Hindawi Publishing Corporation
International Journal of Mathematics and Mathematical Sciences
Volume 2009, Article ID 179481, 6 pages
doi:10.1155/2009/179481

Research Article

On the Relation between the AINV and the FAPINV Algorithms

Davod Khojasteh Salkuyeh and Hadi Roohani

Department of Mathematics, University of Mohaghegh Ardabili, P.O. Box 179, Ardabil, Iran

Correspondence should be addressed to Davod Khojasteh Salkuyeh, khojaste@uma.ac.ir

Received 26 July 2009; Accepted 3 November 2009

Recommended by Victor Nistor

The approximate inverse (AINV) and the factored approximate inverse (FAPINV) are two known algorithms in the field of preconditioning of linear systems of equations. Both of these algorithms compute a sparse approximate inverse of matrix A in the factored form and are based on computing two sets of vectors which are A-biconjugate. The AINV algorithm computes the inverse factors W and Z of a matrix independently of each other, as opposed to the AINV algorithm, where the computations of the inverse factors are done independently. In this paper, we show that, without any dropping, removing the dependence of the computations of the inverse factors in the FAPINV algorithm results in the AINV algorithm.

1. Introduction

Consider the linear system of equations

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

where the coefficient matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular, large, sparse, and $x, b \in \mathbb{R}^n$. Such linear systems are often solved by Krylov subspace methods such as the GMRES (see Saad and Schultz [1], Saad [2]) and the BiCGSTAB (see van der Vorst [3], Saad [2]) methods in conjunction with a suitable preconditioner. A preconditioner is a matrix M such that Mu can be easily computed for a given vector u and system $MAx = Mb$ is easier to solve than (1.1). Usually, to this end one intends to find M such that matrix $M \approx A^{-1}$ ($MA \approx I_n$), where I_n is the identity matrix. There are various methods to compute such an appropriate matrix (see Benzi [4], Benzi and Tuma [5], Saad [2]). The factored approximate inverse (FAPINV) (Lee and Zhang [6, 7], Luo [8–10], Zhang [11, 12]) and the approximate inverse (AINV) (see Benzi and Tuma [13, 14]) are among the algorithms for computing an approximate inverse of A in the factored form. In fact both of these methods compute lower unitriangular matrices W and Z^T and a diagonal matrix $D = \mathrm{diag}(d_1, d_2, \ldots, d_n)$ such that $WAZ \approx D$. In this case, the matrix $M = ZD^{-1}W \approx A$ may be used as a preconditioner for (1.1). It is well-known that the AINV algorithm is free from breakdown for the class of H-matrices [13].

The main idea of the FAPINV algorithm was first introduced by Luo (see Luo [8–10]). Then the algorithm was more investigated by Zhang in [12]. Since in this procedure the factorization is performed in backward direction, we call it BFAPINV (for backward FAPINV) algorithm. In [11], Zhang proposed an alternative procedure to compute the factorization in the forward direction, which we call it FFAPINV (for forward FAPINV) algorithm. In [7], Lee and Zhang showed that the BFAPINV algorithm is free from breakdown for M-matrices. It can be easily seen that the FFAPINV algorithm is free from breakdown for M-matrices, as well. In the left-looking AINV algorithm (see Benzi and Tuma [13, 14]), the inverse factors are computed quite independently. In contrast, in the FFAPINV algorithm, the inverse factors W and Z are not computed completely independently of each other. In this paper, from the FFAPINV algorithm without any dropping, we obtain a procedure which bypasses this dependence. Then we show that this procedure is equivalent to the left-looking AINV algorithm. In the same way one can see that the right-looking AINV algorithm (see Benzi and Tuma [13]) can be obtained from BFAPINV algorithm.

In Section 2, we give a brief description of the FFAPINV algorithm. The main results are given in Section 3. Section 4 is devoted to some concluding remarks.

2. A Review of the FFAPINV Algorithm

Let W and Z be the inverse factors of $A = (a_{ij})$, that is,

$$WAZ = D, \tag{2.1}$$

where $W = (w_1^T, w_2^T, \ldots, w_n^T)^T$, $Z = (z_1, z_2, \ldots, z_n)$, and $D = \mathrm{diag}(d_1, d_2, \ldots, d_n)$, in which w_i's and z_i's are the rows and columns of W and Z, respectively. Using (2.1) we obtain

$$w_i A z_j = \begin{cases} d_i, & i = j, \\ 0, & i \neq j. \end{cases} \tag{2.2}$$

From the structure of the matrices W and Z, we have

$$z_1 = e_1, \quad z_j = e_j - \sum_{i=1}^{j-1} \alpha_i z_i, \quad j = 2, \ldots, n, \tag{2.3}$$

$$w_1 = e_1^T, \quad w_j = e_j^T - \sum_{i=1}^{j-1} \beta_i w_i, \quad j = 2, \ldots, n, \tag{2.4}$$

for some α_i's and β_i's, where e_j is the jth column of the identity matrix.

