Scalable pivoting strategies and orderings for sparse symmetric indefinite problems

Iain. S. Duff* Stéphane Pralet†

We study ordering and pivoting strategies when computing the LDL^T factorization of a symmetric indefinite matrix where L is a lower triangular matrix and D is a block diagonal matrix with 1×1 and 2×2 blocks. We consider direct methods based on a multifrontal technique although most of our comments and analysis apply to other approaches for direct factorization. In the multifrontal scheme, the factorization can be represented by a tree where, at each node, elimination operations are performed within a dense matrix, termed the *frontal matrix*

$$\begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}, \tag{1}$$

where pivots can be chosen only within F_{11} and the Schur complement matrix $F_{22} - F_{21}F_{11}^{-1}F_{12}$ is passed for summation into the frontal matrix of the parent node of the tree. In the symmetric case, $F_{21} = F_{12}^T$. Usually the factorization is computed in two phases. The analysis phase preprocesses the system of equations and is often based purely on matrix structure. The second phase performs the Gaussian elimination. If the threshold tests of the pivoting strategy prevent the selection of some pivots from the F_{11} block, then the factorization can still proceed but their elimination is delayed which will normally increase both storage and work for the factorization.

Firstly we will present new classes of orderings called *constrained orderings* that select pivots during the symbolic Gaussian elimination using two graphs. These orderings are described in [2] but have never been presented. Secondly we will propose new pivoting strategies that combine both numerical and static pivoting and that are well designed for scalable parallel distributed factorizations.

We will use our orderings and our pivoting strategies with a symmetric multifrontal code `MA57` [5] on real test problems that are available from `ftp.numerical.rl.ac.uk/pub/matrices/symmetric/indef/`.

1. Constrained orderings

The main principle of our orderings is to guide the ordering with the preselection of 2×2 and 1×1 pivots. This preselection is based on the work of [1, 2] that computes a symmetric weighted matching from a maximum weighted matching.

Our ordering manipulates two graphs. The first graph is the usual quotient graph used in greedy orderings. It is used to perform the symbolic factorization and to compute metrics. The second graph, the *constraint graph*, is an undirected graph $G_c = (V_f, V_c, E)$ where V_f, the *free variables*, and V_l, the *constrained variables*, are sets of vertices and E is the edge set.

At the beginning of the ordering, a supervariable i is a free supervariable if and only if it corresponds to a large enough diagonal entry, or it corresponds to a 2×2 pivot. Otherwise it belongs to V_c. At each step of the symbolic elimination, we select the best pivot in the set V_f according to a metric related to the quotient graph and each variable in V_c that are adjacent to i are removed from V_c and added to V_f.

We will show that our constrained orderings are faster, need less memory and give reliable estimations.

2. Scalable pivoting strategies

To keep the symmetry while maintaining stability 1×1 and 2×2 pivoting is performed. The criterion of the Duff-Reid algorithm ([3], as modified in [4]) can be used to ensure a growth factor lower than $1/u$ at each

*CERFACS, Toulouse, and Atlas Centre, RAL, Oxon OX11 0QX, England. (`i.s.duff@rl.ac.uk`).
†CNRS-IRIT, ENSEEIHT 2, rue Charles Camichel, BP 7122 - F 31071 Toulouse Cedex 7 (`Stephane.Pralet@enseeiht.fr`).

step of Gaussian elimination. A 1×1 diagonal pivot can be selected if and only if

$$|a_{ii}| \geq u \max_j |a_{ij}| \tag{2}$$

and a 2×2 pivot $P = \begin{pmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{pmatrix}$ can be selected if and only if

$$|P^{-1}| \begin{pmatrix} \max_{k\neq p,q} |a_{pk}| \\ \max_{k\neq p,q} |a_{qk}| \end{pmatrix} \leq \begin{pmatrix} 1/u \\ 1/u \end{pmatrix} \tag{3}$$

where $|P^{-1}|$ denotes the matrix whose values are the absolute values of P^{-1}.

A static approach was proposed by [6] in the context of LU factorization. During Gaussian elimination *"small perturbations"* are added to limit the growth of the factors in order to enhance the backward stability of the algorithm. In our combinations of static and numerical pivoting the factorization does not necessarily follow the analysis and some slight variations are allowed. We want a pivoting strategy with the following features:

(F1) It is easy to decide whether to add a perturbation or not. In particular, in a parallel symmetric indefinite solver, this decision must not involve any limiting extra communication cost or synchronization.

(F2) The perturbations are restricted to the block of fully summed rows/columns in each front.

