SIAM J. NUMER. ANAL.
Vol. 36, No. 3, pp. 864–889

# TRUNCATION STRATEGIES FOR OPTIMAL KRYLOV SUBSPACE METHODS[*]

ERIC DE STURLER[†]

**Abstract.** Optimal Krylov subspace methods like GMRES and GCR have to compute an orthogonal basis for the entire Krylov subspace to compute the minimal residual approximation to the solution. Therefore, when the number of iterations becomes large, the amount of work and the storage requirements become excessive. In practice one has to limit the resources. The most obvious ways to do this are to restart GMRES after some number of iterations and to keep only some number of the most recent vectors in GCR. This may lead to very poor convergence and even stagnation. Therefore, we will describe a method that reveals which subspaces of the Krylov space were important for convergence thus far and exactly how important they are. This information is then used to select which subspace to keep for orthogonalizing future search directions. Numerical results indicate this to be a very effective strategy.

**Key words.** GMRES, GCR, restart, truncation, Krylov subspace methods, iterative methods, non-Hermitian linear systems

**AMS subject classifications.** Primary, 65F10; Secondary, 15A18, 65N22

**PII.** S0036142997315950

## 1. Introduction.

Suppose we need to solve the nonsingular, linear system of equations

$$Ax = b, \tag{1.1}$$

with the initial guess $x_0$ and its residual $r_0 = b - Ax_0$. In the $i$th iteration the GCR method [10] and the GMRES method [22] compute the correction $z_i$ in the $i$th Krylov subspace for $A$ and $r_0$, $K^i(A, r_0) \equiv \operatorname{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{i-1}r_0\}$, that minimizes the norm of the residual $r_i = b - A(x_0 + z_i)$. The relation between the minimal residual correction, $z_i$, and the orthogonality of the new residual to the *shifted Krylov space* $AK^i(A, r_0) \equiv \operatorname{span}\{Ar_0, A^2r_0, \ldots, A^ir_0\}$ is given by the following well-known theorem (e.g., see [21] or [6]).

THEOREM 1.1. *The vector $z_i \in K^i(A, r_0)$ satisfies*

$$z_i = \arg \min_{z \in K^i(A,r_0)} \|b - A(x_0 + z)\|_2$$

*if and only if*

$$r_0 - Az_i \perp AK^i(A, r_0). \tag{1.2}$$

Hence, residual minimizing Krylov subspace methods compute an orthogonal basis for the (shifted) Krylov space. However, for a large number of iterations the orthogonalization is prohibitively expensive in work and in memory requirements, and in practice only incomplete orthogonalization is used. For GMRES one periodically computes the new approximate solution and residual and then restarts, throwing away

[*]Received by the editors February 3, 1997; accepted for publication (in revised form) August 14, 1998; published electronically April 20, 1999.
http://www.siam.org/journals/sinum/36-3/31595.html

[†]Swiss Center for Scientific Computing (SCSC-ETHZ), Swiss Federal Institute of Technology, ETH Zentrum, CH-8092 Zurich, Switzerland (sturler@scsc.ethz.ch).

all previously computed information. For GCR, apart from restarting, one can also limit the orthogonalization of the new search vector to the most recent search vectors; also, here information from earlier iterations is eventually lost. It is well known that truncation and restarting may lead to very poor convergence and even stagnation in cases where the "full" versions converge quite well.

We will now briefly review the mathematical relations for GCR [10] and GMRES [22] solving (1.1).

Given an initial guess $x_0$ and its residual $r_0 = b - Ax_0$, the GCR method computes in $m$ iterations two matrices $U_m = [u_1\, u_2\, \cdots\, u_m]$ and $C_m = [c_1\, c_2\, \cdots\, c_m]$ that satisfy the following relations:

$$\text{range}(U_m) = K^m(A, r_0) = \text{span}\{r_0, Ar_0, A^2r_0, \dots, A^{m-1}r_0\}, \tag{1.3}$$

$$AU_m = C_m, \tag{1.4}$$

$$C_m^H C_m = I_m. \tag{1.5}$$

The minimal residual solution is then computed from the orthogonal projection of $r_0$ onto range$(C_m)$. This gives

$$r_m = (I - C_m C_m^H) r_0, \tag{1.6}$$

$$x_m = x_0 + U_m C_m^H r_0. \tag{1.7}$$

A single iteration of GCR (say $m+1$) can be written as follows:

$$c_{m+1} = (I - C_m C_m^H) A r_m / \|(I - C_m C_m^H) A r_m\|_2, \tag{1.8}$$

$$u_{m+1} = (r_m - U_m C_m^H A r_m) / \|(I - C_m C_m^H) A r_m\|_2, \tag{1.9}$$

$$r_{m+1} = r_m - c_{m+1} c_{m+1}^H r_m, \tag{1.10}$$

$$x_{m+1} = x_m + u_{m+1} c_{m+1}^H r_m. \tag{1.11}$$

In the standard truncated version we orthogonalize $Ar_m$ only on the $l$ most recent vectors; so we replace (1.8) and (1.9) with

$$c_{m+1} = (I - C_{m,l} C_{m,l}^H) A r_m / \|(I - C_{m,l} C_{m,l}^H) A r_m\|_2,$$
$$u_{m+1} = (r_m - U_{m,l} C_{m,l}^H A r_m) / \|(I - C_{m,l} C_{m,l}^H) A r_m\|_2,$$

where $C_{m,l} = [c_{m-l+1}\, \dots\, c_m]$ and $U_{m,l} = [u_{m-l+1}\, \dots\, u_m]$.

The GMRES method computes an implicit representation of the matrices $U_m$ and $C_m$, which leads to a more efficient and robust algorithm. Given an initial guess $x_0$ and its residual $r_0$, the GMRES method computes in $m$ iterations the matrices $W_{m+1} = [w_1\, w_2\, \dots\, w_{m+1}]$, $\bar{H}_m$, $Q_m$, and $\bar{R}_m$ that satisfy the following relations [22]:

$$AW_m = W_{m+1} \bar{H}_m, \quad w_1 = r_0 / \|r_0\|_2, \tag{1.12}$$

$$W_{m+1}^H W_{m+1} = I_{m+1}, \tag{1.13}$$

$$\bar{H}_m = Q_m \bar{R}_m. \tag{1.14}$$

Here $Q_m \in \mathbb{C}^{(m+1)\times(m+1)}$ is the product of the $m$ Givens rotations that reduce the upper Hessenberg matrix $\bar{H}_m$ to the upper-triangular matrix $\bar{R}_m \in \mathbb{C}^{(m+1)\times m}$. Since the last row of $\bar{R}_m$ consists entirely of zeros, we can drop the last row of $\bar{R}_m$ and the last column of $Q_m$ to get the QR-decomposition of $\bar{H}_m$,

$$\bar{H}_m = \bar{Q}_m R_m. \tag{1.15}$$

So $(W_{m+1}\bar{Q}_m)R_m$ is the QR-decomposition of $AW_m$. We can now write the GMRES solution in an equivalent form as the GCR solution.

$$\begin{aligned} r_m &= (I - (W_{m+1}\bar{Q}_m)(W_{m+1}\bar{Q}_m)^H)r_0 \\ &= r_0 - W_{m+1}\bar{Q}_m R_m R_m^{-1} \bar{Q}_m^H \|r_0\|_2 e_1 \\ &= r_0 - W_{m+1}\bar{H}_m R_m^{-1} \bar{Q}_m^H \|r_0\|_2 e_1, \qquad (1.16) \\ x_m &= x_0 + W_m R_m^{-1} \bar{Q}_m^H \|r_0\|_2 e_1. \qquad (1.17) \end{aligned}$$

Comparing the GCR and the GMRES algorithm we see that

$$\begin{aligned} C_m &= W_{m+1}\bar{Q}_m, \\ U_m &= W_m R_m^{-1}. \end{aligned}$$

We would like to have a method that converges like full GMRES, but uses only limited resources. Several methods derived from GMRES or GCR have been proposed recently: GMRESR [25], FGMRES [20], and GCRO [9, 7, 6]. These methods strike a balance between optimality and cost by keeping a limited number of vectors from previous search spaces, hence doing some form of truncation. However, GCRO is the only method that computes the *optimal* solution over the subspace spanned by the new and old search vectors. The GCRO results in [9, 7] show that this form of truncation—maintaining orthogonality to part of the “old” Krylov subspace—in many cases yields very good convergence with significantly reduced costs compared to full GMRES.

However, none of the above-mentioned methods has a mechanism for selecting the best vectors to keep. They merely keep the correction (vector) to the residual in each outer iteration. To find the best truncation, we should not restrict the truncation to discarding or keeping vectors that happen to arise in the iteration, but must consider arbitrary subspaces of the space spanned by all available vectors [7].

The optimal solution in methods like GMRES and GCR is obtained through the orthogonality constraints in (1.2). If we truncate (or restart), we will in general lose orthogonality to the discarded subspace, and this will result in a suboptimal correction. Therefore, we must analyze the error we make by neglecting orthogonality constraints, and truncate such that this error is minimized. In the next section we will derive an equation for the difference between the optimal residual and the residual after truncation. We will refer to this vector as the *residual error*. This equation expresses the residual error using a set of values that are related to the principal angles [2], [13, pp. 584–585] between the subspace discarded by the truncation and the space onto which we compute the orthogonal projection. We will use this equation to decide which subspace of the Krylov space we must keep and which subspace we can discard while minimizing the norm of the residual error or at least bounding it from above.

Recently, several authors have considered keeping some vectors from the search space of a GMRES(m) cycle after a restart. They all concentrate on (absolute) small eigenvalues. Morgan [19] and Chapman and Saad [5] try to remove small eigenvalues from the spectrum within a GMRES(m) iteration. They proceed as follows. First they carry out the Arnoldi iteration in the standard way (1.12)–(1.14). Then they append approximate eigenvectors computed in the previous GMRES(m) cycle to the Arnoldi vectors $W_m$, and use the space spanned by both the Arnoldi vectors and the approximate eigenvectors to compute the minimal residual solution. Finally, they compute new approximations to the eigenvectors and restart. Morgan [19] refers to this method as *augmented GMRES*. In [5] the authors also discuss other enhancements. Baglama, Calvetti, Golub, and Reichel [1] derive a preconditioner from the Arnoldi iteration(s) that moves small eigenvalues away from the origin.

The underlying idea for such strategies is that the worst-case bound on the convergence is reduced by improving the spectrum. However, for nonnormal matrices, typically strongly nonsymmetric matrices, these upper bounds may say very little about the actual convergence. Besides, slow convergence is not necessarily caused by small eigenvalues. When some eigenvalues are located equidistantly on a circle with the origin at the center, GMRES will stagnate for a number of steps equal to the number of eigenvalues on this circle [24], whether they are small or large. In addition, if the eigenvalues cluster around the origin, it may be impossible to even find the smallest eigenvalues to sufficient accuracy. The same holds when the matrix is nonnormal. Moreover, in the nonnormal case slow convergence may not be caused by small eigenvalues at all. In a recent paper, Greenbaum, Pták, and Strakoš [14] show that matrices can be constructed for which full GMRES gives arbitrarily poor (nonincreasing) convergence, irrespective of the eigenvalues. So a favorable spectrum does not help, and improving the spectrum may be useless. Some of these problems are also mentioned in [19] and illustrated in the examples given there.

