ON THE COMPLEXITY OF THE BLOCK LOW-RANK MULTIFRONTAL FACTORIZATION

PATRICK AMESTOY[*], ALFREDO BUTTARI[†], JEAN-YVES L'EXCELLENT[‡], AND THEO MARY[§]

Abstract. Matrices coming from elliptic Partial Differential Equations have been shown to have a low-rank property: well defined off-diagonal blocks of their Schur complements can be approximated by low-rank products and this property can be efficiently exploited in multifrontal solvers to provide a substantial reduction of their complexity. Among the possible low-rank formats, the Block Low-Rank format (BLR) is easy to use in a general purpose multifrontal solver and has been shown to provide significant gains compared to full-rank on practical applications. However, unlike hierarchical formats, such as $\mathcal{H}$ and HSS, its theoretical complexity was unknown. In this paper, we extend the theoretical work done on hierarchical matrices in order to compute the theoretical complexity of the BLR multifrontal factorization. We then present several variants of the BLR multifrontal factorization, depending on the strategies used to perform the updates in the frontal matrices and on the constraints on how numerical pivoting is handled. We show how these variants can further reduce the complexity of the factorization. In the best case (3D, constant ranks), we obtain a complexity of the order of $O(n^{4/3})$. We provide an experimental study with numerical results to support our complexity bounds.

Key words. sparse linear algebra, multifrontal factorization, Block Low-Rank

AMS subject classifications. 15A06, 15A23, 65F05, 65F50, 65N30, 65Y20

1. Introduction.

We are interested in efficiently computing the solution of large sparse systems of linear equations. A sparse linear system is usually referred to as:

$$Ax = b\,, \tag{1}$$

where A is a sparse matrix of order n, x is the unknown vector of size n, and b is the right-hand side vector of size n.

This paper focuses on solving (1) with direct approaches based on Gaussian elimination and more particularly the multifrontal method, which was introduced by Duff and Reid [19] and, since then, has been the object of numerous studies [32, 3, 17].

The multifrontal method achieves the factorization of a sparse matrix A as $A = LU$ or $A = LDL^T$ depending on whether the matrix is unsymmetric or symmetric, respectively. A is factorized through a sequence of operations on relatively small dense matrices called *frontal matrices* or, simply, *fronts*, on which a partial factorization is performed, during which some variables (the fully-summed (FS) variables) are eliminated, i.e. the corresponding factors are computed, and some other variables (the non fully-summed (NFS) variables) are only updated. To know which variables come into which front, and in which order the fronts can be processed, an *elimination* or *assembly tree* [31, 36] is built, which represents the dependencies between fronts.

When system (1) comes from the discretization of an elliptic Partial Differential Equation (PDE), the matrix A, known as stiffness matrix, has been shown to have a low-rank property: conveniently defined off-diagonal blocks of its Schur complements can be approximated by low-rank products [13]. Several formats have been proposed to exploit this property, mainly differing on whether they use a strong or weak admissibility condition (defined in Section 2.3) and on whether they have a nested basis property. The most general of the hierarchical formats is the $\mathcal{H}$-matrix format [12, 26, 14], which is non-nested and strongly-admissible. The $\mathcal{H}^2$-matrix format [14] is its nested counterpart. HODLR matrices [6] are based on the weak admissibility condition and HSS [41, 15] and the closely related HBS [24] formats additionally possess nested basis.

These low-rank formats can be efficiently exploited within direct multifrontal solvers to provide a substantial reduction of their complexity. In comparison to the quadratic complexity of the full-rank solver, most sparse solvers based on hierarchical formats have been shown to possess near-linear complexity. To cite a few, [40, 39, 22] are HSS-based, [24] is HBS-based, [7] is

[*]Université de Toulouse, INPT-IRIT
[†]Université de Toulouse, CNRS-IRIT
[‡]Université de Lyon, INRIA-LIP
[§]Université de Toulouse, UPS-IRIT

HODLR-based, and [34] is $\mathcal{H}^2$-based. Other related work includes [28], a multifrontal solver with front skeletonization, and [37], a Cholesky solver with fill-in compression.

Previously, we have investigated the potential of a so-called Block Low-Rank (BLR) format [1] that, unlike hierarchical formats, is based on a flat, non-hierarchical blocking of the matrix which is defined by conveniently clustering the associated unknowns. While its efficiency has been shown in practice on real applications [1, 2], its theoretical complexity was unknown. Unlike hierarchical formats, it remained to be proved that the BLR format does not only reduce the computations by a constant, i.e., possesses a complexity in $O(n^2)$ just like the full-rank solver.

The main objective of this paper is to compute the complexity of the BLR multifrontal factorization. We will first prove that BLR does provide a non-constant gain, and then show how variants of the BLR approach (introduced in [9, 5]) influence its complexity.

