ELSEVIER

Journal of Computational and Applied Mathematics 69 (1996) 303–318

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

Restarted GMRES preconditioned by deflation

Jocelyne Erhel[a,*], Kevin Burrage[b], Bert Pohl[c]

[a]*INRIA, Campus de Beaulieu, 35042 Rennes, France*
[b]*Department of Mathematics, University of Queensland, Brisbane, 4072, Australia*
[c]*Seminar für Angewandte Mathematik, ETH Zürich, 8092 Zürich, Switzerland*

Received 14 July 1994; revised 4 April 1995

Abstract

This paper presents a new preconditioning technique for the restarted GMRES algorithm. It is based on an invariant subspace approximation which is updated at each cycle. Numerical examples show that this deflation technique gives a more robust scheme than the restarted algorithm, at a low cost of operations and memory.

Keywords: GMRES; Preconditioning; Invariant subspace; Deflation

AMS classification: 65F10; 65F15

1. Introduction

The GMRES algorithm is commonly used to solve large sparse nonsymmetric linear systems. The convergence behaviour of GMRES is related to the eigenvalues and also to the pseudo-eigenvalues (eigenvalues of closed matrices) [15]. Recently, the convergence behaviour of the full-version has been analysed [17] and superlinear convergence has been related to the convergence of Ritz values. However, because of memory requirements, a restarted version must be used in general. It has been observed that the convergence of the restarted algorithm depends heavily on the dimension of the Krylov subspace and may be slower than in the full case [3]. It appears as if the restarting procedure loses information on the smallest Ritz values. An adaptive procedure is proposed in [6] to choose the restart frequency according to the convergence and work requirements.

This paper presents a preconditioning technique which aims at keeping the information when restarting. The idea is to estimate the invariant subspace corresponding to the smallest eigenvalues. Indeed, the rate of convergence is mostly governed by these smallest eigenvalues.

* Corresponding author.

0377-0427/96/$15.00 © 1996 Elsevier Science B.V. All rights reserved
SSDI 0377-0427(95)00047-X

Many authors have proposed preconditioners or hybrid methods based on eigenvalue estimations. For the conjugate gradient algorithm, polynomial preconditioning aims at minimizing a certain norm. Two classical choices lead to the least-squares polynomial or to the minimax polynomial [7, 10]. The quality of the minimax polynomial depends strongly on the eigenvalue estimations. An adaptive procedure which is based on a recursive estimation of the eigenvalues is designed in [1, 13] for both definite and indefinite systems. Polynomial preconditioning was also studied recently for the GMRES algorithm in [19, 5].

Polynomial preconditioning is closely related to hybrid methods which combine, for example, a GMRES algorithm with a Richardson iteration. The idea is to use first GMRES to approximate both the solution and eigenvalues and then to use Richardson iteration using a polynomial derived from the estimated eigenvalues. A survey of hybrid methods which rely on eigenvalue estimations can be found in [16]. These estimations are usually done by the power method or by the Arnoldi technique but they can also be computed from modified moments [9]. Other hybrid solutions do not rely on eigenvalue estimations but use directly a polynomial generated by GMRES itself [16]. An alternative approach discussed in [18, 11] is to build a preconditioner based on the application of GMRES.

In this paper the eigenvalue technique is not used, but rather an invariant subspace approach. This idea has been developed in [4, 14] for the solution of nonlinear parameter-dependent systems of equations, in which a Newton method is used in the invariant subspace corresponding to the eigenvalues of the Jacobian near the unit disk and the usual fixed-point scheme is used in the orthogonal subspace. Therefore, the convergence is accelerated since the eigenvalues in the orthogonal subspace can be made small enough by a deflation approach. This idea has been applied in [8] in the linear case to the iterative methods based on various splittings of the matrix such as Jacobi or Gauss–Seidel splittings. It is shown there in numerical experiments that the relaxation methods can be dramatically accelerated.

However, the GMRES algorithm cannot be easily described as a fixed-point scheme. Here the convergence is related to the smallest eigenvalues. The full-GMRES version behaves as if the smallest eigenvalues were removed after some iterations. But this is no longer true in the restarted case. Therefore, the aim of this paper is to remove them by a preconditioner. After each cycle of GMRES, the preconditioner is updated by pulling out new eigenvalues. This scheme is different from the flexible GMRES method [11] because it executes a true GMRES cycle with a constant preconditioner inside the cycle.

At each restart, new eigenvectors are estimated in order to increase the invariant subspace. The preconditioner is equal to the projected matrix onto the approximated invariant subspace (up to a scaling factor) and is taken as the identity on the orthogonal subspace.

From the optimality of GMRES, this new scheme cannot converge faster than the full-GMRES version since it does not recover all the information kept in the full scheme. But numerical experiments show that for most matrices, it converges quickly whereas the nonpreconditioned version stalls or converges very slowly. It also requires less memory than a full version.

Of course, this technique can be combined with any preconditioner, estimating the invariant subspace of the preconditioned matrix. This approach can be used also to solve consecutive linear systems with the same matrix, as frequently happens in scientific computation. Thus after convergence of the first linear system, a quite accurate invariant subspace may be computed and used to build a roubust preconditioner for the subsequent resolution.

