Parallel Adaptive Deflated GMRES

Désiré Nuentsa Wakam[1], Jocelyne Erhel[1], and William D. Gropp[2]

[1] INRIA Campus de Beaulieu 35042 Rennes Cedex, {desire.nuentsa_wakam, jocelyne.erhel}@inria.fr
[2] NCSA, University of Illinois Urbana-Champaign, wgropp@illinois.edu

Summary. Many scientific libraries are currently based on the GMRES method as a Krylov subspace iterative method for solving large linear systems. The restarted formulation known as GMRES(m) has been extensively studied and several approaches have been proposed to reduce the negative effects due to the restarting procedure. A common effect in GMRES(m) is a slow convergence rate or a stagnation in the iterative process. In this situation, it is less attractive as a general solver in industrial applications. In this work, we propose an adaptive deflation strategy which retains useful information at time of restart to avoid stagnation in GMRES(m) and improve its convergence rate. We give a parallel implementation in the PETSc package. The provided numerical results show that this approach can be effectively used in the hybrid direct/iterative methods to solve large-scale systems.

1 Introduction

The GMRES method due to [11] is widely used, thanks to its monotonic convergence properties, as a Krylov subspace method for solving large and sparse linear systems. Due to memory and computational requirements, the restarted GMRES (noted as GMRES(m)) is generally used. At the time of restart, information from the previous Krylov subspace is discarded and the orthogonality between successive Krylov subspaces is not preserved. The worst case is when the successive generated Krylov subspaces are very close. As a result, there is no significant reduction in the residual norm and the iterative process may stagnate. Deflation techniques are a class of acceleration strategies that collects useful information at the time of restart mainly to avoid this stagnation and improve the convergence rate. The main idea behind these methods is to remove the smallest eigencomponents from the residual vector as they are known to slow down the convergence of GMRES.

In a practical use of a deflation strategy, it is necessary to define the number of eigenvalues to deflate. As the deflation process induces additional operations to GMRES(m), it is interesting as well to know a priori if the deflation will be beneficial. In this work, we propose an adaptive deflated GMRES(m) which aims at enhancing the convergence of GMRES(m) by adaptively extracting the spectral information

R. Bank et al. (eds.), *Domain Decomposition Methods in Science and Engineering XX*, Lecture Notes in Computational Science and Engineering 91,
DOI 10.1007/978-3-642-35275-1_75, © Springer-Verlag Berlin Heidelberg 2013

needed to speedup the convergence. The adaptive strategy is based on a (near) stagnation test which defines if the deflation process is needed or not and if more accurate spectral information are required. Although we use a stagnation test similar to that in [12], our approach is different since we assume that the restart length m is fixed. This work is motivated by the convergence behavior of GMRES when it is used with a Schwarz preconditioner. As the number of subdomains increases, the eigenvalues are less and less clustered. The restarting may have the disadvantage to discard the smallest eigenvalues before their convergence. The proposed adaptive strategy will thus keep these spectral values in the Krylov subspace until their convergence.

The remaining part of this report is organized as follows: in Sect. 2, we first recall the basis of the deflation technique applied as a preconditioner and we derive the adaptive strategy. In Sect. 3, we discuss on the parallel implementation. Section 4 is focused on numerical experiments to show the benefits of this scheme on a real industrial CFD test case.

2 Adaptive Preconditioner for the Deflated GMRES(m)

We are interested in the solution of the linear system

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

The GMRES method is among the best methods to solve this system when the coefficient matrix A is nonsingular and nonsymmetric. For large linear systems, the restarted version should always be used to reduce the memory and computational requirements. The deflated GMRES has been proposed to reduce the negative effects of the restarting procedure. The general idea behind these methods is to add to the Krylov subspace an approximation of the invariant subspace associated to the smallest eigenvalues. In [7], this is carried out by defining a preconditioner that is equal to the projected matrix onto the approximated invariant subspace and is taken as the identity on the orthogonal subspace. Hence, given $U = [u_1, \ldots, u_r] \in \mathbb{R}^{n \times r}$ the r-dimensional basis of the invariant subspace associated to the eigenvalues to deflate, the preconditioner is defined as