We now explain how we reach this objective by shortly describing the contents of each section. Section 2 is devoted to preliminaries; we provide an overview of the Block Low-Rank approximations that can be used within multifrontal solvers. We also present the context of the complexity study, the resolution of elliptic PDEs with the Finite Element (FE) method. Finally, we introduce classical block-admissibility conditions aiming at determining if a block is admissible for low-rank compression. In Section 3, we compute the theoretical bounds on the numerical ranks of the off-diagonal blocks in BLR matrices arising in our context. First, we briefly review the work done on hierarchical matrices and the complexity of their factorization. Then, we explain why applying this work to BLR matrices (which are a very particular kind of hierarchical matrices) does not provide a satisfying result. We then give the necessary ingredients to extend this work to the BLR case. In Section 4, we use the rank bounds computed in Section 3 to compute the theoretical complexity of the standard dense BLR factorization. Then, we explain in Section 5 how the dense BLR factorization can be used within each node of the tree associated with a sparse multifrontal factorization, and we compute the complexity of the corresponding BLR multifrontal factorization. We then present in Section 6 algorithmic variants of the BLR factorization and show how they can further reduce its complexity. In Section 7, we support our theoretical complexity bounds with numerical experiments and analyze the influence on complexity of each variant of the BLR factorization and of two other parameters: the low-rank threshold and the block size. We provide our concluding remarks in Section 8.

2. Preliminaries.

2.1. Block Low-Rank approximations.

In this section, we provide a brief presentation of the BLR format, and explain its main differences with hierarchical formats. A more detailed and formal presentation can be found in Amestoy et al. [1].

Unlike hierarchical formats such as $\mathcal{H}$-matrices, the BLR format is based on a flat, non-hierarchical blocking of the matrix which is defined by conveniently clustering the associated unknowns. Assuming that p such clusters have been defined, and that a permutation P has been defined so that permuted variables of a given cluster are contiguous, a BLR representation $\widetilde{S}$ of a dense matrix S is shown in Equation (2). Subblocks $B_{ij} = (PSP^T)_{ij}$, of size $m_{ij} \times n_{ij}$ and numerical rank k_{ij}^{ε}, are approximated by a low-rank product $\widetilde{B}_{ij} = X_{ij}Y_{ij}^T$ at accuracy ε, where X_{ij} is a $m_{ij} \times k_{ij}^{\varepsilon}$ matrix and Y_{ij} is a $n_{ij} \times k_{ij}^{\varepsilon}$ matrix.

$$\widetilde{S} = \begin{bmatrix} \widetilde{B}_{11} & \widetilde{B}_{12} & \cdots & \widetilde{B}_{1p} \\ \widetilde{B}_{21} & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \vdots \\ \widetilde{B}_{p1} & \cdots & \cdots & \widetilde{B}_{pp} \end{bmatrix} \tag{2}$$

The BLR format is closely related to Block Separable matrices introduced in [16] (Section 5.2) and described in [24, 23]. The main difference is that the off-diagonal blocks of a Block Separable matrix are all assumed to be low-rank, which is equivalent to a BLR matrix with a weak admissibility condition (defined by (Adm_b^w) in Section 2.3). One of the key results of this paper (Lemma 2) is that the number of full-rank blocks on any row can be bounded by a constant, even for a strong admissibility condition for which theoretical bounds on the ranks have been proven.

For the sake of simplicity, and without loss of generality, we will assume in the following that, for a given matrix $\widetilde{S}$, the property $\forall i, j \;\; m_{ij} = n_{ij} = b$ holds, where b, the block size, can depend on the order of the matrix.

Because the multifrontal method relies on dense factorizations, the BLR approximations can be easily incorporated into the multifrontal factorization by representing the frontal matrices as BLR matrices, as will be described in Section 5. In fact, many of the properties we will show in this paper are true for general dense BLR matrices and can be applied to broader contexts than the multifrontal method.

The clustering of the unknowns (noted $\mathcal{I}$) into subdomains that define the blocks of the matrix A is formalized as a subdomain partition $\mathfrak{S}(\mathcal{I}\times\mathcal{I})$ of $\mathcal{I}\times\mathcal{I}$. How this partition is computed in the context of the multifrontal method is explained in Section 5.

In general, hierarchical partitionings are a completely general partitioning of $\mathcal{I}\times\mathcal{I}$. However, a BLR partitioning satisfies the following property:

$$\forall(\sigma\times\tau, \rho\times\upsilon) \in \mathfrak{S}(\mathcal{I}\times\mathcal{I})^2, \quad \sigma\times\upsilon \in \mathfrak{S}(\mathcal{I}\times\mathcal{I}) \quad (\text{and } \rho\times\tau \in \mathfrak{S}(\mathcal{I}\times\mathcal{I}))$$

which is not satisfied by a general hierarchical format as shown in Figure 1(a). This specificity of BLR partitionings is illustrated in Figure 1. In the literature related to hierarchical formats [25], the partitionings are often formalized with so-called *cluster trees* and *block cluster trees* (Figure 2). In the BLR case, we do not need a block cluster tree, because the blocking of A is uniquely defined by the row and column indices cluster trees only (Figure 3). Indeed, the block cluster tree associated to a BLR partitioning is always the complete tree resulting from the block-product of the row and column cluster trees. Note that in Figures 1, 2, and 3 we have assumed, for both the $\mathcal{H}$ and BLR partitionings, that the row and column cluster trees are the same, for the sake of simplicity.