First of all, we see that

$$d_1 = z_1^T A z_1 = e_1^T A e_1 = a_{11}. \tag{2.5}$$

Now let $2 \leq j \leq n$ be fixed. Then from (2.2) and (2.3) and for $k = 1, \ldots, j-1$, we have

$$\begin{aligned} 0 &= w_k A z_j \\ &= w_k A e_j - \sum_{i=1}^{j-1} \alpha_i w_k A z_i \\ &= w_k A_{*j} - \alpha_k w_k A z_k \\ &= w_k A_{*j} - \alpha_k d_k, \end{aligned} \tag{2.6}$$

where A_{*j} is the jth column of A. Therefore

$$\alpha_i = \frac{1}{d_i} w_i A_{*j}, \quad i = 1, \ldots, j-1. \tag{2.7}$$

In the same manner

$$\beta_i = \frac{1}{d_i} A_{j*} z_i, \quad i = 1, \ldots, j-1, \tag{2.8}$$

where A_{j*} is the jth row of A. Putting these results together gives the Algorithm 1 for computing the inverse factors of A.

Some observation can be posed here. It can be easily seen that (see, e.g., Salkuyeh [15])

$$d_j = w_j A z_j = z_j^T A z_j = w_j A z_j = A_{j*} z_j = w_j A_{*j}. \tag{2.9}$$

In this algorithm, the computations for the inverse factors Z and W are tightly coupled. This algorithm needs the columns of the strictly upper triangular part of A for computing Z and the strictly lower triangular part of A for computing W. A sparse approximate inverse of A in the factored form is computed by inserting some dropping strategies in Algorithm 1.

3. Main Results

At the beginning of this section we mention that all of the results presented in this section are valid only when we do not use any dropping. As we mentioned in the previous section the computations for the inverse factors Z and W are tightly coupled. In this section, we extract a procedure from Algorithm 1 such that the computations for the inverse factors are done independently. We also show that the resulting algorithm is equivalent to the left-looking AINV algorithm.

From $WAZ = D$ we have $Z^T A Z = Z^T W^{-1} D$. Obviously, the right-hand side of the latter equation is a lower triangular matrix and $\operatorname{diag}(Z^T A Z) = \operatorname{diag}(Z^T W^{-1} D)$. Therefore

$$z_i^T A z_j = \begin{cases} d_i, & i = j, \\ 0, & i < j. \end{cases} \tag{3.1}$$

(1) $z_1 := e_1$, $w_1 := e_1^T$ and $d_1 := a_{11}$
(2) For $j = 2, \ldots, n$, Do
(3) $z_j := e_j$; $w_j := e_j^T$
(4) For $i = 1, \ldots, j-1$, Do
(5) $\alpha_i := (1/d_i) w_i A_{*j}$; $\beta_i := (1/d_i) A_{j*} z_i$
(6) $z_j := z_j - \alpha_i z_i$; $w_j := w_j - \beta_i w_i$
(7) EndDo
(8) $d_j := w_j A z_j$
(9) EndDo

Algorithm 1: The FFAPINV algorithm without dropping.

Premultiplying both sides of (2.3) by $z_k^T A$, $k = 1, 2, \ldots, j-1$, from the left, we obtain

$$z_k^T A z_j = z_k^T A e_j - \sum_{i=1}^{j-1} \alpha_i z_k^T A z_i. \tag{3.2}$$

Taking into account (3.1), we obtain

$$0 = z_k^T A e_j - \sum_{i=1}^{k-1} \alpha_i z_k^T A z_i - \alpha_k z_k^T A z_k. \tag{3.3}$$

Therefore

$$\alpha_k = \frac{1}{d_k} z_k^T A \left(e_j - \sum_{i=1}^{k-1} \alpha_i z_i \right). \tag{3.4}$$

Hence we can state a procedure for computing the inverse factor Z without need to the inverse factor W as follows:

(1) $z_1 := e_1$, $d_1 := a_{11}$

(2) For $j = 2, \ldots, n$, Do

(3) For $i = 1, \ldots, j-1$, Do

(4) $\alpha_i := (1/d_i) z_i^T A (e_j - \sum_{k=1}^{i-1} \alpha_k z_k)$

(5) EndDo

(6) $z_j := e_j - \sum_{i=1}^{j-1} \alpha_i z_i$

(7) $d_j := z_j^T A z_j$

(8) EndDo

(1) $z_1 := e_1$, $d_1 := a_{11}$
(2) For $j = 2, \ldots, n$, Do
(3) $\quad z_j := e_j$
(4) $\quad$ For $i = 1, \ldots, j-1$, Do
(5) $\qquad \alpha_i := (1/d_i) A_{i*} z_j$
(6) $\qquad z_j := z_j - \alpha_i z_i$
(7) $\quad$ EndDo
(8) $\quad d_j := A_{j*} z_j$
(9) EndDo

Algorithm 2: Left-looking AINV algorithm without dropping.

