Convergence behaviour of deflated GMRES(m) algorithms on AP3000

Kentaro Moriya Takashi Nodera*

(Received 7 August 2000)

Abstract

GMRES(m) method, the restarted version of the GMRES (generalized minimal residual) method, is one of the major iterative methods for numerically solving large and sparse nonsymmetric problems of the form $Ax = b$. However, the information of some eigenvectors that compose the approximation disappears and then the good approximate solution cannot be obtained, because of this restart. Recently, in order to improve such a weak point, some algorithms which

*Keio University, 3-14-1 Hiyoshi Kohoku, Yokohama 223-8522, Japan

[0]See http://anziamj.austms.org.au/V42/CTAC99/Mori for this article and ancillary services, © Austral. Mathematical Soc. 2000. Published 27 Nov 2000.

named MORGAN, DEFLATION and DEFLATED-GMRES algorithm, have been proposed [7, 10, 12]. Those algorithms add the information of eigenvectors that can be obtained in the previous restart frequency. In this paper, we study those algorithms and compare their performances. From the numerical experiments on the distributed memory machine Fujitsu AP3000, we show that DEFLATED-GMRES(m, k) method performs the good reduction of residual norms in these algorithms.

Contents

1 Introduction

We now consider the numerical solution of large and sparse linear systems of equations

$$Ax = b \tag{1}$$

by using of iterative techniques. Numerical formulations of partial differential equation problems give rise to systems of large sparse nonsymmetric linear systems. For these systems robust and fast iterative algorithms must be used. Now, there are quite a few Krylov subspace algorithms for solving the above linear systems, e.g. GMRES, BiCGStab(ℓ) and QMR. Of these algorithms, one kind of the most profitable technique is based on the orthogonal projection, typified by GMRES algorithm.

The GMRES begins with an initial guess x_0 and characterizes the kth iterate as

$$x_k = x_0 + y_k$$

where y_k solves the least squares problem such as

$$\min_{y\in\mathcal{K}_k(r_0)} \|b - A(x_0 + y)\|_2 = \min_{y\in\mathcal{K}_k(r_0)} \|r_0 - Ay\| \tag{2}$$

In equation (2), the initial residual vector $r_0 = b - Ax_0$ and for $v \in \mathcal{R}^n$, $\mathcal{K}_k(v)$ is the Krylov subspace

$$\mathcal{K}_k(v) \equiv \operatorname{span}\{v, Av, A^2v, \ldots, A^{k-1}v\}.$$

At the centre of the usual implementations of GMRES algorithm is Arnordi process, which is given in Saad *et al.* [2], to construct an orthonormal basis for the Krylov subspace. For starting the GMRES step, the process is applied with $v_1 = r_0/\|r_0\|_2$. At each step, the orthonormalization can be done using the modified Gram-Schmidt process. Each $\{v_1, v_2, \ldots, v_n\}$ is an orthonormal basis of $\mathcal{K}_k(r_0)$ that reduces the least squares problem (2) to an upper Hessenberg least squares problem. This is normally solved by QR decomposition with Givens rotations. So full orthogonalization is needed, but more expensive as the subspace are increased. Moreover, the storage requirements also increase. For such reason, we can usually use restarting when the subspace reaches a certain size of m. We call this algorithm the restarted GMRES(m). The disadvantage with restarting is that some information of eigenvector is lost at the time of the restart. The subspace is cast off, and then restarted GMRES algorithms slow down the convergence of residual norm.

In Section 2, we will briefly recall some related properties of deflated GMRES(m) algorithms. Next, in Section 3, we illustrate on some results of numerical experiments on AP3000. At last, we give some concluding remarks.

2 Deflated GMRES(m) Algorithms

In deflation algorithms, there are two distinct ways of developing some knowledge about approximate eigenvectors of AM^{-1} to accelerate convergence.

The first approach is just to add the desired eigenvectors directly to the Krylov subspace. For eigenvalue problems, this is absolutely natural [1]. For solving the linear system (1), this approach has been proposed by Morgan [7]. Morgan injects eigenvectors $u_1, u_2, \ldots, u_p$ into Krylov subspace $\mathcal{K}_k(r_0)$, so the solution x then belong to

$$x_0 + \operatorname{span}\{r_0, Ar_0, \ldots, A^{k-p+1}r_0, u_1, u_2, \ldots, u_p\}$$

The proposed algorithm is called MORGAM(m, k), where m is the dimension of the Krylov subspace and k is the maximal dimension of the invariant subspace.