We define the so-called *sparsity constant*:

$$c_{sp} = \max\left(\max_i \#\{I_j; I_i\times I_j \in \mathfrak{S}(\mathcal{I}\times\mathcal{I})\}, \max_j \#\{I_i; I_i\times I_j \in \mathfrak{S}(\mathcal{I}\times\mathcal{I})\}\right) \tag{3}$$

where $\#E$ denotes the cardinality of a set E (we will use this notation throughout the rest of the article). Thus, the sparsity constant is the maximum number of blocks of a given level in the block cluster tree that are in the same row or column of the matrix. For example, in Figure 1(a), c_{sp} is equal to 2, and in Figure 1(b), it is equal to 4. Under the assumption of a partitioning $\mathfrak{S}(\mathcal{I}\times\mathcal{I})$ defined by a geometrically balanced tree, the sparsity constant can be bound by a constant in $O(1)$ ([25], Lemma 4.5). A geometrically balanced tree is a block cluster tree resulting from a partitioning computed as the intersection between the domain Ω (defined below in (4)) and a hierarchical cubic domain. For a formal description, see Construction 4.3 in [25].

In the following, when referring to the BLR case, we simplify the notation $\mathfrak{S}(\mathcal{I}\times\mathcal{I})$ to $\mathfrak{S}(\mathcal{I})$, as in most cases we do not need a different partitioning of the row and column variables.

Next, we describe the context in which we place ourselves for the complexity study, which is the same as in Hackbusch & Bebendorf [13].

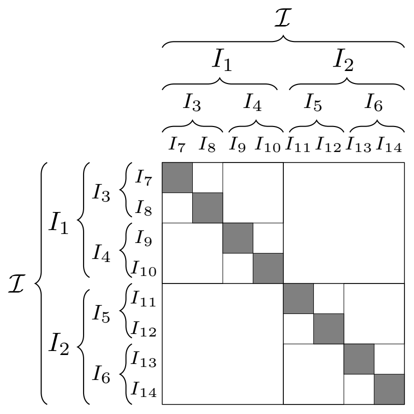
2.2. FE discretization of elliptic PDEs. We consider a Partial Differential Equation of the form:

$$\begin{aligned} Lu &= f \quad \text{in } \Omega \subset \mathbb{R}^d, \Omega \text{ convex }, d \geq 2 \\ u &= g \quad \text{on } \partial\Omega \end{aligned} \tag{4}$$

where L is a uniformly elliptic operator in divergence form:

$$Lu = -\text{div}[C\nabla u + \boldsymbol{c_1} u] + \boldsymbol{c_2}\cdot\nabla u + c_3 u$$

C is a $d\times d$ matrix of functions, such that $\forall x, C(x) \in \mathbb{R}^{d\times d}$ is symmetric positive definite with entries $c_{ij} \in L^\infty(\Omega)$. Furthermore, $\boldsymbol{c_1}(x), \boldsymbol{c_2}(x) \in \mathbb{R}^d$ and $c_3(x) \in \mathbb{R}$.


(a) General $\mathcal{H}$-partitioning

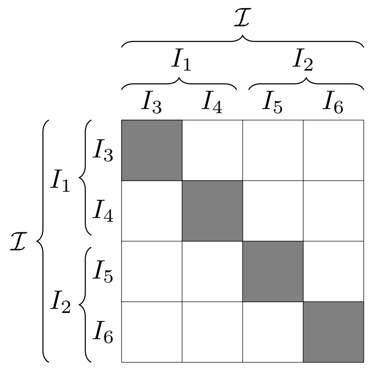

(b) BLR partitioning

FIG. 1. *BLR partitioning: a very particular $\mathcal{H}$-partitioning.*

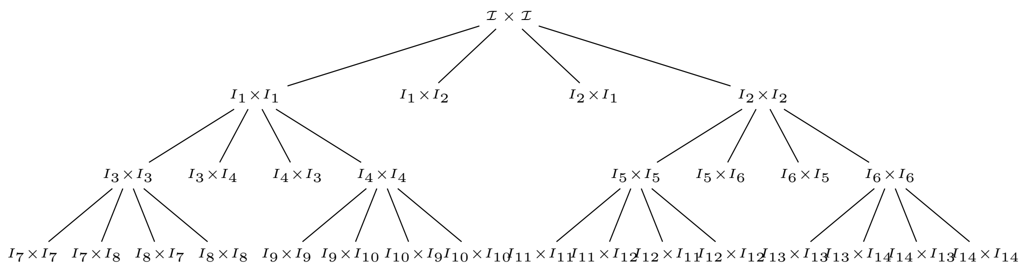


FIG. 2. *Block Cluster Tree associated with the $\mathcal{H}$-partitioning of Figure 1(a)*

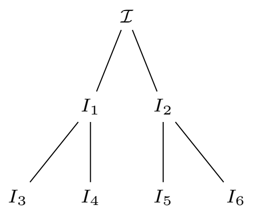


FIG. 3. *Row and column cluster tree associated to the BLR partitioning of Figure 1(b)*

