Grouping variables in Frontal Matrices to improve Low-Rank Approximations in a Multifrontal Solver

Clement Weisbecker, Patrick Amestoy, Cleve Ashcraft, Olivier Boiteau, Alfredo Buttari, Jean-Yves L'Excellent

Grouping variables in Frontal Matrices to improve Low-Rank Approximations in a Multifrontal Solver

List of authors:

C. Weisbecker [1] P. Amestoy [2] C. Ashcraft[3] O. Boiteau[4] A. Buttari[5] J.-Y. L'Excellent[6]

Abstract. Low-rank approximations are commonly used to compress the representation of data structures. The loss of information induced is often negligible and can be controlled. Although the dense internal datastructures involved in a multifrontal method, the so-called *frontal matrices or fronts*, are full-rank, they can be represented by a set of low-rank matrices [1, 2]. Applying to our context the notion of geometric clustering used by [3] to define hierarchical matrices, we show that the efficiency of this representation to reduce the complexity of both the factorization and solve phases strongly depends on how variables are grouped. The proposed approach can be used either to accelerate the factorization and solution phases or to build a preconditioner, and aims at exploiting and completing the features of the `MUMPS` solver for general sparse matrices.

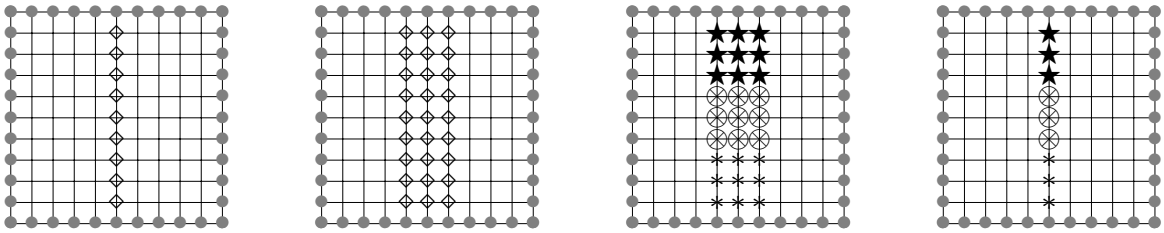
Context and motivation. With a SVD or with rank-revealing QR, a matrix with a *low enough* numerical rank controlled by a parameter ε can be decomposed as a product of two matrices [3] with smaller storage requirements than the initial matrix. This representation (called *low-rank form*) also reduces the complexity of basic algebraic operations.

The objective of this work is to exploit low-rank properties for the direct solution of sparse systems of linear equations through a multifrontal approach (Duff and Reid). This method is based on an assembly tree where each node represents the elimination of a set of pivots of the original matrix called *fully assembled variables*. Those rows and columns define the structure of a dense matrix called *frontal matrix*, and the elimination of the fully assembled variables is achieved through a partial factorization of the frontal matrix. The updates on the remaining variables are also computed and are passed to the parent node for elimination. Although frontal matrices are often full rank, off-diagonal submatrices might be low-rank [1]. One key observation that has motivated our work is that the concept of proximity between variables has to be taken into account to improve the gains resulting from the low-rank representation of the blocks. For fronts resulting from the factorization of Partial Differential Equations matrices of elliptic problems, we have observed that an off-diagonal block has a lower rank if the two sets of variables that it connects are distant from each other in the discretization mesh.

Proposed approach and experiments. Following an algebraic approach, we want to define a grouping strategy of the front variables where variables close to each other are in the same block. This can be achieved by partitioning the subgraph $\mathcal{G}_{FS}$ of the original matrix graph $\mathcal{G}_A$ induced by the fully assembled variables. As $\mathcal{G}_{FS}$ might be disconnected or poorly connected, we compute a well connected halo subgraph $\mathcal{H}$ that catches the geometry of $\mathcal{G}_{FS}$ (Figure 1). For a given depth d, we define the halo nodes as the set of nodes from $\mathcal{G}_A$ at distance less than

[1,2] Université de Toulouse, INPT(ENSEEIHT)-IRIT, France
[3]LSTC, USA [4] EDF R&D, France
[5]CNRS-IRIT, France [6] INRIA-LIP (ENS Lyon), France

d from at least one node in $\mathcal{G}_{FS}$. $\mathcal{H}$ is defined as the subgraph of $\mathcal{G}_A$ induced by the set of halo nodes (Figure 1b). Our grouping then results from partitioning $\mathcal{H}$ (Figures 1c,1d).

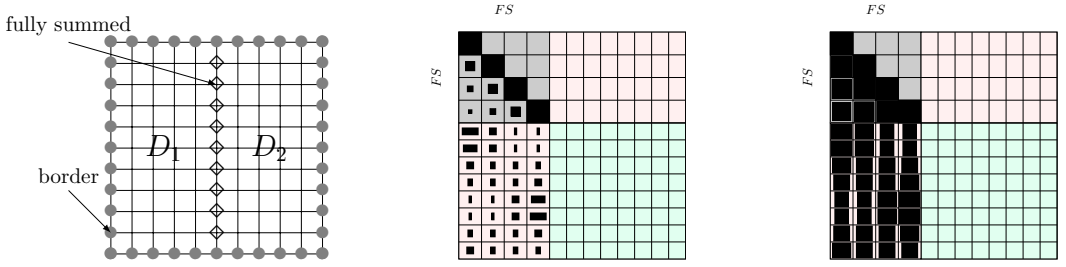


(a) $\mathcal{G}_{FS}$ (b) halo subgraph $\mathcal{H}$ (c) 3 partitions of $\mathcal{H}$ (d) partition of $\mathcal{G}_{FS}$

