} - w_1^{(k)} S_{k+1} - \ldots - w_{m_k}^{(k)} A^{m_k-1} S_{k+1}$
   $R_{k+1} = S_{k+1} + w_1^{(k)} AS_{k+1} + \ldots + w_{m_k}^{(k)} A^{m_k} S_{k+1}$

9. $d_0^{(k+1)} =< \tilde{R}_0, R_{k+1} >_F$
   $\beta_{k+1} = -\dfrac{d_0^{(k+1)} \gamma_{m_k-1}^{(k)}}{d_0^{(k)} w_{m_k}^{(k)}}$
   $Z_{k+1} = R_{k+1} + \beta_{k+1}(Z_{k,0} + w_1^{(k)} Z_{k,1} + \cdots + w_{m_k}^{(k)} Z_{k,m_k})$

10. $n_{k+1} = n_k + m_k$
    $k = k + 1$

end while

5. 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.

Example 1: We compared the performance of the global BCG, global BiCGSTAB and the block BCG algorithms. We used the matrices from the Harwell-Boeing collection: A_1=PDE2961

ETNA
Kent State University
etna@mcs.kent.edu

($N = 2961$) and A_2=Sherman4 ($N = 1024$). The number of nonzero entries of A_1 and A_2 are $nnz(A_1) = 14585$ and $nnz(A_2) = 3786$. The tests were stopped as soon as $max(\| R_k^{(j)} \|_2 / \| R_0^{(j)} \|_2) \leq 10^{-7}$, for $j = 1, \ldots, s$.
The initial guess X_0 was taken to be zero. We set $B = rand(N, s)$, where the Matlab function $rand$ creates an $N \times s$ random matrix with entries uniformly distributed in [0,1].

In Table 5.1, we list the CPU-time (in seconds) obtained with the three algorithms. In parentheses, we give the ratio $s \cdot t(1)/t(s)$, where $t(s)$ is the CPU-time for the global or the block method and $t(1)$ is the CPU-time obtained when applying the corresponding method for one linear system with one right-hand side. Note that the time obtained with one right-hand side solver depends on which right-hand side was used. $t(1)$ was obtained by dividing the time needed for the s right-hand sides by s. We note that a global method is effective if the indicator $s \cdot t(1)/t(s)$ is greater than 1. The maximum number of 500 iterations was allowed for all the algorithms. As mentioned in [18], we used $\tilde{R}_0 = AB$ for the Bl-BCG algorithm.

TABLE 5.1
Runtimes to convergence for Gl-BCG, Gl-BiCGSTAB and Bl-BCG. Matrices A_1 and A_2; $s = 10$ and $s = 20$.

Matrix	s	Gl-BCG	Gl-BiCGSTAB	Bl-BCG
PDE2961 (N=2961)	10	98 (1.39)	40 (1.43)	- -
	20	201 (1.43)	81 (1.45)	-
SHERMAN4 (N=1140)	10	17 (1.37)	10 (1.34)	- -
	20	36 (1.41)	19 (1.39)	24 (1.35)

Table 5.1 shows that Gl-BiCGSTAB returns the best results. Note that, for Sherman4 and $s = 20$, Bl-BCG performs better than Gl-BCG. For the matrix PDE2961, Bl-BCG failed to converge and this was also the case for Sherman4 with $s = 10$.

Example 2: For this experiment we consider the matrix

$$A = \begin{bmatrix} a & 1 & & & & & \\ -1 & a & 1 & & & & \\ & -1 & a & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -1 & a & 1 \\ & & & & & -1 & a \end{bmatrix}$$

with $B = I_{N,s}$, $X_0 = rand(N, s)$, $\tilde{R}_0 = I_{N,s}$, $N = 200$, $s = 6$ and $a = 0$. In Figure 5.1, we plotted the log10 of the Frobenius norm of the residual versus the iterations. As shown in this figure, the Gl-BCG (dashed line) does not converge. Setting $\epsilon = 10^{-8}$, the global Hmrz-stab(solide line) makes jumps of length $m_k = 2$ and we obtain $n_{100} = 200$ with $\|R_{n_{100}}\|_F = 3.14\ 10^{-11}$.
Example 3: For this experiment, the matrix A is the same as in Example 2 with $N = 1000$, $s = 6$, $a = 0$, $X_0 = 0_{N,s}$ and $\tilde{R}_0 = R_0$. Figure 5.2 shows the results obtained with the

ETNA
Kent State University
etna@mcs.kent.edu

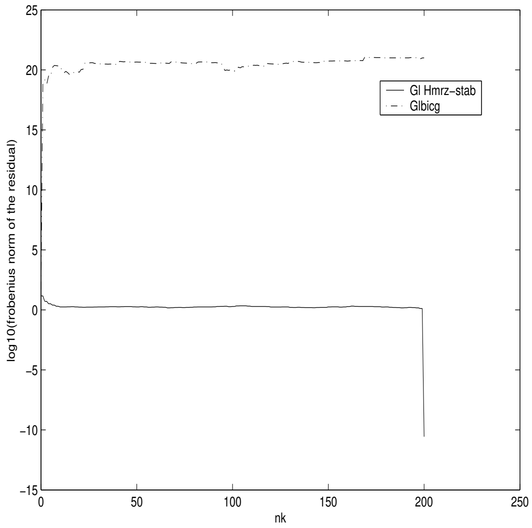

FIG. 5.1. $N = 200$, $s = 6$, $\epsilon = 10^{-8}$

Gl-BICGSTAB (solid line) and the look-ahead Gl-BiCGSTAB (dashed line) algorithms. In this figure we plotted the Frobenius norm (in the logarithmic scale) of the residuals vesus the iterations. With $\epsilon = 10^{-10}$ many jumps are detected in the look-ahead GL-BiCGSTAB algorithm. The first jump is obtained at iteration $n_{16} = 16$ ($n_{17} = 18$), the second one is detected at iteration $n_{23} = 24$ ($n_{24} = 26$). The last jump is of length $m_k = 11$ and is detected at iteration $n_{36} = 51$ ($n_{37} = 62$).