Let us consider a frontal matrix from the elimination tree. Our mixed approach is based on two phases. In the first phase, we perform numerical pivoting in the block of fully summed variables until no remaining variables satisfy the numerical criterion. In the second phase, we eliminate the remaining fully summed variables adding 1×1 perturbations if necessary.

In a parallel distributed environment the fully summed part of large nodes are stored on a single processor, the master. Furthermore the master does not have local access to partially summed rows/columns that are sent directly from the slaves of its child nodes to its own slaves. To avoid extra communications and, even worse, synchronizations we cannot completely check the stability criteria (2) and (3). We will propose an approximation of the off-diagonal information that does not limit the scalability and that will significantly improve the numerical robustness of the factorization. For each fully summed variable of a node p, each slave of its children sends to the master of p the maximum entry that it has computed in its contribution block. Then the master of the parent node will approximate the maximum entry in each row with the maximum quantity that it received from the slaves of its child nodes. Note that it is only an approximation because the child contributions are summed with those from the siblings.

	Iteration 0		Iteration 1		Iteration 2	Factorization Time	
Matrix	`numSEQ`	`mixPAR`	`numSEQ`	`mixPAR`	`mixPAR`	`numSEQ`	`mixPAR` simulation
BRAINPC2	1.6e-15	2.1e-08	1.0e-15	5.7e-15	9.8e-16	0.18	0.11
BRATU3D	2.0e-09	1.7e-05	1.7e-16	1.3e-10	2.3e-16	34.2	9.24
CONT-201	8.8e-11	1.8e-05	1.6e-16	9.4e-09	4.5e-09	5.51	1.94
CONT-300	7.6e-11	1.9e-05	1.9e-16	2.6e-09	3.4e-09	21.1	6.08
cvxqp3	5.2e-11	8.0e-06	2.7e-16	9.3e-13	2.7e-16	9.73	3.08
DTOC	2.1e-16	8.3e-07	2.7e-20	2.1e-13	1.9e-15	29.1	0.41
mario001	6.3e-15	3.1e-08	1.3e-16	2.5e-13	1.3e-16	0.28	0.23
NCVXQP1	4.6e-14	3.3e-13	1.7e-17	4.4e-15	6.1e-17	2.69	1.29
NCVXQP5	2.0e-11	7.5e-08	2.0e-16	1.6e-11	1.5e-14	25.7	23.0
NCVXQP7	9.6e-10	4.3e-06	2.2e-16	2.0e-12	2.7e-16	195.	71.6
SIT100	4.4e-15	2.0e-08	1.4e-16	5.8e-15	1.5e-16	0.13	0.11
stokes128	1.1e-14	4.2e-14	5.5e-16	2.0e-15	1.7e-15	1.14	1.06
stokes64	4.3e-15	1.6e-13	1.5e-15	2.3e-14	2.2e-14	0.33	0.29

Table 1: Component-wise backward error of strategies with static pivoting and factorization time (in seconds). `numSEQ`: sequential approach with numerical pivoting. `mixPAR`: parallel approach combining numerical and static pivoting.

We simulated the parallel factorization in `MA57`. Table 1 shows that our parallel combination of numerical and static pivoting `mixPAR` is numerically robust on our test set. On average, it needs one iteration more to get a similar precision as the sequential `numSEQ` strategy. It is also significantly faster.

References

[1] I. S. Duff and J. R. Gilbert. Maximum-weight matching and block pivoting for symmmetric indefinite systems. In *Abstract book of Householder Symposium XV, June 17-21*, 2002.

[2] I. S. Duff and S. Pralet. Strategies for scaling and pivoting for sparse symmetric indefinite problems. Technical Report TR/PA/04/59, CERFACS, Toulouse, France, 2004. To appear in SIMAX. Also appeared as RAL report RAL-TR-2004-020.

[3] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear systems. *ACM Transactions on Mathematical Software*, 9:302–325, 1983.

[4] I. S. Duff and J. K. Reid. Exploiting zeros on the diagonal in the direct solution of indefinite sparse symmetric linear systems. *ACM Transactions on Mathematical Software*, 22(2):227–257, 1996.

[5] Iain S. Duff. MA57 – A code for the solution of sparse symmetric indefinite systems. *ACM Transactions on Mathematical Software*, 30(2):118–144, 2004.

[6] X. S. Li and J. W. Demmel. Making sparse Gaussian elimination scalable by static pivoting. In *Proceedings of Supercomputing*, Orlando, Florida, November 1998.