The second approach is to explicitly deflate the eigenvectors from the matrix. For example, by solving the equation

$$(I - \sigma uv^T)AM^{-1}x = (I - \sigma uv^T)b,$$

where M is a preconditioner. Also, v is a left eigenvector of AM^{-1} and u is arbitrary vector but is often chosen to be equal to v. This sort of algorithm has been proposed by Erhel *et al.* [10], and Burrage and Erhel [12]. Erhel *et al.* [10] proposed DEFLATED-GMRES(m, k) algorithm. This algorithm is updatethe preconditioner M at each restart. One another algorithm is proposed by Burrage and Erhel [12] which is called DEFLATION(m, k) algorithm. This algorithm is designed a new preconditioning deflation which continuously updates the vectors in U, where the matrix U is a basis of approximated invariant subspace. The detailed derivation of this algorithm is given in their paper [12].

Table 1: AP3000 specification

Interprocessor networks	AP-Net 200MB/sec Barrier synchronization
Cell processor	UltraSPARC IU + FPU
Cache memory	512 KB
Local memory	256 MB

3 Numerical Experiments

In this section we describe numerical experiments which compare parallel performance of the algorithms described in previous section on the test problem and in multiple instruction multiple data (MIMD) machine AP3000 [14]. As a preparation for the experiments, we have a brief introduction of AP3000. This machine is the workstation cluster, and each processor is made by UltraSPARC chip which runs at 300MHz. The specification of AP3000 is given in Table 1. We also use MPI as communicative library [8].

In all test runs, we utilize the components of initial guess $x_0 = 0$ and double precision arithmetic. For the sake of simplicity, we use the simple stopping criterion $(\|r_m\|_2/\|r_0\|_2) \leq 0.1 \times 10^{-12}$, where r_m is the residual vector at mth iteration.

The convergence behaviour of the deflated GMRES(m) algorithms is now

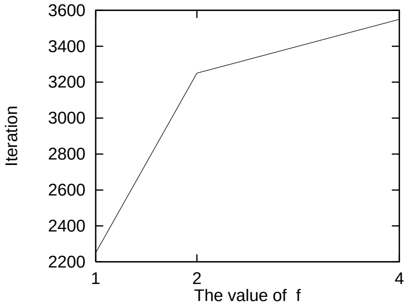


Figure 1: The relation between number of iterations and f for deflated-gmres(50,4) method in example 1.

illustrated by simple example.

3.1 Example 1

We now consider the following linear system of equations $Ax = b$:

$$A = \begin{pmatrix} 1 & 0.1 & & & \\ & 2 & 0.1 & & \mathbf{0} \\ & & \ddots & \ddots & \\ & & & \ddots & 0.1 \\ \mathbf{0} & & & & 16384 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}, \tag{3}$$

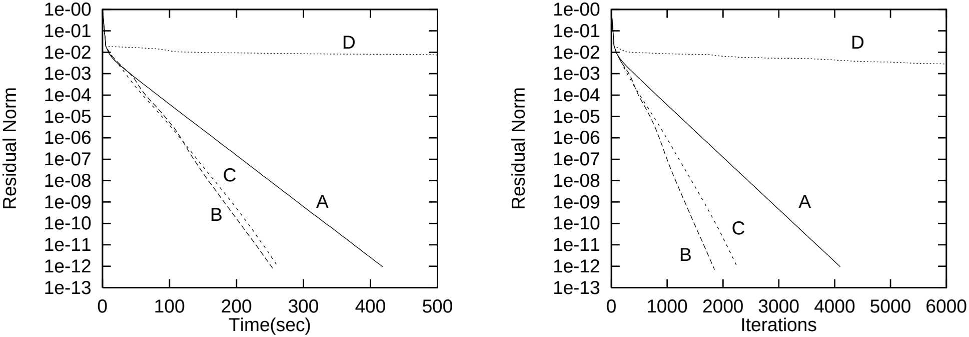
where $A \in R^{16384 \times 16384}$. This example is chosen from Morgan [4, 7]. The coefficient matrix is a triangular matrix and its eigenvalues are therefore real and are diagonal elements. You see that the distribution of eigenvalues are distinct and well separated. This problem is somewhat artificial but has been used. We solve the linear system (3) using standard GMRES(m), MORGAN(m, k), DEFLATED-GMRES(m, k) and DEFLATION(m, k) algorithms. For the deflated GMRES algorithms stated above, these consist of adding approximate eigenvectors obtained from previous Arnordi step to Krylov subspace. The test uses two subspaces dimension of 50 and 70, the last 2 and 4 of which are approximate eigenvectors. For example, these are denoted by deflated GMRES$(50, 2)$, $\cdots$, deflated GMRES$(70, 4)$.