By some modifications this algorithm can be converted in a simple form, avoiding extra computations. Letting $q_i = z_i^T A$, steps (3)–(7) may be written as follows:

(i) $z_j := e_j$

(ii) For $i = 1, \ldots, j-1$, Do

(iii) $\quad \alpha_i := (1/d_i) q_i (e_j - \sum_{k=1}^{i-1} \alpha_k z_k)$

(iv) $\quad z_j := z_j - \alpha_i z_i$

(v) EndDo

(vi) $q_j := z_j^T A$

(vii) $d_j := q_j z_j$.

Obviously the parameter α_i at step (iii) of this procedure can be computed via

$$\alpha_i = \frac{1}{d_i} q_i z_j. \tag{3.5}$$

We have $AZ = W^{-1} D$. This shows that the matrix AZ is a lower triangular matrix. Therefore, since Z is a unit upper triangular matrix, we deduce

$$\alpha_i = \frac{1}{d_i} q_i z_j = \frac{1}{d_i} z_i^T A z_j = \frac{1}{d_i} e_i^T A z_j = \frac{1}{d_i} A_{i*} z_j. \tag{3.6}$$

On the other hand from (2.9), in step (7) of this procedure, we can replace $d_j := q_j z_j$ by $d_j := A_{j*} z_j$. Now by using the above results we can summarized an algorithm for computing Z as in Algorithm 2.

This algorithm is known as the left-looking AINV algorithm . We observe that the left-looking AINV algorithm can be extracted from the FFAPINV algorithm. This algorithm computes Z with working on rows of A. Obviously the factor W can be computed via this algorithm, working on rows of A^T. In the same way, one can obtain the right-looking AINV algorithm from the BFAPINV algorithm.

4. Conclusions

In this paper, we have shown that the AINV and FAPINV algorithms are strongly related. In fact, we have shown that the AINV algorithm can be extracted from the FAPINV algorithm by some modification. Although, without any dropping, the computation of inverse factors of a matrix by the two algorithms is done in different ways, but the results are the same. Hence many of the properties of each of these algorithms are valid for the other one. For example, in (Benzi and Tuma [13]), it has been shown that the right-looking AINV algorithm without any dropping role is well defined for H-matrices. Therefore we conclude that the BFAPINV algorithm is well defined for H-matrices as well.

Acknowledgment

The authors would like to thank one of the referees for helpful suggestions.

References

[1] Y. Saad and M. H. Schultz, "GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 20, no. 3, pp. 856–869, 1986.
[2] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Press, New York, NY, USA, 2nd edition, 1995.
[3] H. A. van der Vorst, "Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 12, no. 2, pp. 631–644, 1992.
[4] M. Benzi, "Preconditioning techniques for large linear systems: a survey," *Journal of Computational Physics*, vol. 182, no. 2, pp. 418–477, 2002.
[5] M. Benzi and M. Tuma, "A comparative study of sparse approximate inverse preconditioners," *Applied Numerical Mathematics*, vol. 30, no. 2-3, pp. 305–340, 1999.
[6] E.-J. Lee and J. Zhang, "A two-phase preconditioning strategy of sparse approximate inverse for indefinite matrices," Tech. Rep. 476-07, Department of Computer Science, University of Kentuky, Lexington, Ky, USA, 2007.
[7] E.-J. Lee and J. Zhang, "Factored approximate inverse preonditioners with dynamic sparsity patterns," Tech. Rep. 488-07, Department of Computer Science, University of Kentuky, Lexington, Ky, USA, 2007.
[8] J.-G. Luo, "An incomplete inverse as a preconditioner for the conjugate gradient method," *Computers & Mathematics with Applications*, vol. 25, no. 2, pp. 73–79, 1993.
[9] J.-G. Luo, "A new class of decomposition for inverting asymmetric and indefinite matrices," *Computers & Mathematics with Applications*, vol. 25, no. 4, pp. 95–104, 1993.
[10] J.-G. Luo, "A new class of decomposition for symmetric systems," *Mechanics Research Communications*, vol. 19, pp. 159–166, 1992.
[11] J. Zhang, *A procedure for computing factored approximate inverse*, M.S. dissertation, Department of Computer Science, University of Kentucky, Lexington, Ky, USA, 1999.
[12] J. Zhang, "A sparse approximate inverse preconditioner for parallel preconditioning of general sparse matrices," *Applied Mathematics and Computation*, vol. 130, no. 1, pp. 63–85, 2002.
[13] M. Benzi and M. Tuma, "A sparse approximate inverse preconditioner for nonsymmetric linear systems," *SIAM Journal on Scientific Computing*, vol. 19, no. 3, pp. 968–994, 1998.
[14] M. Benzi and M. Tuma, "Numerical experiments with two approximate inverse preconditioners," *BIT*, vol. 38, no. 2, pp. 234–241, 1998.
[15] D. K. Salkuyeh, "ILU preconditioning based on the FAPINV algorithm," submitted.