An additional problem of adding approximate eigenvectors to the Krylov subspace after the Arnoldi iteration is that the Arnoldi iteration itself remains a restarted GMRES iteration. This can lead to stagnation problems like those reported for GMRESR in [3, 4]. Appending vectors from the previous GMRES(m) cycle when restarted GMRES stagnates does not bring any improvement, because the residual is orthogonal to these vectors. However, if from the start of the Arnoldi iteration we maintain orthogonality to a selected subspace, we generate a new space orthogonal to this selected subspace. This often leads to much better convergence. Preventing stagnation problems was one of the main motivations for GCRO [7] as an improvement to GMRESR. Also Blom and Verwer report stagnation problems with GMRESR that are remedied by GCRO [3, 4] .

We think our method will be competitive for a wide range of problems. However, especially for strongly nonsymmetric systems, we expect the method introduced here to be better than the methods mentioned above. In the present paper we will outline our approach and illustrate it for five test problems, three of which are taken from [19], to compare with augmented GMRES.

In section 2 we derive the theory on which our truncation algorithm is based, in section 3 we discuss the implementation, in section 4 we present our numerical results, and in section 5 we give some conclusions and outline future work.

## 2. Neglecting orthogonality and optimal truncation.

We will now derive a set of equations for the residual error.

Consider the following case. We have already computed the optimal approximation to $b$ in range$(C)$, and the new residual is given by $r = b - CC^H b$, where $C$ is a matrix with orthonormal columns. Now let $F$ be a matrix with full column rank, and let $\dim(\text{range}(C) \oplus \text{range}(F)) = \dim(\text{range}(C)) + \dim(\text{range}(F))$. Then the QR-decomposition $QR = F - CC^H F$ yields the best approximation to $r$ in the space range$(C) \oplus$ range$(F)$, $QQ^H r$. On the other hand, the QR-decomposition $F = WS$ yields the best approximation to $r$ in the subspace range$(F)$, $WW^H r$. The difference between these two approximations is the *residual error* $e$. The residual error depends on the *principal angles* [2], [13, pp. 584–585] between the subspaces range$(C)$ and range$(F)$, and one way to analyze the consequences of neglecting orthogonality is to study these principal angles. However, we follow a slightly different, but equivalent, strategy. Instead of looking at the principal angles we look at the (length of the) residual error $e$. This approach is cheaper.[1]

Definition 2.1. *Let the matrices $C \in \mathbb{C}^{n\times k}$, $F \in \mathbb{C}^{n\times m}$, and the vector $r \in \mathbb{C}^n$ be given, such that*

$$C^H C = I_k, \tag{2.1}$$
$$C^H r = 0, \tag{2.2}$$
$$\operatorname{rank}(F) = m, \tag{2.3}$$
$$\operatorname{rank}(F - CC^H F) = m. \tag{2.4}$$

*Furthermore, let*

$$F = CB + QR, \tag{2.5}$$
$$Q^H Q = I_m, \tag{2.6}$$

*where $B = C^H F$ and $R$ is upper-triangular. Using $B$ and $R$ we define*

$$Z = BR^{-1} \quad = \quad (C^H F)(Q^H F)^{-1}, \tag{2.7}$$
$$K = Z^H Z, \tag{2.8}$$

*and we denote the singular value decomposition of $Z$ by*

$$Z = Y_Z \Sigma_Z V_Z^H. \tag{2.9}$$

*$Y_Z = [y_1\ y_2\ \ldots\ y_k]$ and $V_Z = [v_1\ v_2\ \ldots\ v_m]$ are ordered so as to follow the convention that*

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p,$$

*where $p = \min(k, m)$. Also let*

$$F = WS, \tag{2.10}$$
$$W^H W = I_m, \tag{2.11}$$

*and $S$ be upper-triangular. Finally, let*

$$r_1 = (I - QQ^H)r, \tag{2.12}$$
$$r_2 = (I - WW^H)r, \tag{2.13}$$

*and let the residual error be $e = r_2 - r_1$.*

In Definition 2.1, $r_1$ is the residual corresponding to the best approximation to $r$ in the space $\operatorname{range}(C) \oplus \operatorname{range}(F)$, whereas $r_2$ is the residual corresponding to the best approximation to $r$ in the space $\operatorname{range}(F)$ ignoring the orthogonality to $\operatorname{range}(C)$. The residual error is the difference between $r_2$ and $r_1$. Note that $C^H Q = O$ and that from (2.3)–(2.4) we know that $R$ and $S$ are nonsingular.

Theorem 2.2. *The residual error $e$ is given by*

$$r_2 - r_1 = \sum_{i=1}^{p} \left( \frac{\nu_i \sigma_i^2}{1+\sigma_i^2} Q v_i - \frac{\nu_i \sigma_i}{1+\sigma_i^2} C y_i \right), \tag{2.14}$$

[1]In practice we often do not have $F$ explicitly available, and in our approach we do not need to orthogonalize $F$.

*where $\nu_i = v_i^H Q^H r$, and the norm of the residual error is given by*

$$\|r_2 - r_1\|_2 = \left( \sum_{i=1}^{p} \frac{|\nu_i|^2 \sigma_i^2}{1 + \sigma_i^2} \right)^{1/2} . \tag{2.15}$$

*Proof.* Equations (2.12)–(2.13) give

$$r_2 - r_1 = QQ^H r - WW^H r. \tag{2.16}$$

From (2.5) and (2.10) we can derive

$$W = CBS^{-1} + QRS^{-1},$$

which leads to

$$WW^H r = CB(S^H S)^{-1} R^H Q^H r + QR(S^H S)^{-1} R^H Q^H r, \tag{2.17}$$

using $C^H r = 0$ from Definition 2.1. Again using (2.5)–(2.6) and (2.10)–(2.11), we see that

$$\begin{aligned}(WS)^H(WS) = (CB + QR)^H(CB + QR) \quad &\Leftrightarrow \\ S^H S = B^H B + R^H R,&\end{aligned}$$

so that

$$\begin{aligned} R(S^H S)^{-1} R^H &= (R^{-H}(S^H S)R^{-1})^{-1} \\ &= (R^{-H} B^H B R^{-1} + I)^{-1} \\ &= (Z^H Z + I)^{-1} \\ &= (K + I)^{-1}. \end{aligned} \tag{2.18}$$

Note that all inverses above are well defined since $R$ and $S$ are nonsingular by definition. Substituting (2.18) into (2.17) gives

$$\begin{aligned} WW^H r &= CBR^{-1} R(S^H S)^{-1} R^H Q^H r + QR(S^H S)^{-1} R^H Q^H r \\ &= CZ(I + K)^{-1} Q^H r + Q(I + K)^{-1} Q^H r, \end{aligned} \tag{2.19}$$

and then substituting (2.19) into (2.16) gives

$$\begin{aligned} r_2 - r_1 &= QQ^H r - CZ(I + K)^{-1} Q^H r - Q(I + K)^{-1} Q^H r \\ &= QK(I + K)^{-1} Q^H r - CZ(I + K)^{-1} Q^H r. \end{aligned} \tag{2.20}$$

Using (2.9) we can rewrite (2.20) as

(2.21)
$$r_2 - r_1 = QV_Z(\Sigma_Z^H \Sigma_Z)(I + \Sigma_Z^H \Sigma_Z)^{-1} V_Z^H Q^H r - CY_Z \Sigma_Z (I + \Sigma_Z^H \Sigma_Z)^{-1} V_Z^H Q^H r.$$

From $\nu_i = v_i^H Q^H r$ we have

$$V_Z^H Q^H r = \sum_{i=1}^{m} \nu_i e_i, \tag{2.22}$$

where $e_i$ is the $i$th Cartesian basis vector. Finally, the substitution of (2.22) into (2.21) gives

$$r_2 - r_1 = \sum_{i=1}^{p} \left( \frac{\nu_i \sigma_i^2}{1+\sigma_i^2} Q v_i - \frac{\nu_i \sigma_i}{1+\sigma_i^2} C y_i \right),$$

and the norm of the residual error follows immediately from the orthogonality of $C$ and $Q$. $\square$

Theorem 2.2 indicates that for $\sigma_i = 0$, corresponding to a direction in range($F$) orthogonal to range($C$), the associated component in the residual error is zero, and for $\sigma_i \to \infty$, corresponding to a direction in range($F$) that becomes dependent with range($C$), the associated component in the residual error equals the optimal correction. Thus, no correction is made in that direction.

We have derived equations for the residual error that show the consequences of neglecting the orthogonality to range($C$), that is, the consequences of discarding range($C$) by truncation or restart. In the next subsection we show how Theorem 2.2 can be used to obtain the (components of the) residual error in the case of discarding an arbitrary subspace of range($C$). This will then be used to select subspaces to discard or to keep in order to maintain good convergence at low cost.

**Optimal truncation.** We will now use the results of the previous subsection to determine which subspace of range($C$) should be kept and what can be discarded. We consider computing the residual $r_3$ while maintaining orthogonality to the subspace range($CT$) and neglecting orthogonality to the subspace range($CT_c$). By defining $T$ appropriately we can select arbitrary subspaces of range($C$). $T_c$ is the complement of $T$ (see below). In the following we use the notation $[X|Y]$ to indicate the matrix that is formed by appending the columns of $Y$ to the matrix $X$; we will use a similar notation for appending rows to a matrix.

DEFINITION 2.3. *Let $C$, $F$, $Q$, and $r$ be as in Definition* 2.1, *and let the matrix $[T|T_c]$ be a square, unitary matrix such that* $\operatorname{rank}([T|T_c]) = \operatorname{rank}(C)$, *and $T \in \mathbb{C}^{k\times l}$. Now let $\bar{F} = [CT|F]$, $\bar{C} = CT_c$, and $\bar{Q} = [CT|Q]$, and let*

$$\bar{B} = \bar{C}^H \bar{F} = [0|T_c^H B], \tag{2.23}$$

$$\bar{R} = \bar{Q}^H \bar{F} = \left[\begin{array}{c|c} I & T^H B \\ \hline 0 & R \end{array}\right]. \tag{2.24}$$

*Using $\bar{B}$ and $\bar{R}$, we define*

$$\bar{Z} = \bar{B}\bar{R}^{-1} \quad = \quad [0|T_c^H Z] \tag{2.25}$$

(2.26)

*and $\bar{K} = \bar{Z}^H \bar{Z}$. We denote the singular value decomposition of $\bar{Z}$ by*

$$\bar{Z} = Y_{\bar{Z}} \Sigma_{\bar{Z}} V_{\bar{Z}}^H, \tag{2.27}$$

*where $Y_{\bar{Z}} = [\bar{y}_1\ \bar{y}_2\ \dots\ \bar{y}_{k-l}]$ and $V_{\bar{Z}} = [\bar{v}_1\ \bar{v}_2\ \dots\ \bar{v}_{m+l}]$ are ordered such as to follow the convention that*

$$\bar{\sigma}_1 \geq \bar{\sigma}_2 \geq \cdots \geq \bar{\sigma}_{\min(k-l,m+l)}.$$

*Furthermore, let*

$$\bar{F} = \bar{W}\bar{S}, \tag{2.28}$$

$$\bar{W}^H \bar{W} = I_{m+l}, \tag{2.29}$$

*where $\bar{S}$ is upper-triangular, and let*

$$r_3 = (I - \bar{W}\bar{W}^H)r. \tag{2.30}$$

*Finally, let the residual error from discarding* range$(CT_c)$ *be given by* $\bar{e} = r_3 - r_1$.

