Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

The design and use of a sparse direct solver for skew symmetric matrices[☆]

Iain S. Duff

Rutherford Appleton Laboratory, Chilton, Oxon OX11 0QX, UK

A R T I C L E I N F O

Article history:
Received 24 July 2007
Received in revised form 12 December 2007

MSC:
65F05
65F50

Keywords:
Direct solver for sparse skew symmetric matrices
Gaussian elimination
Multifrontal method
Matching
Graph collapsing
Convection-diffusion equations
Preconditioning
GMRES

A B S T R A C T

We consider the LDL^{T} factorization of sparse skew symmetric matrices. We see that the pivoting strategies are similar, but simpler, to those used in the factorization of sparse symmetric indefinite matrices, and we briefly describe the algorithms used in a forthcoming direct code based on multifrontal techniques for the factorization of real skew symmetric matrices. We show how this factorization can be very efficient for preconditioning matrices that have a large skew component.

1. Introduction

We are concerned with the solution of

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

when A is an $n \times n$ sparse matrix and x and b are vectors of length n.

We are particularly interested in the case when A is skew symmetric although we will also consider the case when A is a general matrix. We study pivoting strategies for computing the LDL^{T} factorization of a real skew symmetric matrix where L is a lower triangular matrix and D is a block diagonal matrix with 2×2 blocks. We consider direct methods based on a multifrontal technique although many of our comments and analysis apply to other approaches for direct factorization.

We discuss some elementary properties of skew symmetric matrices in Section 2. We consider the factorization of skew symmetric matrices in Section 3 and show some numerical results from a prototype code in Section 4. We then indicate in Section 5 the potential power of our skew solver when used as a preconditioner. We present some conclusions in Section 6.

2. Skew symmetric matrices

A skew symmetric matrix can be defined by the relationship

$$A = -A^{\mathrm{T}} \tag{2.2}$$

[☆] This work was supported by the EPSRC Grant GR/S42170.
E-mail address: i.s.duff@rl.ac.uk.

0377-0427/$ – see front matter Crown Copyright © 2008 Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2008.05.016

so that,

$$a_{ij} = -a_{ji}, \quad \text{for } i \neq j,$$

and

$$a_{ii} = 0, \quad \text{for all } i,$$

so that the diagonal is all zero.

Skew symmetric matrices have many very interesting properties. We list a few of these below, some of which will be important to us when designing and implementing our sparse factorization. Each statement can be proved with a one or two line proof that we leave as a simple exercise for the reader.

- Skew symmetric matrices of odd order are singular.
- Skew symmetric matrices are normal.
- The inverse of a skew symmetric matrix is skew symmetric.
- Skew symmetric matrices have all eigenvalues on the imaginary axis.
- Any matrix is uniquely decomposable into the sum of a symmetric and a skew symmetric matrix.
- If K is skew symmetric, $x^{\mathrm{T}}Kx = 0$, for all x.

We note that the counterpart for complex matrices, called skew Hermitian,

$$A = -A^H$$

allows the diagonal entries to be pure imaginary. As we will see later, our factorization techniques rely heavily on the presence of a zero diagonal and so we will not consider complex matrices in this paper nor will we consider any possible extension of our techniques to that case. We note, however, that if A is skew Hermitian, then iA is Hermitian and suggest that this might be the way to treat such matrices for factorization and equation solving.

Skew symmetric matrices are often ignored by numerical analysts; for example there is no reference to them in two of the standard texts for numerical linear algebra [17,12].

3. Factorization of skew symmetric matrices

Clearly it is not possible to use a 1×1 pivot from the diagonal because all such entries are zero. However, we can use pivots of the form

$$P = \begin{pmatrix} 0 & p \\ -p & 0 \end{pmatrix}, \tag{3.3}$$

which are themselves skew 2×2 blocks. Pivots of this form, where both diagonal entries are zero are termed *oxo* pivots. They also occurred in our earlier work on symmetric indefinite pivoting [6]. If we choose a pivot of this kind and we perform Gaussian elimination using this pivot then the remaining reduced matrix will also be skew symmetric [13] and, in particular, will continue to have an all zero diagonal. Note that this is another way to observe that an odd skew symmetric matrix is singular since there must be a 1×1 diagonal pivot with the value zero.

We can then use the same numerical pivoting as is the case for *oxo* pivots in the indefinite case. Thus, if our potential 2×2 pivot (3.3) comes from rows s and t at stage k of Gaussian elimination, and we define

$$\alpha_k = \max_{j \neq t} |a_{sj}^{(k)}|, \qquad \beta_k = \max_{j \neq s} |a_{tj}^{(k)}|,$$

where the reduced matrix has entries $a_{ij}^{(k)}$, then our threshold test is

$$\max(\alpha_k, \beta_k)/|p| \leq 1/u, \tag{3.4}$$

where $0 < u \leq 0.5$ to ensure that a pivot can always be chosen when the matrix is fully summed. This threshold pivoting is as used ubiquitously in sparse factorization where values near zero maintain sparsity but give a potentially more unstable factorization whereas higher values of the threshold may cause more fill-in in the factors. Our normal default threshold is to take u equal to 0.01.

