Parallel Arnoldi Method for the Construction of a Krylov Subspace Basis: an Application in Magnetohydrodynamics

J.G.L. Booten [1], P.M. Meijer [2], H.J.J. te Riele [1] and H.A. van der Vorst [3]

[1] Center for Mathematics and Computer Science,
P.O. Box 94079, 1090 GB Amsterdam, the Netherlands
[2] FOM Institute for Plasmaphysics,
P.O. Box 1207, 3430 BE Nieuwegein, the Netherlands
[3] Mathematical Institute, University of Utrecht,
P.O. Box 80.010, 3508 TA Utrecht, the Netherlands

Abstract. In a recent article [6] a new method was proposed for computing internal eigenvalues of symmetric matrices. In the present paper we extend these ideas to non-hermitian eigenvalue problems and apply them to a practical example from the field of magnetohydrodynamics (MHD). The method is very suitable for an efficient parallel implementation. We give some results for the time-consuming kernels of the underlying orthogonalization process, the Arnoldi method, obtained for an MHD problem on a distributed memory multiprocessor.

1 Introduction

The numerical solution of large-scale eigenvalue problems has many applications in both science and engineering. For a recent overview of some application areas, see [9]. In the field of linear magnetohydrodynamics (MHD), which concerns the study of the interaction of an ionized gas, a so-called plasma, with a magnetic field, the finite element discretization of the partial differential MHD equations gives rise to a generalized non-hermitian eigenvalue problem (see e.g. [1]). Over the last years the demand for solving increasingly larger eigenvalue problems in MHD is growing. Meanwhile considerable progress has been made in the techniques to implement numerical algorithms for large eigenvalue problems on parallel computers. For a recent review in this area, see [2].

The most commonly used algorithms to solve non-hermitian eigenvalue problems are referred to as Arnoldi algorithms. The basic idea is to reduce a general non-hermitian large matrix A to upper Hessenberg form H, after which the eigenvalues are determined for the much smaller projected matrix H. The eigenvalues of this matrix tend to converge to the dominant eigenvalues of the original matrix A. For obvious reasons the method is usually applied in an iterative way. The basis upon which the upper Hessenberg matrix is built is the so-called Krylov subspace basis. The eigenvalues of H are the Ritz values of A with respect to the Krylov subspace. The orthogonal projection method of the original matrix onto the subspace is known as the Arnoldi method. The Arnoldi method is the most time-consuming part in the Arnoldi-based algorithms for solving non-hermitian eigenproblems. It should be noted that the Arnoldi method is also applied as the orthogonalization scheme in the well-known GMRES algorithm [8] for solving general nonsymmetric large linear systems of equations.

In many applications one is not particularly interested in the dominant part of the eigenvalue spectrum, but only in an internal branch. A commonly followed procedure [9] is to shift the original matrix by a value close to the eigenvalues in the interesting part of the spectrum, invert the shifted matrix in order to make the desired eigenspectrum dominant and perform the standard Arnoldi algorithm for the inverted matrix. In MHD the interesting part of the spectrum is the so-called Alfvén branch. In [5] this part of the spectrum was computed applying the implicitly restarted Arnoldi method [10] with a complex shift and invert strategy. The most expensive part in the shift-and-invert Arnoldi method [9] is inverting the shifted matrix: first one has to perform an LU-decomposition of the shifted matrix, followed by solving lower and upper triangular systems of equations. Furthermore, the latter operations make this method unfavourable for an efficient implementation on a distributed memory computer [2].

In a recent paper [6] Paige, Parlett and Van der Vorst demonstrated that the zeros of the iteration polynomial associated with the minimum residual approximation to the solution of a symmetric indefinite system of equations converge slowly, but monotonically to eigenvalues of the coefficient matrix close to the origin. The computation of these eigenvalue approximations, introduced by the authors of [6] as the *harmonic* Ritz values, offers new possibilities for the determination of internal eigenvalues. In the present paper we have investigated whether these ideas could be extended to the non-hermitian case. In the next section we briefly discuss the method and illustrate it with an example from MHD. In the final section we describe the implementation of the time-consuming kernels of the Arnoldi method on a distributed memory machine for the MHD-type matrices. Also timing results are presented.