We say that the subspace range$(CT)$ is kept and that the subspace range$(CT_c)$ is discarded. Note that $\bar{F} = \bar{C}\bar{B} + \bar{Q}\bar{R}$ (cf. Definition 2.1). The residual $r_3$ corresponds to the best approximation to $r$ in the space range$(CT) \oplus$ range$(F)$. Following Theorem 2.2 we can derive an equation for the residual error $\bar{e}$ depending on $T$. From this equation we can derive what the best choice for $T$ is.

THEOREM 2.4. *The residual error $\bar{e}$ is given by*

$$r_3 - r_1 = \sum_{i=1}^{\min(k-l,m+l)} \left( \frac{\bar{\nu}_i \bar{\sigma}_i^2}{1+\bar{\sigma}_i^2} \bar{Q}\bar{v}_i - \frac{\bar{\nu}_i \bar{\sigma}_i}{1+\bar{\sigma}_i^2} \bar{C}\bar{y}_i \right), \tag{2.31}$$

*where $\bar{\nu}_i = \bar{v}_i^H \bar{Q}^H r$, and its norm is given by*

$$\|r_3 - r_1\|_2 = \left( \sum_{i=1}^{\min(k-l,m+l)} \frac{|\bar{\nu}_i|^2 \bar{\sigma}_i^2}{1+\bar{\sigma}_i^2} \right)^{1/2}. \tag{2.32}$$

*Proof.* The proof follows immediately from Theorem 2.2. □

Analogous to (2.15) the norm of the residual error is determined by the singular values of $\bar{Z}$ and by the values $\bar{\nu}_i$. The smaller the singular values are, the smaller the residual error will be. If we can bound the singular values from above by some small value, then the error cannot be large. Likewise, if we want to maintain orthogonality to a subspace of dimension $l$, then we should truncate such that the $l$ largest singular values from $Z$ are removed in $\bar{Z}$. For the moment we ignore the fact that the coefficient $\bar{\nu}_i$ may be very small, in which case the size of $\bar{\sigma}_i$ does not matter.

So, we want to choose $T_c$ (and hence $T$) such that the maximum singular value of $\bar{Z}$ is minimized. How to achieve this is indicated by the following min-max theorem, which is an obvious variant of Theorems 3.1.2 and 3.3.15 in [15, p. 148 and pp. 177–178].

THEOREM 2.5. *Let $Z$ and its singular value decomposition be as in Definition* 2.1, *and let $T_c$ be as in Definition* 2.3. *Then*

$$\min_{\substack{S \subset \mathbb{C}^k \\ \dim(S) = k-l}} \max_{\substack{x \in S \\ \|x\|_2 = 1}} \|x^H Z\|_2 = \begin{cases} \sigma_{l+1} & \textit{if } l+1 \le p \\ 0 & \textit{if } l+1 > p, \end{cases} \tag{2.33}$$

*and the minimum is found for*

$$S = \operatorname{span}\{y_{l+1}, y_{l+2}, \ldots, y_k\}. \tag{2.34}$$

*This is equivalent to*

$$\min_{\substack{T_c \in \mathbb{C}^{k\times(k-l)} \\ T_c^H T_c = I_{k-l}}} \max_{\substack{\xi \in \mathbb{C}^{k-l} \\ \|\xi\|_2 = 1}} \|(T_c \xi)^H Z\|_2 = \begin{cases} \sigma_{l+1} & \textit{if } l+1 \le p \\ 0 & \textit{if } l+1 > p, \end{cases} \tag{2.35}$$

*and the minimum is found for $T_c$, such that*

$$\text{range}(T_c) = S = \text{span}\{y_{l+1}, y_{l+2}, \dots, y_k\}, \tag{2.36}$$

$$x = T_c\xi. \tag{2.37}$$

*Proof.* The proof is a variant of the proofs of Theorems 3.1.2 and 3.3.15 in [15, p. 148 and pp. 177–178]. $\square$

Since $\bar{Z} = [0|T_c^H Z]$ and $\|(T_c\xi)^H Z\|_2 = \|\xi^H(T_c^H Z)\|_2 = \|\xi^H \bar{Z}\|_2$, it is clear that the choice for $T_c$ in (2.36) minimizes the maximum singular value of $\bar{Z}$.

Now from Theorem 2.5 the most obvious choices for the optimal truncation $T$ and its complement $T_c$ are

$$T = [y_1\, y_2\, \dots\, y_l], \tag{2.38}$$

$$T_c = [y_{l+1}\, y_{l+2}\, \dots\, y_k]. \tag{2.39}$$

For this particular choice of $T$ and $T_c$, we can derive the singular value decomposition of $\bar{Z}$ immediately from the singular value decomposition of $Z$. From (2.25) and the singular value decomposition of $Z$ (Definition 2.1) we get $\bar{Z} = Y_{\bar{Z}}\Sigma_{\bar{Z}}V_{\bar{Z}}^H$, where

$$Y_{\bar{Z}} = [e_1\, e_2\, \dots\, e_{p-l}|\star], \tag{2.40}$$

$$\Sigma_{\bar{Z}} = \text{diag}(\sigma_{l+1}, \sigma_{l+2}, \dots, \sigma_p, 0, \dots, 0)_{(\text{k-l}) \times (\text{m+l})}, \tag{2.41}$$

$$V_{\bar{Z}} = \left[\begin{array}{c|c|c} 0 & v_{l+1}\, v_{l+2}\, \dots\, v_p & \star \\ \hline \star & 0 & 0 \end{array}\right] \tag{2.42}$$

(see [8]). Here the $\star$ symbols denote any submatrices that satisfy the respective conditions that $Y_{\bar{Z}}$ and $V_{\bar{Z}}$ be unitary matrices.

We can now give the following theorem about the residual error $\bar{e} = r_3 - r_1$ and its norm.

THEOREM 2.6. *The residual error $\bar{e} = r_3 - r_1$ that results from the truncation defined by the matrix $T$ from* (2.38) *is given by*

$$r_3 - r_1 = \sum_{i=l+1}^{p} \left( \frac{\nu_i\sigma_i^2}{1+\sigma_i^2} Qv_i - \frac{\nu_i\sigma_i}{1+\sigma_i^2} Cy_i \right), \tag{2.43}$$

*and its norm is given by*

$$\|r_3 - r_1\|_2 = \left( \sum_{i=l+1}^{p} \frac{|\nu_i|^2\sigma_i^2}{1+\sigma_i^2} \right)^{1/2}. \tag{2.44}$$

*Proof.* The proof follows from Theorems 2.2 and 2.4, using $\bar{Q}$, $V_{\bar{Z}}$, $\Sigma_{\bar{Z}}$, $\bar{C}$, and $Y_{\bar{Z}}$ from Definition 2.3, and (2.40)–(2.42). We also use the possibility to take $Y_{\bar{Z}} = I$; see (2.40). For details we refer to [8]. $\square$

**The effect of restarting GMRES and selecting the subspace to keep.** We will now analyze the residual error that results from restarting GMRES, and select the subspace to keep after $m$ iterations of GMRES. The implementation will be given in the next section.

Given some iteration $s < m$, our analysis gives the following information. First, how much worse the convergence would have been after $m$ iterations, if we had restarted after $s$ iterations, that is, discarded range$(AW_s)$. Second, which subspace from the first $s$ iterations we should have kept, in order to have in the remaining $(m-s)$-iterations convergence as close as possible to that of the full $m$ GMRES iterations.

After $m$ iterations of GMRES, starting with $w_1 = r_0/\|r_0\|$, we have from (1.12) $AW_m = W_{m+1}\bar{H}_m$, and (1.15) gives the orthonormal basis for range$(AW_m)$:

$$W_{m+1}\bar{Q}_m = W_{m+1}[q_1\, q_2\, \cdots\, q_m]. \tag{2.45}$$

Furthermore, we have $AW_s = W_{s+1}\bar{H}_s$, and the residual $r_s$ is given by (1.16):

$$\begin{aligned} r_s &= r_0 - W_{s+1}\bar{Q}_s\bar{Q}_s^H W_{s+1}^H r_0 \\ &= W_{s+1}(I - \bar{Q}_s\bar{Q}_s^H)\|r_0\|_2 e_1 \\ &= W_{s+1}\tilde{q}_{s+1}\tilde{q}_{s+1}^H\|r_0\|_2 e_1.\,^{2} \end{aligned} \tag{2.46}$$

We define $\rho_s = (\tilde{q}_{s+1}^H e_1)\|r_0\|_2\, \tilde{q}_{s+1}$. Then $r_s = W_{s+1}\rho_s$, which we can also write as $r_s = W_{m+1}\rho_s$ with some abuse of notation.[3]

Now, consider a restart of GMRES with $r_s$ as initial residual and making $m-s$ iterations. Clearly, range$(AW_m) =$ range$(AW_s) \oplus AK^{m-s}(A, r_s)$. Using (2.45)–(2.46) and following the notation of Definition 2.1, we take

$$C = W_{s+1}\bar{Q}_s = W_{m+1}[q_1\, \cdots\, q_s], \tag{2.47}$$

$$Q = W_{m+1}[q_{s+1}\, \ldots\, q_m]. \tag{2.48}$$

For $F$ we can take any basis of $AK^{m-s}(A, r_s)$, because any matrix whose columns form a basis for range$(F)$ gives the same matrix $Z$. Let $F = MS$, with S invertible; then we have

$$Z = (C^H F)\,(Q^H F)^{-1} = (C^H M) S\, S^{-1}(Q^H M)^{-1} = (C^H M)\,(Q^H M)^{-1}.$$

So $F$ can be represented implicitly by

$$F = W_{m+1}[(\bar{H}_{s+1}\rho_s)\,(\bar{H}_{s+2}\bar{H}_{s+1}\rho_s)\,\ldots\,(\bar{H}_m\cdots\bar{H}_{s+2}\bar{H}_{s+1}\rho_s)].$$

In practice we generate $F$ by an Arnoldi iteration with $\bar{H}_m$ and $\rho_s$. Now, following Definition 2.1, we compute $B = C^H F$ and $R = Q^H F$. The optimal residual after $m$ GMRES iterations is given by $r_1$ in Definition 2.1. The residual after making first $s$ GMRES iterations, restarting, and then making another $m-s$ GMRES iterations is given by $r_2$ in Definition 2.1. The difference between the two residuals, the residual error $e$, is given by Theorem 2.2, where $r = r_s$, and $Z = BR^{-1}$, with $B$ and $R$ computed as above. The singular value decomposition of $Z$ not only describes the loss of convergence because of restarting (discarding $C$), according to Theorem 2.2, but it also indicates which vectors from the first $s$ iterations we should have kept for good convergence in the remaining $m-s$ iterations, according to Theorems 2.5 and 2.6. Of course, we can choose any $s < m$, and we can do the analysis for several values of $s$ if we want.