Another advantage of this method is the easy resolution of the preconditioner which requires merely level 2 dense BLAS operations. Moreover, the algorithm only requires a sparse matrix–vector product and can be applied to the so-called matrix-free versions of GMRES where the matrix is not stored.

This paper is organized as follows. In Section 2, the GMRES variants are given. Section 3 is devoted to the design of the preconditioner and to implementation issues; whereas, Section 4 describes numerical experiments.

2. Definitions and notations

In this section we recall the basic GMRES algorithms [12] and describe our version with variable preconditioning.

Thus consider the linear system $Ax = b$ with $x, b \in \mathbb{R}^n$ and with a nonsingular nonsymmetric matrix $A \in \mathbb{R}^{n \times n}$. The Krylov subspace $K(k; A; r_0)$ is defined by $K(k; A; r_0) = \text{span}(r_0, Ar_0, \ldots, A^{k-1}r_0)$. The GMRES algorithm uses the Arnoldi process to construct an orthonormal basis $V_k = [v_1, \ldots, v_k]$ for $K(k; A; r_0)$. The full GMRES method allows the Krylov subspace dimension to increase up to n and always terminates in at most n iterations. The restarted GMRES method on the other hand restricts the Krylov subspace dimension to a fixed value m and restarts the Arnoldi process using the last iterate x_m as an initial guess. It may stall if for example full GMRES converges at the nth iteration. Below is the algorithm for the restarted GMRES noted GMRES(m).

Algorithm GMRES(m)
ε is the tolerance for the residual norm;
convergence := false;
choose x_0;
until convergence **do**
  $r_0 = b - Ax_0$;
  $\beta = \| r_0 \|$;
  $v_1 := r_0/\beta$;
  for $j = 1, \ldots, m$ **do**
    $p := Av_j$;
    for $i = 1, \ldots, j$ **do**
      $h_{ij} := v_i^T p$;
      $p := p - h_{ij} v_i$;
    endfor
    $h_{j+1,j} := \| p \|$;
    $v_{j+1} := p/h_{j+1,j}$;
    compute $s = \| b - Ax_j \|$;
    if $s < \varepsilon$ **then**
      solve $\min_{y_j \in \mathbb{R}^j} \| \beta e_1 - \bar{H}_j y_j \|$;
      $x_j := x_0 + V_j y_j$;
      convergence := true;
      exit;
    endif;
  endfor;

solve $\min_{y_m \in \mathbb{R}^m} \| \beta e_1 - \bar{H}_m y_m \|$;
$x_m := x_0 + V_m y_m$;
if $\| b - Ax_m \| < \varepsilon$ **then**
convergence := true;
else
$x_0 = x_m$;
endif
enddo

For $1 \leqslant k \leqslant m$, for the matrix $\bar{H}_k = (h_{ij})$ is an upper Hessenberg matrix of order $(k+1) \times k$ and we get the fundamental relation

$$AV_k = V_{k+1} \bar{H}_k.$$

The GMRES algorithm computes $x = x_0 + V_k y_k$ where y_k solves the least-squares problem $\min_{y_k \in \mathbb{R}^k} \| \beta e_1 - \bar{H}_k y_k \|$. Usually a QR factorisation of $\bar{H}_k$ using Givens rotations is used to solve this least-squares problem.

The linear system can be preconditioned either at left or at right solving, respectively, $M^{-1}Ax = M^{-1}b$ or $AM^{-1}(Mx) = b$, where M is the preconditioning matrix.

Here we define a new scheme where the preconditioner is allowed to vary from one cycle to another, as opposed to a flexible scheme where the preconditioner changes inside a cycle. Since we still apply GMRES in each cycle, the results for GMRES(m) are still valid.

Algorithm PRECGMRES(m)
ε is the tolerance for the residual norm;
convergence := false;
choose x_0;
choose M;
until convergence **do**
$r_0 = b - Ax_0$;
Arnoldi process applied to AM^{-1} to compute V_m;
solve $\min_{y_m \in \mathbb{R}^m} \| \beta e_1 - \bar{H}_m y_m \|$;
$x_m := x_0 + M^{-1} V_m y_m$;
if $\| b - Ax_m \| < \varepsilon$ **then**
convergence := true;
else
$x_0 = x_m$;
update M
endif;
enddo

We chose a right preconditioning in order to guarantee a nonincreasing residual;

Proposition 2.1. *In the algorithm* PRECGMRES(m), *the norm of the residual* $r = b - A * x$ *is nonincreasing.*

Proof. This is true for GMRES(m) with a constant preconditioner. Here the algorithm always minimizes $\| b - Ax \|$, whatever the matrix M is. □

Remark 2.2. This is not true for a left-preconditioner since the residual is multiplied by M^{-1} and we actually observed an increase in the residual $\| b - Ax \|$ at each restart with a left-preconditioner.