$$M_D^{-1} \equiv I_n + U(|\lambda_n| T_r^{-1} - I_r)U^T, \quad T = U^T B U, \tag{2}$$

where λ_n is the largest eigenvalue in magnitude, I_n and I_r are the identity matrices and B the initial preconditioned matrix. Since M_D^{-1} is nonsingular, the eigenvalues of the resulted matrix $M_D^{-1}B$ or BM_D^{-1} are $\lambda_{r+1}, \ldots, \lambda_n, |\lambda_n|$ with a multiplicity at least r. It is therefore expected to get a faster convergence rate with this preconditioner since the r smallest eigencomponents that slow down the convergence are deflated. This assumes that U is a good approximation of the basis of the selected invariant subspace. For large matrices however, the cost of accurately computing U (as suggested in [7] and later in [4]) may induce a significant overhead. This process should be carried out only when it is necessary, for instance to avoid stagnation.

Algorithm 4 DGMRES(m,k,r): Restarted GMRES with adaptive deflation

input (m, $itmax$, ε, k, smv, bgv, $rmax$);
Set $B \equiv AM^{-1}$, M^{-1} is any external preconditioner
$r_0 = b - Ax_0$; $U = [\,]$; $M_D = I$; $it = 0$; $r = 0$;
while ($\|r_0\| > \varepsilon$)
  Arnoldi process on B to get $BM_D^{-1}V_m = V_{m+1}\bar{H}_m$. See [11]
  $x_m = x_0 + M_D^{-1}M^{-1}V_m y_m$, y_m solution of $min\|\beta e_1 - \bar{H}_m y_m\|_2$;
  $r_m = b - Ax_m$, $it \leftarrow it + m$;
  If ($\|r_m\| > \varepsilon$ and $it < itmax$) **then**
    $Iter = m * log(\frac{\varepsilon}{\|r_m\|})/log(\frac{\|r_m\|}{\|r_0\|})$;
    If(($Iter > smv * (itmax - it)$ **and** $r < rmax$) **then**
      Compute k Schur vectors of B noted X. See [7]
      Orthogonalize X against U
      Compute $T = \begin{bmatrix} U & X \end{bmatrix}^T B \begin{bmatrix} U & X \end{bmatrix} \equiv \begin{pmatrix} U^TBU & U^TBX \\ X^TBU & X^TBX \end{pmatrix}$
      Increase U by X; $r \leftarrow r + k$;
      If($Iter > bgv * (itmax - it)$) **then**
        Improve U as indicated in [4, Sect. 3]
      EndIf
      Factorize T Set $M_D^{-1} \equiv I_n + U(|\lambda_n|T^{-1} - I_r)U^T$
    End If
  End If
  $x_0 = x_m$, $r_0 = r_m$
end while

We thus propose here an adaptive strategy that detects a near-stagnation in the iterative process or a slow reduction in the residual norm. This approach is based upon the work by Sosonkina et al. [12] in which the Krylov subspace is adaptively increased along the cycles of GMRES(m); Here, we find it natural to enrich the subspace with the eigencomponents that slow down the convergence. The main steps are given in Algorithm 4. First, m steps of the Arnoldi process are performed to compute the orthonormal basis V_m. It also creates an upper Hessenberg matrix $H_m = V_m^T BV_m$ which is the restriction of B onto the $m-$dimensional Krylov subspace. Then, a least-squares problem is solved to minimize the residual norm in the Krylov subspace. At the time of restart, if the desired residual norm is not achieved, a stagnation test is computed to determine if a deflation process could be beneficial to accelerate the convergence. This test considers the convergence rate over the previous restart cycles and evaluates the number of iterations ($Iter$) needed to achieve the desired accuracy. If $Iter$ is greater than the remaining number of steps (bounded by a small multiple smv of the number of iterations allowed), then data are computed to update the preconditioner associated to the deflation process. This test is therefore used to reduce the iteration counts in GMRES(m). To detect a near-stagnation, we use another test which considers a large multiple bgv of the remaining number of steps. In this case, a harmonic projection is carried out to accurately compute the eigenvalues and continuously update the previous estimation of U.

