ELSEVIER Journal of Computational and Applied Mathematics 67 (1996) 15–41

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Nested Krylov methods based on GCR

E. de Sturler*

Faculty of Technical Mathematics and Informatics, Delft University of Technology, Mekelweg 4, Delft, Netherlands

Received 7 June 1993; revised 15 November 1994

Abstract

Recently the GMRESR method for the solution of linear systems of equations has been introduced by Vuik and Van der Vorst (1991). Similar methods have been proposed by Axelsson and Vassilevski (1991) and Saad (1993) (FGMRES[1]). GMRESR involves an outer and an inner method. The outer method is GCR, which is used to compute the optimal approximation over a given set of search vectors in the sense that the residual is minimized. The inner method is GMRES, which computes a new search vector by approximately solving the residual equation. This search vector is then used by the outer algorithm to compute a new approximation. However, the optimality of the approximation over the space of search vectors is ignored in the inner GMRES algorithm. This leads to suboptimal corrections to the solution in the outer algorithm. Therefore, we propose to preserve the orthogonality relations of GCR in the inner GMRES algorithm. This gives optimal corrections to the solution and also leads to solving the residual equation in a smaller subspace and with an "improved" operator, which should also lead to faster convergence. However, this involves using Krylov methods with a singular, nonsymmetric operator. We will discuss some important properties of this. We will show by experiments that in terms of matrix–vector products, this modification (almost) always leads to better convergence. Because we do more orthogonalizations, it does not always give an improved performance in time. This depends on the costs of the matrix–vector products relative to the costs of the orthogonalizations. Of course, we can also use methods other than GMRES as the inner method. Methods with short recurrences like BiCGSTAB seem especially interesting. The experimental results indicate that, especially for such methods, it is advantageous to preserve the orthogonality in the inner method.

Keywords: Nonsymmetric linear systems; Iterative solvers; Krylov subspace; GCR; GMRES; GMRESR; BiCGSTAB

AMS classification: 65F10

1. Introduction

For the solution of systems of linear equations the so-called Krylov subspace methods are very popular. However, for general matrices no Krylov method can satisfy a global optimality requirement

* Correspondence address: Interdisciplinary Project Center for Supercomputing, Swiss Federal Institute of Technology, IPS-ETHZ, CH-8092 Zürich, Switzerland. E-mail: sturler@ips.id.ethz.ch.

[1] Since FGMRES and GMRESR are very similar, the ideas presented will be relevant for FGMRES as well.

SSDI 0377-0427(94)00123-5

and have short recurrences [7]. Therefore, either restarted or truncated versions of optimal methods, such as GMRES [14], may be used. Alternatively, one may use methods with short recurrences, which do not satisfy a global optimality requirement, such as BiCG [8], BiCGSTAB [17], BiCGSTAB(l) [15], CGS [16] or QMR [10], which satisfies some quasi-minimization requirement. Recently Vuik and Van der Vorst introduced a new type of method, GMRESR [18], which is a nested GMRES method. Among various other nice properties, GMRESR is flexible in the sense that a global minimization over some specific part of the Krylov space is done, and the dimension of this space and the vectors that span it are controlled by the algorithm. The GMRESR algorithm (with GMRES as inner method) is given in Fig. 1.

The GMRESR algorithm consists of GCR [6], Fig. 2, as the outer algorithm and m steps of GMRES as an inner method for the computation of $\mathscr{P}_{m,k}(A)r_{k-1}$. The inner GMRES method computes a new search vector by approximately solving the residual equation, $Ae_k = r_k$, and then the outer GCR algorithm minimizes the residual over the new search vector and all previously kept search vectors u_i. The algorithm can be explained as follows.

Suppose we are given the system of equations $Ax = b$, where A is a real, nonsingular, linear operator and b is a given right-hand side. Furthermore, we have the two matrices

$$U_k = (u_1 \; u_2 \; \ldots \; u_k) \quad \text{and} \quad C_k = AU_k \tag{1}$$

with the property that $C_k^{\mathrm{T}} C_k = I_k$. Consider the following minimization problem:

$$\min_{x \in \mathrm{range}(U_k)} \|b - Ax\|_2. \tag{2}$$

The solution of this minimization problem is given by

$$x = U_k C_k^{\mathrm{T}} b, \tag{3}$$

and $r = b - Ax$ satisfies

$$r = b - C_k C_k^{\mathrm{T}} b, \quad r \perp \mathrm{range}(C_k). \tag{4}$$

GMRESR:

1. Select x_0, m, tol;
   $r_0 = b - Ax_0$, $k = 0$;
2. **while** $\|r_k\|_2 >$ tol **do**
   $k = k + 1$;
   $u_k = \mathscr{P}_{m,k}(A)r_{k-1}$;
   $c_k = Au_k$;
   for $i = 1, \ldots, k-1$ **do**
   $\alpha_i = c_i^{\mathrm{T}} c_k$;
   $c_k = c_k - \alpha_i c_i$;
   $u_k = u_k - \alpha_i u_i$;
   $u_k = u_k / \|c_k\|_2$;
   $c_k = c_k / \|c_k\|_2$;
   $x_k = x_{k-1} + (c_k^{\mathrm{T}} r_{k-1}) u_k$;
   $r_k = r_{k-1} - (c_k^{\mathrm{T}} r_{k-1}) c_k$;

$\mathscr{P}_{m,k}(A)r_k$ indicates the GMRES polynomial that is implicitly constructed in m steps of GMRES.

Fig. 1. The GMRESR algorithm.

GCR:

1. Select x_0, tol;
   $r_0 = b - Ax_0$, $k = 0$;
2. **while** $\|r_k\|_2 >$ tol **do**
   $k = k + 1$;
   $u_k = r_{k-1}$;
   $c_k = Au_k$;
   for $i = 1, k-1$ **do**
   $\alpha_i = c_i^{\mathrm{T}} c_k$;
   $c_k = c_k - \alpha_i c_i$;
   $u_k = u_k - \alpha_i u_i$;
   $u_k = u_k / \|c_k\|_2$;
   $c_k = c_k / \|c_k\|_2$;
   $x_k = x_{k-1} + (c_k^{\mathrm{T}} r_{k-1}) u_k$;
   $r_k = r_{k-1} - (c_k^{\mathrm{T}} r_{k-1}) c_k$;

Fig. 2. The GCR algorithm.

We have constructed the inverse of A over the subspace range(C_k). This inverse is given by

$$A^{-1}C_kC_k^{\mathrm{T}} = U_kC_k^{\mathrm{T}}. \tag{5}$$

This principle underlies the GCR method, shown in Fig. 2, where for the preconditioned case A denotes the explicitly preconditioned operator. In GCR the matrices U_k and C_k are constructed such that the range(U_k) equals the Krylov space $K^k(A, r_0) = \mathrm{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\}$, where $r_0 = b - Ax_0$ and x_0 is an initial approximation to the solution. Provided GCR does not break down, it is a finite method and at step k it solves the minimization problem (2), with range(U_k) = $K^k(A, r_0)$. It is obvious from (1)–(2) that in the step $u_{k+1} = r_k$ (in GCR) we can replace r_k by any other vector and the algorithm still works in the sense that it solves the minimization problem (2). In this case range(U_k) = $K^k(A, r_0)$ no longer holds. In general the better the choice of u_{k+1} the faster the convergence. The optimal choice, of course, is $u_{k+1} = e_k$, where e_k is the error in x_k. Therefore, the key idea is to choose a suitable approximation to the error e_k. In order to find approximations to e_k, we use the relation

$$Ae_k = r_k, \tag{6}$$

and *any* method which gives an approximate solution to this equation can be used to find good choices for u_{k+1}. We may also vary such methods from one step to the next and we can regard them as preconditioners. These observations were made in [18] and lead to the formulation of the GMRESR method, shown in Fig. 1, and its variants.

However, since we already have an optimal $x_k \in \mathrm{range}(U_k) \Leftrightarrow (b - Ax_k) \perp \mathrm{range}(AU_k)$, we need an approximation u_{k+1} to e_k, such that $c_{k+1} = Au_{k+1}$ is also orthogonal to range(C_k). This is done explicitly by the orthogonalization loop in the outer (GCR) algorithm. Because this is not taken into account in the inner method, the "wrong" minimization problem is solved, since it also considers spurious corrections.

Consider step $k + 1$ in GMRESR. We have computed the matrices U_k, C_k and the vectors x_k and r_k. Then, in the inner GMRES loop (after m iterations) we solve the minimization problem

$$\min_{y \in \mathbb{R}^m} \| r_k - AV_m y \|_2, \tag{7}$$

where V_m is an orthogonal matrix such that range(V_m) = $K^m(A, r_k)$ (see [14]), and in the outer loop we set

$$\begin{aligned} u_{k+1} &= V_m y - U_k C_k^{\mathrm{T}} AV_m y / \| (I - C_k C_k^{\mathrm{T}}) AV_m y \|_2, \\ c_{k+1} &= (I - C_k C_k^{\mathrm{T}}) AV_m y / \| (I - C_k C_k^{\mathrm{T}}) AV_m y \|_2, \\ r_{k+1} &= r_k - (r_k^{\mathrm{T}} c_{k+1}) c_{k+1}. \end{aligned} \tag{8}$$

It is obvious that for optimality we should have solved the minimization problem

$$\min_{y \in \mathbb{R}^m} \| r_k - (I - C_k C_k^{\mathrm{T}}) AV_m y \|_2, \tag{9}$$

which is equivalent to

$$\min_{y \in \mathbb{R}^m} \| (I - C_k C_k^{\mathrm{T}}) [r_k - AV_m y] \|_2, \tag{10}$$

since $r_k \perp \text{range}(C_k)$. The results of the minimization problems (7) and (10) are equal only if $\text{range}(AV_m) \perp \text{range}(C_k)$, which is generally not the case. Also in GMRESR we search in principle the entire Krylov space $K(A, r_k) = \text{span}\{r_k, Ar_k, A^2 r_k, \dots\}$ for an approximation $u_{k+1} \approx e_k$, whereas

$$e_k \in K(A, r_k) \cap A^{-1}\text{range}(C_k)^{\perp}, \tag{11}$$

and therefore

$$r_k \in K(A, Ar_k) \cap \text{range}(C_k)^{\perp}. \tag{12}$$

Another disadvantage of GMRESR is that the inner loop is essentially a restarted GMRES with the same operator A. It therefore also displays some of the problems of restarted GMRES; most notably, it can have the tendency to stagnate in the inner loop (see Section 4). From this we infer that it is favourable to preserve the GCR orthogonality relations also in the inner (GMRES) method, at least for the speed of convergence, as we show in Theorem 2.2 in the next section. From (8), (9), (11) and (12) we come to the idea of using $(I - C_k C_k^{\mathrm{T}})A$ as the operator in the Krylov method in the inner loop. This would generate corrections to the residual $c_{k+1} \in K(A, b) \cap \text{range}(C_k)^{\perp}$. But it will search for updates to the approximation $u_{k+1} \in \{r_k, (I - C_k C_k^{\mathrm{T}})Ar_k, \dots\} \subset \text{range}(C_k)^{\perp}$, whereas in general $e_k \notin \text{range}(C_k)^{\perp}$. We should note, however, that corrections to the residual c_{k+1} correspond to corrections to the approximation

$$u_{k+1} = A^{-1}c_{k+1} \in K(A, r_k) \cap A^{-1}\text{range}(C_k)^{\perp},$$

which corresponds to (11). This may seem like a problem, since we have introduced the operator A^{-1}. But over the space $\text{span}\{(I - C_k C_k^{\mathrm{T}})Ar_k, [(I - C_k C_k^{\mathrm{T}})A]^2 r_k, \dots\}$ the inverse of A is known, because we know the inverse of A over $\text{range}(C_k)$ (5):

$$\begin{aligned} A^{-1}(I - C_k C_k^{\mathrm{T}})A &= A^{-1}A - A^{-1}C_k C_k^{\mathrm{T}}A \\ &= I - U_k C_k^{\mathrm{T}}A. \end{aligned} \tag{13}$$

In Krylov methods the solution itself is not needed to continue the iteration, only the residual. Therefore, we can use a Krylov method for the residual and find the corresponding updates to the solution vector using (13). This leads to an extension of the GMRESR method, which has an improved performance for many problems. In fact, this leads to a nested method, where the "inner" Krylov method computes an approximation to the error in the "outer" Krylov method and the information acquired in the "outer" method is used to speed up the convergence in the "inner" method.