At first, we will decide how many eigenvalues are deflated at each restart. Therefore, we investigate the relation between iteration and the number of eigenvectors f which added by each restart frequency. Figure 1 shows representative plots of above relations. From this figure, we can get the result of minimum iteration at $f = 1$. So we set $f = 1$.

The convergence results of numerical experiments for these algorithms are given in Table 2. Runs for which convergence was not possible in 30 minutes are labelled by $(\ldots)$. In this table, it can be seen that the convergence for MORGAM(m, k) and DEFLATED-GMRES(M, k) algorithms are always better than for the standard restarted GMRES(m) algorithm when measured in terms of both time and iterations. On the other hand, the DEFLATION(m, k) algorithm could not converge at all for this problem.

TABLE 2: The numerical result in Example 1 (time: computational time(sec), iter: the number of iterations).

Algorithm	time	iter
GMRES(50)	418	4100
GMRES(70)	450	3220
MORGAN(48,2)	$\underline{255}$	1848
MORGAN(46,4)	255	1746
MORGAN(68,2)	367	1748
MORGAN(66,4)	315	1396
DEFLATED-GMRES(50,2)	358	3250
DEFLATED-GMRES(50,4)	261	2250
DEFLATED-GMRES(70,2)	330	2240
DEFLATED-GMRES(70,4)	260	1680
DEFLATION(50,2)	. . .	. . .
DEFLATION(50,4)	. . .	. . .
DEFLATION(70,2)	. . .	. . .
DEFLATION(70,4)	. . .	. . .



(a) Time vs. Residual norm

(b) Iterations vs. Residual norm

Figure 2: The convergence history of the residual norm in Example 1, A: gmres(50), B: morgan(48,2), C: deflated-gmres(50,4), D: deflation(50,4).

Figure 2 shows the graph of time vs. residual and of iterations vs. residual norm, respectively. From this figure, MORGAN(m,k) and DEFLATED-GMRES(m,k) algorithms have a fairly good convergence. Both algorithms have similar convergence slopes toward the final phase of the iteration. It can be seen that deflation is effective in speedup convergence early in the solution process. But, the convergence behaviour of DEFLATION(m,k) algorithm seems fairly flat and nearly stagnate for a large period of the convergence history.

3.2 Example 2

We consider the following three-dimensional problem [3]

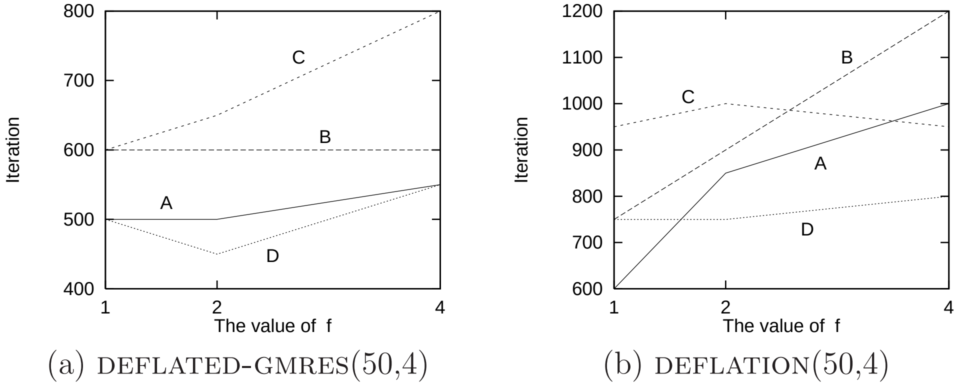
$$\begin{aligned} -u_{xx} - u_{yy} - u_{zz} + Ru_x &= g(x,y,z) \text{ on } \Omega, \\ u(x,y)|_{\partial\Omega} &= 0, \end{aligned}$$

where $\Omega \equiv [0,1]\times[0,1]\times[0,1]$, and R is a Reynolds number. The right hand side is $g(x,y,z)$ is chosen so that the solution is

$$u(x,y,z) = \exp(xyz)\sin(\pi x)\sin(\pi y)\sin(\pi z).$$

We discretize using central finite difference scheme on the uniform $80\times 80\times 80$ grid, producing a linear system of order $n = 51200$.