3 Implementation Notes

We now give some details about the implementation of Algorithm 4 on distributed-memory computers. The programming model is SPMD (Single Program Multiple Data) and communications are done using the message-passing interface (MPI). The adjacency graph of the input sparse matrix is first built. PARMETIS is then used to partition the vertices of the graph into D disjoint vertices. From this partitioning, the matrix is distributed such that each processor holds a contiguous chunk of rows corresponding to the vertices it owns. The right hand side and all other vectors (Krylov basis, invariant basis) are distributed accordingly. Note that the goal of this data distribution is to get a good load balance and to minimize communication during matrix-vector multiply and preconditioning steps. When the additive Schwarz preconditioner is used, an overlapping partitioning can be defined by taking recursively adjacent vertices from the initial disjoint partitions.

The main parallel operations in Algorithm 4 so far are the matrix-vector multiply, scalar products, and the application of M^{-1} and M_D^{-1}. M^{-1} can be any parallel preconditioner as long as it implements the basic operation $v_j \leftarrow M^{-1}v_i$. In our tests, the restricted additive Schwarz has been used as defined in [5]. It is then necessary in the setup phase to factorize in each process the block matrices A_p corresponding to the restriction of A onto the defined subdomains. M_D^{-1} is applied to a distributed vector v_j in a straightforward manner given the data distribution described above. This implies r all-to-all communications to compute the projection onto the invariant subspace. There is no additional communication for the other terms since the $r \times r$ dense matrix T is owned by each process.

We provide an implementation of this method using the PETSc package (see [3]). The original implementation of the built-in *KSP GMRES* has been modified to provide the data needed for the deflation and to apply the resulting preconditioner to generate the Krylov basis. Although the current presentation does not discuss the choice of side of preconditioning, the implementation does define left and right preconditioning. Note that the current adaptive preconditioning can be associated with any other preconditioner available in the package or defined by the end user since we provide generic interface similar to the other Krylov subspace methods in the package. The resulted KSP module (named as DGMRES) is available in PETSc release 3.2.

4 Numerical Experiments

This section presents some numerical results to prove the efficiency of the proposed approaches. The test problem arises from design optimization in computational fluid dynamics. The physical model is a 3D flow simulation in a jet engine compressor rotor. The physical equations are the Reynolds-Averaged Navier-Stokes for compressible flows, discretized using the finite volume method as presented by Aubert et al. [2]. The matrices have been extracted from the software Turb'Opty^{TM} designed by the FLUOREM company. They are also available in the University of

Florida sparse matrix collection (see [6]) under the name *RM07R* in the FLUOREM group. The matrix is nonsymmetric and indefinite with a size 272,635 and 37,355,908 nonzero entries. Other test cases can be found in [8].

With this test case so far, previous studies have shown the limits of some existing solvers in terms of memory usage and numerical accuracy (see [9]). Pacull et al. [10] have proved as well the instability of the ILU factorization to approximate the solution of linear subsystems. In our hybrid approach, we therefore rely on a direct solver within each subdomain, such as MUMPS [1].