3. The construction of the preconditioner

In this section, we describe how to build and to update the preconditioner and we discuss the convergence properties. The objective is to remove the smallest eigenvalues of A which are known to slow down the convergence of GMRES and to replace them by real positive eigenvalues equal to the largest modulus of the eigenvalues. The eigenvalues of the preconditioned matrix will be a multiple eigenvalue equal to this largest modulus and the eigenvalues of the original matrix which are not removed. We chose the largest modulus instead of 1 for scaling reasons because we do not know the range of the eigenvalues and 1 could be smaller than all the eigenvalues.

In the sequel, we assume that all eigenvalues of A are nondefective, or in other words that A is diagonalizable in $\mathbb{C}$. Let $|\lambda_1| \leqslant |\lambda_2| \leqslant \cdots \leqslant |\lambda_n|$ be the eigenvalues of A.

Let us assume that P is an invariant subspace of dimension r corresponding to the smallest r eigenvalues of A and let U be an orthonormal basis of P. The preconditioner will be based on a deflation technique such that the linear system is solved exactly in the subspace P. Now we give the main theoretical result of the paper.

Theorem 3.1. *If* $T = U^{\mathrm{T}}AU$ *and* $M = I_n + U(1/|\lambda_n| T - I_r)U^{\mathrm{T}}$, *then* M *is nonsingular and* $M^{-1} = I_n + U(|\lambda_n| T^{-1} - I_r)U^{\mathrm{T}}$ *and the eigenvalues of* AM^{-1} *are* $\lambda_{r+1}, \lambda_{r+2}, \ldots, \lambda_n, |\lambda_n|, |\lambda_n|$ *with a multiplicity at least* r.

Proof. Let $Z = (U, W)$ be an orthonormal basis of $\mathbb{R}^n$. In this basis, A is similar to a matrix $\tilde{A}$:

$$\tilde{A} = \begin{pmatrix} T & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{pmatrix}, \tag{1}$$

where $T = U^{\mathrm{T}}AU$ is the restriction of A onto the subspace P.

Now, since M can be written as $M = 1/|\lambda_n| UTU^{\mathrm{T}} + (I_n - UU^{\mathrm{T}})$, it is similar in the basis Z to the matrix $\tilde{M}$:

$$\tilde{M} = \begin{pmatrix} T/|\lambda_n| & 0 \\ 0 & I_{n-r} \end{pmatrix}, \tag{2}$$

where I_{n-r} is the identity matrix. Since T is nonsingular, $\tilde{M}$ is also nonsingular and its inverse is easily computed by

$$\tilde{M}^{-1} = \begin{pmatrix} |\lambda_n| T^{-1} & 0 \\ 0 & I_{n-r} \end{pmatrix}, \tag{3}$$

Now, if we come back to the original basis, we get the expression for M^{-1}:

$$M^{-1} = I_n + U(|\lambda_n| T^{-1} - I_r)U^T. \tag{4}$$

The preconditioned matrix AM^{-1} is therefore similar in the basis Z to the matrix $\tilde{E} = \tilde{A}\tilde{M}^{-1}$:

$$\tilde{E} = \begin{pmatrix} |\lambda_n| I_r & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{pmatrix},$$

so that its eigenvalues are $|\lambda_n|$ and the remaining eigenvalues of A, that is to say the eigenvalues of $\tilde{A}_{22}$. □

However, we do not know exactly the invariant subspace P but only an approximation $\bar{P}$. Using an orthonormal basis $(\bar{U}, \bar{W})$ we get now the same preconditioner M but a nonzero block in the $(2,1)$ position of the matrices $\bar{A}$ and $\bar{E}$:

$$\bar{A} = \begin{pmatrix} \bar{T} & \bar{A}_{12} \\ \bar{A}_{21} & A_{22} \end{pmatrix}, \qquad \bar{E} = \begin{pmatrix} |\bar{\lambda}_n| I_r & \bar{A}_{12} \\ |\bar{\lambda}_n| \bar{A}_{21} \bar{T}^{-1} & \bar{A}_{22} \end{pmatrix}, \tag{5}$$

where $\bar{T} = \bar{U}^T A \bar{U}$ and $|\bar{\lambda}_n|$ approximates the largest eigenvalue. This gives a perturbed matrix where the perturbation is given by the block $|\bar{\lambda}_n| \bar{A}_{21} \bar{T}^{-1}$. If this block is small enough, the eigenvalues of AM^{-1} are closed to $|\bar{\lambda}_n|$ and to the eigenvalues of $\bar{A}_{22}$ (recall that the eigenvalues are supposed to be nondefective), and we can still expect an improved convergence rate for this preconditioned GMRES.

Remark 3.2. Formula (4) shows that the preconditioner M^{-1} is merely a rank-r update of the identity matrix of the form $I + UXU^T$, where $X = |\lambda_n| T^{-1} - I_r$ is a dense matrix of order r, easy to compute. Thus the product by a vector $M^{-1} w$ can be implemented by dense BLAS2 operations of order $n \times r$, which is inexpensive and efficient.