2 Arnoldi Method for Internal Eigenvalues

If we perform i steps of the standard Arnoldi process [3, 9] with a general $n \times n$ matrix A and a unit two-norm starting vector v_1, we obtain an orthonormal set of basis vectors $v_1, v_2, \ldots, v_i, v_{i+1}$. The first i vectors span the Krylov subspace $K_i(A; v_1)$. Define the $n \times j$ matrix $V_j = [v_1, \ldots, v_j]$. Then from the Arnoldi process the following relation can be derived [9]:

$$AV_i = V_{i+1}\hat{H}_i, \tag{1}$$

with $\hat{H}_i$ an $(i+1) \times i$ upper Hessenberg matrix, whose upper $i \times i$ part we denote as H_i. Using the orthonormality of the basis, we obtain:

$$V_i^H A^H A V_i = \hat{H}_i^H V_{i+1}^H V_{i+1} \hat{H}_i = \hat{H}_i^H \hat{H}_i \equiv M_i^2. \tag{2}$$

From this equation we see that the columns of $AV_iM_i^{-1}$ form an orthonormal basis for $AK_i(A; v_1)$, since we have

$$M_i^{-1} V_i^H A^H A V_i M_i^{-1} = I_i, \tag{3}$$

with I_i the $i \times i$ identity matrix. Using this basis and the relation

$$V_i^H A V_i = V_i^H V_{i+1} \hat{H}_i = H_i \tag{4}$$

it is straightforward to obtain the orthogonal projection of A^{-1} onto $AK_i(A; v_1)$ (see also [6]):

$$M_i^{-1}(AV_i)^H A^{-1} AV_i M_i^{-1} = M_i^{-1} H_i^H M_i^{-1}. \tag{5}$$

Solving the eigenvalue problem for the projected matrix $M_i^{-1} H_i^H M_i^{-1}$, or, which is similar, for the matrix $M_i^{-2} H_i^H$, gives the Ritz values of A^{-1} with respect to $AK_i(A; v_1)$. These Ritz values should be good approximations for the extreme eigenvalues of A^{-1}. Their reciprocals are the harmonic Ritz values and should therefore be good approximations for the eigenvalues of A closest to the origin.

Note that the procedure described above enables us to compute approximations for eigenvalues close to any point in the spectrum of A, simply by performing the method with a shifted matrix $A - \sigma I$. Also note that we avoid explicit inverting of a matrix, which makes the method more appropriate for an efficient parallel implementation than the shift-and-invert method.

We have tested the method for a small problem from MHD (order 416) taken from [5]. We used subspace iteration, restarting at each step with a linear combination of four eigenvectors associated with the four eigenvalues closest to the shift. We were able to reproduce the spectrum obtained in [5]. However, the Krylov dimension had to be set at 125 and convergence turned out to be very slow as displayed in fig. 1. A similar behavior was observed in the symmetric case [6], apart from the fact that the convergence is far from monotonic in the present example.

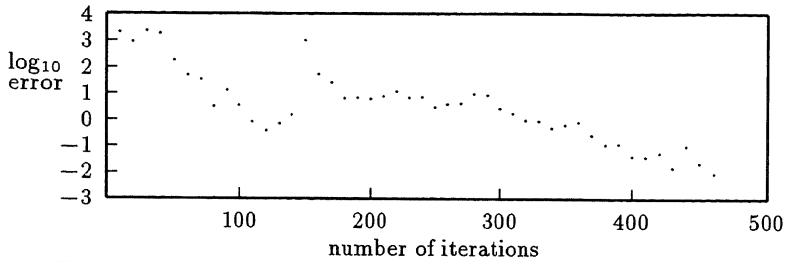

Fig. 1. Typical convergence behavior for a harmonic Ritz value, computed with the present Arnoldi method.