At first, as same as the Example 1, we will decide to how many eigenvalues are deflated at each restart. We investigate the relation between iterations



(a) DEFLATED-GMRES(50,4) (b) DEFLATION(50,4)

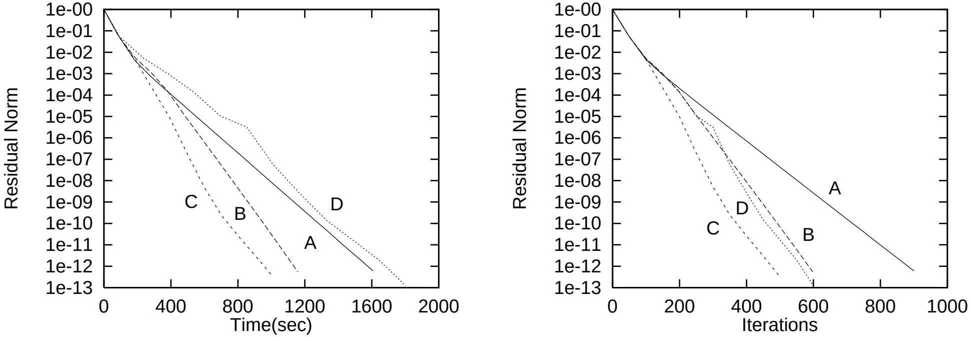
FIGURE 3: The relation between number of iterations and f for DEFLATED-GMRES(50,4) method and DEFLATION(50,4) method in Example 2, A: $R = 1.0$, B: $R = 10.0$, C: $R = 100.0$, D: $R = 1000.0$.

and number of deflated eigenvalues f. In Figure 3, we show the relation between iterations and f for DEFLATED-GMRES$(50, 4)$ and DEFLATION(50,4) algorithms. In this figure, we can get the minimum number of iterations, when we use $f = 1$ in most cases. So we set $f = 1$.

The convergence results of numerical experiments are given in Table 3 for various values of R. We also compare the standard GMRES(m) with incomplete LU (ILU) decomposition in our experiments. The ILU decomposition is frequently used for preconditioning technique, but its process includes forward and backward substitution (see Bruaset [6]). This is particularly difficult to implement on MIMD parallel computer. Bastian *et al.* [5] proposed some convenient techniques (i.e. block divided method) for parallelizing the ILU preconditioner for the linear system coming from finite difference discretization of elliptic PDE problem (see Nodera [11]). However, since the communication overhead is significantly large in some case, the ILU preconditioner can not often perform well. So, we use the modified parallelization technique of ILU decomposition which based on the original block divided method. This new algorithm decreases the communication overhead and determines the appropriate band-size. In our numerical tests, we implement the ILU decomposition using the block divided technique. The detailed implementation of ILU decomposition is given by Moriya *et al.* [13].

This table shows the convergence of deflated GMRES algorithms. Mostly DEFLATED-GMRES(m, k) algorithm is better convergence in terms of computational time. Especially, for low Reynolds numbers DEFLATED-GMRES(m, k) algorithm converges rather fast compared to the other algorithms. However, the standard GMRES(m) with ILU decomposition fairly good convergence in case of higher order Reynolds number such as $R = 100$ and 1000. For DEFLATION(m, k) algorithm, the eigenvalue calculation procedure takes more time than the other deflation algorithms. It is the cause that its computational time has not been reduced.

Figure 4 shows the reduction of in residual norm as deflated GMRES(m, k) and standard GMRES(m) algorithm for $R = 1$. As we observed, all the curves, except the standard GMRES(50) curve, have similar convergence slopes at the early phase of iteration. The first about 100 steps of all the algorithms are identical. Differences appear at around this step 100, each algorithms



(a) Time vs. Residual norm

(b) Iterations vs. Residual norm

Figure 4: The convergence history of the residual norm in Example 2 ($R = 1.0$), A: GMRES(50), B: MORGAN(48,2), C: DEFLATED-GMRES(50,4), D: DEFLATION(50,4).