4.1 Benefits of the Deflated Restarting

We now give the main benefits of using the deflated GMRES with the additive Schwarz method (ASM). It is known that one level ASM is a weak preconditioner when the number of subdomains D gets large. The size of the Krylov subspace m could then be increased to enhance the robustness of the global method. However, choosing a good size m of the Krylov subspace is a trial-and-error process. With the adaptive deflation, we show experimentally that the method is robust for various values of m and D. Moreover, using a large number of subdomains reduces the memory required to handle the submatrices by the direct solver. Hence it is expected that the time to factorize these matrices and the memory required will get smaller as D increases. This is reported in the last column of Table 1. We also report the number of matrix-vector multiplies and the global CPU time with respect to the number of subdomains D. We then compare the restarted version (GMRES(m)) with the deflated version (DGMRES(m,k)), where $m = 48$ and 64. A dash in a field means that the relative residual norm of 10^{-8} is not reached after 2500 iterations. It can be observed that DGMRES provides reliable and faster convergence than the classical restarted GMRES. It also gives a faster method since significantly fewer iterations are needed. Furthermore, the method reveals a substantial acceleration as the number of processors increases. Note that without the deflation, this acceleration will not be obtained since the number of matrix-vector multiplies increases hugely with the subdomains. For instance, this behavior can be seen with GMRES(64) when using $D = 16$ and $D = 32$.

Table 1. RM07R : Benefits of using DGMRES with an additive Schwarz preconditioner and an overlap of 1. The deflation process reduces the total number of iterations and helps to use a large number of subdomains and thus a large number of processors. Here, the number of processors is indeed equal to the number of subdomains.

D	GMRES(48)		DGMRES(47,1)			GMRES(64)		DGMRES(63,1)		
	Matvecs	Time	Matvecs	Time	r	Matvecs	Time	Matvecs	Time	r
16	551	230	212	173.4	3	355	193.8	208	168.9	2
32	-	-	533	109.2	4	2217	244.6	455	94.6	7
64	-	-	410	56.8	4	-	-	453	50.8	7
128	-	-	791	51.5	15	-	-	638	44.3	8

4.2 Adaptive DGMRES and Full GMRES

From the robustness standpoint, the full GMRES approach is more reliable than the restarted version even with the deflation process. However as the size of the basis grows, it should be more sensitive to round-off errors. To illustrate this behavior, we consider two formulations of the Arnoldi process, namely the classical Gram-Schmidt (*CGS*) and the modified Gram-Schmidt (*MGS*) algorithms. The former is sometimes preferred since it provides good kernel operations in parallel environments. In the PETSc package, for instance, it is used by default in the GMRES implementation as the orthogonalization method with a possible iterative refinement strategy. In Fig. 1, the residual history is displayed with respect to the number of matrix-vector products. The method stops when the relative residual norm is 10^{-10}. It can then be noticed that with *CGS*, stagnation occurs in the full GMRES (in solid line) due to severe cancellation in the algorithm and consequently a loss of orthogonality. This does not happen when the basis is small since the round-off errors are not propagated very far and DGMRES (dash-dotted line) converges at the desired accuracy even with *CGS*. Note that although good accuracy is finally achieved in

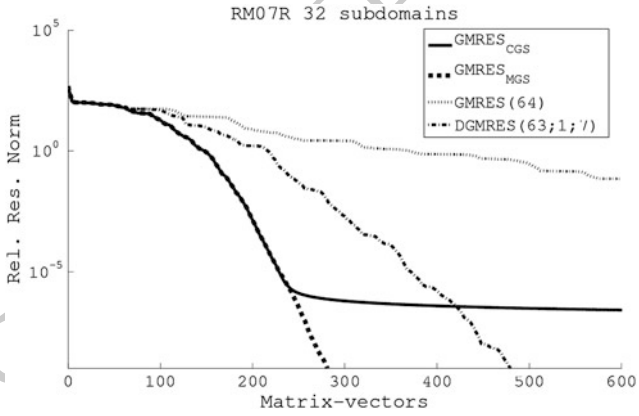

Fig. 1. Convergence of full GMRES, GMRES(m) and DGMRES(m, k, r) with classical Gram-Schmidt(*CGS*) and modified Gram-Schmidt (*MGS*) orthogonalization scheme. k is the number of eigenvalues to extract at each detected stagnation and r is the total number of eigenvalues extracted at the convergence. Thirty two subdomains are used in the additive Schwarz method with a 1-overlap

full GMRES with *MGS* (dashed line), it will require much more memory to store all the vectors of the growing Krylov basis (265 vectors in this case). In DGMRES, the Krylov basis is stored just for one cycle. Only the invariant basis U is stored over the restart cycles together with vectors $M^{-1}AU$ to reduce the matrix-vector counts. Thus in this example, only $63 + 7 \times 2 = 77$ vectors are stored. Note also that this number can be further reduced by using a smaller Krylov basis since convergence is still good, as shown in Table 1.