In this article we will consider as inner methods GMRES and BiCGSTAB. BiCGSTAB has the advantage that its cost per iteration is low, so that we have a relatively cheap method to compute approximations to the error. Using GMRES in the inner method has the following attractive features. First, at the end of the inner iteration the error is minimized over the space spanned by the search vectors of both the outer method and the inner method, as shown in Theorem 2.2. Second, using a few parameters and truncation, we can control the sizes of the Krylov subspaces in both the inner and the outer iteration and hence optimize the length of recurrences and memory requirements versus convergence speed: see [4, 9].

In the next section we will discuss the implications of the orthogonalizations in the inner method more formally. We will prove that it leads to an optimal approximation over the space spanned by both the outer and the inner iteration vectors. It also introduces a potential problem: the possibility of breakdown in the generation of the Krylov space in the inner iteration, since we iterate with a singular operator. We will show, however, that such a breakdown is rare, because it can only happen after a specific (generally large) number of iterations. Furthermore, we will also show how to remedy such a breakdown. In most of the following discussion we will consider m (not fixed) steps of GMRES to be the inner method. The optimality of GMRES and the fact that for a nonsingular operator GMRES does not break down permit some statements about optimality. These characteristics of GMRES also isolate the breakdown of the Krylov space generated by $(I - C_k C_k^T)A$ from the other possibilities of breakdown in various Krylov methods.

2. Consequences of inner orthogonalization

This section involves a theoretical discussion of optimality, the possibility of breakdown, and the continuation after breakdown. For the practical use of the methods this will in general not be too important, and readers interested mainly in using these methods might want to skip this section, and proceed to Section 3.

We will now formally define the elements of our discussion, repeating some of the previous section for the sake of clarity.

Definition 2.1. The matrix A is a nonsingular linear operator and the vector b is the given right-hand side of the system of equations $Ax = b$. We define the *Krylov space* $K(B, x) = \text{span}\{x, Bx, B^2x, \dots\}$ and the *Krylov subspace* $K^s(B, x) = \text{span}\{x, Bx, B^2x, \dots, B^{s-1}x\}$ for any linear operator B and any vector x. Furthermore, the matrices U_k and C_k satisfy the relations

$$U_k = (u_1 \ u_2 \ \cdots \ u_k), \tag{14}$$

$$C_k = AU_k, \tag{15}$$

with the property

$$C_k^T C_k = I_k, \tag{16}$$

and $u_i, c_i \in K(A, b)$. We also have the vectors x_k and r_k satisfying

$$x_k = \arg \min_{x \in \text{range}(U_k)} \|b - Ax\|_2 \Leftrightarrow \tag{17}$$

$$x_k = U_k C_k^T b. \tag{18}$$

$$r_k = b - C_k C_k^T b, \tag{19}$$

$$r_k \perp \text{range}(C_k). \tag{20}$$

Next, we define the operators

$$P_{C_k} = C_k C_k^T \tag{21}$$

and

$$A_{C_k} = (I - P_{C_k})A. \tag{22}$$

Because $\text{range}(C_k) \subset K(A,b)$, $K(A,b) \cap \text{range}(C_k)^{\perp}$ is an invariant subspace of A_{C_k}. We use m (not fixed) steps of the GMRES algorithm to compute the optimal approximation to r_k in the space $\text{span}\{A_{C_k}r_k, A_{C_k}^2 r_k, \ldots, A_{C_k}^m r_k\} = K^m(A_{C_k}, A_{C_k}r_k)$. This leads to the optimal approximation to the solution over the (global) space

$$\text{range}(U_k) + A^{-1}K^m(A_{C_k}, A_{C_k}r_k)$$

as follows.

Theorem 2.2. *Let $A, U_k, C_k, r_k, x_k, A_{C_k}$ and P_{C_k} be as in Definition 2.1. Let $\{r_k, A_{C_k}r_k, A_{C_k}^2 r_k, \ldots, A_{C_k}^m r_k\}$ be independent and $\{v_1, \ldots, v_{m+1}\}$ be an orthonormal basis for $K^{m+1}(A_{C_k}, r_k)$, with $v_1 = r_k/\|r_k\|_2$, generated by m steps of GMRES. This defines the relation $A_{C_k}V_m = V_{m+1}\bar{H}_m$; see [14]. Let y be defined by*

$$y = \arg\min_{\tilde{y}\in\mathbb{R}^m} \|r_k - A_{C_k}V_m\tilde{y}\|_2 = \arg\min_{\tilde{y}\in\mathbb{R}^m} \|r_k - V_{m+1}\bar{H}_m\tilde{y}\|_2. \tag{23}$$

Then the minimal residual solution of the (inner) GMRES method, $A^{-1}A_{C_k}V_m y$, gives the outer approximation

$$x_{k+1} = x_k + A^{-1}A_{C_k}V_m y, \tag{24}$$

which is also the solution to the global minimization problem

$$\min\{\|b - A\tilde{x}\|_2 : \tilde{x} \in \text{range}(U_k) \oplus \text{range}(V_m)\}. \tag{25}$$

Proof. The solution x_{opt} to the global minimization problem satisfies

$$x_{\text{opt}} = \arg\min\{\|b - A\tilde{x}\|_2 : \tilde{x} \in \text{range}(U_k) \oplus \text{range}(V_m)\} \Leftrightarrow$$

$$x_{\text{opt}} = U_k z + V_m y,$$

where

$$y, z = \arg\min\{\|b - AU_k z - AV_m y\|_2 : z \in \mathbb{R}^k, y \in \mathbb{R}^m\}. \tag{26}$$

Furthermore, we have by construction of GMRES that $A_{C_k}V_m = V_{m+1}\bar{H}_m \Leftrightarrow AV_m = P_{C_k}AV_m + V_{m+1}\bar{H}_m$, so that the minimization problem in (26) is equivalent to

$$y, z = \arg\min\{\|b - C_k(z + C_k^{\mathrm{T}}AV_m y) - V_{m+1}\bar{H}_m y\|_2 : z \in \mathbb{R}^k, y \in \mathbb{R}^m\}. \tag{27}$$

Because $\text{range}(C_k) \perp \text{range}(V_{m+1})$, the minimization (27) can be solved by two separate minimizations for z and y, respectively:

$$z + C_k^{\mathrm{T}}AV_m y = C_k^{\mathrm{T}}b \Leftrightarrow z = C_k^{\mathrm{T}}(b - AV_m y), \tag{28}$$

and by (19),

$$y = \arg\min_{\tilde{y}\in\mathbb{R}^m} \|(I - C_k C_k^T)b - V_{m+1}\bar{H}_m\tilde{y}\|_2 \Leftrightarrow$$

$$y = \arg\min_{\tilde{y}\in\mathbb{R}^m} \|r_k - V_{m+1}\bar{H}_m\tilde{y}\|_2 . \tag{29}$$

This results in

$$x_{\text{opt}} = U_k z + V_m y, \tag{30}$$

where y is given by (29), which is equivalent to (23), and z is given by (28). For x_{k+1} we have, using (18), (23) and (24),

$$\begin{aligned} x_{k+1} &= x_k + A^{-1}(I - C_k C_k^T)AV_m y \quad \text{(23) and (24)} \\ &= U_k C_k^T b + V_m y - U_k C_k^T AV_m y \quad \text{(18)} \\ &= U_k C_k^T (b - AV_m y) + V_m y \\ &= U_k z + V_m y. \qquad \square \end{aligned} \tag{31}$$

It also follows from this theorem that the GCR optimization (in the outer iteration) is given by (24), so that the residual computed in the inner GMRES iteration equals the residual of the outer GCR iteration:

$$r_{k+1} = b - Ax_{k+1} = b - Ax_k - A_{C_k}V_m y = r_k - A_{C_k}V_m y = r_k - V_{m+1}\bar{H}_m y. \tag{32}$$

So the outer method only needs to compute the new u_{k+1} and c_{k+1} by

$$c_{k+1} = (A_{C_k}V_m y)/\|A_{C_k}V_m y\|_2 = ((I - C_k C_k^T)AV_m y)/\|(I - C_k C_k^T)AV_m y\|_2, \tag{33}$$

$$u_{k+1} = (A^{-1}A_{C_k}V_m y)/\|A_{C_k}V_m y\|_2 = V_m y - U_k C_k^T AV_m y/\|(I - C_k C_k^T)AV_m y\|_2, \tag{34}$$

where $A_{C_k}V_m y$ and $A^{-1}A_{C_k}V_m y = (I - U_k C_k^T A)V_m y$ have been computed already as the residual update and the solution in the inner iteration; see (23) and (24). The outer GCR method consequently becomes very simple. We will now consider the possibility of breakdown, generating a Krylov space with a singular operator. The following theorem may not be a surprising one, but it is given because it captures the essence of the following discussion.

Theorem 2.3. *Let B be a linear operator such that $\{y, By, B^2y, \ldots, B^m y\}$ is independent and $K^{m+1}(B, y)$ is an invariant subspace of B, so that $\{y, By, B^2y, \ldots, B^{m+1}y\}$ is dependent. Then:*

(1) *B is nonsingular over $K^{m+1}(B, y)$ and therefore has an inverse over this space if and only if $\{By, B^2y, \ldots, B^{m+1}y\}$ is independent.*

(2) *Let $B^{m+1}y$ be given by*

$$B^{m+1}y = \sum_{i=0}^{m} \alpha_i B^i y, \tag{35}$$

where $B^0 = I$. *Then* B *is nonsingular over* $\operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$, *and therefore* B *has an inverse over this space if and only if in* (35) $\alpha_j \neq 0$ *and* $\alpha_i = 0$, $0 \leqslant i \leqslant j-1$.

(3) *Let* $B^{m+1} y$ *be given by* (35), *and* $\alpha_j \neq 0$, *and* $\alpha_i = 0$, $0 \leqslant i \leqslant j-1$. *Then* B *is singular over* $\operatorname{span}\{B^{j-1} y, B^j y, \ldots, B^m y\}$. *More specifically*

$$\exists z \in \operatorname{span}\{B^j y, \ldots, B^m y\} \text{ such that } Bz = B^j y,$$

and therefore $(B^{j-1} y - z) \in \operatorname{null}(B)$.