Table 3: The numerical result in Example 2 (time: computational time (second), iter: the number of iterations).

Algorithm	R							
	1.0		10.0		100.0		1000.0	
	time	iter	time	iter	time	iter	time	iter
GMRES(50)	1607	900	1233	700	1461	800	1183	650
GMRES(70)	1705	700	2046	840	2041	840	1694	700
GMRES(50)+ILU	1515	450	1373	400	$\underline{877}$	250	$\underline{693}$	200
GMRES(70)+ILU	1658	350	1679	350	1367	280	1346	280
MORGAN(48,2)	1158	598	1360	698	1845	948	1696	848
MORGAN(46,4)	1314	646	1419	696	1944	946	1871	896
MORGAN(68,2)	1462	558	1856	698	2199	838	2253	838
MORGAN(66,4)	1701	626	1923	696	2697	976	2452	906
DEFLATED-GMRES(50,2)	$\underline{963}$	500	1258	650	1568	800	1171	600
DEFLATED-GMRES(50,4)	1008	500	$\underline{1195}$	600	1221	600	1025	500
DEFLATED-GMRES(70,2)	1270	490	1425	560	1970	770	1786	700
DEFLATED-GMRES(70,4)	1299	490	1493	560	1490	560	1503	560
DEFLATION(50,2)	1730	600	2194	750	2557	850	2631	900
DEFLATION(50,4)	1802	600	2103	700	2899	950	2258	750
DEFLATION(70,2)	2099	560	2938	770	3163	840	2905	770
DEFLATION(70,4)	2417	630	2687	700	3342	840	3568	910

converge quite different ways. It is interesting to note that in this case DEFLATED-GMRES(50,4) algorithm performs better convergence history than the others.

3.3 Example 3

We consider a model of the 3-dimensional Navier-Stokes problem stated as follows [9].

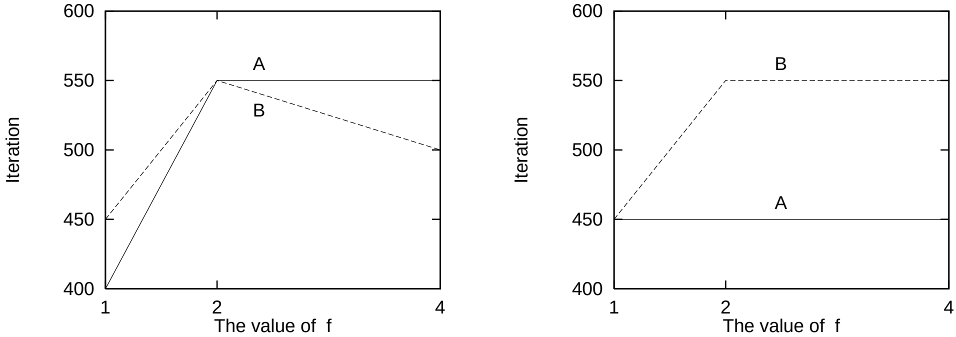
$$a_1 u_{xx} + a_2 u_{yy} + a_3 u_{zz} + R(a_4 u_x + a_5 u_y + a_6 u_z) + a_7 u = g(x, y, z) \quad \text{on } \Omega \quad (4)$$

where

$$\begin{aligned} a_1 &= 2 + \sin(2\pi x)\cos(2\pi y)\cos(2\pi z) \\ a_2 &= 2 + \cos(2\pi x)\sin(2\pi y)\cos(2\pi z) \\ a_3 &= 2 + \cos(2\pi x)\cos(2\pi y)\sin(2\pi z) \\ a_4 &= \sin(4\pi x), \quad a_5 = \sin(4\pi y), \quad a_6 = \sin(4\pi z) \\ a_7 &= \sin(2\pi x)\sin(2\pi y)\sin(2\pi z) \end{aligned}$$

and the parameter R simulates a Reynolds number. We pose equation (4) on $\Omega = [1, 0] \times [1, 0] \times [1, 0]$ with Dirichlet boundary conditioned and discretize using central difference scheme on the uniform $64 \times 64 \times 64$ grid, producing a linear system of order of $n = 262144$. The right-hand side is determined so that the exact solution is

$$u(x, y, z) = \sin(2\pi x)\cos(2\pi y)\sin(2\pi z).$$



(a) DEFLATED-GMRES(50,4) (b) DEFLATION(50,4)

Figure 5: The relation between number of iterations and f for DEFLATED-GMRES(50,4) method and DEFLATION(50,4) method in Example 3, A: $R = 1.0$, B: $R = 10.0$.