5 Conclusion

We have designed an adaptive deflation strategy that can be used for preconditioned GMRES. We show in this paper that the proposed algorithm can be used to improve the robustness and reduce both CPU time and memory required by hybrid solvers based on a one level additive Schwarz method. We have implemented this method in the new module DGMRES of the PETSc library.

Acknowledgments This work is funded by the French National Agency of Research under the contract ANR-TLOG07-011-03 LIBRAERO. The work of the first author was done while visiting the NCSA at Urbana-Champaign in the context of the Joint laboratory INRIA-University of Illinois. Experiments in this paper have been carried out using the *parapide* cluster in the GRID'5000 experimental testbed (see https://www.grid5000.fr). We thank the referees for providing many instructive comments.

Bibliography

[1] Patrick R. Amestoy, Iain S. Duff, Jean-Yves L'Excellent, and Jacko Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM J. Matrix Anal. Appl.*, 23(1):15–41 (electronic), 2001. ISSN 0895–4798.

[2] S. Aubert, J. Tournier, M. Rochette, J. Blanche, M. N'Diaye, S. Melen, M. Till, and P. Ferrand. Optimization of a gas mixer using a new parametric flow solver. In *Proceedings of the ECCOMAS Computational Fluid Dynamics Conference, Swansea, UK*, 2001.

[3] Satish Balay, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.0.0, Argonne National Laboratory, 2008.

[4] Kevin Burrage and Jocelyne Erhel. On the performance of various adaptive preconditioned GMRES strategies. *Numer. Linear Algebra Appl.*, 5(2):101–121, 1998. ISSN 1070–5325.

[5] Xiao-Chuan Cai and Marcus Sarkis. A restricted additive Schwarz preconditioner for general sparse linear systems. *SIAM J. Sci. Comput.*, 21(2):792–797, 1999.

[6] Timothy A. Davis and Yifan Hu. The University of Florida Sparse Matrix Collection. *ACM Transactions on Mathematical Software*, 38(1), 2011.

[7] Jocelyne Erhel, Kevin Burrage, and Bert Pohl. Restarted GMRES preconditioned by deflation. *J. Comput. Appl. Math.*, 69(2):303–318, 1996. ISSN 0377–0427.

[8] Désiré Nuentsa Wakam and François Pacull. Memory efficient and robust hybrid algebraic solvers for large CFD linear systems. *Computer and Fluids*, submitted, 2011. special issue of ParCFD2011.

[9] Désiré Nuentsa Wakam, Jocelyne Erhel, Edouard Canot, and Guy-Antoine Atenekeng Kahou. A comparative study of some distributed linear solvers

on systems arising from fluid dynamics simulations. In *Parallel Computing: From Multicores and GPU's to Petascale*, volume 19 of *Advances in Parallel Computing*, pages 51–58. IOS Press, 2010.

[10] F. Pacull, S. Aubert, and M. Buisson. Study of ILU factorization for schwarz preconditioners with application to computational fluid dynamics. In *Proceedings of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*. Civil-Comp Press, Stirlingshire, UK, 2011. doi: doi:10.4203/ccp.95.39.

[11] Youcef Saad and Martin H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comp.*, 7(3):856–869, 1986.

[12] Maria Sosonkina, Layne T. Watson, Rakesh K. Kapania, and Homer F. Walker. A new adaptive GMRES algorithm for achieving high accuracy. *Numer. Linear Algebra Appl.*, 5(4):275–297, 1998.