[2] $\tilde{q}_{s+1}$ is the last column of the matrix generated by the first $s$ Givens rotations; $\tilde{q}_{s+1} \neq q_{s+1}$ since $\tilde{q}_{s+1}$ will be changed by the next Givens rotation.

[3] We will assume every vector suitably adjusted by adding zeros at the end. Likewise, we will assume every matrix suitably adjusted by adding zero rows at the bottom.

We will use the information from the singular value decomposition of $Z$ as follows. A large singular value $\sigma_i$ indicates that the corresponding vector, $Cy_i = W_{s+1}\bar{Q}_s y_i$, was important for convergence in iterations $s+1, \ldots, m$. Our strategy is to keep such vectors on the assumption that they are also important for subsequent iterations. Of course, strictly speaking, we do not know this in advance. A small singular value indicates that the corresponding vector was not important for convergence in iterations $s+1, \ldots, m$. Our strategy is to discard such vectors on the assumption that they are not important for subsequent iterations either. Keeping these vectors would mean to waste storage and CPU-time.

Finally, given some value $s$, we do not test the importance of any subspaces generated in the last $m-s$ iterations. Just as in GCR(k) we could keep the last $k \leq (m-s)$ orthonormal basis vectors from range$(AW_m)$, $W_{m+1}[q_{m-k+1} \ldots q_m]$. This would be useful for systems that are "close" to Hermitian. In the next section (Implementation) we will show how to do this.

The truncation algorithm outlined here is easily extended to the GCR algorithm. From (1.8)–(1.11) we have after $m$ iterations of GCR $A[r_0 \ldots r_{m-1}] = C_m R_m$ and

$$[r_0 \ldots r_m] \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & -1 \end{bmatrix} = C_m D_m,$$

where $R_m = C_m^H A[r_0 \ldots r_{m-1}]$ and $D_m = \operatorname{diag}(c_1^H r_0, \ldots, c_m^H r_{m-1})$. This gives

$$AC_{m-1} = C_m R_m \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & -1 \end{bmatrix} D_{m-1}^{-1} = C_m \bar{H}_{m-1},$$

where $\bar{H}_{m-1}$ is an upper Hessenberg matrix. Let $R_m = [\rho_1 \ldots \rho_m]$; then the residual after $s$ iterations is given by $r_s = U_{s+1}\rho_{s+1}$ and $Ar_s = C_{s+1}\rho_{s+1}$. Now we can compute $F$ as

$$F = C_m \left[\rho_{s+1} \; \bar{H}_{s+1}\rho_{s+1} \; \ldots \; (\bar{H}_{m-1} \ldots \bar{H}_{s+1})\rho_{s+1}\right],$$

and we can take $C = C_s$, and $Q = [c_{s+1} \ldots c_m]$. Note that $C^H F$ and $Q^H F$ are just the top $s$ rows and bottom $m-s$ rows of $F$; so $Z$ can easily be computed. This also holds if we compute $F$ by an Arnoldi iteration.

## 3. Implementation.

In the previous section we have analyzed how the loss of orthogonality influences the convergence, and we have shown how to use this analysis to select subspaces to which we should maintain orthogonality. We will use the GCRO method [9, 7] to implement this, because on the one hand GMRES is more efficient and more stable than GCR, and on the other hand GCR offers the flexibility to optimize over arbitrary subspaces. Therefore we think the GCRO scheme is a good compromise.

The implementation of GCRO has been extensively described in [6, 7], including efficient implementations, so we will only outline a straightforward implementation that includes the generalized truncation. At some point we have the matrix $C_k = [c_1\, c_2\, \dots\, c_k]$, which contains the search vectors to which we maintain orthogonality. Furthermore, we have the matrix $U_k = [u_1\, u_2\, \dots\, u_k]$, such that $C_k = AU_k$, and the approximate solution $x$ and residual $r = b - Ax$, such that (see (2.1)–(2.2))

$$C_k^H C_k = I_k,$$
$$C_k^H r = 0.$$

We set $w_1 = r/\|r\|_2$, and we compute in $m$ GMRES iterations

$$AW_m = C_k B_m + W_{m+1}\bar{H}_m, \qquad \text{where} \tag{3.1}$$
$$B_m = C_k^H A W_m, \tag{3.2}$$
$$C_k^H W_{m+1} = O, \tag{3.3}$$
$$\bar{H}_m = W_{m+1}^H A W_m. \tag{3.4}$$

To compute (3.1) we multiply each new $w_i$ by $A$, starting with $w_1$, and then orthogonalize $Aw_i$ first on $C_k$ and then on $W_i$, using the modified Gram–Schmidt algorithm. Next, using (1.16)–(1.17), we compute the approximate solution of the residual equation, $\tilde{x}$, and its residual, $\tilde{r}$. Let $g_m = R_m^{-1}\bar{Q}_m^H\|r\|_2 e_1$; then

$$\tilde{r} = r - W_{m+1}\bar{H}_m g_m, \tag{3.5}$$
$$\tilde{x} = W_m g_m - U_k B_m g_m. \tag{3.6}$$

The computational cost for this part is the same as for GMRES(m), $m$ matrix-vector products, $\frac{1}{2}(m+1)(m+2)$ ddots, and $\frac{1}{2}m(m+5)$ daxpys, plus $km$ ddots and $k(m+1)$ daxpys for the orthogonalization on $C_k$.

Now we choose a value $s$, and we compute the matrix $Z$ to select the subspace of range$(AW_s)$ to which we will maintain orthogonality in the next iterations. We compute $\rho_s = (I - \bar{Q}_s\bar{Q}_s^H)\|r\|_2 e_1$ (see (2.46)), where $\bar{Q}_s^H\|r\|_2 e_1$ is already available from the GMRES iteration, and we multiply by $\bar{Q}_s$ using the first $s$ Givens rotations from the GMRES iteration. We use an Arnoldi iteration with $\bar{H}_m$ and $\rho_s$ to generate $M$ such that $F = W_{m+1}M$. From (2.47) and (2.48) we have $C = W_{m+1}[q_1\, \dots\, q_s]$ and $Q = W_{m+1}[q_{s+1}\, \dots\, q_m]$, so that $[C|Q] = W_{m+1}\bar{Q}_m$, and

$$[C|Q]^H F = \bar{Q}_m^H M = \left[\frac{B}{R}\right],$$

which is computed by applying the Givens rotations that define $\bar{Q}_m^H$ to $M$. Finally, we only have to compute $Z = BR^{-1}$ and its singular value decomposition. As will be clear from the numerical tests, the matrices $Z$ are generally very small, and hence the singular value decomposition incurs only negligible cost. Moreover, in the computation of $Z$, the matrix $W_{m+1}$ is never used explicitly; we only use the matrix $\bar{H}_m$ and the implicitly stored Givens rotations. We can do this analysis for several values of $s$ to select an appropriate subspace. We will maintain orthogonality to the selected subspace of range$(AW_s)$ by appending its basis vectors to $C_k$.

Suppose we want to keep a subspace of dimension $p_1$ from the space generated in the first $s$ GMRES iterations, where $p_1$ is either fixed or derived from the singular values of $Z$. In addition, as we do not test the subspace generated in the remaining $m - s$ iterations, we may want to keep the last $p_2 \leq m - s$ orthonormal basis vectors of range$(AW_m)$: $W_{m+1}q_{m-p_2+1}, \dots, W_{m+1}q_m$; see (2.45). This might, for example, be useful for nearly Hermitian systems. These vectors will be discarded by the next truncation if not useful. Including the correction to the residual from the GMRES iteration,[4] we will add $p_1 + p_2 + 1$ vectors to the matrices $U_k$ and $C_k$.

If the new number of vectors in $C$ will exceed the chosen maximum, we have to truncate first. From (3.1) and the QR-decomposition of $\bar{H}_m$ (1.15), we see that

$$AW_m = C_k B_m + (W_{m+1}\bar{Q}_m)R_m.$$

Thus, following Definition 2.1, we compute the matrix $\widehat{Z} = B_m R_m^{-1}$ and its singular value decomposition $\widehat{Z} = \widehat{Y}\widehat{\Sigma}\widehat{V}^H$ to determine the truncation. Since we orthogonalize on range$(C_k)$ while generating $W_{m+1}$, the truncation analysis using $\widehat{Z}$ has a slightly different interpretation than presented in section 2. This can be avoided by neglecting $C_k$ in the Arnoldi iteration and appending the vectors afterwards to the Krylov subspace over which we minimize. However, this typically will result in worse convergence behavior. For a discussion, see [8].

Let $\widehat{Y} = [\hat{y}_1\; \hat{y}_2\; \ldots\; \hat{y}_k]$, and suppose we want to keep a subspace of dimension $l$. Then the $l$ vectors from $C_k$ and $U_k$ we want to keep are

$$C_l = C_k[\hat{y}_1\; \hat{y}_2\; \ldots\; \hat{y}_l], \tag{3.7}$$

$$U_l = U_k[\hat{y}_1\; \hat{y}_2\; \ldots\; \hat{y}_l]. \tag{3.8}$$

We implement this truncation in a special way, because we still need all $u_i$ to construct the new $u$ vectors that we want to keep from the GMRES iteration (see below). Moreover, often $l$ is almost as large as $k$, and then our implementation is cheaper than a straightforward one. Consider discarding the vector $C_k\hat{y}_k$. This means we need to maintain orthogonality to range$(C_k[\hat{y}_1\; \ldots\; \hat{y}_{k-1}])$. However, the basis vectors of this space are not important; we just want to keep a matrix $C_{k-1}^{(1)}$ such that $C_{k-1}^{(1)} \perp C_k\hat{y}_k$. We can do this using $k-1$ Givens rotations $G_1, \ldots, G_{k-1}$ such that $G_1 \ldots G_{k-1}e_k = \hat{y}_k$. This gives

$$C_k G_1 \ldots G_{k-1} G_{k-1}^H \ldots G_1^H \widehat{Y}_k e_k = C_k G_1 \ldots G_{k-1} e_k,$$

so that dropping the last column vector of $C_k^{(1)} = C_k G_1 \ldots G_{k-1}$ results in the required truncation. Now let $C_{k-1}^{(1)}$ be $C_k^{(1)}$ with the last column vector removed. Note that

$$G_{k-1}^H \ldots G_1^H \widehat{Y}_k = \left[\begin{array}{c|c} \widehat{Y}_{k-1}^{(1)} & 0 \\ \hline 0 & 1 \end{array}\right].$$

So we have $C_k\widehat{Y}_{k-1} = C_{k-1}^{(1)}\widehat{Y}_{k-1}^{(1)}$. Now discarding the vector $C_k\hat{y}_{k-1} = C_{k-1}^{(1)}\widehat{Y}_{k-1}^{(1)}e_{k-1}$ can be done in the same way using $k-2$ Givens rotations, and so on, until we have discarded $p_3 = k - l$ vectors. Counting a Givens rotation applied to $C_k$ or $U_k$ as two daxpys, we need only $2p_3(2k - p_3 - 1)$ daxpys (for small $p_3$). Let $G$ be the product of all Givens rotations. Then we construct the two matrices

$$\begin{aligned} \widehat{C}_k &= C_k G, \\ \widehat{U}_k &= U_k G, \end{aligned} \tag{3.9}$$

[4] We could discard this vector after the update, but from our earlier work [7, 9] this vector seems to be very effective.

whose first $l$ columns span the same subspaces as $C_l$ and $U_l$, respectively. Note that all computations are done in place. After computing the matrices $\widehat{C}_k$ and $\widehat{U}_k$, we can overwrite the vectors $c_{l+1}, \dots, c_k$.