We consider the resolution of problem (4) by the Finite Element (FE) method. Let $\mathcal{D} = H_0^1(\Omega)$ be the domain of definition of operator L. We consider a FE discretization, with step size h, that defines the associated approximation of $\mathcal{D}$, the space $\mathcal{D}_h$. Let $n = N^d = \dim \mathcal{D}_h$ be its dimension and $\{\varphi_i\}_{i\in\mathcal{I}}$ the basis functions, with $\mathcal{I} = [1, n]$ the index set. Similarly as in Hackbusch & Bebendorf [13], we assume that a quasi-uniform and shape-regular triangulation is used. We define X_i, the support of φ_i, and generalize the definition of support to subdomains:

$$X_\sigma = \bigcup_{i\in\sigma} X_i$$

We note J the bijection defined by:

$$J : \begin{array}{ccc} \mathbb{R}^n & \to & \mathcal{D}_h \\ x & \mapsto & \sum_{i\in\mathcal{I}} x_i\varphi_i \end{array}$$

To compute an approximated solution of Equation (4), we solve the discretized problem (1) where A is the stiffness matrix defined by $A = J^*LJ$. We assume that (1) is solved using the multifrontal method to factorize A. We also define $B = J^* L^{-1} J$ and $M = J^* J$. B is the Galerkin discretization of L^{-1} and M the mass matrix.

A matrix of the form

$$S = A_{\Psi\Psi} - A_{\Psi\Phi} A_{\Phi\Phi}^{-1} A_{\Phi\Psi} \tag{5}$$

for some $\Phi, \Psi \subset \mathcal{I}$ such that $\Phi \cup \Psi = \mathcal{I}$ is called a Schur complement of A. One of the main results of Bebendorf ([11], Section 3) states that the Schur complements of A can be approximated if an approximant of the inverse stiffness matrix A^{-1} is known.

Therefore, we are interested in finding $\widetilde{A}^{-1}$, approximant of the inverse stiffness matrix A^{-1}. The following result from FE theory will be used ([13], Subsection 5.2): the discretization of the inverse of the operator is approximated by the inverse of the discretized operator, i.e.,

$$\|A^{-1} - M^{-1} B M^{-1}\|_2 \leq O(\varepsilon_h) \tag{6}$$

where ε_h is the accuracy associated with the step size h of the FE discretization. In the following, for the sake of simplicity, we assume that the low-rank threshold ε is set to be equal to ε_h.

Then, assuming we can find $\widetilde{M}^{-1}$ and $\widetilde{B}$, approximants of the inverse mass matrix M^{-1} and of the B matrix, we have ([13], Subsection 5.3):

$$\begin{aligned} M^{-1}BM^{-1} \; - \; \widetilde{M}^{-1}\widetilde{B}\widetilde{M}^{-1} \; = \; & (M^{-1} - \widetilde{M}^{-1})BM^{-1} \\ & + \; \widetilde{M}^{-1}(B - \widetilde{B})M^{-1} \; + \; \widetilde{M}^{-1}\widetilde{B}(M^{-1} - \widetilde{M}^{-1}) \end{aligned} \tag{7}$$

Thus $M^{-1}BM^{-1}$ can be approximated by $\widetilde{M}^{-1}\widetilde{B}\widetilde{M}^{-1}$ and therefore so can A^{-1}.

2.3. Block-admissibility condition. In the following, we will use a key concept called the *admissibility* condition. The block-admissibility condition determines whether a block $\sigma \times \tau$ is admissible for low-rank compression. The standard block-admissibility condition, also called *strong* block-admissibility, is the following:

$$\sigma \times \tau \text{ is admissible } \Leftrightarrow \max(\operatorname{diam}(X_\sigma), \operatorname{diam}(X_\tau)) \leq \eta \operatorname{dist}(X_\sigma, X_\tau) \tag{Adm_b^s}$$

where $\eta > 0$ is a fixed parameter. Condition (Adm_b^s) formalizes the intuition that the rank of a block $\sigma \times \tau$ is correlated to the distance between X_σ and X_τ: the greater the distance, the weaker the interaction, the smaller the rank; that distance is to be evaluated relatively to the subdomain diameters.

The η parameter controls how strict we are in considering a block admissible. The smaller the η, the fewer admissible blocks. On the contrary, if we choose

$$\eta_{max} = \max_{\substack{\sigma,\tau \in \mathfrak{S}(\mathcal{I}) \\ \operatorname{dist}(X_\sigma, X_\tau) > 0}} \frac{\max(\operatorname{diam}(X_\sigma), \operatorname{diam}(X_\tau))}{\operatorname{dist}(X_\sigma, X_\tau)} \tag{8}$$

then condition (Adm_b^s) can be simplified in the following condition, that we call *least-restrictive* strong block-admissibility:

$$\sigma \times \tau \text{ is admissible } \Leftrightarrow \operatorname{dist}(X_\sigma, X_\tau) > 0 \tag{Adm_b^{lrs}}$$

Finally, there is an even less restrictive admissibility condition, called *weak* block-admissibility:

$$\sigma \times \tau \text{ is admissible } \Leftrightarrow \sigma \neq \tau \tag{Adm_b^w}$$

With the weak admissibility, even blocks that correspond to neighbors (subdomains at distance zero) are admissible, as long as they are not self-interactions (i.e., the diagonal blocks).

The proofs in [13], on which this paper is based, rely on the strong admissibility. In [27], it is shown that using the weak block-admissibility condition instead leads to a smaller constant in the complexity estimates. The extension to the weak admissibility condition in the BLR case is out of the scope of this paper. Therefore, we assume that a strong block-admissibility condition is used for computing our theoretical complexity bounds, that we will simply note (Adm_b).