Proof. (1) Assume that B is nonsingular over $K^{m+1}(B, y)$. Then $\dim BK^{m+1}(B, y) = \dim K^{m+1}(B, y)$, and hence $\{By, B^2 y, \ldots, B^{m+1} y\}$ is independent. Conversely, assume that $\{By, B^2 y, \ldots, B^{m+1} y\}$ is independent. Then $\forall x \in BK^{m+1}(B, y)$, $x = \sum_{i=0}^{m} \alpha_i B^{i+1} y$, where the α_i are uniquely determined, and we can construct $\tilde{x} = \sum_{i=0}^{m} \alpha_i B^i y$, so that $B\tilde{x} = x$. Because the α_i are unique and both $\{y, B_y, \ldots, B^m y\}$ and $\{By, B^2 y, \ldots, B^{m+1} y\}$ are independent, this defines a one-to-one correspondence and hence B must be nonsingular over $K^{m+1}(B, y)$.

(2) First assume that in (35) $\alpha_j \neq 0$ and $\alpha_i = 0$, $0 \leqslant i \leqslant j-1$. Then $B^{m+1} y \notin \operatorname{span}\{B^{j+1} y, \ldots, B^m y\}$ since $\{B^j y, \ldots, B^m y\}$ is independent. Let $\bar{y} = B^j y$, then $\{\bar{y}, B\bar{y}, \ldots, B^{m-j}\bar{y}\}$ is independent, $K^{m+1-j}(B, \bar{y})$ is an invariant subspace of B and $\{B\bar{y}, B^2\bar{y}, \ldots, B^{m+1-j}\bar{y}\}$ is independent. By case (1) of this theorem we can conclude that B is nonsingular over $K^{m+1-j}(B, \bar{y}) = \operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$, and therefore B has an inverse over this space.

Second assume that $\operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$ is an invariant subspace of B and B is nonsingular over this space. Then obviously $B^{m+1} y \in \operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$ so that in (35) $\alpha_i = 0$, $0 \leqslant i \leqslant j-1$. Again let $\bar{y} = B^j y$ so that $\{\bar{y}, B\bar{y}, \ldots, B^{m-j}\bar{y}\}$ is independent and $K^{m+1-j}(B, \bar{y})$ is an invariant subspace of B. Then by case (1) of this theorem $\{B\bar{y}, B^2\bar{y}, \ldots, B^{m+1-j}\bar{y}\} = \{B^{j+1} y, B^{j+2} y, \ldots, B^{m+1} y\}$ is also independent. Therefore $B^{m+1} y \notin \operatorname{span}\{B^{j+1} y, B^{j+2} y, \ldots, B^m y\}$ so that $\alpha_j \neq 0$.

(3) Since B is nonsingular over $\operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$, B has an inverse over $\operatorname{span}\{B^j y, B^{j+1} y, \ldots, B^m y\}$, so that case (3) of the theorem follows trivially. □

Now although GMRES is still optimal in the sense that at each iteration it computes the minimal residual solution over the generated Krylov subspace, the generation of the Krylov space itself, from a singular operator, may break down. The following simple example shows that this may happen before the solution is found, even though both the solution and the right-hand side are in the range of the given (singular) operator. Define the matrix A:

$$A = (e_2 \; e_3 \; e_4 \; 0),$$

where e_i denotes the ith Cartesian basis vector. Note that $A = (I - e_1 e_1^T)(e_2 e_3 e_4 e_1)$, which is the same type of operator as A_{C_k}, an orthogonal projection times a nonsingular operator. Now consider the system of equations $Ax = e_3$. Then GMRES (or any other Krylov method) will search for a solution in the space

$$\operatorname{span}\{e_3, Ae_3, A^2 e_3, \ldots\} = \operatorname{span}\{e_3, e_4, 0, 0, \ldots\},$$

so we have a breakdown of the Krylov space and the solution is not contained in it, even though the solution exists and is an element of the range of A ($Ae_2 = e_3$), and the right-hand side is in the

orthogonal complement of null(A). So the singular nonsymmetric case is quite different from the symmetric one. We will now define breakdown of the Krylov space for the inner GMRES iteration.

Definition 2.4. *Let* A, U_k, C_k, A_{C_k} and r_k be as in Definition 2.1. Let $\{r_k, A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\}$ be independent and $\{v_1, v_2, \ldots, v_m\}$ be an orthonormal basis for $K^m(A_{C_k}, r_k)$ with $v_1 = r_k/\|r_k\|_2$, generated by $m-1$ steps of GMRES. We say we have a *breakdown* of the Krylov subspace if $A_{C_k}v_m \in \text{range}(V_m)$, since this implies we can no longer expand the Krylov subspace. We call it a *lucky breakdown* if $v_1 \in \text{range}(A_{C_k}V_m)$, because we then have found the solution (the inverse of A is known over the space range$(A_{C_k}V_m)$). We call it a *true breakdown* if $v_1 \notin \text{range}(A_{C_k}V_m)$, because then the solution is not contained in the Krylov subspace.

The following theorem relates true breakdown to the invariance of the sequence of subspaces in the inner method for the operator A_{C_k}. Case (3) indicates that it is always known whether a breakdown is true or lucky.

Theorem 2.5. *Let* A, U_k, C_k, A_{C_k} *and* r_k *be as in Definition* 2.1. *Let* $\{r_k, A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\}$ *be independent and* $\{v_1, v_2, \ldots, v_i\}$ *be an orthonormal basis for* $K^i(A_{C_k}, r_k)$, *for* $i = 1, \ldots, m$, *with* $v_1 = r_k/\|r_k\|_2$, *generated by* $m-1$ *steps of GMRES. Then at step* m:

(1) *A true breakdown occurs if and only if* range$(A_{C_k}V_{m-1})$ *is an invariant subspace of* A_{C_k}.

(2) *A true breakdown occurs if and only if* $A_{C_k}v_m \in \text{range}(A_{C_k}V_{m-1})$.

(3) *A breakdown occurs if and only if we can define* H_m *by* $A_{C_k}V_m = V_mH_m$. *Furthermore, it is a true breakdown if and only if* H_m *is singular.*

Proof. (1) Assume that a true breakdown occurs. By assumption, $\{r_k, \ldots, A_{C_k}^{m-1}r_k\}$ is independent and breakdown implies that $K^m(A_{C_k}, r_k)$ is an invariant subspace of A_{C_k}. So $v_1 \notin \text{range}(A_{C_k}V_m)$ implies that A_{C_k} is singular over $K^m(A_{C_k}, r_k)$. By Theorem 2.3 $\{A_{C_k}r_k, \ldots, A_{C_k}^m r_k\}$ is dependent, and therefore $A_{C_k}^m r_k \in \text{span}\{A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\} = \text{range}(A_{C_k}V_{m-1})$, and range$(A_{C_k}V_{m-1})$ is an invariant subspace of A_{C_k}.

Conversely assume that $\text{range}(A_{C_k}V_{m-1}) = \text{span}\{A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\}$ is an invariant subspace of A_{C_k}. Then $\text{span}\{r_k, \ldots, A_{C_k}^{m-1}r_k\}$ is an invariant subspace of A_{C_k}, so a breakdown will occur. Furthermore, $\text{range}(A_{C_k}V_m) = \text{span}\{A_{C_k}r_k, \ldots, A_{C_k}^m r_k\}$, and since $A_{C_k}^m r_k \in \text{span}\{A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\}$ by assumption, we have $\text{range}(A_{C_k}V_m) = \text{range}(A_{C_k}V_{m-1})$. Because $\{r_k, \ldots, A_{C_k}^{m-1}r_k\}$ is independent by assumption, $r_k \notin \text{range}(A_{C_k}V_m)$, so that a true breakdown occurs.

(2) Assume a true breakdown occurs. Then by case (1) of this proof range$(A_{C_k}V_{m-1})$ is an invariant subspace and hence $\text{range}(A_{C_k}V_m) = \text{range}(A_{C_k}V_{m-1})$, so that $A_{C_k}v_m \in \text{range}(A_{C_k}V_{m-1})$.

Assume, on the other hand, that $A_{C_k}v_m \in \text{range}(A_{C_k}V_{m-1})$. Then $A_{C_k}v_m \in \text{span}\{A_{C_k}r_k, \ldots, A_{C_k}^{m-1}r_k\} \subset \text{range}(V_m)$, so that a breakdown occurs. Furthermore $A_{C_k}v_m \in \text{range}(A_{C_k}V_{m-1})$ implies $\text{range}(A_{C_k}V_m) = \text{range}(A_{C_k}V_{m-1})$. Because $\{r_k, \ldots, A_{C_k}^{m-1}r_k\}$ is independent by assumption, $r_k \notin \text{range}(A_{C_k}V_{m-1}) = \text{range}(A_{C_k}V_m)$. Consequently a true breakdown occurs.

(3) Assume first that a breakdown occurs. Then by Definition 2.4, $A_{C_k}v_m \in \text{range}(V_m)$, GMRES defines the relation $A_{C_k}V_{m-1} = V_m\bar{H}_{m-1}$ and therefore we can define an $H_m \in \mathbb{R}^{m\times m}$ such that $A_{C_k}V_m = V_mH_m$.

Second, assume that an H_m exists that satisfies $A_{C_k}V_m = V_m H_m$. Then $A_{C_k}v_m \in \text{range}(V_m)$, so that a breakdown occurs. Assume that a true breakdown occurs. Let H_m be nonsingular. Then $\exists x$ such

that $H_m x = e_1$, where e_1 denotes the first Cartesian basis vector, so that

$$v_1 = V_m H_m x = A_{C_k} V_m x \Rightarrow v_1 \in \text{range}(A_{C_k} V_m).$$

This contradicts our assumptions, which implies that H_m must be a singular.

Assume that H_m is singular. Then $\dim V_m H_m = \dim A_{C_k} V_m < m$, which implies that $\{A_{C_k} r_k, \ldots, A_{C_k}^m r_k\}$ is dependent. This in turn leads to $\text{range}(A_{C_k} V_m) = \text{span}\{A_{C_k} r_k, \ldots, A_{C_k}^{m-1} r_k\} = \text{range}(A_{C_k} V_{m-1})$, so that a true breakdown occurs (see above or Theorem 2.3). □

From the previous theorem and its proof one can already conclude that a true breakdown occurs if and only if A_{C_k} is singular over $K^m(A_{C_k}, r_k)$. From Definition 2.1 we know $\text{null}(A_{C_k}) = \text{range}(U_k)$. We will make this more explicit in the following theorem, which relates true breakdown to the intersection of the inner search space and the outer search space.

Theorem 2.6. *Let A, U_k, C_k, A_{C_k} and r_k be as in Definition 2.1. Let $\{r_k, A_{C_k} r_k, \ldots, A_{C_k}^{m-1} r_k\}$ be independent and $\{v_1, v_2, \ldots, v_i\}$ be an orthonormal basis for $K^i(A_{C_k}, r_k)$ for $i = 1, \ldots, m$, with $v_1 = r_k / \| r_k \|_2$, generated by $m - 1$ steps of GMRES. A true breakdown occurs at step m if and only if*

$$\exists u \neq 0, u \in \text{range}(V_m) \text{ such that } u \in \text{range}(U_k).$$

Proof. Let $u \neq 0$, $u \in \text{range}(V_m)$, and $u \in \text{range}(U_k)$. By Definition 2.1, $A_{C_k} u = 0$; consequently $\dim(\text{range}(A_{C_k} V_m)) < m \Rightarrow \{A_{C_k} r_k, \ldots, A_{C_k}^m r_k\}$ is dependent. This implies (see the proof of Theorem 2.5 or Theorem 2.3) that $\text{range}(A_{C_k} V_{m-1})$ is an invariant subspace of A_{C_k}, so that a true breakdown occurs.