Now we add the vectors that we want to keep, and we compute the new approximate solution and residual. Note that the new approximate solution amounts to a residual minimization over the subspace $\mathrm{range}(C_k) \oplus \mathrm{range}(AW_m)$. We first compute the new $u_i$ using the storage for the vectors $c_{l+1}, \dots, c_k$ as temporary storage, so as not to overwrite the vectors in $\widehat{U}_k$ until we have computed all new $u_i$. Then we compute the new $c_i$. Let the singular value decomposition of $Z$ be given by $Z = Y\Sigma V^H$ with $Y = [y_1 \ \dots \ y_s]$ and let $\bar{Q}_m = [q_1 \ \dots \ q_m]$ (represented by Givens rotations). We set

$$\begin{array}{rcll} u_{l+1} &=& \tilde{x}/\|r-\tilde{r}\|_2, & \\ u_{l+1+i} &=& (W_m - \widehat{U}_k G^H B_m) R_m^{-1} y_i & \text{for } i = 1, \dots, p_1, \\ u_{l+1+p_1+j} &=& (W_m - \widehat{U}_k G^H B_m) R_m^{-1} e_{m-p_2+j} & \text{for } j = 1, \dots, p_2. \\ c_{l+1} &=& (r-\tilde{r})/\|r-\tilde{r}\|_2, & \\ c_{l+1+i} &=& W_{s+1}\bar{Q}_s y_i = W_{m+1}\bar{Q}_m y_i & \text{for } i = 1, \dots, p_1, \\ c_{l+1+p_1+j} &=& W_{m+1} q_{m-p_2+j} & \text{for } j = 1, \dots, p_2. \end{array} \tag{3.10}$$

We can also compute the new $u_i$ in place with a slightly more complicated algorithm. Finally, we compute the new residual and approximate solution,

$$r = r - c_{l+1}c_{l+1}^H r, \tag{3.11}$$

$$x = x + u_{l+1}c_{l+1}^H r. \tag{3.12}$$

The costs of this part are $p_1(2s+k-2) + p_2(2m+k-2)$ daxpys. By construction $r \perp c_{l+1+i}$, for $i = 0, \dots, p_1+p_2$, and the vectors $c_{l+1+i}$ for $i = 1, \dots, p_1+p_2$ are mutually orthogonal. Hence we only have to orthogonalize $c_{l+1}$ against the vectors $c_{l+1+i}$, for $i = 1, \dots, p_1+p_2$, and adapt $u_{l+1}$ accordingly. This costs $(p_1+p_2)$ ddots and $2(p_1+p_2)$ daxpys. Occasionally, (partial) reorthogonalization of the vectors $c_i$ and/or the residual may be useful.

There is a large degree of freedom in choosing the parameters. The best implementation would be to adapt them dynamically. However, in our implementation we use fixed parameters. Below we give the approximate cost for the case where we keep $p$ vectors from the subspace generated in $m$ (GMRES) iterations, and after the first $(1+\frac{k}{p})m$ iterations, we discard $p$ vectors from $C_k$ every $m$ iteration. Here $k$ is the maximum number of vectors in $C$. The first $(1+\frac{k}{p})m$ iterations cost

| | |
|---|---|
| $(1+\frac{k}{p})m$ | **matrix-vector products**, |
| $(1+\frac{k}{p})\frac{1}{2}[m(m+k+4p+1)+pk]$ | **daxpys**, |
| $(1+\frac{k}{p})[\frac{1}{2}m(m+k+3)+p]$ | **ddots**; |

after that, every $m$ iterations cost

| | |
|---|---|
| $m$ | **matrix-vector products**, |
| $\frac{1}{2}m(m+2k+4p+1)+5kp$ | **daxpys**, |
| $\frac{1}{2}m(m+2k+3)+p$ | **ddots**. |

**4. Numerical results.** We will describe five numerical tests. They are derived from two-dimensional, vertex-centered, finite volume discretizations of convection-diffusion equations. For the first two tests, Problem 1 and Problem 2, we applied ILU(0) preconditioning [17, 18], and we iterated until we achieved a reduction of the residual norm by a factor of $10^{-10}$. The other three tests are variants of the same convection-diffusion problem, Problem 3, that differ only in the speed of convection. These tests come from [19], where Morgan compares his *augmented GMRES* (see Introduction) to restarted GMRES and to QMR [12] and TFQMR [11]. To compare with Morgan's results we also solve these tests without preconditioning and to about the same precisions as in [19].

We will compare the convergence of restarted GMRES(m) with truncated GCRO for several sets of parameters. We compare in two ways. The first is the speed of convergence that can be obtained using a given amount of memory, typically expressed as the (maximum) number of vectors that must be stored. The second is the speed of convergence that can be obtained given the (maximum) dimension of the subspace over which we minimize in any iteration.

Furthermore, for Problem 1 we show that our analysis of the residual error from restarting explains the convergence of restarted GMRES very well. We analyze the convergence of GMRES(m) by studying the singular values of the matrix $Z$ associated with a restart at iteration $s$, for $s = 1, \ldots, m-1$. For large singular values, (2.15) indicates a large potential loss of convergence. This especially holds for $s$ close to $m$. For small singular values, (2.15) indicates that only a small loss of convergence could result from discarding the corresponding directions. Of course, for values of $s$ close to $m$, we have only a small subspace to measure the convergence loss from restarting. More important, we cannot determine from within one GMRES(m) cycle which directions really will be the most important to maintain orthogonality to in the next GMRES(m) cycle. Nevertheless, the results from our numerical tests clearly show that our "singular value analysis" provides important information. For the truncation in GCRO we carry out the analysis for only one value of $s$.

For Problem 3 we also compare with Morgan's augmented GMRES and with TFQMR [11], BiCGstab [23], and the truncated GCR variant described in [16]. This GCR variant discards the vector $c_i$ that made the smallest contribution to the orthogonalization of the most recent vector $c_k$. The method does not consider spaces but only the vectors that happen to appear in the iteration. Moreover, to determine which vector should be removed, it uses only the orthogonalization of the most recent vector $c_k$, which may not involve that much information. Furthermore, the method suffers from the known disadvantages of GCR compared with GMRES (stability problems and expensive in daxpys).

**Problem 1.** The first problem, taken from [7], is defined by the discretization of

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

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

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

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. The step-size in the $x$-direction is 1/199, and in the $y$-direction it is 4/199.

In Figures 4.1–4.3 we plot the residual norm against the number of iterations (matrix-vector products) for GMRES(m), for $m = 10$, 20, and 50, and for full GMRES, which converges in 130 iterations. At each iteration we also give the largest

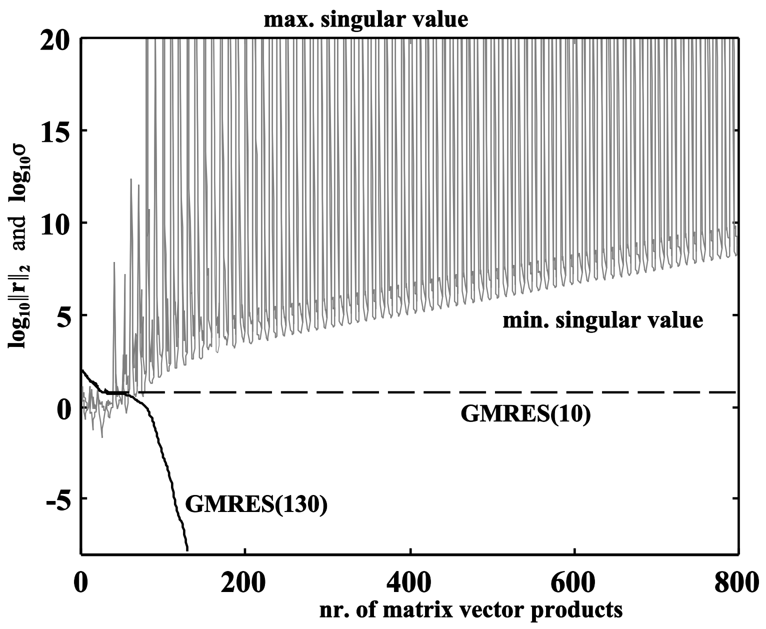

FIG. 4.1. *Analysis and convergence of GMRES*(10) *for Problem* 1.

and smallest singular values associated with a restart at that iteration. They are computed as follows. After the $i$th GMRES(m) cycle, we compute $Z$ and its singular value decomposition for each iteration $s = 1, \dots, m-1$, and we plot the largest and the smallest singular value of $Z$ at iteration $(i-1)m+s$.

For $m = 10$, see Figure 4.1, we see in the first few iterations small singular values and some convergence. Then the singular values become extremely large and GMRES(10) stagnates. The large size of the minimum singular value indicates that the new search space is (almost) entirely contained in the old search space. For $m = 20$ (see Figure 4.2), we see initially the same behavior as for $m = 10$. The large maximum singular values indicate that important directions (subspaces) are lost through restarting; however, the small minimum singular values show that the residual is changed, even if its norm does not decrease much. Hence, after a large number of iterations the components of the residual that cause stagnation have been removed; the singular values become relatively small, and the method starts to converge. There are still some occasional peaks of the maximum singular value; these are accompanied by short stagnations of the convergence. For $m = 50$ (see Figure 4.3), we see large singular values and stagnation immediately after each restart, but towards the end of a cycle of 50 iterations, the maximum singular value goes down and the method makes some progress. Nevertheless, GMRES(50) converges very slowly and, in fact, reaches the required tolerance hardly faster than GMRES(20). Compared to full GMRES, both GMRES(20) and GMRES(50) show very poor convergence.