Note that in Figure 1, the $\mathcal{H}$ and BLR partitionings were illustrated for the weak admissibility case, for the sake of simplicity.

3. From Hierarchical to BLR bounds.

The existence of $\mathcal{H}$-matrix approximants of the Schur complements of A has been shown in [13, 11]. In this section, we summarize the main ideas of the proof and give the necessary ingredients to extend it to the BLR case. The reader can refer to Hackbusch & Bebendorf [13, 10, 11] for the details of the proof for hierarchical matrices.

3.1. $\mathcal{H}$-admissibility and properties.

The admissibility of a partition can now be defined based on the block-admissibility condition. In the case of hierarchical matrices, the $\mathcal{H}$-admissibility condition is defined as:

$$\mathfrak{S}(\mathcal{I}\times\mathcal{I}) \text{ is admissible } \Leftrightarrow \forall \sigma\times\tau\in\mathfrak{S}(\mathcal{I}\times\mathcal{I}),\quad \begin{array}{l}(Adm_b) \text{ is satisfied}\\ \text{or } \min(\#\sigma,\#\tau)\le c_{min}\end{array} \qquad (Adm_{\mathcal{H}})$$

where c_{min} is a positive constant. The partitioning associated with the $\mathcal{H}$-admissibility condition ($Adm_{\mathcal{H}}$) can thus roughly be obtained by the following algorithm: for a given initial partition, for each block $\sigma\times\tau$, if (Adm_b) is satisfied, the block is admissible and is added to the final partition; if not, the block is subdivided, until either (Adm_b) is satisfied or the block becomes small enough. This often leads to non-uniform partitionings, such as the example shown on Figure 1(a).

We note $\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r)$ the set of hierarchical matrices such that r is the maximal rank of the blocks defined by the admissible partition $\mathfrak{S}(\mathcal{I}\times\mathcal{I})$.

In Hackbusch & Bebendorf [13, 10, 11], the proof that the Schur complements of A possess $\mathcal{H}$-approximants is derived using (6).

It is first established that B and M^{-1} possess $\mathcal{H}$-approximants ([13], Theorems 3.4 and 4.3). More precisely, they can be approximated with accuracy ε by $\mathcal{H}$-matrices $\widetilde{B}$ and $\widetilde{M}^{-1}$ such that

$$\widetilde{B}\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r_G) \qquad (9)$$

$$\widetilde{M}^{-1}\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), |\log\varepsilon|^d) \qquad (10)$$

where $\mathfrak{S}(\mathcal{I}\times\mathcal{I})$ is an $\mathcal{H}$-admissible partition and r_G is the rank resulting from the approximation of the degenerate Green function's kernel. r_G can be shown to be small for many problem classes [13, 10].

Then, the following $\mathcal{H}$-arithmetics theorem is used.

Theorem 1 ($\mathcal{H}$-matrix product, Theorem 2.20 in Grasedyck & Hackbusch [25]). *Let H_1 and H_2 be two hierarchical matrices of order n, such that $H_1\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r_1)$ and $H_2\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r_2)$. Then, their product is also a hierarchical matrix and it holds:*

$$H_1H_2\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), c_{sp}\max(r_1,r_2)\log n)$$

In Theorem 1, c_{sp} is the sparsity constant, defined by (3).

Then, using the fact that $r_G > |\log\varepsilon|^d$ [13], and applying (6), (7), and Theorem 1, it is established ([13], Theorem 5.4) that

$$\widetilde{A}^{-1}\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r_{\mathcal{H}}),\quad \text{with } r_{\mathcal{H}} = c_{sp}^2 r_G\log^2 n \qquad (11)$$

Furthermore, if an approximant $\widetilde{A}^{-1}$ exists, then for any $\Phi\subset\mathcal{I}$, an approximant of $A_{\Phi\Phi}^{-1}$ must also exist, since $A_{\Phi\Phi}$ is simply the restriction of A to the subdomain X_Φ [11].

Thus, using (5), in combination to the fact that the stiffness matrix A can also be approximated by $\widetilde{A}\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), O(1))$, the existence of $\widetilde{S}$, $\mathcal{H}$-approximant of any Schur complement S of A, is guaranteed by (11) and it is shown [11] that the maximal rank of the blocks of $\widetilde{S}$ is $r_{\mathcal{H}}$, i.e.

$$\widetilde{S}\in\mathcal{H}(\mathfrak{S}(\mathcal{I}\times\mathcal{I}), r_{\mathcal{H}}) \qquad (12)$$

Finally, it can be shown that the complexity of factorizing an $\mathcal{H}$-matrix of order m and of maximal rank r is [25, 26]:

$$\mathcal{C}(m) = O(c_{sp}^2 r^2 m\log^2 m) \qquad (13)$$

Equation (13) relies on the assumption that the factorization is fully-structured, i.e. the compressed form $\widetilde{A}$ of A is available at no cost.

To conclude, in the $\mathcal{H}$ case, applying (13) to the (dense) factorization of $\widetilde{S}$ leads to a cost which is almost linear when $r = O(1)$ and almost in $O(mN^2)$ when $r = O(N)$. As will be explained in Section 5, both cases lead to near-linear complexity of the multifrontal (sparse) factorization [39].