3.1. Computing the invariant subspace

The GMRES algorithm provides the Hessenberg matrix $H_k = V_k^T A V_k$ which is the restriction of A onto the Krylov subspace $K(k; A; r_0)$. The eigenvalues of H_k are called the Ritz values.

Let us assume that H_k is decomposed into the Schur form with the eigenvalues ordered by increasing values with the Schur vectors S corresponding to the m smallest eigenvalues. Then the vectors $U = V_k S$ approximate the Schur vectors of A corresponding to the smallest eigenvalues of A. Since the largest Ritz value approximates the largest eigenvalue of A, we can therefore construct a matrix M.

After each restart we estimate new Ritz values which approximate the eigenvalues of AM^{-1} which in turn approximate the eigenvalues of $\tilde{A}_{22}$ and hence the remaining eigenvalues of A. We increase the size of the invariant subspace to get a more powerful preconditioner by adding new Schur vectors. In order to avoid loss of orthogonality, these vectors are orthogonalized against the previous basis U.

In some sense, this algorithm recovers the superlinear convergence of the full-GMRES version which behaves as if the smallest eigenvalues were removed. This approach has some merit when

dealing with a restarted version. In this case, the preconditioner keeps the information on the smallest Ritz values which would be lost by restarting.

Currently a fixed number l of eigenvalues are pulled out after each restart. More precisely, if H_m has complex conjugate eigenvalues or eigenvalues of the same modulus, then all the corresponding Schur vectors are extracted. Hence the number of extracted eigenvalues may vary in some extent from one restart to another.

Below is the new restarted GMRES, right-preconditioned by deflation, called DEFLGMRES(m, l):

Algorithm DEFLGMRES(m, l)
ε is the tolerance for the residual norm;
convergence := false;
choose x_0;
$M := I_n$;
$U := \{\ \}$;
until convergence **do**
  $r_0 = b - Ax_0$;
  Arnoldi process applied to AM^{-1} to compute V_m;
  solve $\min_{y_m \in \mathbb{R}^m} \| \beta e_1 - \bar{H}_m y_m \|$;
  $x_m := x_0 + M^{-1} V_m y_m$;
  if $\| b - Ax_m \| < \varepsilon$ convergence := true;
  else
    $x_0 = x_m$;
    compute l Schur vectors of H_m noted S_l;
    orthogonalize $V_m S_l$ against U;
    increase U by $V_m S_l$;
    $T := U^T A U$;
    $M^{-1} := I_n + U(|\lambda_n| T^{-1} - I_{r(j)}) U^T$;
  endif
enddo

Where $r(j)$ is the current dimension of the invariant subspace P at the jth cycle. Here S_l denotes a set of l Schur vectors in the Hessenberg matrix H_m of order m corresponding to its l smallest eigenvalues. The basis U is increased by adding the new Schur vectors $V_m S_l$ after they have been orthogonalized against the previous basis U.

Remark 3.3. In fact, the matrix M^{-1} is not computed but its expression is directly used to multiply by it.

Remark 3.4. If the vectors $V_m S_l$ are linearly dependent of U, then the orthogonalization behaves as if we added random vectors.

3.2. Implementation issues and complexity analysis

This new scheme involves the computation of the precondioner after each restart and the resolution of the preconditioned system at each iteration. Assume that at each restart, until

convergence, always l vectors $u = (u_1, \ldots, u_l)$ are added to the basis U of the approximate invariant subspace; then at the jth restart the matrix $T = U^T AU$ is of order $r(j) = l \times j$ and the matrix U has $r(j)$ vectors of size n.

To store the preconditioner, we need to store actually the basis U and also the matrix AU to save operations when computing $T = U^T(AU)$. So the memory cost for P restarts in the scheme DEFLGMRES(m, l) is

$$W_{\text{Total}} = 2nPl. \tag{6}$$

This can be bounded by stopping the deflation after a fixed number of restarts and by keeping the same preconditioner afterwards.

The total cost for P restarts in the scheme DEFLGMRES is

$$C_{\text{Total}} = \sum_{j=0}^{P-1} (C_{\text{Arnoldi}} + C_{\text{Prec}}(j) + C_{\text{Basis}}(j)), \tag{7}$$

where C_{Arnoldi} represents the cost for one restart, $C_{\text{Prec}}(j)$ is the cost to solve the preconditioned system and $C_{\text{Basis}}(j)$ is the cost to increase the basis at each restart.

All costs are evaluated in number of floating-point operations. For simplicity, we neglect the terms which are independent of n. The cost for a matrix–vector product is $2\tau n$ where τ is the mean number of nonzeros per row.