3 Parallel Implementation

We have implemented the basic operations of the Arnoldi method on a parallel distributed memory system. Results are given for the application to the MHD-type matrices, which are of block-tridiagonal shape (with rather dense blocks). The numerical experiments were carried out on a Parsytec GC_{el} (located at SARA, the Academic Computer Center Amsterdam), a distributed memory machine consisting of 512 nodes, each having 4 Mbytes local memory. The processors were configured in a 2d-grid topology.

The blocks in the MHD matrices are of the order $16N_F$, with N_F the number of Fourier modes, see also [7]. Let N_G denote the number of grid points in the finite element discretization of the MHD equations, which is the number of rows

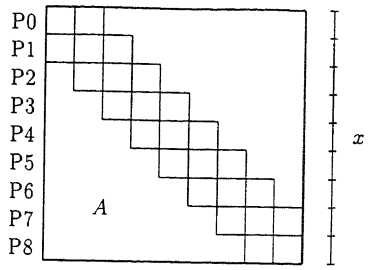

Fig. 2. Data layout on the processors.

of blocks in the matrices. Then the order of the matrices is given by $N = 16N_F N_G$. We store an equal number of rows of blocks on each processor and for vectors x we assign the corresponding segment to the processors. We choose the problem size such that the number of processors divides N_G exactly. An example of the data layout is given in fig. 2 with $N_G = 9$ and a 3×3 processor grid.

The time-consuming kernels in the Arnoldi method consist of matrix-vector products, inner products and vector updates. The parallel implementation of the matrix-vector products for the MHD-type of matrices does not pose any serious problems, since a strong data locality can be preserved and each processor only has to perform nearest neighbor communication. Therefore, a pipeline has to be embedded in the 2d-grid topology. Only four communication steps have to be performed to evaluate the matrix-vector product, independent of the number of processors.

In the evaluation of the vector updates no communication is involved and thus the implementation of this kernel is optimal.

However, for the inner products the situation is entirely different. Inner products are needed on all processors. Each processor has to evaluate its local part, after which the local inner products have to be accumulated on one processor for instance and the global inner product has to be broadcasted again to all processors. Therefore, when going to larger processor grids, the inner products can be a serious bottleneck and a considerable degradation of the performance of the algorithm can occur, as we shall illustrate below. The accumulation of local inner products on a $p \times p$ grid requires p communication steps when p is even and $p + 1$ steps when p is odd. This is shown in fig. 3 for a 5×5 grid. Broadcasting of the global inner product requires the same number of communication steps. This can be illustrated by reversing the direction of the arrows in fig. 3.

In table 1 we present the execution times for the matrix-vector product (MV) and the inner product (IP) obtained for three fairly large MHD problems ($N = 20480$, $N = 40960$ and $N = 81920$) on different processor grids. These problems could not be dealt with on smaller grids than the ones shown, because this would exceed the local processor memory. Also the execution rates (in Mflops) are given. The number of floating point operations for the matrix-vector product equals $4 \times 2 \times (16N_F)^2(3N_G - 2)$ and for the inner product $4 \times 2 \times 16N_F N_G$. The factor 4 is due to the (double) complex arithmetic, which we use.

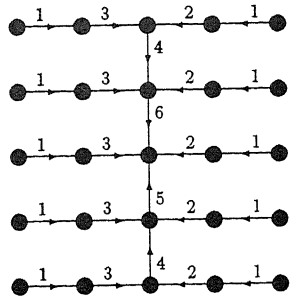

Fig. 3. Accumulation of local inner products on a 2d-grid topology.

Table 1. Timing results for the matrix-vector product (MV) and the inner product (IP) for different MHD problems on a $p_1 \times p_2$ processor grid.