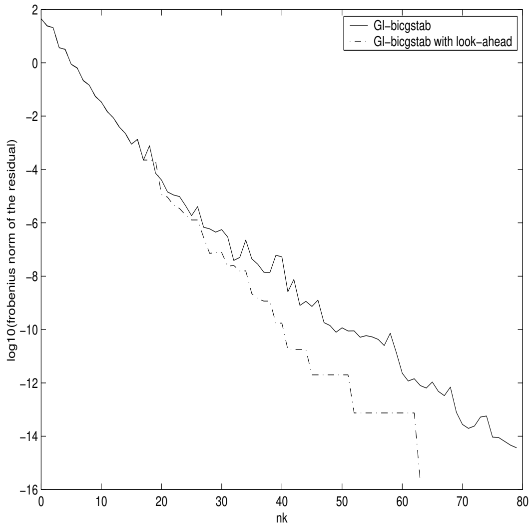

FIG. 5.2. $N = 1000$, $s = 6$ *and* $\epsilon = 10^{-10}$

Example 4: For this last experiment, the matrix $A = PDE2961$ is taken from the Harwell Boeing collection ($N = 2961$). The nonzero entries of A are $nnz(A_4) = 14585$. We used

ETNA
Kent State University
etna@mcs.kent.edu

$s = 10$, $B = I_{N,s}$, $X_0 = 0_{N,s}$ and $\tilde{R}_0 = R_0$. Figure 5.3 reports on the results obtained with the Gl-BICGSTAB (solid line) and the look-ahead Gl-BiCGSTAB (dashed line) algorithms. We plotted the Frobenius norm of the residuals versus the iterations.

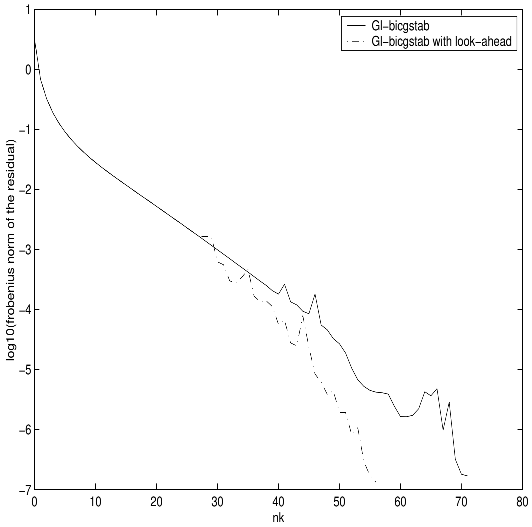

FIG. 5.3. $N = 2961$, $s = 10$, $\epsilon = 10^{-10}$

REFERENCES

[1] C. BREZINSKI, M. REDIVO ZAGLIA AND H. SADOK, *Avoiding breakdown and near-breakdown in Lanczos-type algorithms*, Numerical Algorithms, 1 (1991), pp. 261–284.

[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 BiCGSTAB 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] A. DRAUX, *Polynomes Orthogonaux Formels.(French) [Formal orthogonal polynomials] Applications.*, LNM 974. Springer, Berlin Heidelberg New York, 1983.

[6] A.EL GUENNOUNI, K. JBILOU AND H. SADOK *A block version of BiCGSTAB for linear systems with multiple right-hand sides*,Elec. Trans. Numer. Anal., 16 (2003), pp. 129–142. http://etna.mcs.kent.edu/vol.16.2003/pp129-142.dir/pp129-142.pdf.

[7] 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.

[8] R. FREUND AND M. MALHOTRA, *A Block-QMR Algorithm for Non-Hermitian Linear Systems with multiple Right-Hand Sides*, Linear Alg. Appl., 254 (1997), pp. 119–157.

[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] 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*, Comp. Meth. Appl. Mech. Eng., 146 (1997), pp. 173–196.

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

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

[13] Y. SAAD AND M.H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856–869.

[14] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, PWS publishing, New York, 1995.
[15] M. SADKANE, *Block Arnoldi and Davidson methods for unsymmetric large eigenvalue problems*, Numer. Math., 64 (1993), pp. 687–706.
[16] W. SCHONAUER, *Scientific Computing on Vector Computers*, North-Holland, Amsterdam, New York, Tokyo, 1997.
[17] V. SIMONCINI, *A stabilized QMR version of block BICG*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 419–434.
[18] V. SIMONCINI AND E. GALLOPOULOS, *An Iterative Method for Nonsymmetric Systems with Multiple Right-hand Sides*, SIAM J. Sci. Comp., 16 (1995), pp. 917–933.
[19] C. SMITH, A. PETERSON AND R. MITTRA, *A Conjugate Gradient Algorithm for Treatement of Multiple Incident Electromagnetic Fields*, IEEE Transactions On Antennas and Propagation, 37 (1989), pp. 1490–1493.
[20] B. VITAL, *Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multiprocesseur*, Ph.D. thesis, Univérsité de Rennes, Rennes, France, 1990.
[21] H.A. 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. Comp. Appl. Math., 18 (1987), pp. 249–263.
[22] H.A. VAN DER VORST, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems*, SIAM J. Sci. Stat. Comp., 12 (1992), pp. 631–644.