Note that this is exactly a sparse generalization, using a threshold, of the stability test of [2]. We note that his analysis shows that partial pivoting for skew symmetric matrices is more stable than that for symmetric indefinite systems although the potential for growth is similar in both cases.

Since all pivots are of the form (3.3) and the stability is related to the value of p, it makes sense to use the idea of [8] (that we will call CMP after [10]) to first compress the graph by identifying a pair of nodes of the graph with an *oxo* pivot and then obtain the reduced graph by fusing the two nodes. We follow the earlier work by identifying the blocks using a symmetrized version of `MC64` as discussed in [5,8]. The added benefit of this approach is that the 2×2 blocks thus chosen have as off-diagonal entries a set that maximises their product. It is thus likely that these pivots will satisfy the threshold test (3.4).

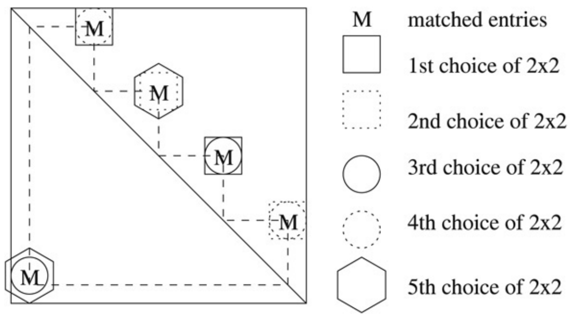

Fig. 3.1. Example of selection of 2×2 pivots in an odd cycle (from [8]).

Indeed, in our current implementation, we use a static pivoting strategy similar to that adopted in [9]. They found, similarly to [14,16], that a preprocessing strategy involving the preselection of large entries as potential pivots is very beneficial in enabling reasonable stability for static pivoting.

We briefly discuss our matching and graph collapsing algorithm in Section 3.1 and the subsequent factorization in Section 3.2.

3.1. Matching and graph collapsing

If the matrix is structurally nonsingular, MC64 will identify a set of n entries, no two in the same row or the same column, such that the product of the moduli of these entries is maximised over all possible such sets. We then follow [5] by using an algorithm similar to that of [9] to identify 2×2 blocks by decomposing the permutation into cycles. Clearly for skew symmetric matrices there are no cycles of length one. Cycles of length two correspond immediately to a 2×2 block. Longer cycles of even parity of length $2k$, say, can be divided into k 2×2 blocks and cycles of odd parity, $2k + 1$, say, can be split into k 2×2 blocks and an odd entry. As this entry corresponds to a zero diagonal (see Fig. 3.1), it is not suitable as a pivot. In this prototype implementation we assign it to the root node where we know it will combine with other single blocks to permit the factorization to complete (we assume that the original matrix is structurally nonsingular).

3.2. Numerical factorization

Our strategy is then to perform an ordering to reduce fill-in on the matrix corresponding to the reduced graph and then to expand the ordering to the original matrix.

We then use this ordering and the corresponding assembly tree to perform the numerical factorization using a multifrontal scheme very similar to that used in the symmetric indefinite case [4]. In particular, we use essentially the static pivoting strategy of [10] where we always choose the pivots as recommended by the analysis so that the work and storage required for the factorization are as predicted in the analysis. At the root node we employ the threshold pivoting strategy (3.4), where u could be set to 0.5.

4. Numerical results from a prototype code

We have developed a prototype code for the direct solution of skew symmetric systems using a multifrontal method with the approach described in Section 3. We call this prototype code SK57 for the purposes of this paper, but we intend to include a more polished version in 2007 under the name MA63.

All of the experimental runs for this paper are performed on a 3.05 Mhz Dell Precision 650 Workstation with 4 Gbytes of RAM. The programs used are coded in Fortran and compiled with the NAG Fortran compiler although they have also all been run using the public domain g95 compiler.

When comparing our new code against other codes, we run into two problems. First there is a lack of skew symmetric problems in the usual sparse matrix data bases. There are only two in the Rutherford–Boeing test set [7], and these two are all that appear in the collection of Tim Davis in Florida [3]. The second issue is that we do not know of any other skew symmetric code with which to compare our new code.

To resolve the first problem we have generated random skew matrices of various orders and densities. There is not an option in the HSL code YM11 to generate skew matrices, so we generate a symmetric matrix in triangular form, remove the diagonal, and interpret the form as skew. We have run with a range of values for the matrix order and the number of entries. We only show four sets here (each is the average of five runs) but they are representative of a much larger set.