The cost for Q restarts of the classical GMRES(m) is then

$$\begin{aligned} Q \times C_{\text{Arnoldi}} &= Q * 2(\tau nm + nm(m+1) + 2nm) \\ &= Q * 2nm(\tau + m + 3). \end{aligned}$$

The preconditioned system can be solved using dense BLAS2 primitives, neglecting the cost for computing X (see Remark 3.1). We obtain a complexity of about

$$C_{\text{Prec}}(j) = 4nmr(j). \tag{8}$$

At each cycle, the new vectors u must be computed and orthogonalized with the previous set of vectors in order to increase the basis U. Also, the matrix $T = U^T AU$ must be updated and factorized, using the block decomposition

$$T = \begin{pmatrix} U^T AU & U^T Au \\ u^T AU & u^T Au \end{pmatrix}.$$

This gives

$$C_{\text{Basis}}(j) = 2nl(\tau + m + l + 4r(j)). \tag{9}$$

The global complexity for P restarts of the new scheme is

$$C_{\text{Total}} = P * 2n(m(\tau + m + 3) + l(\tau - l) + l(m + 2l)P). \tag{10}$$

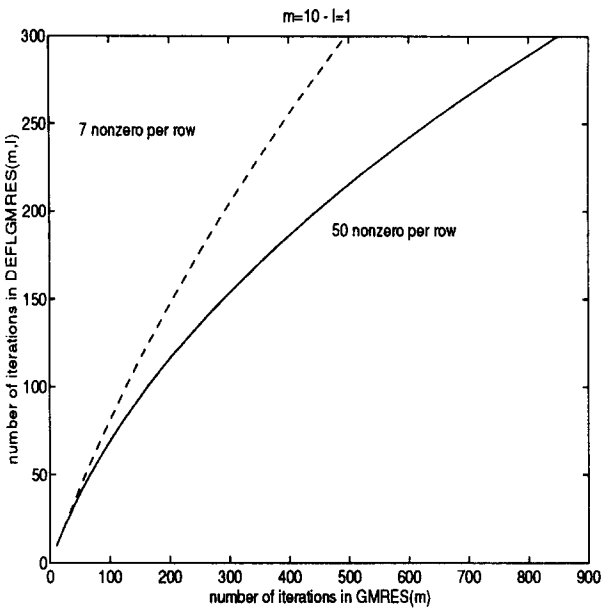

Fig. 1. Cost evaluation for GMRES(m) and DEFLGMRES(m, l).

Hence, the scheme will perform better than the classical restarted GMRES scheme if

$$C_{\text{Total}} < Q \times C_{\text{Arnoldi}}. \tag{11}$$

Fig. 1 plots the curve where both costs are the same for $m = 10$ and $l = 1$ and for two values of τ, $\tau = 7$ and $\tau = 50$. Under each curve, DEFLGMRES(m, l) is more efficient than GMRES(m) and above the curve GMRES(m) is more efficient. It can be seen that a modest acceleration in convergence is sufficient to obtain good performances.

4. Numerical results

We have tested the algorithm using Matlab and the template of GMRES provided in netlib [2]. The matrices are taken from [3] and have the form $A = SDS^{-1}$ with A, S, $D \in \mathbb{R}^{100 \times 100}$ and with $S = (1, \beta)$ a bidiagonal matrix with 1 on the diagonal and β on the upper subdiagonal.

The system $Ax = b$ is solved for right-hand sides $b = (1, \ldots, 1)^T$ and GMRES starts with $x_0 = 0$. The tolerance ε for convergence is set to 10^{-8}. In the deflated version, noted DEFLGMRES(1), one real or two complex conjugate eigenvalues are pulled out at each restart ($l = 1$ or 2). The total number of extracted eigenvalues is bounded by r. Once this number is reached, the preconditioning is no more updated.

Results for six different examples of dimension 100 are presented with the following characteristics:

No.	β	D	condition number $\kappa(S)$
1	0.9	diag(1, 2, ... , 100)	18.334
2	1.1	diag(1, 2, ... , 100)	151570
3	0.9	diag(1, 100, 200, ... , 10000)	
4	0.9	diag($-10, -9, \ldots, -1, 1, 2, \ldots, 90$)	
5	0.9	D_1	
6	0.9	D_2	

where D_1 and D_2 are block diagonal matrices with 2×2 blocks. D_1 has 12 complex eigenvalues and other eigenvalues are evenly distributed: $1 \pm i, 2 \pm 2i, 3 \pm 3i, -3 \pm i, -2 \pm 2i, -1 \pm 3i$, [13 ... 100]. D_2 has only complex eigenvalues: $a \pm ai$, $a - 26 \pm ai$, $a = 1, \ldots, 25$.

Figs. 2–7 show the convergence rate for full-GMRES noted full, GMRES(10) noted restarted, DEFLGMRES(10, 1) noted deflated. GMRES(10) converges only for example 1 and very slowly for example 4. It stalls for examples 2, 3, 5, 6 respectively because of ill-conditioned eigenvalues ($\kappa(S)$ is large), a large interval of eigenvalues and complex eigenvalues. Convergence is slow in example 4 because of negative eigenvalues. On the other hand, DEFLGMRES(10, 1) converges for all examples exept example 6 where full-GMRES converges at the nth iteration. GMRES(m) converges slowly for all examples except example 6 when increasing sufficiently m, as shown in Tables 1–6. These tables compare the number of iterations and number of operations, given in the column flops in millions of operations. The memory requirements, measured by the number of vectors of