Assume that a true breakdown occurs. Then by Theorem 2.5, $\text{range}(A_{C_k} V_{m-1})$ is an invariant subspace of A_{C_k}, which implies that $\{A_{C_k} r_k, \ldots, A_{C_k}^m r_k\}$ is dependent. Therefore,

$$\exists\, u \neq 0, u \in \text{span}\{r_k, \ldots, A_{C_k}^{m-1} r_k\} = \text{range}(V_m): \quad A_{C_k} u = 0.$$

Hence

$$u \in \text{null}(A_{C_k}) \Rightarrow u \in \text{range}(U_k). \qquad \square$$

The following theorem indicates that breakdown cannot occur in the inner GMRES method before the total number of iterations exceeds the dimension of the Krylov space $K(A, b)$. This means that, in practice, a breakdown will be rare.

Theorem 2.7. *Let A, U_k, C_k, A_{C_k} and r_k be as in Definition 2.1. Let $m = \dim(K(A, b))$; that is, $m = \max\{s: \{b, Ab, \ldots, A^{s-1} b\}$ is independent$\}$. Define $P_s(A)b = \sum_{i=0}^{s} \gamma_i A^i b$, with $\gamma_s \neq 0$, and let*

$$u_i = P_{l_i - 1}(A) b, \quad i = 1, \ldots, k,$$

$$l = \max_{i = 1, \ldots, k} l_i < m.$$

Then $r_k = (I - P_{C_k}) b = P_l(A) b$. We call l the total number of iterations. If

$$j < m - l$$

then

$\{r_k, A_{C_k} r_k, \ldots, A_{C_k}^j r_k\}$ *is independent,*

and therefore no breakdown occurs in the first j steps of GMRES.

Proof. By definition, $c_i = Au_i \Rightarrow c_i \in \operatorname{span}\{Ab, \ldots, A^l b\}$. Further, $r_k = P_l(A)b \Rightarrow A_{C_k} r_k = (I - P_{C_k}) P'_{l+1}(A)b = P_{l+1}(A)b$ and analogously we find $A_{C_k}^i r_k = P_{l+i}(A)b$ for $i = 1, \ldots, j$. Thus we have

$$\{r_k, A_{C_k} r_k, \ldots, A_{C_k}^j r_k\} = \{P_l(A)b, P_{l+1}(A)b, \ldots, P_{l+j}(A)b\}$$

is independent by definition of m. Therefore no breakdown can occur. □

We will now show how a true breakdown can be overcome. There are basically two ways to continue.

In the inner iteration: by finding a suitable vector to expand the Krylov subspace.

In the outer iteration: by computing the solution of the inner iteration just before breakdown and continuing by making one LSQR step (see below) in the outer iteration.

2.1. Continuation in the inner iteration

The following theorem indicates how we can find a suitable vector to expand the Krylov subspace.

Theorem 2.8. *Let* A, U_k, C_k, A_{C_k} *and* r_k *be as in Definition* 2.1, *let* $\{r_k, A_{C_k} r_k, \ldots, A_{C_k}^{m-1} r_k\}$ *be independent and* $\{v_1, v_2, \ldots, v_i\}$ *be an orthonormal basis for* $K^i(A_{C_k}, r_k)$, *for* $i = 1, \ldots, m$, *with* $v_1 = r_k / \|r_k\|_2$, *generated by* $m - 1$ *steps of GMRES. If a true breakdown occurs then*

$$\exists c \in \operatorname{range}(C_k) \text{ such that } A_{C_k} c \notin \operatorname{range}(A_{C_k} V_{m-1}),$$

which implies

$$\exists c_i \in \{c_1, \ldots, c_k\} \text{ such that } A_{C_k} c_i \notin \operatorname{range}(A_{C_k} V_{m-1}).$$

Proof. By Theorem 2.5 we have $A_{C_k} v_m \in \operatorname{range}(A_{C_k} V_{m-1})$. Now assume that we would stop after the last regular GMRES step, step $(m-1)$, and compute r_{k+1} using (32) and c_{k+1} using (33). Then from $r_{k+1} = r_k - A_{C_k} V_{m-1} y = \|r_k\|_2 v_1 - V_m \bar{H}_{m-1} y$, we have $r_{k+1} \in \operatorname{range}(V_m)$ and therefore $A_{C_k} r_{k+1} \in \operatorname{range}(A_{C_k} V_{m-1})$. Because A is a nonsingular operator $r_{k+1} \in \operatorname{span}\{Ar_{k+1}, \ldots, A^p r_{k+1}\}$, for some $p \Rightarrow r_{k+1} = \sum_{j=1}^p \alpha_j A^j r_{k+1} \Rightarrow r_{k+1} = (I - P_{C_k}) r_{k+1} = \sum_{j=1}^p \alpha_j (I - P_{C_k}) A^j r_{k+1}$ and since $r_{k+1} \neq 0$, $r_{k+1} \perp \operatorname{range}(C_k)$ and $r_{k+1} \perp \operatorname{range}(A_{C_k} V_{m-1})$:

$$\exists j \text{ such that } (I - P_{C_k}) A^j r_{k+1} \neq 0 \text{ and } (I - P_{C_k}) A^j r_{k+1} \notin \operatorname{range}(A_{C_k} V_{m-1}).$$

We also have $A_{C_k} r_{k+1} \in \operatorname{range}(A_{C_k} V_{m-1})$. Now let s be such that

$$(I - P_{C_k}) A^s r_{k+1} \notin \operatorname{range}(A_{C_k} V_{m-1})$$

and

$$\forall j,\ 1 \leqslant j < s:\quad (I - P_{C_k})A^j r_{k+1} \in \text{range}(A_{C_k} V_{m-1}).$$

Then

$$(I - P_{C_k})A^s r_{k+1} = (I - P_{C_k})A(I - P_{C_k})A^{s-1} r_{k+1} + (I - P_{C_k})AP_{C_k}A^{s-1} r_{k+1},$$

where by assumption $(I - P_{C_k})A^{s-1} r_{k+1} \in \text{range}(A_{C_k} V_{m-1}) \Rightarrow (I - P_{C_k})A(I - P_{C_k})A^{s-1} r_{k+1} \in \text{range}(A_{C_k} V_{m-1})$. This implies that $(I - P_{C_k})AP_{C_k}A^{s-1} r_{k+1} \notin \text{range}(A_{C_k} V_{m-1})$ and since $P_{C_k}A^{s-1} r_{k+1} \in \text{range}(C_k)$, we have

$$\exists c \in \text{range}(C_k) \text{ such that } A_{C_k} c \notin \text{range}(A_{C_k} V_{m-1}),$$

which in turn implies

$$\exists c_i \in \{c_1, \ldots, c_k\} \text{ such that } A_{C_k} c_i \notin \text{range}(A_{C_k} V_{m-1}). \qquad \square$$

Not any $c \in \text{range}(C_k)$ will do, as is shown by the following example. Let $u_1 = b$. Then

$$A_{C_k} r_k = A_{C_k}(b - C_k C_k^{\mathrm{T}} b) = -A_{C_k} C_k C_k^{\mathrm{T}} b.$$

Therefore, we must try the c_i until one of them works.

2.2. *Continuation in the outer iteration*

Another way to continue after a true breakdown in the inner GMRES iteration is to compute the inner iteration solution just before the breakdown and apply an LSQR switch (see below) in the outer GCR iteration. The following theorem states the reason why the LSQR switch must be applied.

Theorem 2.9. *Let A, U_k, C_k, A_{C_k} and r_k be as in Definition 2.1, let $\{r_k, A_{C_k} r_k, \ldots, A_{C_k}^{m-1} r_k\}$ be independent and $\{v_1, v_2, \ldots, v_i\}$ be an orthonormal basis for $K^i(A_{C_k}, r_k)$, for $i = 1, \ldots, m$, with $v_1 = r_k / \| r_k \|_2$, generated by $m - 1$ steps of GMRES. Let a true breakdown occur at step m, and assume that we compute the solution at the previous GMRES step, switch to the outer method, and make one GCR step (24), (32), (33) and (34). So we compute $r_{k+1} = r_k - V_m \bar{H}_{m-1} y$ and $c_{k+1} = A_{C_k} V_{m-1} y / \| A_{C_k} V_{m-1} y \|$. Then we will have stagnation in the next inner GMRES iterations; that is,*

$$r_{k+1} \perp \text{span}\{A_{C_{k+1}} r_{k+1}, A_{C_{k+1}}^2 r_{k+1}, \ldots\}.$$

Proof. We have $c_{k+1} \in \text{range}(A_{C_k} V_{m-1})$. Furthermore, $r_{k+1} \in \text{range}(V_m)$ and $A_{C_k} r_{k+1} \in \text{range}(A_{C_k} V_{m-1})$ by Theorem 2.5. $A_{C_{k+1}} = (I - c_{k+1} c_{k+1}^{\mathrm{T}}) A_{C_k}$ and from this and Theorem 2.5 it follows that $\text{range}(A_{C_k} V_{m-1})$ is also an invariant subspace of $A_{C_{k+1}}$ and $A_{C_{k+1}} r_{k+1} \in \text{range}(A_{C_k} V_{m-1})$. $\square$

The reason for the stagnation is that the new residual r_{k+1} remains in the same Krylov space $K(A_{C_k}, r_k)$, which contains a nonzero $u \in \text{range}(U_k)$. So we have to "leave" this Krylov space. We can do this using the so-called LSQR switch, which was introduced in [18] to remedy stagnation in

the inner method. As in the GMRESR method, stagnation in the inner method will result in a breakdown in the outer GCR method, because the residual cannot be updated. The following theorem indicates that the LSQR switch always works.

Theorem 2.10. *If stagnation occurs in the inner GMRES method, that is*

$$\min_{\tilde{y}\in\mathbb{R}^{m-1}} \|r_{k+1} - A_{C_k} V_{m-1}\tilde{y}\|_2 = \|r_{k+1}\|_2,$$

then we can continue (LSQR switch) by setting

$$c_{k+2} = \gamma(I - C_{k+1}C_{k+1}^{\mathrm{T}})AA^{\mathrm{T}}r_{k+1}$$

and

$$u_{k+2} = \gamma A^{-1}(I - C_{k+1}C_{k+1}^{\mathrm{T}})AA^{\mathrm{T}}r_{k+1},$$

where γ is a normalization constant. This leads to

$$r_{k+2} = r_{k+1} - (r_{k+1}^{\mathrm{T}}c_{k+2})c_{k+2}$$

and

$$x_{k+2} = x_{k+1} + (r_{k+1}^{\mathrm{T}}c_{k+2})u_{k+2},$$

which always gives an improved approximation. Therefore, these vectors can also be used as the start vectors for a new Krylov subspace in the inner GMRES method if desired.