Figure 5 shows that the relation between number of iterations and f for DEFLATED-GMRES(50,4) and DEFLATION(50,4) algorithm in Example 3. From this figure, we will set $f = 1$.

Table 4 shows the results of numerical experiments for various value of R. Run for which convergence was not possible in 70 minutes are labelled by $(\ldots)$. In this table, it can be seen that the convergence of deflated GMRES(m, k) algorithm is better convergence in terms of the computational time. Most significantly, the DEFLATED-GMRES$(50, 4)$ and DEFLATED-GMRES(70,4) algorithms gave somewhat better residual reduction in some experiments. In case of $R = 100$ and $R = 1000$, the standard GMRES(m) algorithm has failed to converge, but the most of both MORGAN(m, k) and DEFLATED-GMRES(m, k) algorithms have fairly good converged. For the Reynolds number $R = 1000$, all algorithms fail to converge.

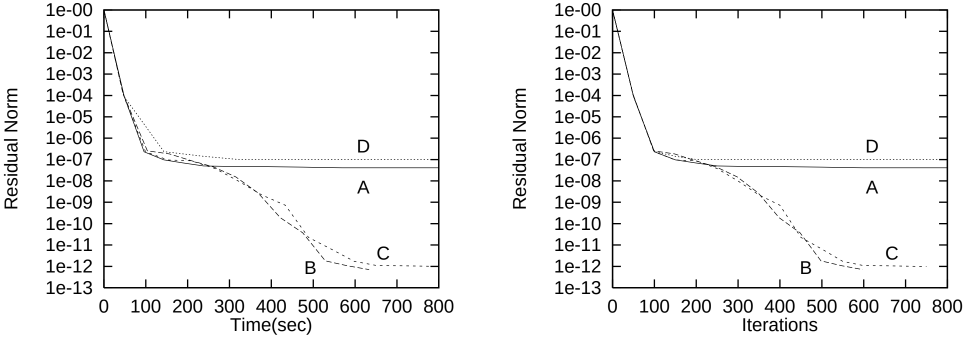
Figure 6 plotted the convergence history for these runs at $R = 100$. In this figure, we can see that cases where an initial decrease in the residual norm is followed by stagnation. The first 100 steps of all the algorithms have a similar convergence slopes toward the final phase of iterations. Differences appear at around step 100. MORGAN(48,2) and DEFLATED-GMRES(50,4) algorithms are still identical until final convergence step. At around step 100, the standard GMRES(50) and DEFLATION$(50, 4)$ algorithms are referred to as stalling. This is sometime occurs for the case of indefinite matrix. This figure also shows that the MORGAN(48,2) and DEFLATED-GMRES(50,4) algorithms keep the residual size better behaved than the standard GMRES(m) and DEFLATION(m, k) algorithms over the course of run. Especially, MORGAN$(48, 2)$ is slightly better than DEFLATED-GMRES$(50, 4)$ algorithm.

4 Concluding Remark

We have compared in detailed the convergence behaviour of three different types of deflated GMRES algorithms for solving a linear system of equation (1) and have exposed some important features of deflated GMRES algorithms. Numerical experiments have shown that the DEFLATED-GMRES(m, k) algorithm is competitive with and more efficient than the standard GMRES(m)

TABLE 4: The numerical result in Example 3 (time: computational time (second), iter: the number of the iterations).