In Figure 4.4 we compare truncated GCRO, referred to as *GCROT*, for three sets of parameters with GMRES(m). GCROT($m$,$k_{max}$,$k_{new}$,$s$,$p_1$,$p_2$) indicates the following implementation. The parameter $m$ indicates the number of GMRES iterations, $k_{max}$ indicates the maximum number of vectors in $C_k$ and $U_k$ that can be kept, and

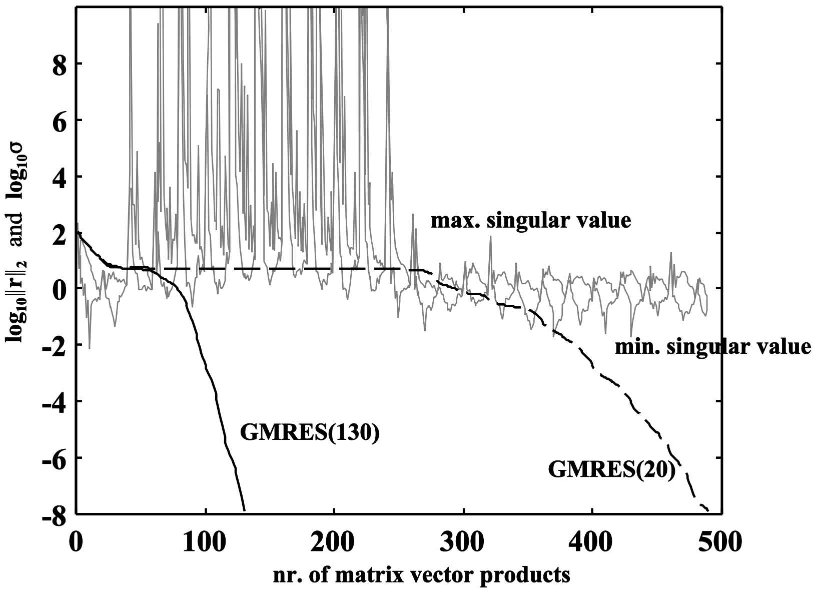

FIG. 4.2. *Analysis and convergence of GMRES*(20) *for Problem* 1.

$k_{new}$ indicates the number of vectors that is stored after truncating $C_k$ and adding the new vectors. The parameter $s$ indicates the dimension of the subspace from the GMRES iteration from which we select $p_1$ vectors to be appended to $C_k$. We also append the last $p_2$ orthonormal basis vectors from range$(AW_m)$. The number of vectors to be stored is at most $m + 2k_{max}$, and the dimension of the subspace over which we minimize is at most $m + k_{max}$. If we only keep the residual correction from the GMRES iteration, $c_{l+1}$ in (3.10), we omit the last three parameters and just write GCROT$(m,k_{max},k_{new})$. We only truncate $C_k$ and $U_k$ in that case. The singular value decompositions involve matrices of size $s \times (m - s)$ to select a subspace from the GMRES iteration and at most $k_{max} \times m$ for the truncation of $C_k$ and $U_k$. For our tests these are very small matrices. Moreover, their size is independent from the problem size and the number of iterations.

We see that using only twenty vectors GCROT(6,7,7,3,1,0) converges almost twice as fast as GMRES(50), and using 50 vectors GCROT converges three times as fast as GMRES(50) and almost as fast as full GMRES. If we use a few additional vectors, the convergence of GCROT improves gradually. In contrast, if we increase $m$ from 20 to 50, the convergence of GMRES(m) improves only marginally. GMRES(50) minimizes over subspaces of dimension 50. GCROT(6,7,7,3,1,0) minimizes over subspaces of dimension thirteen at most. These results show that our method converges very well using a sequence of subspaces of very small dimension.

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

$$-1000(u_{xx} + u_{yy}) + 2e^{4(x^2+y^2)}u_x - 2e^{4(x^2+y^2)}u_y = 0$$

on $[0,1] \times [0,1]$. The boundary conditions are $u = 1$ on $y = 0$, $u = 0$ on $y = 1$, $u' = 0$ on $x = 0$, and $u' = 0$ on $x = 1$, where $u'$ denotes the (outward) normal derivative. The step-size in both $x$-direction and $y$-direction is 1/199.

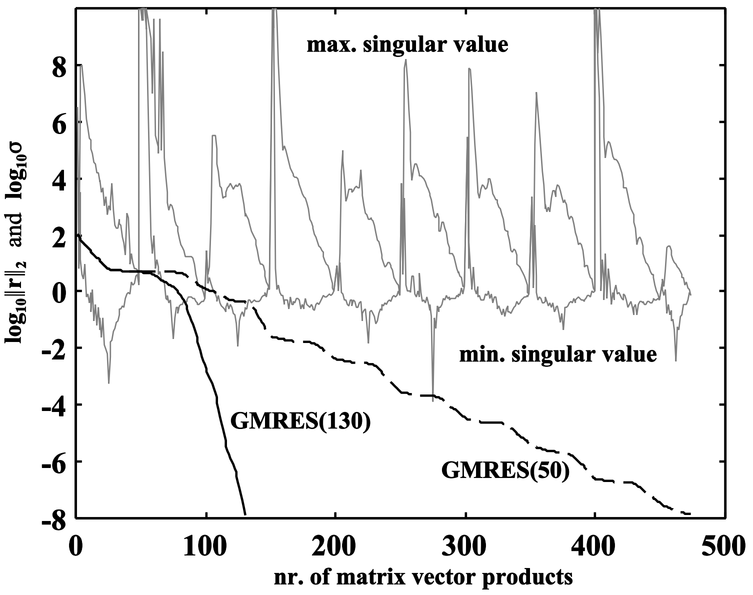

FIG. 4.3. *Analysis and convergence of GMRES*(50) *for Problem* 1.

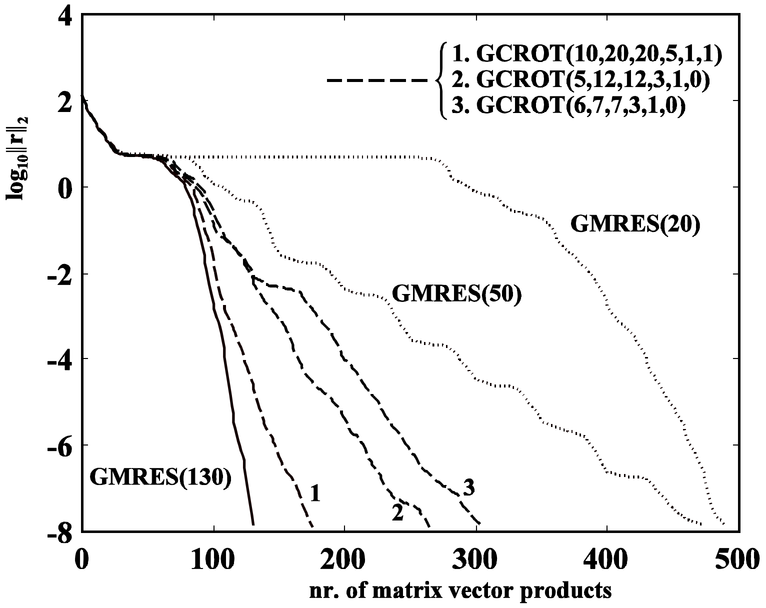

FIG. 4.4. *Convergence of GMRES(m) and GCROT for Problem* 1.

In Figure 4.5 we show the convergence of GMRES(m) for several values of $m$. For small $m$ GMRES(m) converges very slowly, and even GMRES(50) still needs more than 600 iterations. Also interesting is that GMRES(150) is not faster than GMRES(100): both need about 325 iterations. See [8] for an explanation. In practice GMRES(100) or GMRES(150) will hardly ever be used because of the huge memory

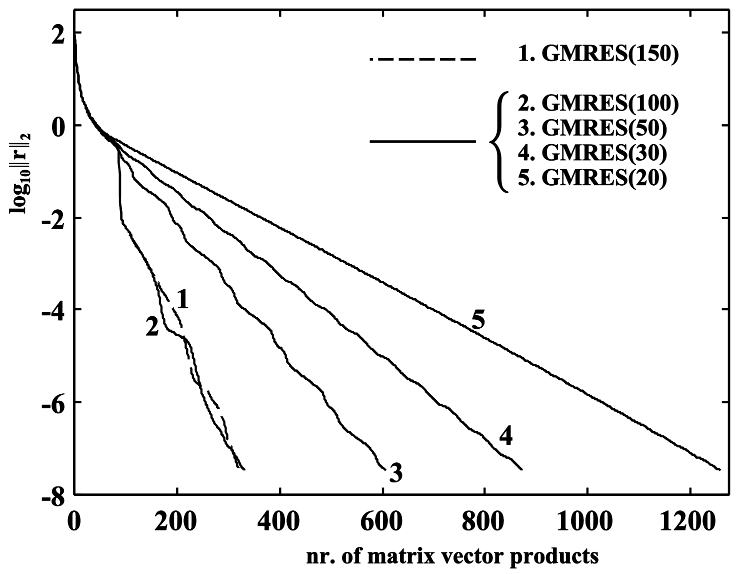

FIG. 4.5. *Convergence of GMRES(m) for Problem* 2.

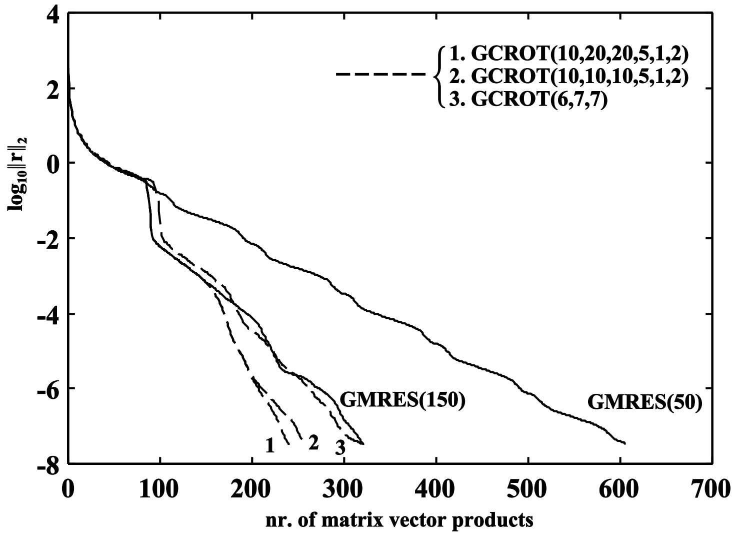

FIG. 4.6. *Convergence of GMRES(m) and GCROT for Problem* 2.

requirements and the overhead in vector operations.

In Figure 4.6 we compare the convergence of GMRES(m) with the convergence of GCROT for three sets of parameters. With merely 20 vectors and minimizing over subspaces of dimension 13 only, GCROT converges as fast as GMRES(150). Using 50 vectors and minimizing over subspaces of dimension 30, GCROT converges about 40% faster than GMRES(150). The convergence of GMRES(150) shows that, at least for the first 150 iterations, GCROT(10,20,20,5,1,2) and GCROT(10,10,10,5,1,2) converge optimally.