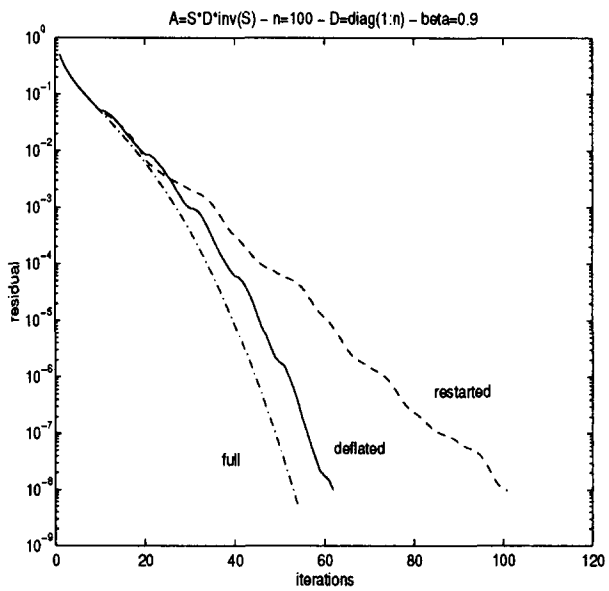

Fig. 2. Convergence rates for example 1.

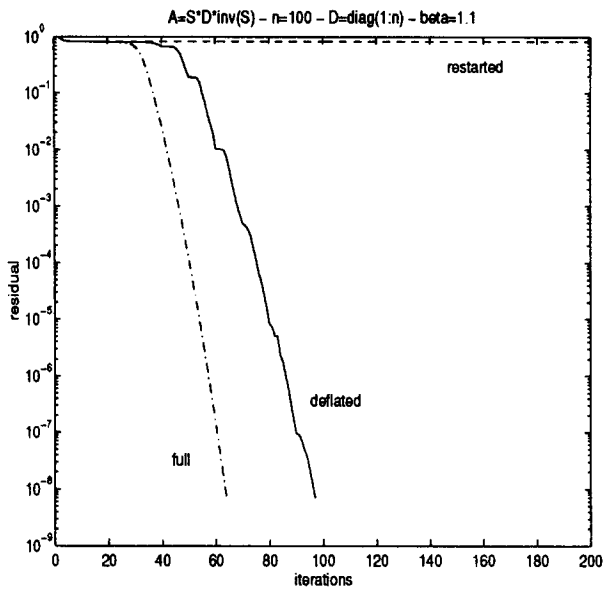

Fig. 3. Convergence rates for example 2.

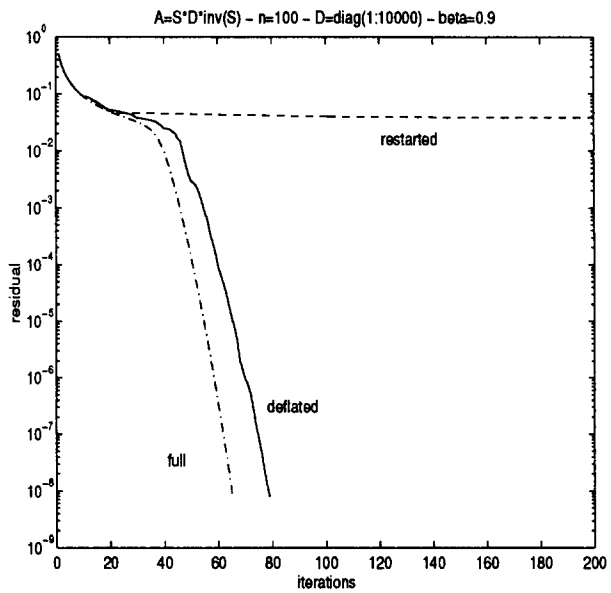

Fig. 4. Convergence rates for example 3.

size n, are given in the full case by the number of iterations, for GMRES(m) by m and for DEFLGMRES(m, l) by $m + 2r$ (recall that r is the total number of extracted eigenvalues and that one or two eigenvalues are pulled out at each cycle).

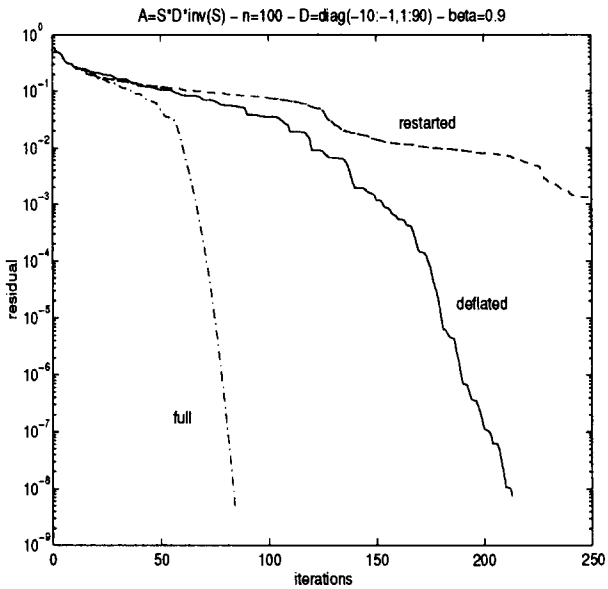

Fig. 5. Convergence rates for example 4.