Proof. (Following the proof in [18]).

$$c_{k+2}^{\mathrm{T}}r_{k+1} = \gamma r_{k+1}^{\mathrm{T}}AA^{\mathrm{T}}r_{k+1},$$

because $r_{k+1} \perp \mathrm{range}(C_{k+1})$, and so

$$c_{k+2}^{\mathrm{T}}r_{k+1} = \gamma\|A^{\mathrm{T}}r_{k+1}\|_2^2 \neq 0.$$

So, this step will always give an improvement to the residual. This also proves that $(I - C_{k+1}C_{k+1}^{\mathrm{T}})AA^{\mathrm{T}}r_{k+1} \notin \mathrm{range}(A_{C_k}V_{m-1})$, because $r_{k+1} \perp \mathrm{range}(A_{C_k}V_{m-1})$, so that we may also use this vector to build a Krylov subspace in the inner iteration after a true breakdown in the previous inner iteration. □

3. Implementation

A straightforward implementation of the GCR method and an orthogonalizing inner method will obviously result in a large number of vector updates, inner products and vector scalings. We will therefore describe an implementation which greatly reduces this work. This implementation is mainly due to Fokkema [9]. First we will consider the outer GCR method, next the inner GMRES method, and finally the implementation of BiCGSTAB as the inner algorithm.

Instead of the matrices U_k and C_k (see Definition 2.1) we will use in the actual implementation the matrices $\hat{U}_k$, $\bar{C}_k$, N_k, Z_k and d_k which are defined below.

Definition 3.1. *Let A, U_k, C_k, b and r_k be as in Definition 2.1. Then $\hat{U}_k$, $\bar{C}_k$, N_k, Z_k and d_k are defined as follows:*

$$C_k = \bar{C}_k N_k, \tag{36}$$

where

$$N_k = \operatorname{diag}(\|\bar{c}_1\|_2^{-1}, \|\bar{c}_2\|_2^{-1}, \ldots, \|\bar{c}_k\|_2^{-1}), \tag{37}$$

$$A\hat{U}_k = \bar{C}_k Z_k, \tag{38}$$

where Z_k is assumed to be upper-triangular. Finally d_k is defined by the relation

$$r_k = b - \bar{C}_k d_k. \tag{39}$$

From this the approximate solution x_k, corresponding to r_k, is implicitly represented as

$$x_k = \hat{U}_k Z_k^{-1} d_k. \tag{40}$$

Using this relation, x_k can be computed at the end of the complete iteration or before truncation (see the end of this section). This implicit representation of U_k saves all the intermediate updates of previous u_k to a new u_{k+1}, which saves about 50% of the computational costs in the outer GCR iteration.

3.1. Initialization and restart

If the method is started with an initial guess $x_0 \neq 0$, instead of $x_0 = 0$ as defined in Definition 2.1, we must compute $r_0 = b - Ax_0$ and change the updates to

$$r_k = r_0 - \bar{C}_k d_k \quad \text{and} \quad x_k = x_0 + \hat{U}_k Z_k^{-1} d_k.$$

The rest of the algorithm remains the same. Also after a restart (see Section 3.7) we compute a new result by adding a correction to the previously computed x_k.

3.2. GMRES as inner iteration

Assume we have made k outer iterations, so that $\hat{U}_k$, $\bar{C}_k$ and r_k are given. Then, in the inner GMRES iteration, the orthogonal matrix V_{m+1} is constructed such that $\bar{C}_k^{\mathrm{T}} V_{m+1} = O$ and

$$AV_m = \bar{C}_k B_m + V_{m+1} \bar{H}_m, \tag{41}$$

$$B_m = N_k^2 \bar{C}_k^{\mathrm{T}} AV_m. \tag{42}$$

This algorithm is equivalent to the usual GMRES algorithm, except that the vectors Av_i are first orthogonalized on $\bar{C}_k$. From (41) and (42) it is obvious that $AV_m - \bar{C}_k B_m = A_{C_k} V_m = V_{m+1} \bar{H}_m$, cf.

Theorem 2.2. Next, we compute y according to (23), and we set (cf. (33) without normalization)

$$\bar{c}_{k+1} = V_{m+1}\bar{H}_m y, \tag{43}$$

$$\hat{u}_{k+1} = V_m y. \tag{44}$$

This gives $A\hat{u}_{k+1} = AV_m y = \bar{C}_k B_m y + V_{m+1}\bar{H}_m y = \bar{C}_k B_m y + \bar{c}_{k+1}$, so if we set

$$z_{k+1} = \begin{pmatrix} B_m y \\ 1 \end{pmatrix},$$

then the relation $A\hat{U}_{k+1} = \bar{C}_{k+1} Z_{k+1}$ is again satisfied. It follows from Theorem 2.2 that the new residual of the outer iteration equals the final residual of the inner iteration, $r_{k+1} = r_m^{\text{inner}}$, and is given by

$$r_{k+1} = r_k - \bar{c}_{k+1}, \tag{45}$$

so that $d_{k+1} = 1$. Obviously the residual norm only needs to be computed once. If we replace the new residual of the outer iteration r_{k+1} by the residual of the inner iteration r_m^{inner}, then we get from (45) an important relation that holds more generally,

$$\bar{c}_{k+1} = r_k - r_m^{\text{inner}}. \tag{46}$$

This relation is important, since in general (when other Krylov methods are used for the inner iteration) $\bar{c}_{k+1}$ or c_{k+1} cannot be computed from u_{k+1}, because u_{k+1} is not computed explicitly, nor does a relation like (43) exist. However, the (inner) residual is always known, because it is needed for the inner iteration itself. See also the part on BiCGSTAB. Therefore in our current implementation (45) is replaced by

$$r_{k+1} = r_m^{\text{inner}} = r_k - V_{m+1}\bar{H}_m y, \tag{47}$$

and (43) by (46). Finally, we need to compute the new coefficient of N_{k+1}, $\|\bar{c}_{k+1}\|_2^{-1}$, in order to satisfy the relations in Definition 3.1. The outer iteration then consists only of (45) or (47), $d_{k+1} = 1$, the computation of $\|\bar{c}_{k+1}\|_2^{-1}$ and (40) at the end.

3.3. *Algorithms for nested GMRES with inner orthogonalization*

In Figs. 3 and 4 we give an outline of the inner GMRES algorithm for A_{C_k} and two versions of the outer GCR algorithm: GCRO (because of the implicit inner orthogonalization) and one generic version with an arbitrary inner method, with only the requirement that on output the relation $A\hat{u}_{k+1} = r_k - r^{\text{inner}} + \bar{C}_k Z_{1..k,k+1}$ holds. The former version explicitly uses the relations that hold after an inner iteration of (m steps of) GMRES to make a few more optimizations. However, for some problems (with GMRES for A_{C_k} as the inner method) the generic version turns out to be slightly more stable and is then to be preferred. For the experiments discussed later, we used the generic version, even with an inner GMRES for A_{C_k}.

3.4. *BiCGSTAB as inner iteration*

For the sake of simplicity we will only consider nonpreconditioned BiCGSTAB here (or explicitly preconditioned). The algorithm as given in [17] is listed in Fig. 5. The algorithm contains

GMRES with A_{C_k} (m steps):
(after k outer iterations)
1. Select tol;
   $v_1 = r_k / \|r_k\|_2$;
2. **for** $j = 1, \ldots, m$ **do**
   $v_{j+1} = Av_j$;
   for $i = 1, \ldots, k$ **do**
   $b_{i,j} = c_i^T v_{j+1}$;
   $v_{j+1} = v_{j+1} - b_{i,j} c_i$;
   for $i = 1, \ldots, j$ **do**
   $h_{i,j} = v_i^T v_{j+1}$;
   $v_{j+1} = v_{j+1} - h_{i,j} v_i$;
   $h_{j+1,j} = \|v_{j+1}\|_2$;
   $v_{j+1} = h_{j+1,j}^{-1} v_{j+1}$;
$y_m = \arg\min_y \| \|r_k\|_2 e_1 - \bar{H}_m y \|_2$;
$\hat{u}_{k+1} = V_m y_m$;
$r^{\text{inner}} = r_k - V_{m+1} \bar{H}_m y_m$;
$Z_{1..k,k+1} = B_m y_m$;

Fig. 3. The inner GMRES algorithm with A_{C_k}.

GCRO, generic version:
1. Select x_0, tol;
   $r_0 = b - Ax_0$, $k = 0$;
2. **while** $\|r_k\|_2 >$ tol **do**
   call inner(
   output:
   $\hat{u}_{k+1}$,
   r^{inner},
   $Z_{1..k,k+1}$);
   $\bar{c}_{k+1} = r_k - r^{\text{inner}}$;
   $(N_{k+1})_{k+1} = \|\bar{c}_{k+1}\|_2^{-1}$;
   $Z_{k+1,k+1} = 1$;
   $d_{k+1} = (\bar{c}_{k+1}^T r_k) \cdot (N_{k+1})_{k+1}^2$;
   $r_{k+1} = r_k - d_{k+1} \bar{c}_{k+1}$;
   $k = k + 1$;
$x_k = x_0 + \hat{U}_k Z_k^{-1} d_k$;

GCRO, with GMRES:
1. Select x_0, tol;
   $r_0 = b - Ax_0$, $k = 0$;
2. **while** $\|r_k\|_2 >$ tol **do**
   call gmres(
   output:
   $\hat{u}_{k+1}$,
   r^{inner},
   $\|r^{\text{inner}}\|_2$,
   $Z_{1..k,k+1}$);
   $\bar{c}_{k+1} = r_k - r^{\text{inner}}$;
   $(N_{k+1})_{k+1} = (\|r_k\|^2 - \|r^{\text{inner}}\|^2)^{-1/2}$;
   $Z_{k+1,k+1} = 1$;
   $d_{k+1} = 1$;
   $r_{k+1} = r^{\text{inner}}$;
   $k = k + 1$;
$x_k = x_0 + \hat{U}_k Z_k^{-1} d_k$;

Fig. 4. GCRO: a generic version and a special version for inner GMRES with A_{C_k}.

two matrix–vector products with A, which must be replaced by A_{C_k}. The coefficients of the orthogonalizations must be saved to satisfy relation (38) at the end of the iteration. This avoids the explicit correction of x_i, which will become $\hat{u}_{k+1}$ in the outer iteration. The implementation is as follows. On initialization we set $z_{k+1} = 0$, $x_0 = 0$ and hence $r_0 = r_k^{\text{outer}}$. After the statement $v_i = Ap_i$ we add

$$y_1 = N_k^2 \bar{C}_k^T v_i; \quad v_i = v_i - \bar{C}_k y_1,$$

BiCGSTAB:

1. Select x_0, tol;
   $r_0 = b - Ax_0$, $\hat{r}_0 = r_0$;
   $\rho_0 = \alpha = \omega_0 = 1$;
   $v_0 = p_0 = 0$, $i = 0$;
2. **while** $\|r_i\|_2 >$ tol **do**

   $i = i + 1$;
   $\rho_i = (\hat{r}_0, r_{i-1})$;
   $\beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
   $p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1}v_{i-1})$;
   $v_i = Ap_i$;

   $\alpha = \rho_i/(\hat{r}_0, v_i)$;
   $s = r_{i-1} - \alpha v_i$;
   $t = As$;

   $\omega_i = (t, s)/(t, t)$;
   $x_i = x_{i-1} + \alpha p_i + \omega_i s$;
   $r_i = s - \omega_i t$;

Fig. 5. The BiCGSTAB algorithm.