**Problem 3.** This problem is taken from [19]. The problem is defined by the discretization of

$$u_{xx} + u_{yy} + Du_x = -(41)^2$$

on $[0, 1] \times [0, 1]$. The boundary conditions are $u = 0$ on all boundaries. The step-size in both $x$-direction and $y$-direction is $1/41$. The problem involves three test cases, namely $D = 1$, $D = 41$, and $D = 41^2$.

In [19] Morgan compares GMRES(25) with two versions of *augmented GMRES* by looking at the residual norm after a fixed number of iterations (matrix-vector products). The first version uses 21 normal Arnoldi vectors and 4 approximate eigenvectors. The use of an approximate eigenvector requires storing two vectors, and so this version requires 29 vectors. In each cycle it minimizes over a subspace of dimension 25, and it takes 21 matrix-vector products to generate this space. We will refer to this version as A-GMRES(29). The second version uses 17 normal Arnoldi vectors and 4 approximate eigenvectors, and so this version requires 25 vectors. In each cycle it minimizes over a subspace of dimension 21, and it takes 17 matrix-vector products to generate this space. We will refer to this version as A-GMRES(25).

We will compare GCROT with full GMRES, A-GMRES, GMRES(25), TFQMR [11], BiCGstab [23], and the GCR variant suggested in [16], referred to as *GCR-plus(m)*. The results for A-GMRES are taken from [19]. For TFQMR and BiCGstab we use a random left starting vector, and we report the iteration counts corresponding to the norm of the true residual. We use the TFQMR implementation from QMR-pack [12, 11]. However, we had to implement a simple restarting strategy for TFQMR, because the method occasionally stopped prematurely (since we wanted the norm of the true residual to converge). It was not possible in these cases to simply solve to a slightly higher precision. We use TFQMR and BiCGstab to compare the convergence of GCROT with the convergence of two robust, nonrestarted, nontruncated methods, and to show that careful truncation of GMRES-like methods leads to at least as good a convergence. However, for completeness we give in Table 4.1 an overview of the average number of vector operations (daxpys and ddots) per iteration/matrix-vector product for the methods that we compare GCROT with. For GCROT we can derive this number from the overview of the costs at the end of section 3.

Unfortunately, the test problems that we took from [19] for comparison with A-GMRES are small, very sparse compared to, e.g., finite element problems with multiple variables, and nonpreconditioned. This makes a comparison by floating-point operations (flops) rather unfavorable for the GMRES-like methods. However, it should be clear from Table 4.1 and the iteration counts that methods like GCROT and A-GMRES are quite competitive for problems where the matrix-vector product and the preconditioner are expensive. This is very common in practice. In many problems matrix-vector product and preconditioner together cost about the same number of flops as 40 to 80 vector operations. Moreover, many computers achieve a much higher flop rate for vector operations than for (irregular) sparse matrix-vector products and backward and forward substitution in ILU-type preconditioners.

Finally, TFQMR and BiCGstab need to store approximately eight vectors; thus, in memory requirements they can be significantly cheaper than the other methods.

***D* = 1.** Morgan reports that after 200 matrix-vector products A-GMRES(25) has reached an accuracy of $\log_{10} \|r\|_2 = -11.92$, A-GMRES(29) of $\log_{10} \|r\|_2 = -12.55$,

Table 4.1
*The average number of daxpys and ddots per matrix-vector product (and preconditioner) for the methods discussed. GMRES(m) restarts every m iterations. GCRplus(m) maintains orthogonality to m basis vectors of the Krylov subspace. A-GMRES(m,j) uses m Arnoldi vectors and j approximate eigenvectors. N is the total number of iterations; we assume for GCRplus(m) that $N \gg m$.*

| Method | daxpy | ddot |
|---|---|---|
| Full GMRES | $\frac{1}{2}(N+5)$ | $\frac{1}{2}(N+2)$ |
| GMRES(m) | $\frac{1}{2}(m+5)$ | $\frac{1}{2}(m+2)$ |
| GCRplus(m) | $(2m+2)$ | $(m+1)$ |
| A-GMRES(m,j) | $\frac{1}{2}(m+5\frac{j^2}{m}+6j+5)$ | $\frac{1}{2}(m+4j+\frac{j^2}{m}+4)$ |
| TFQMR | 5 | 2.5 |
| BiCGstab | 3 | 2.5 |

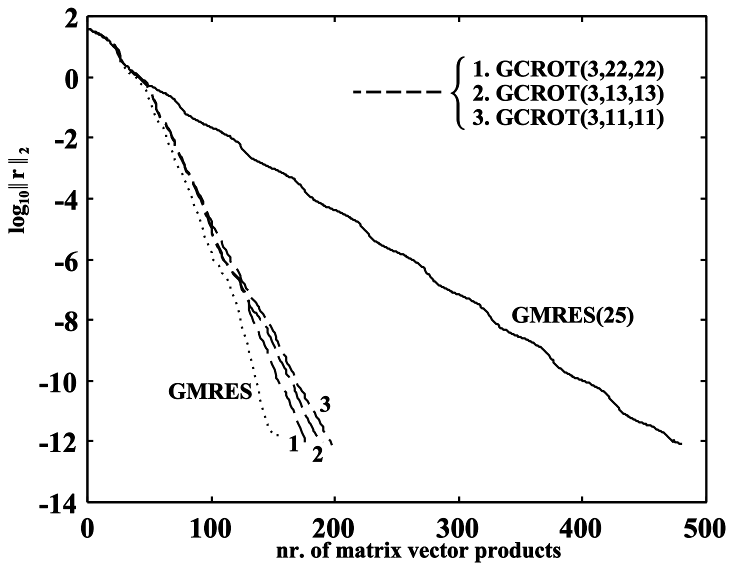

Fig. 4.7. *Convergence of GCROT for Problem* 3 *with* $D = 1$.

and GMRES(25) of $\log_{10}\|r\|_2 = -3.89$.[5] Starting with the same initial residual, we have iterated until GCROT reached an absolute accuracy of $\log_{10}\|r\|_2 = -12.00$; the results are given in Figure 4.7. For comparison we also include the results for full GMRES. All GCROT versions converge in about the same number of iterations as the A-GMRES versions with the same memory requirements. This convergence is close to optimal as can be seen from comparison with full GMRES.

Table 4.2 gives a comparison between full GMRES, GMRES(25), A-GMRES(29), GCROT, GCRplus(13), which needs 26 vectors, GCRplus(15), which needs 30 vectors, TFQMR, and BiCGstab. We see that A-GMRES(29) and GCROT behave roughly the same. Since the matrix is almost symmetric, removing small eigenvalues is more or less the optimal strategy. Nevertheless, GCROT converges as well as A-GMRES, even

[5]In our tests GMRES(25) reached an accuracy of around $-4.3$ after 200 iterations. This is probably due to small differences in discretization. It is the only significant difference we found. To reach an accuracy of $-6.0$, GMRES(25) took about the same number of iterations in our tests, as Morgan indicates [19].

TABLE 4.2
*The number of matrix-vector products to reach* $\log_{10}\|r\|_2 = -6.00$ *and* $\log_{10}\|r\|_2 = -12.0$ *for Problem* 3 *with* $D = 1$*. For A-GMRES, which was stopped after a fixed number of iterations, we give that number of iterations and in brackets the accuracy reached. GCRplus did not converge, and we give the highest accuracy reached and the number of iterations to reach this.*

| Method | Number of matrix-vector products | |
|---|---|---|
| | $\log_{10}\|r\|_2 = -6$ | $\log_{10}\|r\|_2 = -12$ |
| Full GMRES | 102 | 159 |
| GMRES(25) | 278 | 473 |
| A-GMRES (29) | 116 | 200 (-12.55) |
| GCROT(3,22,22) | 110 | 176 |
| GCROT(3,13,13) | 111 | 190 |
| GCROT(3,11,11) | 116 | 197 |
| GCRplus(15) | 116 | 189 (-11.15) |
| GCRplus(13) | 122 | 209 (-11.15) |
| TFQMR | 172 | 406 |
| BiCGstab | 138 | 234 |

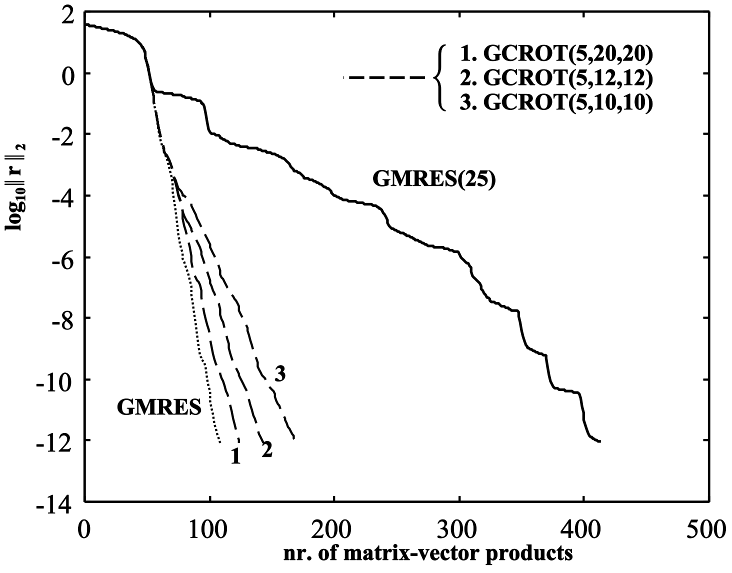

FIG. 4.8. *Convergence of GCROT for Problem* 3 *with* $D = 41$*.*

though GCROT does not look at eigenvalues. BiCGstab works reasonably well, which is to be expected for this type of problem. Surprisingly, TFQMR takes about twice the number of iterations, and the method has problems reaching the chosen accuracy. GCRplus initially converges well, but does not converge to $\log_{10}\|r\|_2 = -12.0$. We give the highest accuracy reached in brackets, together with the number of matrix-vector products to reach this accuracy. Compared with GCROT, GCRplus takes more iterations, whereas the residual norm is an order of magnitude larger.