3.2. Why this result is not suitable to compute a complexity bound for BLR. One might think that, since BLR is a specific type of $\mathcal{H}$-matrix, the previous result can be used to derive the complexity of the BLR factorization. However, the bound obtained by simply applying $\mathcal{H}$-matrix theory to BLR is useless, as explained below.

Applying the result on $\mathcal{H}$-matrices to BLR is equivalent to bounding *all the ranks* k_{ij}^{ε} by *the same bound* r, the maximal rank. The problem is that this necessarily implies $r = b$, because there will always be some blocks of size b such that $\mathrm{dist}(X_\sigma, X_\tau) = 0$ (i.e., non-admissible blocks, which will be considered full-rank). Thus, the best we can say about a BLR matrix is that it belongs to $\mathcal{H}(\mathfrak{S}(\mathcal{I}), b)$, which is obvious and overly pessimistic.

In addition, with a BLR partitioning, the sparsity constant c_{sp} (defined by (3)) is not bounded, as it is equal to $p = m/b$. Thus, (13) leads to a factorization complexity bound in $O((m/b)^2 b^2 m \log^2 m) = O(m^3 \log^2 m)$, even worse than the full-rank factorization.

3.3. BLR-admissibility and properties. To compute a meaningful complexity bound for BLR, we divide the BLR blocks into two groups: the blocks who satisfy the block-admissibility condition (whose rank r can be bounded by a meaningful bound), and those who do not, which we assume are left in full-rank form. We show that the number of non-admissible blocks in A can be asymptotically negligible, provided an appropriate partitioning $\mathfrak{S}(\mathcal{I})$. This leads us to introduce the notion of BLR-admissibility of a partition $\mathfrak{S}(\mathcal{I})$, and we establish for such a partition a bound on the maximal rank of the admissible blocks.

In the following, we note $\mathcal{B}_A$ the set of admissible blocks. We also define

$$N_{na} = \max_{\sigma \in \mathfrak{S}(\mathcal{I})} \#\{\tau \in \mathfrak{S}(\mathcal{I}), \sigma \times \tau \notin \mathcal{B}_A\} \tag{14}$$

the maximum number of non-admissible blocks on any row. Note that, because we have assumed for simplicity that the row and column partitioning are the same, N_{na} is also the maximum number of non-admissible blocks on any column.

We then recast the $\mathcal{H}$-admissibility of a partition to the BLR uniform blocking. We propose the following BLR-admissibility condition:

$$\mathfrak{S}(\mathcal{I}) \text{ is admissible } \Leftrightarrow N_{na} \leq q \tag{Adm_{BLR}}$$

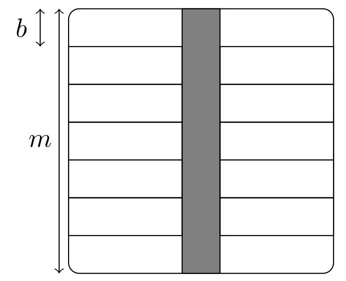
where q is a positive constant. With (Adm_{BLR}), we want the number of blocks (on any row or column) that are not admissible (and thus whose rank is not bounded by r), to be itself bounded by q.

For example, if the least-restrictive strong block-admissibility condition (Adm_b^{lrs}) is used, (Adm_{BLR}) means that a partition is admissible if for any subdomain, its number of neighbors (i.e. number of subdomains at distance zero) is smaller than q. The BLR-admissibility condition is illustrated in Figure 4, where we have assumed that (Adm_b^{lrs}) is used for simplicity. In Figure 4(a), the vertical subdomain (in gray) is at distance zero of $O(m/b)$ blocks and thus N_{na} is not constant. In Figure 4(b), the maximal number of blocks at distance zero of any block is at most 9 and thus the partition is BLR-admissible for $q \geq 9$. Note that if a general strong admissibility condition (Adm_b^s) is used, the same reasoning applies, as in Figure 4(b), N_{na} only depends on η and d, which are both constant.

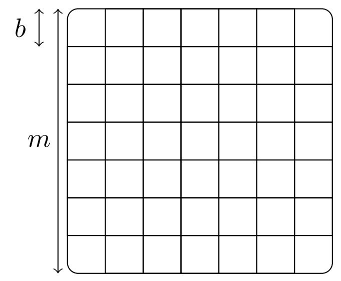
We note $\mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r, q)$ the set of BLR matrices such that r is the maximal rank of the admissible blocks defined by the BLR-admissible partition $\mathfrak{S}(\mathcal{I})$.

We now prove the following lemma.

LEMMA 2. *Let $\mathfrak{S}(\mathcal{I} \times \mathcal{I})$ be a given $\mathcal{H}$-partitioning and let $\mathfrak{S}(\mathcal{I})$ be the corresponding BLR partitioning obtained by refining the $\mathcal{H}$ one. Let us note $N_{na}^{(\mathcal{H})}$ and $N_{na}^{(BLR)}$ the value of N_{na} for the $\mathcal{H}$ and BLR partitionings, respectively. Then: (a) Provided $b \geq c_{min}$, it holds $N_{na}^{(BLR)} \leq N_{na}^{(\mathcal{H})}$;*



(a) Example of a non-BLR-admissible partition. Here, the gray subdomain has $N_{na} = O(m/b) \neq O(1)$ neighbors.