BiCGSTAB in GCR with orthogonalization:
(after k outer iterations)

1. Select x_0, maxit, tol, rtol;
   $r_0 = b - Ax_0$, $\hat{r}_0 = r_0$;
   $\rho_0 = \alpha = \omega_0 = 1$;
   $v_0 = p_0 = 0$, $i = 0$;
2. **while** $\|r_i\|_2 > \max(\text{rtol} * \|r_0\|_2, \text{tol})$
   and $i <$ maxit
   do
   $i = i + 1$;
   $\rho_i = (\hat{r}_0, r_{i-1})$;
   $\beta = (\rho_i/\rho_{i-1})(\alpha/\omega_{i-1})$;
   $p_i = r_{i-1} + \beta(p_{i-1} - \omega_{i-1}v_{i-1})$;
   $v_i = Ap_i$;
   $y_1 = N_k^2 \bar{C}_k^{\mathrm{T}} v_i$; $v_i = v_i - \bar{C}_k y_1$;
   $\alpha = \rho_i/(\hat{r}_0, v_i)$;
   $s = r_{i-1} - \alpha v_i$;
   $t = As$;
   $y_2 = N_k^2 \bar{C}_k^{\mathrm{T}} t$; $t = t - \bar{C}_k y_2$;
   $\omega_i = (t, s)/(t, t)$;
   $x_i = x_{i-1} + \alpha p_i + \omega_i s$;
   $r_i = s - \omega_i t$;
   $z_{k+1} = z_{k+1} + \alpha y_1 + \omega_i y_2$;

Fig. 6. The BiCGSTAB algorithm in GCR with orthogonalization.

and after $t = As$ we add

$$y_2 = N_k^2 \bar{C}_k^{\mathrm{T}} t; \quad t = t - \bar{C}_k y_2 .$$

Further, at the end of the iteration, we set $z_{k+1} = z_{k+1} + \alpha y_1 + \omega_i y_2$. From this it is easily verified that the relation $Ax_i = r_0 - r_i + \bar{C}_k z_{k+1}$ is satisfied at the end of each iteration. If some stopping criterion is satisfied after step i we set

$$\hat{u}_{k+1} = x_i, \tag{48}$$

$$\bar{c}_{k+1} = r_0 - r_i = r_k^{\text{outer}} - r_i^{\text{inner}}, \quad \text{cf. (46)}, \tag{49}$$

$$z_{k+1} = \begin{pmatrix} z_{k+1} \\ 1 \end{pmatrix}, \tag{50}$$

and the (outer iteration) relation (38) is again satisfied. In this case we have to compute the residual update explicitly. By construction the columns of $\bar{C}_{k+1}$ are orthogonal. So we only have to compute $\|\bar{c}_{k+1}\|_2^{-1}$ and set

$$N_{k+1} = \operatorname{diag}(\|\bar{c}_1\|_2^{-1}, \ldots, \|\bar{c}_{k+1}\|_2^{-1}), \tag{51}$$

$$d_{k+1} = \|\bar{c}_{k+1}\|_2^{-2} \bar{c}_{k+1}^{\mathrm{T}} r_k, \tag{52}$$

$$r_{k+1} = r_k - d_{k+1}\bar{c}_{k+1}. \tag{53}$$

After this the relations of Definition 3.1 are again satisfied. Experience with BiCGSTAB as inner iteration indicates that limiting the number of inner iterations and using a large relative tolerance (with respect to the outer residual norm) gives the optimal interplay between the outer and inner iteration. An often employed strategy is to have a maximum of 25 inner iterations and a relative tolerance of 10^{-2}. The inner algorithm is given in Fig. 6. Note that we aligned the algorithms in Figs. 5 and 6 to make the extra steps obvious. The additional work for BiCGSTAB in GCRO with orthogonalization (assuming k outer search vectors) is $2k$ inner products and $2k + 2$ vector updates and some scalar work.

3.5. Truncation in general

When the number of outer iterations becomes large, the multiplication by A_{C_k} will be expensive, and we may run out of memory. Therefore, we consider the truncation of the outer iteration. We will give only a general description here. For a more detailed discussion see [4, 9]. We choose a matrix $W_l \in \mathbb{R}^{k\times l}$, where $l < k$, such that

$$W_l^{\mathrm{T}} W_l = I_l. \tag{54}$$

There are now two ways to implement the truncation:

(1) First, we compute the current approximation according to (40): $x_k = \hat{U}_k Z_k^{-1} d_k$, then we compute the new matrices U_l and C_l as follows

$$C_l = C_k W_l = \bar{C}_k N_k W_l = \bar{C}_k \bar{W}_l \quad \text{where } \bar{W}_l = N_k W_l, \tag{55}$$

and

$$U_l = U_k W_l = \hat{U}_k Z_k^{-1} \bar{W}_l. \tag{56}$$

In order to satisfy the relations of Definition 3.1 we set

$$\begin{aligned} &x_0 = x_k, \quad r_0 = r_k, \\ &\bar{C}_l = C_l, \quad \hat{U}_l = U_l, \\ &Z_l = I_l, \quad d_l = 0, \quad N_l = \operatorname{diag}(1, \ldots, 1), \end{aligned} \tag{57}$$

and we replace relation (39) by $r_l = r_0 - \bar{C}_l d_l$. After this we have to add corrections to x_0 so that (40) must be replaced by $x_l = x_0 + \hat{U}_l Z_l^{-1} d_l$.

(2) The second method does not require the computation of x_k and is therefore more compatible with the overall algorithm. Instead we use w_1 to compute an implicit representation of x_k:

$$w_1 = \delta^{-1} N_k^{-1} d_k \quad \text{where } \delta = \| N_k^{-1} d_k \|_2. \tag{58}$$

The other vectors of W_l can then be chosen freely as long as they satisfy (54). Furthermore, we compute $\bar{C}_l$, $\hat{U}_l$, N_l, Z_l and r_l according to (55)–(57). Finally, we set $d_l = \delta e_1$, so that the relation $r_l = b - \bar{C}_l d_l$ is satisfied:

$$r_l = b - \bar{C}_l d_l = b - \bar{C}_k N_k w_1 \delta = b - \bar{C}_k N_k \delta^{-1} N_k^{-1} d_k \delta = b - \bar{C}_k d_k = r_k.$$

Then x_l is implicitly represented by $x_l = \hat{U}_l Z_l^{-1} d_l$ ($= U_l d_l$).

3.6. Parallel implementation

Both the GMRESR method and the method just described (GCRO) will run efficiently on (large) distributed memory computers, if they use a variant of GMRES that reduces the cost of global communication in the inner products, as described in [3, 2]. Furthermore, GCRO with inner GMRES has the advantage that it converges in almost the same number of iterations as (full) GMRES (see Section 4). However, it has fewer inner products, which require expensive global communication. GCRO with inner GMRES also uses much less memory than GMRES for an equal number of iterations (matrix–vector products). Because the cost of global communication is often the bottleneck for fast parallel implementations of GMRES on large distributed memory computers, and on these computers memory restrictions can be a severe constraint, GCRO with inner GMRES will perform much better than GMRES for many problems on large distributed memory computers.

3.7. Convergence check at the end

Since the residual is never explicitly computed from the approximate solution, it is advisable to check the true residual at the end, when the solution is finally computed. If the norm of the true residual turns out to be larger than the prescribed tolerance, we first reorthogonalize the true residual on $\bar{C}_k$ and compute the corresponding approximate solution. Generally this is sufficient to get the norm of the true residual below the prescribed tolerance. If it is not, then the process can simply be restarted, while keeping the old $\bar{C}_k$ and $\hat{U}_k$ in order to preserve optimality in these directions. Only a few inner iterations were needed in our test cases.

4. Numerical experiments

We will discuss the results of three numerical experiments which concern the solution of two-dimensional convection–diffusion problems on regular grids, discretized by a finite volume technique, resulting in a pentadiagonal matrix. The systems are preconditioned with either ILU(0) applied to the scaled system; see [5] and [11], or with Saad's ILUT preconditioner from the package SPARSEKIT2 (in netlib) [12], which uses a drop tolerance and allows higher level fill-in.

Because we propose the new method (GCRO) to improve the convergence of GMRESR, we use the first two problems to compare the convergence of the following GMRES-like methods:

- (full) GMRES,
- GMRESR(m): m indicates the number of inner GMRES iterations for each outer iteration,
- GCRO(m), which is GCR with m GMRES iterations for A_{C_k} as inner method.

The examples illustrate the typical differences of these methods, the stagnation in the inner loop of GMRESR(m) that slows down the convergence, the optimality of GMRES in matrix–vector products and its high computational cost, and the near-optimal convergence of GCRO(m) and its much lower computational cost compared with GMRES. The examples illustrate the potential of GCRO(m) very well.

The last problem is used to illustrate that instead of a GMRES-like method in the inner loop we can also use other methods. Apart from the GMRES-like methods given above, we compare the following methods based on BiCGSTAB:

- BiCGSTAB,
- GMRESRSTAB: GCR with BiCGSTAB as inner method,
- GCROSTAB: GCR with BiCGSTAB for A_{C_k} as inner method.

This example also shows that the GCRO approach may improve the convergence of BiCGSTAB-like methods. The example also indicates a problem that may arise in using such methods in the GMRESR approach.

We will compare the convergence of these methods both with respect to time (on one processor of a Convex C3840) and with respect to the number of matrix–vector products. This makes sense since the main trade-off between (full) GMRES, the GCRO variants, and the GMRESR variants is fewer iterations against less work per iteration. Which method converges faster in time, then, highly depends on the relative cost of the matrix–vector product and preconditioning.

Problem 1. The first problem is defined by the discretization of

$$-(u_{xx} + u_{yy}) + bu_x + cu_y = 0$$

on $[0,1] \times [0,4]$, where

$$b(x,y) = \begin{cases} 200 & \text{for } 0 \leqslant y \leqslant 1, \\ -200 & \text{for } 1 < y \leqslant 2, \\ 200 & \text{for } 2 < y \leqslant 3, \\ -200 & \text{for } 3 < y \leqslant 4, \end{cases}$$

and $c = 200$. The boundary conditions are $u = 1$ on $y = 0$, $u = 0$ on $y = 4$, $u' = 0$ on $x = 0$ and $u' = 0$ on $x = 1$, where u' denotes the (outward) normal derivative, see Fig. 9. The stepsize in the x-direction is $1/99$ and in the y-direction it is $4/199$.

The convergence history for Problem 1 is given in Figs. 7 and 8 for (full) GMRES, GCRO(m) and GMRESR(m), for $m = 5$ and $m = 10$.