***D* = 41.** Morgan reports that after 200 matrix-vector products A-GMRES(25) has reached an accuracy of $\log_{10}\|r\|_2 = -11.05$, A-GMRES(29) of $\log_{10}\|r\|_2 = -11.19$, and GMRES(25) of $\log_{10}\|r\|_2 = -4.15$. Starting with the same initial residual, we have iterated until GCROT reached an absolute accuracy of $\log_{10}\|r\|_2 =$

Table 4.3

*The number of matrix-vector products to reach* $\log_{10}\|r\|_2 = -6.00$ *and* $\log_{10}\|r\|_2 = -12.0$ *for Problem 3 with* $D = 41$*. For A-GMRES, which was stopped after a fixed number of iterations, we give that number of iterations and in brackets the accuracy reached. GCRplus did not converge, and we give the highest accuracy reached and the number of iterations to reach this.*

| Method | Number of matrix-vector products | |
|---|---|---|
| | $\log_{10}\|r\|_2 = -6$ | $\log_{10}\|r\|_2 = -12$ |
| Full GMRES | 79 | 108 |
| GMRES(25) | 300 | 411 |
| A-GMRES (29) | 134 | 200 (-11.19) |
| GCROT(5,20,20) | 86 | 124 |
| GCROT(5,12,12) | 95 | 143 |
| GCROT(5,10,10) | 105 | 169 |
| GCRplus(15) | 122 | 214 (-11.26) |
| GCRplus(13) | 184 | 294 (-11.55) |
| TFQMR | 125 | 264 |
| BiCGstab | 122 | 196 |

$-12.00$; the results are given in Figure 4.8. For comparison we also include the results of full GMRES. All three versions of GCROT reach an accuracy of $\log_{10}\|r\|_2 = -12.0$ in significantly fewer iterations than 200 and hence are much faster than A-GMRES. Indeed, A-GMRES(25) and A-GMRES(29) take about 40% more matrix-vector products than GCROT(5,10,10) and over 50% more matrix-vector products than GCROT(5,12,12). Once more the GCROT versions stay close to full GMRES in convergence.

Table 4.3 compares full GMRES, GMRES(25), A-GMRES(29), GCROT, GCRplus(13), GCRplus(15), TFQMR, and BiCGstab. We see that for stronger convection, and hence a more nonsymmetric matrix, GCROT converges much better than A-GMRES and GCRplus. GCRplus does not converge to the required precision and takes almost twice the number of iterations of GCROT only to reach an order of magnitude lower precision. Also BiCGstab and TFQMR take significantly more matrix-vector products than GCROT.

***$D = 41^2$*.** Morgan reports that after 500 matrix-vector products A-GMRES(25) has reached an accuracy of $\log_{10}\|r\|_2 = -9.49$, A-GMRES(29) of $\log_{10}\|r\|_2 = -9.80$, and GMRES(25) of $\log_{10}\|r\|_2 = -7.01$. Starting with the same initial residual, we have iterated until GCROT reached an absolute accuracy of $\log_{10}\|r\|_2 = -10.00$; the results are given in Figure 4.9. For comparison we also include the results of full GMRES. All four versions of GCROT reach an accuracy of $\log_{10}\|r\|_2 = -10.0$ in approximately 500 iterations, and so they are about as fast as the A-GMRES versions. Again all versions are quite close to full GMRES in convergence.

Table 4.4 compares full GMRES, GMRES(25), A-GMRES(29), GCROT, GCRplus(13), GCRplus(15), TFQMR, and BiCGstab. We see that all GMRES-like methods show about the same convergence, and all remain relatively close to the convergence of full GMRES. This is not surprising, because even full GMRES does not show superlinear convergence. BiCGstab does not converge at all, not even to low precision. TFQMR initially converges well, but it takes about 70% more iterations than GCROT, A-GMRES, or GCRplus to reach the final precision.

**Summary.** For $D = 1$ and $D = 41^2$ the convergence of A-GMRES is almost optimal, and GCROT performs equally well. For $D = 41$ A-GMRES is significantly slower than full GMRES and here GCROT performs much better than A-GMRES. GCROT shows also in this case close to optimal convergence. GCRplus converges well

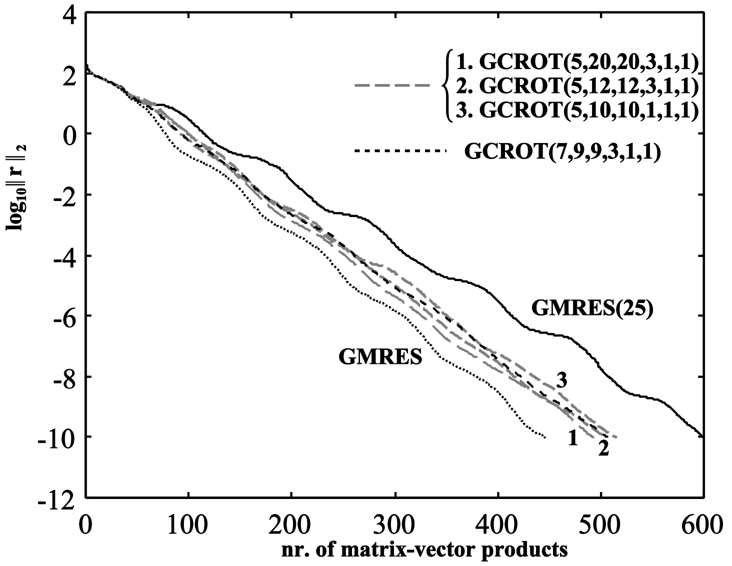

FIG. 4.9. *Convergence of GCROT for Problem* 3 *with* $D = 41^2$.

TABLE 4.4
*The number of matrix-vector products to reach* $\log_{10} \|r\|_2 = -6.00$ *and* $\log_{10} \|r\|_2 = -10.0$ *for Problem* 3 *with* $D = 41^2$. *For A-GMRES, which was stopped after a fixed number of iterations, we give that number of iterations and in brackets the accuracy reached. BiCGstab did not converge.*

| Method | Number of matrix-vector products | |
|---|---|---|
| | $\log_{10} \|r\|_2 = -6$ | $\log_{10} \|r\|_2 = -10$ |
| Full GMRES | 308 | 446 |
| GMRES(25) | 441 | 634 |
| A-GMRES (29) | 326 | 500 (-9.80) |
| GCROT(5,20,20,3,1,1) | 327 | 493 |
| GCROT(5,12,12,3,1,1) | 337 | 505 |
| GCROT(7,9,9,3,1,1) | 347 | 507 |
| GCRplus(15) | 321 | 498 |
| GCRplus(13) | 319 | 489 |
| TFQMR | 354 | 849 |
| BiCGstab | $\infty$ | $\infty$ |

for $D = 1$ and $D = 41^2$ if the tolerance for convergence is not too small. However, the method consistently has problems reaching high accuracy. For $D = 41$ GCRplus is much slower than GCROT. Also for other problems GCRplus seems unreliable. For example, for Problem 1 the method does not converge for several values of $m$, even though it does converge for some smaller values of $m$. Moreover, the method is significantly more expensive than GCROT in vector operations per matrix-vector product. Our tests show that GCROT converges as well as or significantly better than the nontruncated methods BiCGstab and TFQMR. Especially for the last variant of Problem 3, with very strong convection, BiCGstab and TFQMR converge poorly. Moreover, TFQMR typically takes significantly more iterations when high accuracy is required, and BiCGstab seems less robust. Whether GCROT is faster than these methods in time depends, apart from the iteration counts, on the relative costs of the vector operations compared with the costs of the matrix-vector product and preconditioner.

**5. Conclusions.** We have derived a set of equations that describe how the convergence of restarted or truncated Krylov subspace methods is influenced by the loss of orthogonality to certain subspaces of the generated Krylov space. These equations indicate which subspace was most important for convergence, and how important. We use these equations to derive a "truncated GMRES" that is based on the GCRO method.

Our numerical tests show that our strategy is very effective. Indeed, we obtain close to optimal convergence for all tests. Moreover, the method compares very favorably with other methods. Finally, we would like to point out that truncated GCRO often shows very good convergence using as few as 20 or 25 vectors. The faster convergence and better robustness of truncated GCRO compared with TFQMR and BiCGstab, which need about eight vectors, may well be worth the extra memory for hard problems.

Several improvements are possible for our current implementation. It would be better to use dynamically adapted parameters to control the truncation. In addition, we should test whether it is important for the truncation to use the $\nu_i$ in (2.14)–(2.15) as well.

**Acknowledgments.** We thank the anonymous referees for their helpful comments to improve the manuscript, and for pointing out reference [16].

REFERENCES

[1] J. Baglama, D. Calvetti, G. H. Golub, and L. Reichel, *Adaptively preconditioned GMRES algorithms*, SIAM J. Sci. Comput., 20 (1998), pp. 243–269.

[2] Å. Björck and G. Golub, *Numerical methods for computing angles between linear subspaces*, Math. Comp., 27 (1973), pp. 579–594.

[3] J. G. Blom and J. Verwer, *Algorithm* 758*; VLUGR*2*: a vectorizable adaptive-grid solver for PDEs in* 2*D*, ACM Trans. Math. Software, 22 (1996), pp. 302–328.

[4] J. G. Blom and J. Verwer, *Vlugr*3*: a vectorizable adaptive grid solver for PDEs in* 3*D, Part* I*: Algorithmic aspects and applications*, Appl. Numer. Math., 16 (1994), pp. 129–156.

[5] A. Chapman and Y. Saad, *Deflated and augmented Krylov subspace techniques*, Numer. Linear Algebra Appl., 4 (1997), pp. 43–66.

[6] E. De Sturler, *Iterative Methods on Distributed Memory Computers*, Ph.D. thesis, Delft University of Technology, Delft, The Netherlands, 1994.

[7] E. De Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 15–41.

[8] E. De Sturler, *Truncation Strategies for Optimal Krylov Subspace Methods*, Tech. Report TR-96-38, Swiss Center for Scientific Computing, Swiss Federal Institute of Technology Zurich (SCSC-ETH Zurich), ETH Zentrum, CH-8092, Switzerland, 1996; available online at http://www.cscs.ch/Official/PubTR.html.

[9] E. De Sturler and D. R. Fokkema, *Nested Krylov methods and preserving the orthogonality*, in Sixth Copper Mountain Conference on Multigrid Methods, N. D. Melson, T. A. Manteuffel, and S. F. McCormick, eds., NASA Conference Publication 3224, Part 1, NASA Langley Research Center, Hampton, VA, 1993, pp. 111–125.

[10] 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), pp. 345–357.

[11] R. W. Freund, *A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems*, SIAM J. Sci. Statist. Comput., 14 (1993), pp. 470–482.

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

[13] G. Golub and C. Van Loan, *Matrix Computations*, 2nd ed., The Johns Hopkins University Press, Baltimore, MD, 1989.

[14] A. Greenbaum, V. Pták, and Z. Strakoš, *Any nonincreasing convergence curve is possible for GMRES*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 465–469.

[15] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, New York, 1993.

[16] C. P. Jackson and P. Robinson, *A numerical study of various algorithms related to the preconditioned conjugate gradient method*, Internat. J. Numer. Methods Engrg., 21 (1985), pp. 1315–1338.

[17] J. Meijerink and H. 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), pp. 148–162.

[18] J. A. Meijerink and H. A. Van der Vorst, *Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems*, J. Comput. Phys., 44 (1981), pp. 134–155.

[19] R. B. Morgan, *A restarted GMRES method augmented with eigenvectors*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1154–1171.

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

[21] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing, Boston, 1996.

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

[23] 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), pp. 631–644.

[24] H. A. Van der Vorst and C. Vuik, *The superlinear convergence behaviour of GMRES*, J. Comput. Appl. Math., 48 (1993), pp. 327–341.

[25] H. A. Van der Vorst and C. Vuik, *GMRESR: A family of nested GMRES methods*, Numer. Linear Algebra with Appl., 1 (1994), pp. 369–386.