To resolve the second problem, we use the HSL code MA48 that is designed for general sparse matrices and so can solve skew symmetric systems.

From the results of Table 4.1 we see that the analyse times for SK57 are much less than for MA48 by a factor of up to nearly 15. This is because SK57 uses all the technology of multifrontal methods and, in particular, does not have to perform

Table 4.1
Comparison of MA48 and SK57 on skew systems

Order		3000	3000	5000	5000
Entries		197000	297000	195000	395000
Time for analysis					
	MA48	1.68	1.92	8.42	9.90
	SK57	0.26	0.34	0.73	0.67
Time for factorize					
	MA48	25.11	28.66	104.67	126.74
	SK57	17.09	19.68	71.83	77.06
Time for solve					
	MA48	0.03	0.03	0.07	0.08
	SK57	0.05	0.05	0.12	0.12
Entries in factors					
	MA48	8010	8388	20007	22499
	SK57	4255	4338	11199	11865
Scaled residual					
	MA48	5.3^{-14}	3.7^{-14}	1.4^{-13}	8.5^{-14}
	SK57	2.2^{-11}	7.3^{-12}	6.3^{-12}	3.5^{-11}

Times in seconds on our DELL workstation.

a numerical factorization during the analysis phase which is the case with MA48. The factorization time is also much quicker as might be expected because the skew solver only performs elimination operations on half the matrix. Likewise the storage is greatly reduced, not quite by half as MA48 has the freedom to choose unsymmetric pivots and can potentially gain a little on fill-in because of this. MA48 does well in the solve phase because the skew code needs to process the factors twice. While the scaled residual of SK57 is acceptable for most purposes, we see that the a priori selection of pivots means that we do not obtain as good a scaled residual as when threshold pivoting (with $u = 0.1$ in Eq. (3.4)) is used throughout.

5. The use of a skew symmetric solver as a preconditioner

Our logic in this section is that if a matrix is close to being skew symmetric, say of the form

$$A = \alpha S + K, \tag{5.5}$$

with S symmetric and K skew, where $\|S\|$ and $\|K\|$ are similar and $\alpha \ll 1$, then it makes sense to approximate the solution of (1.1) by a direct solution using K and then to complete the solution of (1.1) using an iterative method. In our case, we use the implementation of GMRES [15] by Frayssé, Giraud, Gratton, and Langou [11].

We thus solve $Ax = b$, where A is as in (5.5) using a straight application of GMRES on the matrix preconditioned by K^{-1} or at least by the factors of K^{-1} obtained from our direct skew solver. In the experiments that follow we use preconditioning from the right but we have tried left and two-sided preconditioning with essentially identical results.

For our tests, we again use random matrices and have again averaged the results, this time over three runs. From the results in Table 5.2 we see that our scheme is indeed very good even for a not too small value of α.

Of course, any square matrix (A) can be split into the sum of a symmetric (S) and a skew matrix (K) and that decomposition is unique, viz.

$$A = S + K = \frac{1}{2}(A + A^{\mathrm{T}}) + \frac{1}{2}(A - A^{\mathrm{T}}),$$

so the algorithm that we have just outlined could be considered for solving a general unsymmetric system. However, there is no guarantee that either component is nonsingular so our attempts at using this as a general strategy did not meet with much success. We should note that we also tried to use a direct method on the symmetric part as a preconditioner and this was just as unsuccessful.

We then performed some experiments by permuting and scaling A to try to obtain a healthily nonsingular skew component. The algorithm that we use to do this first selects a transversal using MC64 and then further permutes the columns so that no transversal is on the diagonal of the permuted matrix. We then scan the columns in order and follow chains of columns determined by the maximum transversal entries. We scale the first column in the chain by one and then scale each column in the chain in the following way until we return to the first column of the chain (this corresponds to finding cycles in the permutation). From a current column in the chain, the next column in the chain is the one corresponding to the row of the transversal entry in the current column. We let the scaling on the current column be sc_1 and the value of the transversal entry a_1. If the symmetrically placed entry to this in the next column has value a_2, then we choose the scaling for the next column to be $-(a_1 \times sc_1)/a_2$. Thus we generate skew pairs in the matrix so that the skew part of the matrix is at least structurally nonsingular (if A is) and has a good chance (because of the use of MC64) of being nonsingular. Sad to say, this also only works marginally better than an unpreconditioned run of GMRES and not always so.

Of course the unique skew component of a general matrix will not necessarily be the best skew approximation for the matrix (of course the other component will no longer be symmetric) so there may be better ways to select a skew matrix to use as a preconditioner, for example one could just take the lower triangle of the matrix to represent the skew component.