Fig. 7 shows that (full) GMRES converges fastest (in iterations), which is of course to be expected, followed by GCRO(5), GCRO(10), GMRESR(10), and GMRESR(5). From Fig. 7 we can see that GCRO(m) converges much smoother and much faster than GMRESR(m) and follows the convergence of (full) GMRES quite well. The vertical "steps" of GMRESR(m) are caused by the optimization in the outer iteration, which does not involve a matrix–vector product. This figure clearly shows that the GMRESR variants suffer severely from stagnation in the inner iteration, which makes the convergence very slow. Stagnation in the inner iteration occurs frequently in GMRESR(m) and often destroys the superlinear convergence behaviour, at least during certain stages of the convergence history. The reason is probably that the inner iteration of GMRESR (m) is essentially a restarted GMRES. Fig. 8 gives the convergence with respect to time. GCRO(5) is the fastest, which is not surprising in view of the fact that it follows the convergence of (full) GMRES quite closely, but has much lower cost per iteration. GCRO(10) is also faster than (full) GMRES,

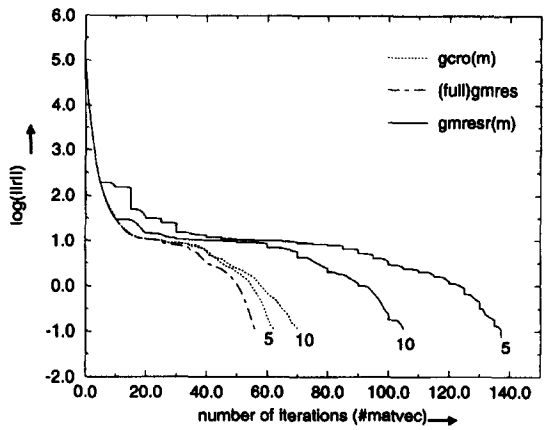

Fig. 7. Convergence vs. number of iterations for Problem 1.

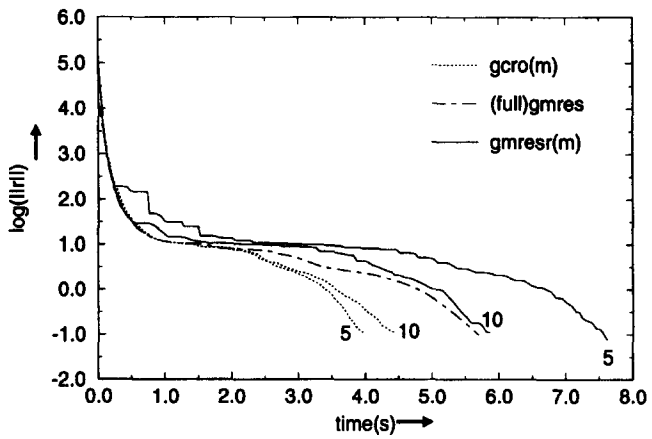

Fig. 8. Convergence vs. time for Problem 1.

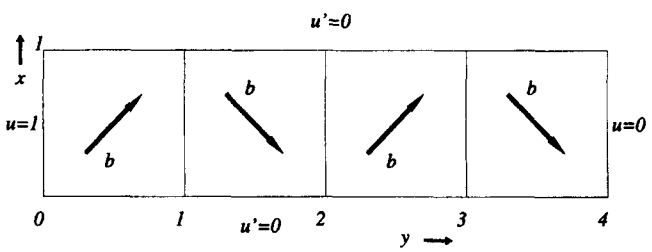

Fig. 9. Problem 1.

which is still faster than GMRESR(10) and GMRESR(5). This shows that, for this example, the improvement in convergence for the GCRO variants outweighs the extra work in orthogonalizations compared to the GMRESR variants. In fact, the effects of stagnation are so strong for this example that the GMRESR variants are even slower than (full) GMRES. Apart from their fast

convergence in time, the GCRO variants also have the advantage of much lower memory requirements than (full) GMRES (less than half in this example); this, of course, also holds for GMRESR.

Problem 2. The second problem is defined by the discretization of

$$-(u_{xx} + u_{yy}) + bu_x + cu_y = 0$$

on $[-1,1] \times [-1,1]$, where $b(x,y) = 400y(1-x^2)$ and $c(x,y) = -400x(1-y^2)$. The boundary conditions are $u = 1$ on $x = 1$ and $y = -1$, and $\partial u/\partial n = 1 - u$ on $x = -1$ and $y = 1$, where $\partial u/\partial n$ denotes the outward normal derivative; see Fig. 10. The stepsize is 1/200 in both the x- and y-direction. For this type of problem ILU(0) is not a good preconditioner. Therefore, we used the ILUT preconditioner described in [12], with a drop tolerance of 0.0001 and a maximum fill-in of 5 in both the lower- and upper-triangular factor.

The results of Problem 2 are given in Figs. 12 and 13. In this example, the difference between (full) GMRES and GCRO(5) in convergence against number of matrix–vector products is again very

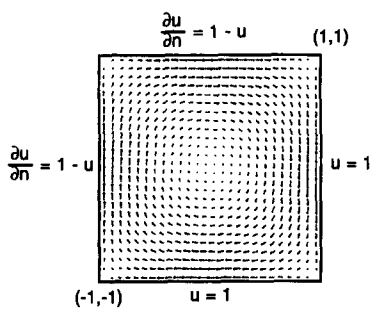

Fig. 10. Problem 2.

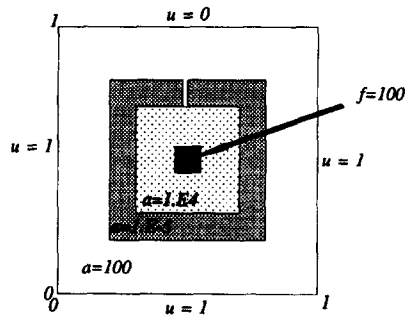

Fig. 11. Problem 3.

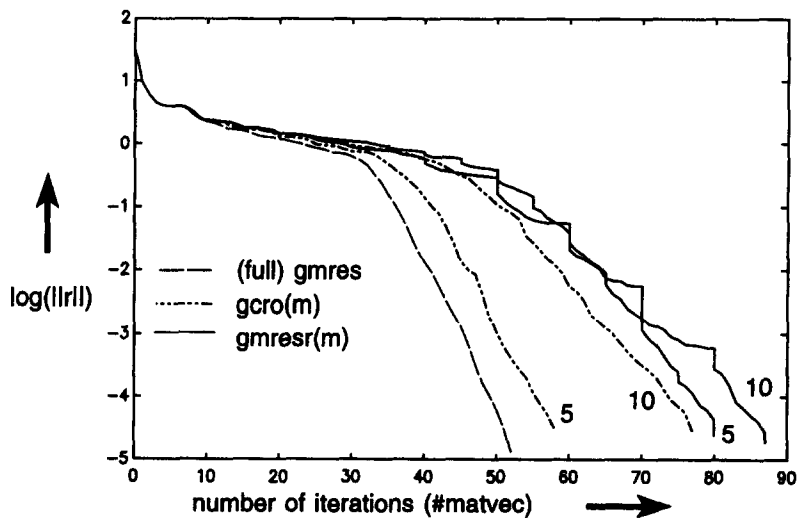

Fig 12. Convergence vs. number of iterations for Problem 2.

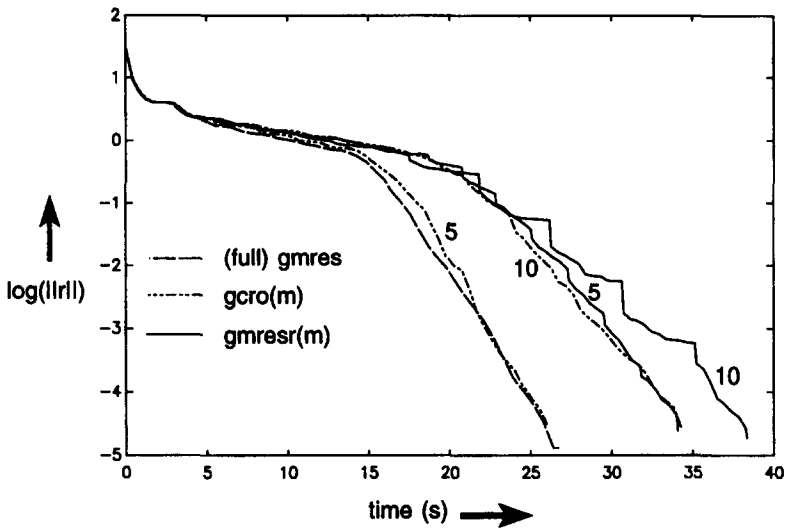

Fig. 13. Convergence vs. time for Problem 2.

small; GCRO(5) converges (almost) exactly like (full) GMRES. This is explained by the "global" optimization in GCRO(m) over both the inner and the outer search vectors (the latter form some "best" sample of the entire, previously searched Krylov subspace). In fact, we may view GCRO(m) as a semi-full GMRES. Fig. 12 shows again the tendency of GMRESR(m) to stagnate in the inner iteration (especially for $m = 10$), which causes it to converge much slower than GCRO(m). Fig. 13 shows that GCRO(5) and (full) GMRES are the fastest in CPU time. We also see (as in the previous example) that GCRO(5) converges faster than GCRO(10). It seems that a good interaction between the inner and outer iteration is important for good convergence.

Problem 3. The third problem is taken from [17]. We solve the equation

$$-(au_x)_x - (au_y)_y + bu_x = f$$

on the unit square, where b is given by $b(x, y) = 2e^{2(x^2+y^2)}$. We have Dirichlet boundary conditions, see Fig. 11. The functions a and f are also given in Fig. 11: $f = 0$ everywhere, except for the small square in the centre, where $f = 100$. The stepsize in both the x- and y-directions is 1/128. An ILU(0) preconditioner was used (as in [17]). The results of Problem 3 are given in Figs. 14–16.

First, we consider the GMRES variants. Fig. 14 gives the convergence history for (full) GMRES, GCRO(m) and GMRESR(m). For this problem, we used $m = 10$ because it was the optimal choice for both GMRESR(m) and GCRO(m), and we used $m = 50$ to highlight the difference in convergence behaviour in the inner iteration of GMRESR(m) and GCRO(m). GMRESR(50) stagnates in the inner GMRES iteration, whereas GCRO(50) displays almost the same convergence behaviour as GCRO(10) and (full) GMRES.

Next, we consider BiCGSTAB as inner method to illustrate that methods other than GMRES can be used successfully as inner iteration. This also indicates the potential of GCRO to improve (accelerate) the convergence of methods like BiCGSTAB. In Fig. 15 the convergence history is

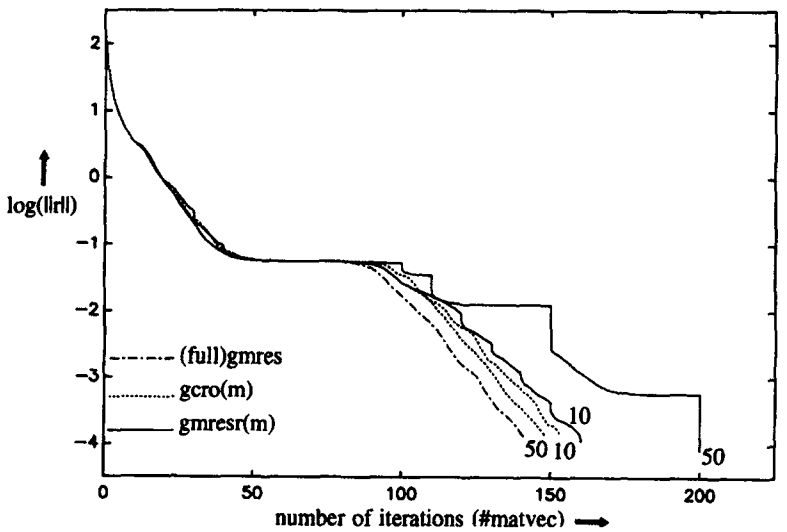

Fig. 14. Convergence vs. number of iterations for Problem 3.