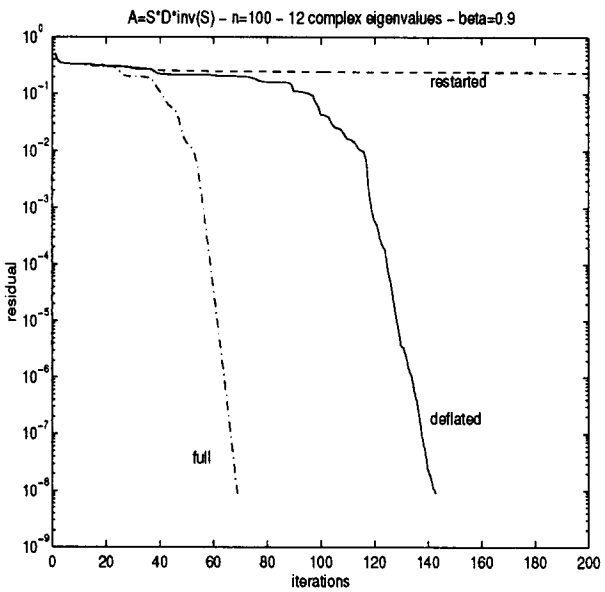

Fig. 6. Convergence rates for example 5.

These examples show that our new deflated scheme DEFLGMRES(m, l) is more robust than GMRES(m) which stalls in most examples for small m and is more efficient when both converge. For a same memory cost, DEFLGMRES has a lower operation count than GMRES. However,

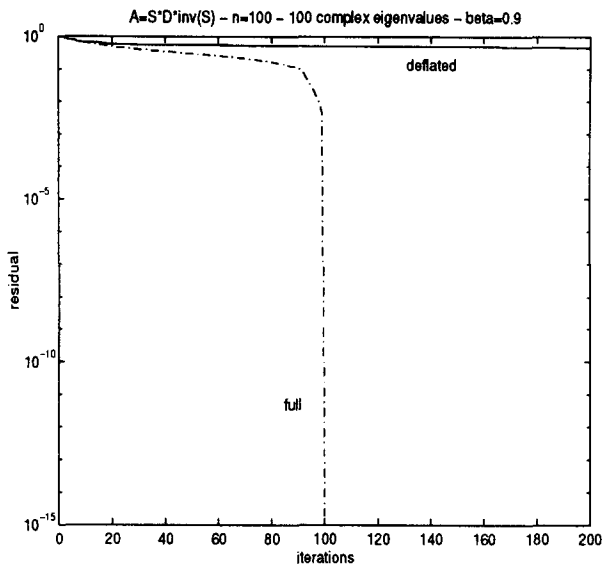

Fig. 7. Convergence rates for example 6.

Table 1
Results for example 1

Method	r	Iterations	Flops
FULL	0	54	1.2167
GMRES(10)	0	101	1.5601
GMRES(20)	0	96	1.5494
DEFLGMRES(10, 1)	1	97	1.5629
DEFLGMRES(10, 1)	2	81	1.4105
DEFLGMRES(10, 1)	3	70	1.2820
DEFLGMRES(10, 1)	4	64	1.2528
DEFLGMRES(10, 1)	5	63	1.2993
DEFLGMRES(10, 1)	6	62	1.3395

a sufficient number of eigenvalues must be pulled out to ensure convergence in difficult cases. Though DEFLGMRES is slower and requires more operations than full-GMRES, it requires also less memory. In example 3, full-GMRES and DEFLGMRES need respectively 65 and 24 vectors of length n. Moreover, the operation counts should become favorable to DEFLGMRES compared to full-GMRES when increasing n, thus the number of iterations.

5. Perspectives

This paper presents a new GMRES scheme defining a variable preconditioner based on the estimation of Schur vectors and on deflation techniques. Examples presented here show that this

Table 2
Results for example 2

Method	r	Iterations	Flops
FULL	0	64	1.5694
GMRES(10)	0	∞	*
GMRES(30)	0	> 500	*
GMRES(40)	0	157	3.0861
DEFLGMRES(10, 1)	6	∞	*
DEFLGMRES(10, 1)	8	98	2.3070
DEFLGMRES(10, 1)	13	97	2.4898

Table 3
Results for example 3

Method	r	Iterations	Flops
FULL	0	65	1.6069
GMRES(10)	0	∞	*
GMRES(20)	0	> 500	*
GMRES(40)	0	237	4.6692
DEFLGMRES(10, 1)	5	86	1.7015
DEFLGMRES(10, 1)	7	79	1.6989

Table 4
Results for example 4

Method	r	Iterations	Flops
FULL	0	84	2.3981
GMRES(10)	0	∞	*
GMRES(50)	0	> 500	*
GMRES(60)	0	300	7.1154
DEFLGMRES(10, 1)	10	∞	*
DEFLGMRES(10, 1)	12	321	7.4932
DEFLGMRES(10, 1)	16	238	6.2022
DEFLGMRES(10, 1)	21	213	6.0091

preconditioned restarted scheme converges whereas the unpreconditioned restarted scheme stalls or converges much slower. The memory requirements may be bounded to a small number of vectors. The scheme allows matrix-free versions of GMRES and makes only use of dense matrix–vector multiplications.