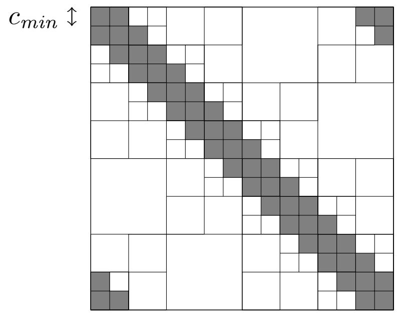
(b) Example of a BLR-admissible partition when $q \geq N_{na} = 9 = O(1)$, the maximal number of neighbors of any block.

FIG. 4. *Illustration of the BLR-admissibility condition.*

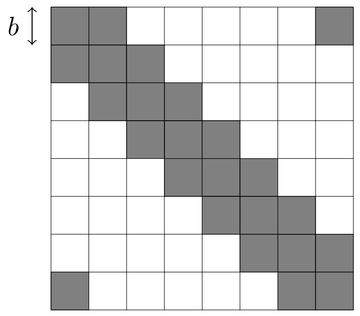
(b) Under the assumption that $\mathfrak{S}(\mathcal{I} \times \mathcal{I})$ is defined by a geometrically balanced block cluster tree, it holds $N_{na}^{(\mathcal{H})} = O(1)$.

Proof. (a) We provide Figure 5 (where non-admissible blocks are in gray) to illustrate the following proof. The BLR partitioning is simply obtained by refining the $\mathcal{H}$ one. Since non-admissible $\mathcal{H}$-blocks are of size $c_{min} \leq b$, they will not be refined, and thus the BLR refining only adds more admissible blocks to the partitioning. Furthermore, if c_{min} is strictly inferior to b, the non-admissible $\mathcal{H}$-blocks will be merged as a single BLR-block of size b and thus $N_{na}^{(BLR)}$ may in fact be smaller than $N_{na}^{(\mathcal{H})}$. (b) Since all non-admissible blocks necessarily belong to the same level of the block cluster tree (the last one), it holds by definition that $N_{na}^{(\mathcal{H})} \leq c_{sp}$. We conclude with the fact that in the $\mathcal{H}$ case, the sparsity constant is bounded for geometrically balanced block cluster trees [25]. ∎

As a corollary, we assume in the following that the partition $\mathfrak{S}(\mathcal{I})$ is defined by a geometrically balanced cluster tree and is thus admissible for $q = N_{na} = O(1)$.



(a) Example of $\mathcal{H}$-partitioning, with $N_{na}^{(\mathcal{H})} = 4 \leq c_{sp} = 6 = O(1)$. Here, we have assumed the bottom-left and top-right blocks are non-admissible for illustrative purposes.



(b) BLR refining of the $\mathcal{H}$-partitioning 5(a). $c_{sp} = O(m/b)$ is not bounded anymore but $N_{na}^{(BLR)} = 3 = O(1)$ remains constant.

FIG. 5. *Illustration of Lemma 2 (proof of the boundedness of N_{na}).*

The next step is to find BLR approximants of B and M^{-1}. In the Appendix, we give the constructions (35) and (36) of $\widetilde{B}$ and $\widetilde{M}^{-1}$, BLR approximants of B and M^{-1}, that satisfy:

$$\widetilde{B} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_G, N_{na}) \tag{15}$$

$$\widetilde{M}^{-1} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), 0, N_{na}) \tag{16}$$

(15) and (16) are the BLR equivalents of (9) and (10), respectively. It now remains to derive a BLR arithmetic property similar to Theorem 1.

In the Appendix, we prove the following theorem.

Theorem 3 (BLR matrix product). *If $A \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_A, q_A)$ and $B \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_B, q_B)$ are BLR matrices then their product $P = AB$ is a BLR matrix such that*

$$P \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_P, q_P)$$

with $r_P = c_{sp} \min(r_A, r_B) + q_A r_B + q_B r_A$ and $q_P = q_A q_B$.

Note that the sparsity constant c_{sp} is not bounded but only appears in the term $c_{sp} \min(r_A, r_B)$ that will disappear when one of r_A or r_B is zero.

Since A^{-1} can be approximated by $M^{-1}BM^{-1}$ (equation (6)), applying Theorem 3 on (15) and (16) leads to

$$\widetilde{A}^{-1} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), N_{na}^2 r_G, N_{na}^3) \tag{17}$$

and from this approximant of A^{-1} we can derive an approximant of $A_{\Phi\Phi}^{-1}$ for any $\Phi \subset \mathcal{I}$.

The stiffness matrix A satisfies the following property:

$$\forall \sigma, \tau \in \mathfrak{S}(\mathcal{I}),\ A_{\sigma\tau} \neq 0 \Leftrightarrow \mathrm{dist}(\sigma, \tau) = 0$$

and thus $A \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), 0, N_{na})$. A fortiori, for any $\Phi, \Psi \subset \mathcal{I}$, we have $A_{\Phi\Psi} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), 0, N_{na})$.

Therefore applying Theorem 3 on (5) and (17) implies in turn:

$$\widetilde{S} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), N_{na}^4 r_G, N_{na}^5) \tag{18}$$

As a result, there are at most $N_{na}^5 = O(1)$ non-admissible blocks that are not considered low-rank candidates and are left full-rank.