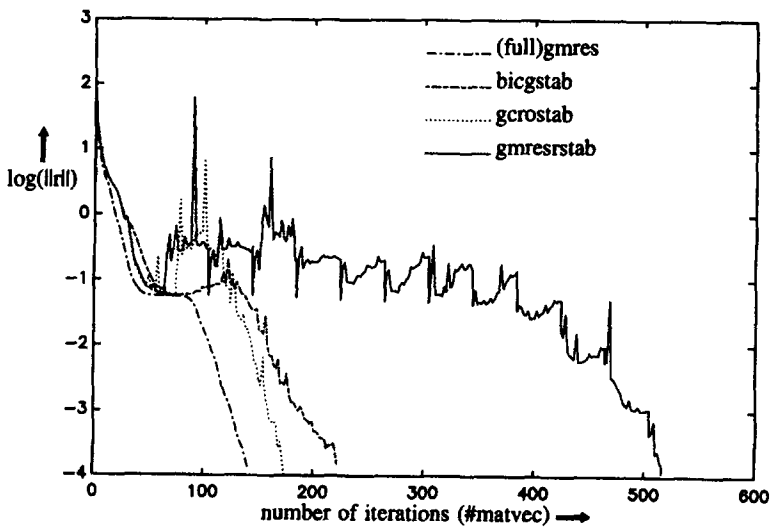

Fig. 15. Convergence for BiCGSTAB variants vs. number of iterations for Problem 3.

given for (full) GMRES (for reference), BiCGSTAB, and BiCGSTAB as the inner method GMRESR (GMRESRSTAB) and in GCRO (GCROSTAB). We use different parameters l GMRESRSTAB and GCROSTAB, because they behave quite differently, and for both metho the optimal parameters (within a reasonable region) should be used to make a fair comparison. T following strategies gave the best results for the respective BiCGSTAB variants:

- For GMRESRSTAB the inner iteration was ended after either 20 steps or a relative reduction the residual norm by a factor 0.01.

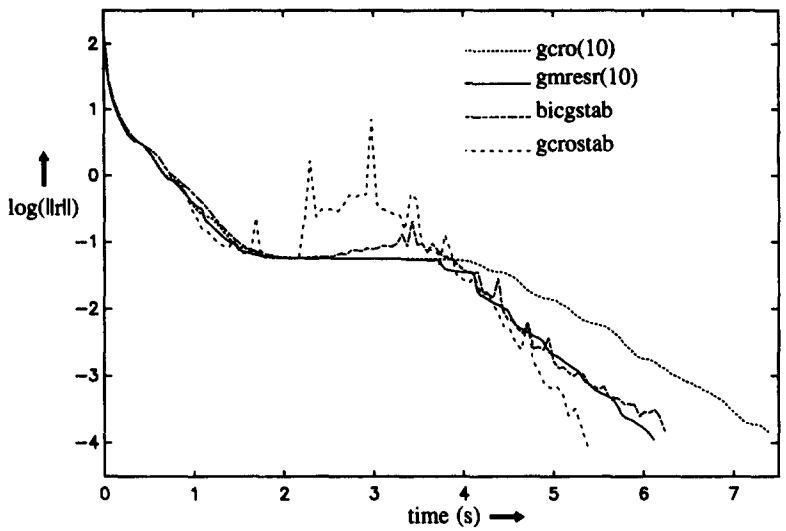

Fig. 16. Convergence vs. time for Problem 3.

• For GCROSTAB the inner iteration was ended after 25 steps or a relative reduction of the residual norm by a factor of 0.01.

The convergence behaviour of GMRESRSTAB for this example is somewhat typical for GMRESRSTAB in general (albeit very bad in this particular case). A reason for this erratic behaviour may be that the convergence of BiCGSTAB depends on the (implicit) (bi-)orthogonality to some basis. This relation is destroyed after each outer iteration. Now if the inner iteration does not yield a good approximation, the outer iteration will not give much improvement either and the method becomes "trapped". At the start, the same seems to hold for GCROSTAB. However, after a few outer GCR iterations the "improved" operator A_{C_k} somehow yields a better convergence than BiCGSTAB by itself. We have also observed this for other tests, although it may also happen that GCROSTAB converges worse than BiCGSTAB.

In Fig. 16 the convergence versus CPU time is given for the methods with the best performance (in time), GCROSTAB, BiCGSTAB, GCRO(10) and GMRESR(10). GCROSTAB gives the best convergence in time. It is approximately 20% faster than BiCGSTAB, notwithstanding the extra work in orthogonalizations. Although GCRO(10) converges in fewer iterations than GMRESR(10), in time GMRESR(10) is faster. So in this case the decrease in iterations does not outweigh the extra work in orthogonalizations. For completeness we mention that GMRESRSTAB took almost 15 s to converge, whereas (full) GMRES took about 20 s.

5. Conclusions

From the GMRESR methods we have derived a modified set of methods, which preserve the optimality of the outer method in the inner iteration. This optimality is lost in the inner iteration of

GMRESR since it essentially uses "restarted" (inner) iterations which cannot take advantage of the "convergence history". Therefore, GMRESR(m) may converge very slowly and lose the superlinear convergence behaviour of (full) GMRES, due to poor convergence of the inner, restarted GMRES, and this might repeat itself at each inner iteration.

In contrast, the GCRO variants exploit the "convergence history" to generate a search space in the inner iteration that does not interfere with the previous minimization of the error over the space of outer search vectors. If then GMRES is used as the inner method (GCRO(m)), we do a global minimization of the error over both the inner search space and the outer search space. The set of outer search vectors is a sample of the entire, previously searched Krylov subspace. From this point of view, we may say that GCRO(m) is a semi-full GMRES. This probably accounts for the smooth convergence, the preservation of superlinear convergence (if GMRES converges superlinearly), and the absence of stagnation, which may occur in the inner method of GMRESR. In many practical problems, the convergence behaviour in iterations of GCRO(m) is almost the same as that of (full) GMRES. Apparently the subset of Krylov subspace vectors that is maintained approximates the entire Krylov subspace that has been generated sufficiently well. Because GCRO(m) requires fewer inner products and needs less storage than (full) GMRES for the same number of iterations, it is better suited for implementation on large, distributed memory, parallel computers.

Although there is the possibility of breakdown in the inner method for GCRO, this seems to occur rarely, as is indicated by Theorem 2.7; indeed, it has never happened in any of our experiments. Moreover, in case the generation of the Krylov subspace does break down, we have suggested two ways to continue.

With respect to the performance of the discussed methods, we see that GCRO(m) (almost) always converges in fewer iterations than GMRESR(m). Because GCRO(m) is on the average more expensive per iteration, this does not always lead to faster convergence in time. This depends on the costs of the matrix–vector product and preconditioner compared to the costs of the orthogonalizations. Our experiments, with a relatively inexpensive matrix–vector product and preconditioner, show that even in this case the GCRO variants are very competitive with other solvers. However, especially when the matrix–vector product and preconditioner are expensive or when not enough memory is available for (full) GMRES, GCRO(m) is very attractive. For both GMRESR(m) and GCRO(m) it appears that a small number of inner iterations works best.

GCRO with BiCGSTAB also seems to be a good method, especially when a large number of iterations is necessary, or when the available memory is small relative to the problem size. GMRESR with BiCGSTAB does not seem to work well. This may be caused by the fact that after one outer iteration the restarted BiCGSTAB has lost the (implicit) bi-orthogonality relations constructed in the previous iteration(s).

Acknowledgements

The author wishes to acknowledge Shell Research B.V. and STIPT for the financial support of his research. Furthermore, the author wishes to acknowledge the anonymous referees for their help in the presentation of this article.

References

[1] O. Axelsson and P.S. Vassilevski, A black box generalized conjugate gradient solver with inner iterations and variable-step preconditioning, *SIAM J. Matrix Anal. Appl.* **12** (1991) 625–644.

[2] E. De Sturler, A parallel restructured version of GMRES(*m*), Technical Report 91–85, Faculty of Technical Mathematics and Informatics, Delft University of Technology, Delft, Netherlands, 1991.

[3] E. De Sturler, A parallel variant of GMRES(*m*), in: J.J.H. Miller and R. Vichnevetsky, Eds., *Proc. 13th IMACS World Congr. on Computation and Applied Mathematics*, Dublin, Ireland (Criterion Press, Dublin, 1991) 682–683.

[4] E. De Sturler and D.R. Fokkema, Nested Krylov methods and preserving the orthogonality, in: N. Duane Melson, T.A. Manteuffel and S.F. McCormick, Eds., *6th Copper Mountain Conf. on Multigrid Methods*, NASA Conf. Publication 3224, Part 1, Hampton, VA (NASA Langley Research Center, 1993) 111–125.

[5] J.J. Dongarra, I.S. Duff, D.C. Sorensen and H.A. Van der Vorst, *Solving Linear Systems on Vector and Shared Memory Computers* (SIAM, Philadelphia, PA, 1991).

[6] S.C. Eisenstat, H.C. Elman and M.H. Schultz, Variational iterative methods for nonsymmetric systems of linear equations, *SIAM J. Numer. Anal.* **20** (1983) 345–357.

[7] V. Faber and T. Manteuffel, Necessary and sufficient conditions for the existence of a conjugate gradient method, *SIAM J. Numer. Anal.* **21** (1984) 352–362.

[8] R. Fletcher, Conjugate gradient methods for indefinite systems, in: G.A. Watson, Ed., *Numerical Analysis Dundee* 1975, Lecture Notes in Math., Vol. 506 (Springer, Berlin, 1976) 73–89.

[9] D.R. Fokkema, Hybrid methods based on the GCR principle, to appear.

[10] R.W. Freund and N.M. Nachtigal, QMR: a quasi minimal residual method for non-Hermitian linear systems, *Numer. Math.* **60** (1991) 315–339.

[11] J.A. Meijerink and H.A. Van der Vorst, An iterative solution method for linear equations systems of which the coefficient matrix is a symmetric *M*-matrix, *Math. Comp.* **31** (1977) 148–162.

[12] Y. Saad, ILUT: a dual threshold incomplete ILU factorization, Technical Report 92-38, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, 1992; *Linear Algebra Appl.*, to appear.

[13] Y. Saad, A flexible inner–outer preconditioned GMRES algorithm, *SIAM J. Sci. Statist. Comput.* **14** (1993) 461–469.

[14] Y. Saad and M. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.* **7** (1986) 856–869.

[15] G.L.G. Sleijpen and D.R. Fokkema, BiCGstab(l) for linear equations involving matrices with complex spectrum, *Elec. Trans. Numer. Anal. (ETNA)* **1** (1993) 11–32.

[16] P. Sonneveld, CGS, a fast Lanczos-type solver for nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.* **10** (1989) 36–52.

[17] H.A. 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) 631–644.

[18] H.A. Van der Vorst and C. Vuik, GMRESR: a family of nested GMRES methods, Technical Report 91-80, Faculty of Technical Mathematics and Informatics, Delft University of Technology, Delft, Netherlands, 1991.