This new scheme must be compared to other preconditioning techniques, such as flexible GMRES, GMRESR, on large sparse matrices.

Table 5
Results for example 5

Method	r	Iterations	Flops
FULL	0	69	1.7620
GMRES(10)	0	∞	*
GMRES(40)	0	∞	*
GMRES(50)	0	447	9.6937
DEFLGMRES(10, 1)	8	∞	*
DEFLGMRES(10, 1)	12	195	4.5698
DEFLGMRES(10, 1)	17	143	3.7658

Table 6
Results for example 6

Method	r	Iterations	Flops
FULL	0	100	3.1897
GMRES(10)	0	∞	*
DEFLGMRES(10, 1)	26	∞	*

Further work needs to be done in considering different strategies for updating the preconditioning matrix, including the development of an adaptive approach. Other possible approaches include a version where the dimension of U does not increase very much but approximates an invariant subspace more accurately.

Finally, it is intended to extend this work to a parallel implementation in a MIMD environment.

Acknowledgements

Part of this work was done while Kevin Burrage was a guest professor of the Seminar für Angewandte Mathematik at ETH Zürich and while Jocelyne Erhel and Bert Pohl were Ethel Raybould Fellows in the Department of Mathematics at the University of Queensland in Australia. Kevin Burrage gratefully acknowledges the financial support provided by the Intel Corporation.

References

[1] S.F. Ashby, Polynomial preconditioning for conjugate gradient methods, Ph.D. Thesis, Univ. of Illinois at Urbana-Champaign, 1987.

[2] R. Barret, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* (SIAM/netlib, Philadelphia, 1993).

[3] Y. Huang and H. van der Vorst, Some observations on the convergence behavior of GMRES, Technical Report 89-09, Faculty of technical mathematics and informatics, Delft, 1989.
[4] H. Jarausch and W. Mackens, Numerical treatment of bifurcation problems by adaptive condensation, in: Weber, Kupper and H.D. Mittelmann, Eds., *Numerical Methods for Bifurcation Problems* (Birkhäuser, Basel, 1987).
[5] W. Joubert, A robust GMRES-based adaptive polynomial preconditioning algorithm for nonsymmetric linear systems, *SIAM J. Sci. Comput.* **15** (1994) 427–439.
[6] W. Joubert, On convergence behaviour of the restarted gmres algorithm for solving nonsymmetric linear systems, *J. Numer. Linear Algebra Appl.* **1** (1994) 427–448.
[7] C.A. Michelli, O.G. Johnson and G. Paul, Polynomial preconditioners for conjugate gradient calculations, *SIAM J. Numer. Anal.* **20**(2) (1983) 362–376.
[8] B. Pohl, K. Burrage and J. Erhel, A deflation technique for linear systems of equations, Technical Report 94-02, ETH, Zurich, June 1994.
[9] L. Reichel, D. Calvetti and G.H. Golub, An adaptive Chebyshev iterative method for nonsymetric linear systems based on modified moments, *Numer. Math.* **67** (1994) 21–40.
[10] Y. Saad, Practical use of polynomial preconditioning for the conjugate gradient method, *SIAM J. Sci. Statist. Comput.* **6**(4) (1985) 865–881.
[11] Y. Saad, A flexible inner-outer preconditioned GMRES algorithm, *SIAM J. Sci. Statist. Comput.* **14**(2) (1993) 461–469.
[12] Y. Saad and M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.* **7** (1986) 856–869.
[13] P.E. Saylor, S.F. Ashby and T.A. Manteuffel, Adaptive polynomial preconditioning for Hermitian indefinite linear systems, *BIT* **29** (1989) 583–609.
[14] G.M. Shroff and H.B. Keller, Stabilization of unstable procedures: the recursive projection method, *SIAM J. Numer. Anal.* **30**(4) (1993) 1099–1120.
[15] L.N. Trefethen, N.M. Nachtigal and S.C. Reddy, How fast are non-symmetric matrix computations?, *SIAM J. Matrix Anal. Appl.* **13**(3) (1992) 778–795.
[16] L.N. Trefethen, N.M. Nachtigal and L. Reichel, A hybrid GMRES algorithm for nonsymmetric linear systems, *SIAM J. Matrix Anal. Appl.* **13**(3) (1992) 796–825.
[17] H.A. van der Vorst and C. Vuik, The superlinear convergence behaviour of GMRES, *J. Comput. Appl. Math.* **48** (1993) 327–341.
[18] H.A. van der Vorst and C. Vuik, GMRES: a family of nested GMRES methods, *J. Num. Linear Algebra Appl.* **1** (1994) 369–386.
[19] M.B. van Gijzen, A polynomial preconditioner for the GMRES algorithm, *J. Comput. Appl. Math.* **59** (1) (1995) 91–107.