Figure 1: Halo-based partitioning of $\mathcal{G}_{FS}$ ($d = 1$)

The same grouping strategy is applied to the non-fully assembled variables of the front (the grouping of these variables could also be inherited from the grouping of fully assembled variables in fronts that are higher in the tree). The depth d has to be small enough to ensure that $\mathcal{H}$ is much smaller than $\mathcal{G}_A$ and correctly captures the geometry of $\mathcal{G}_{FS}$. This grouping will induce a blocking of the frontal matrix where off-diagonal blocks will be represented by low-rank forms. Because we target a parallel multifrontal solver with advanced numerical features, this simple, regular blocking was chosen over more complicated ones such as hierarchical matrices [3]. Still, it is general enough to handle various classes of problems.

The proposed approach was first experimented on an artificial test case defined over a 2D domain discretized with a regular mesh. A separator of the mesh is computed and the nodes are classified into separator nodes, border nodes and subdomain nodes (nodes in D_i), as shown in Figure 2a. Once the variables of the two subdomains have been eliminated, the separator has to be eliminated and the border is updated. Such a process is in practice repeated at each node of the assembly tree. Figure 2 plots the results measured on a problem generated using the GetFEM++[7] toolbox to define the matrix associated to the laplacian operation (P1 elements and Dirichlet conditions) over a 2D domain of size 250000. Inside each block, a black box is drawn whose size is proportional to the block's rank.



(a) The domain (b) Halo-based partitioning (c) Random partitioning

Figure 2: Domain and Block ranks (density proportionnal to the rank, $\varepsilon = 10^{-14}$)

We observe that although the front has full rank, a good grouping (compare Figures 2b and 2c) can reveal low-rank blocks. It has to be noted that the obtained gains are not affected by the relative order of blocks nor by the order of variables within each block. Experiments also

[7] `http://download.gna.org/getfem/html/homepage/`

show that the reduction of rank is relatively insensitive to the size of the blocks. We thus have the freedom to choose a different block size in order to satisfy other possible constraints (e.g. cache and BLAS effects, granularity for parallelism and communication). Low-rank properties can be exploited through different strategies that can be thought of as incremental modifications of an existing multifrontal solver such as `MUMPS`[8]. The partial factorization of a frontal matrix is done by panels of columns, with three main steps associated to each panel: factorization of a diagonal block, triangular solves to apply the transformation to the panel (i.e., to each block below the diagonal block) and right-looking updates. Putting a block into low-rank form (*demoting* a block) can be done at any of these steps. This choice affects the behavior of the method. The earlier we demote a block, the larger will be the gains but the more approximated the resulting updating operations will be. Depending on this choice, we defined three strategies. In **strategy 1**, the blocks are demoted once the front is completely processed. This only affects the storage of the factors (stored in low-rank form). Thus, it reduces the operation and memory complexity of the solution phase. (The solve complexity is the same as the storage complexity.) **Strategy 2**, instead, uses low-rank operations for updating the block of border variables. A reduction in the number of operations during factorization will thus occur. The factorization might however be less accurate with some possible error propagation. **Strategy 3** also uses low-rank operations during the panel factorization. A larger reduction in the number of operations can thus be expected. The table presents numerical results with $\varepsilon = 10^{-14}$. After resolution, we observed a backward error of the same order of ε. The potential of this approach is also illustrated in terms of storage for the factors (reduced by 81%) and operation count for the factorization (reduced by up to 82%). Preliminary results with MUMPS on real-world matrices from EDF (generated with *Code_Aster*[9]) have confirmed that low-rank representations can be exploited in the highest levels of the assembly tree, with slightly smaller but still comparable gains.

Strategy	1	2	3
storage (gain in %)	81	81	81
#ops (gain in %)	0	68	82

The next step is to understand how the physical problem and domain influence the efficiency of the method, and perform a deeper study of grouping methods for irregular problems (e.g. one could use the numerical values in a frontal matrix to help detecting variables with small interactions, taking into account interactions in the partially filled graph rather than in the initial matrix). We will report more experiments on real problems of different nature at the conference.

References.

[1] S. Chandrasekaran, P. Dewilde, M. Gu and N. Somasunderam, *On the Numerical Rank of the Off-Diagonal Blocks of Schur Complements of Discretized Elliptic PDEs*, SIAM Journal on Matrix Analysis and Applications, 31 (2010), pp. 2261-2290.

[2] J. Xia, S. Chandrasekaran, M. Gu and X. S. Li, *Superfast Multifrontal Method for Large Structured Linear Systems of Equations*, SIAM J. Mat. Anal. Appl., 31 (2009), pp. 1382-1411.

[3] M. Bebendorf, *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems (Lecture Notes in Comp. Science and Engineering)*, Springer, 1 ed., 2008.

[8] `http://http://graal.ens-lyon.fr/MUMPS/` or `http://mumps.enseeiht.fr/`

[9] `http://www.code-aster.org`