			MV		IP	
N_F	N_G	$p_1 \times p_2$	time[ms]	Mflops	time[ms]	Mflops
5	256	8×4	4111	9.5	7.46	22.0
		8×8	2061	19.0	5.39	30.4
		16×8	1036	37.9	5.42	30.2
		16×16	522	75.1	6.15	26.6
5	512	8×8	4111	19.1	8.16	40.1
		16×8	2061	38.1	6.81	48.1
		16×16	1036	75.8	6.84	47.9
		32×16	519	151.3	9.00	36.4
10	512	16×16	4077	77.1	8.23	79.6
		32×16	2048	153.4	9.69	67.6

From the table we clearly see that the execution time for the MV product scales almost perfectly linear with N_G and N_F^2 if we keep the number of processors fixed. If we compare the performances of the MV product for a fixed problem size ($N_F = 5, N_G = 512$) we observe an almost ideal relative speed-up of 7.92 in going from 64 to 512 processors. For the largest problem size we obtained a maximum performance of 153.4 Mflops for the MV product, which corresponds to 38% of the peak performance of the Parsytec.

For the inner products the performance is much worse as can be seen from the last two columns in the table. If we increase the number of processors for a fixed problem, the execution time decreases initially. However, at a certain stage the communication starts to dominate the computation in the evaluation of the inner product and the execution time increases again. Note that for small processor grids the performance of the IP is better than for the MV product, because only the BLAS level 1 library is available on the Parsytec at the moment.

As we illustrated in section two, the dimension of the Krylov subspace in the Arnoldi algorithm for computing internal eigenvalues has to be rather high. This means that for the construction of the last basis vectors a large number of inner products (and vector updates) have to be evaluated corresponding to only one matrix-vector product. Consider for instance the largest MHD problem in table 1 and the construction of a Krylov subspace basis of the order 100. From the timings of the matrix-vector product and the inner product on the largest processor grid (last row of the table) we observe that the evaluation of the inner products for constructing the last vector of the basis amounts to roughly half the time of the matrix-vector product. This would certainly lead to a serious degradation of the algorithm. A way to get around this difficulty is to create a Krylov subspace basis first and orthogonalize the basis afterwards, see e.g. [2]. A similar procedure has been applied in the s-step Arnoldi algorithm described by Kim and Chronopoulos [4]. The number of global communications due to the inner products is brought down from $O(i^2)$ to $O(i)$, i being the Krylov dimension.

References

1. J. Cullum, W. Kerner, R. Willoughby: A generalized nonsymmetric Lanczos procedure. Computer Physics Communications 53, 19–48 (1989)
2. J.W. Demmel, M.T. Heath, H.A. van der Vorst: Parallel numerical linear algebra. Acta Numerica 2, 111–197 (1993)
3. G.H. Golub, C.F. van Loan: Matrix Computations, 2nd ed. Baltimore: The Johns Hopkins University Press 1989
4. S.K. Kim, A.T. Chronopoulos: An efficient parallel algorithm for extreme eigenvalues of sparse nonsymmetric matrices. The International Journal of Supercomputer Applications 6, 98–111 (1992)
5. M.N. Kooper, H.A. van der Vorst, S. Poedts, J.P. Goedbloed: Application of the implicitly updated Arnoldi method with a complex shift and invert strategy in MHD. FOM Preprint PP 93/061 (1993)
6. C.C. Paige, B.N. Parlett, H.A. van der Vorst: Approximate solutions and eigenvalue bounds from Krylov subspaces. To appear. Linear Algebra and its Applications (1994)
7. S. Poedts, P.M. Meijer, J.P. Goedbloed, H.A. van der Vorst, A. Jacoby: Parallel magnetohydrodynamics on the CM-5. This conference.
8. Y. Saad, M.H. Schultz: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM Journal on Scientific and Statistical Computing 7, 856–869 (1986)
9. Y. Saad: Numerical methods for large eigenvalue problems. Manchester: Manchester University Press 1992
10. D.C. Sorensen: Implicit application of polynomial filters in a k-step Arnoldi method. SIAM Journal on Matrix Analysis and Applications 13, 357–385 (1992)