Table 5.2
Number of iterations for preconditioned GMRES

α		0.0	10^{-8}	10^{-2}	10^{-1}	1.0
Order	Entries					
100	400	1	2	9	21	100
200	1200	1	2	9	23	203
300	1800	1	2	21	27	313

Finally, there are practical cases where the skew part should be dominant, for example in convection-diffusion equations that are highly convection dominated. We have done some initial experiments on this with Andy Wathen of Oxford but without a great deal of success. On further investigation it turned out that, although he was modelling highly convective equations, the skew part of the matrix was not that dominant because of the artificial viscosity terms to avoid problems with spurious oscillations of the numerical solution. We plan to investigate further the use of a skew preconditioner on formulations of highly convective equations where this has not been done.

We do note, however, that this approach and the code are not applicable in recent work on using Hermitian and skew-Hermitian splitting methods (for example, [1]) even in cases when the matrices are real. This is so because, in spite of the titles of the papers, the iteration matrices are of the form

$$(\alpha I - S)$$

with S skew so that they are neither skew nor do they have zero diagonals, which is crucial to our approach and software.

6. Conclusions

We have presented pivoting strategies for skew symmetric matrices and shown their efficacy in a prototype code. We have illustrated the strengths and weaknesses of such a factorization and have shown how it could be useful in preconditioning systems, particularly those with a large skew component.

Acknowledgements

I would like to thank John Reid of the Rutherford Appleton Laboratory for his comments on an early draft of this manuscript. I would also like to thank the anonymous referee for his or her comments.

References

[1] Z.-Z. Bai, G.H. Golub, M.K. Ng, Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems, SIAM Journal of Matrix Analysis and Applications 24 (2003) 603–626.
[2] J.R. Bunch, A note on the stable decomposition of skew-symmetric matrices, Mathematics of Computation 38 (158) (1982) 475–479.
[3] T.A. Davis, University of Florida sparse matrix collection. http://www.cise.ufl.edu/research/sparse/matrices/, 2004.
[4] I.S. Duff, MA57 – A code for the solution of sparse symmetric indefinite systems, ACM Transactions on Mathematical Software 30 (2) (2004) 118–144.
[5] I.S. Duff, J.R. Gilbert, Maximum-weight matching and block pivoting for symmetric indefinite systems, in: Abstract book of Householder Symposium, vol. XV, June 17–21, 2002.
[6] I.S. Duff, N.I.M. Gould, J.K. Reid, J.A. Scott, K. Turner, Factorization of sparse symmetric indefinite matrices, IMA J. Numerical Analysis 11 (1991) 181–204.
[7] I.S. Duff, R.G. Grimes, J.G. Lewis, The Rutherford-Boeing sparse matrix collection, Tech. Rep. RAL-TR-97-031, Rutherford Appleton Laboratory, Oxfordshire, England, also Technical Report ISSTECH-97-017 from Boeing Information & Support Services, Seattle and Report TR/PA/97/36 from CERFACS, Toulouse, 1997.
[8] I.S. Duff, S. Pralet, Experiments in preprocessing and scaling symmetric problems for multifrontal solutions, Tech. Rep. WN/PA/04/17, CERFACS, Toulouse, France, February 2004.
[9] I.S. Duff, S. Pralet, Strategies for scaling and pivoting for sparse symmetric indefinite problems, SIAM Journal of Matrix Analysis and Applications 27 (2) (2005) 313–340.
[10] I.S. Duff, S. Pralet, Towards stable mixed pivoting strategies for the sequential and parallel solution of sparse symmetric indefinite systems, SIAM Journal of Matrix Analysis and Applications 29 (3) (2007) 1007–1024.
[11] V. Frayssé, L. Giraud, S. Gratton, J. Langou, A set of GMRES routines for real and complex arithmetics on high performance computers, Technical Report TR/PA/03/3, CERFACS, Toulouse, France, January 2003.
[12] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., The John Hopkins University Press, Baltimore, MD, 1996.
[13] N.J. Higham, Accuracy and Stabilty of Numerical Algorithms, 2nd ed., SIAM, Philadelphia, 2002.
[14] X.S. Li, J.W. Demmel, SuperLU_DIST: A scalable distributed-memory sparse direct solver or unsymmetric linear systems, ACM Transactions on Mathematical Software 29 (2) (2003) 110–140.
[15] Y. Saad, M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal of Scientific and Statistical Computing 7 (1986) 856–869.
[16] O. Schenk, K. Gärtner, On fast factorization pivoting methods for sparse symmetric indefinite systems, Tech. Rep. CS-2004-004, CS Department, University of Basel, August 2004.
[17] G.W. Stewart, Introduction to Matrix Computations, Academic Press, New York, NY, 1973.