Algorithm	R							
	1.0		10.0		100.0		1000.0	
	time	iter	time	iter	time	iter	time	iter
GMRES(50)	523	550	528	550	...	...	...	...
GMRES(70)	620	490	622	490	...	...	...	...
GMRES(50)+ILU	717	400	700	400	...	...	...	...
GMRES(70)+ILU	643	280	642	280	...	...	...	...
MORGAN(48,2)	467	448	471	448	633	598	...	...
MORGAN(46,4)	541	496	542	496	705	646	...	...
MORGAN(68,2)	587	418	582	418	716	488	...	...
MORGAN(66,4)	599	416	707	486	...	...	...	...
DEFLATED-GMRES(50,2)	568	550	577	550	...	...	...	...
DEFLATED-GMRES(50,4)	428	400	489	450	820	750	...	...
DEFLATED-GMRES(70,2)	662	490	669	490	1502	1120	...	...
DEFLATED-GMRES(70,4)	583	420	585	420	488	350	...	...
DEFLATION(50,2)	764	500	677	450	...	...	...	...
DEFLATION(50,4)	709	450	707	450	...	...	...	...
DEFLATION(70,2)	848	420	992	490	3872	1820	...	...
DEFLATION(70,4)	844	420	1003	490	...	...	...	...



(a) Time vs. Residual norm

(b) Iterations vs. Residual norm

Figure 6: The convergence history of the residual norm in Example 3 ($R = 100.0$), A: gmres(50), B: morgan(48,2), C: deflated-gmres(50,4), D: deflation(50,4).

in most cases. Here we comment that it seems likely that DEFLATED-GMRES(50,4) is better reduction of residual norm than the other algorithms.

There still remain several problems to be studied. For example, how to choose m adaptively during the process of iterations, and how to choose the total number of eigenvalues are deflated, and how many eigenvalues that are deflated at each restart, when A is general nonsymmetric coefficient matrix. For determining m automatically when to restart, an approach of the adaptive procedure of GMRES and deflated GMRES algorithms will be given in Tsuno *et al.* [15] and Moriya *et al.* [16], respectively.

References

[1] J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University Press, 1965.

[2] Y. Saad and M.K. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

[3] W. Schönauer. *Scientific Computing on Vector Computers*. North Holland, 1987.

[4] R. Morgan. Computing interior eigenvalues of large matrices. *Linear Algebra and its Appl.*, 154,156:289–309,1991.

[5] P. Bastian and G. Horton. Parallelization of robust multigrid methods: ILU factorization and frequency decomposition Method. *SIAM J. Sci. Stat. Comput.*, 12:1457–1470, 1991.

[6] A.M. Bruaset. *A Survey of Preconditioned Iterative Methods*. Pitman Res. Note in Math., No. 32, Longman, U. K., 1995.

[7] R. Morgan. A restarted GMRES method augmented with eigenvectors. *SIAM J. Matrix Anal. App.*, 16:1154–1171, 1995.

[8] Message Passing Interface Forum, MPI: A message-passing standard, May (1995). http://www.mcs.anl.gov/mpi/

[9] G.L.G. Sleijpen and H.A. Van der Vorst. An overview of approaches for the stable computation of hybrid BiCG methods. *Appl. Numer. Math.*, 19:235–254, 1995.

[10] J. Erhel, K. Burrage, and B. Pohl. Restarted GMRES preconditioned by deflation. *Journal of Comput. and Appl. Math.*, 69:303–318, 1996.

[11] T. Nodera and N. Tsuno. The parallelization of the incomplete LU factorization on AP1000. Springer *Lect. Notes in Comp. Sci.*, 1470:788–792, 1998.

[12] K. Burrage and J. Erhel. On the performance of various adaptive preconditioned GMRES strategies. *Numer. Linear Algebra with Appl.*, 5:101–121, 1998.

[13] K. Moriya and T. Nodera. Parallelization of ILU decomposition for elliptic boundary value problem of the PDE on AP3000. Springer, *Lect. Notes on Comp. Sci.*, 1615:344–353, 1999.

[14] Fujitsu Lab.: Parallel server AP3000: http://www.fujitsu.co.jp/hypertext/Products/Info_process/hpc/ap3000-e. C1000 C999 C1006 C1003 C1007 C1007 C997, C1000, C1003 C1001 C1011 C997, C1000, C1000 C1008 C997, C1000, C1000, C1000 C1008 C1001

[15] N. Tsuno and T. Nodera. The speedup of the GMRES(m) method using the early restarting procedure, (in Japanese). *Transaction of Information Processing Society of Japan*, 40:(4)1760–1773, 1999. C1016

[16] K. Moriya and T. Nodera. The DEFLATED-GMRES(m, k) method with switching the restart frequency dynamically. submitted to *Numer. Linear Algebra with Appl.*, 1999. C1016