The rest are low-rank and their rank is bounded by $N_{na}^4 r_G$. In addition to the bound r_G, which is already quite large [10], the constant N_{na}^4 can be very large. However, our bound is extremely pessimistic. In Section 7, we will experimentally validate that, in reality, the ranks are much smaller. Similarly, the bound N_{na}^5 on the number of non-admissible blocks is also very pessimistic.

In conclusion, the ranks are bound by $O(r_G)$, i.e. the BLR bound only differs from the hierarchical one by a constant.

In the following, the bound $N_{na}^4 r_G$ will be simply referred to as r.

4. Complexity of the dense BLR factorization.

We first present the standard (dense) BLR factorization algorithm in Subsection 4.1. We then compute the dense BLR complexity in Subsection 4.2. We will extend the computation of the complexity to the sparse multifrontal case in Subsection 5.2.

Note that, as long as a bound on the ranks holds, similar to the one we have established in Section 3, the complexity computations reported in this section hold, and thus, the following results may be applicable to a broader context than the resolution of discretized PDEs.

In Algorithm 1, operations on non-admissible blocks are omitted for the sake of simplicity (but are taken into account in the complexity computations).

4.1. Standard BLR factorization.

In order to perform the LU or LDL^T factorization of a BLR matrix, the standard block LU or LDL^T factorization has to be modified so that the low-rank subblocks can be exploited to perform fast operations. Many such algorithms can be defined depending on where the compression step is performed. We present, in Algorithm 1, a version where the compression is performed after the so-called Solve step.

As described in detail in [1], this algorithm is fully compatible with threshold partial pivoting. The pivots are selected inside the BLR blocks; to assess their quality, they are compared to the pivots of the entire column. Therefore, in practice, to perform numerical pivoting, the Solve step is merged with the Factor step and done in full-rank (i.e. before the Compress). The pivots that are too small are delayed to the next BLR block, with a mecanism similar to the delayed pivoting

Algorithm 1 Dense BLR LDL^T (left-looking) factorization: standard UFSC variant.

```
1: {Input: a $p \times p$ block matrix $F$ of order $m$; $F = [F_{ij}]_{i=1:p,j=1:p}$}
2: for $k = 1$ to $p$ do
3:   for $i = k$ to $p$ do
4:     for $j = 1$ to $k - 1$ do
5:       Update $F_{ik}$:
6:       Inner Product: $\widetilde{C}_{ik}^{(j)} \leftarrow X_{ij}(Y_{ij}^T D_{jj} Y_{kj}) X_{kj}^T$
7:       Outer Product: $C_{ik}^{(j)} \leftarrow \widetilde{C}_{ik}^{(j)}$
8:       $F_{ik} \leftarrow F_{ik} - C_{ik}^{(j)}$
9:     end for
10:  end for
11:  Factor: $F_{kk} = L_{kk} D_{kk} L_{kk}^T$
12:  for $i = k + 1$ to $p$ do
13:    Solve: $F_{ik} \leftarrow F_{ik} L_{kk}^{-T} D_{kk}^{-1}$
14:  end for
15:  for $i = k + 1$ to $p$ do
16:    Compress: $F_{ik} \approx \widetilde{F}_{ik} = X_{ik} Y_{ik}^T$
17:  end for
18: end for
```

between panels (and fronts) in the full-rank case. Note that pivoting may slightly perturb the initial clustering of the variables shown in Equation (2). These details are omitted in Algorithm 1 for the sake of clarity.

This algorithm will be referred to as UFSC (standing for Update, Factor, Solve, and Compress), to indicate the order in which the steps are performed. Note that UFSC is the left-looking equivalent of the algorithm presented in Amestoy et al. [1]. Thanks to the flexibility of the BLR format, it is possible to easily define two other variants, CUFS and FSUC, which target different objectives [1]. In particular, we will also study in Section 6 the complexity of the CUFS variant, where the compression is performed earlier, before the solve, leading to a further use of the low-rank property of the blocks.

We recall that we denote the low-rank form of a block B by $\widetilde{B}$. Thus, the Outer Product on line 7 consists in decompressing the low-rank block $\widetilde{C}_{ik}^{(j)}$ into the corresponding full-rank block $C_{ik}^{(j)}$.

We present Algorithm 1 and its variants in their LDL^T version, but they can be easily adapted to the unsymmetric case. Note that the complexity of the BLR factorization is the same in LU or LDL^T, up to a constant.

4.2. Computation of the dense BLR complexity. First, we compute the complexity of factorizing a dense frontal matrix of order m. The cost of the main steps Factor, Solve, Compress, Inner and Outer Product necessary to compute the factorization of a matrix of order m are shown in Table 1 (third column). This cost depends on the type (full-rank or low-rank) of the block(s) on which the operation is performed (second column). Note that the Inner Product operation can take the form of a product of two full-rank blocks (FR-FR), two low-rank blocks (LR-LR) or a low-rank block and a full-rank one (LR-FR). In the last two cases, the Inner Product yields a low-rank block that is decompressed by means of the Outer Product operation. We note b the block size and $p = m/b$ the number of blocks per row and/or column.

We assume here that the cost of compressing an admissible block is $O(b^2 r)$. For example, this is true when the Compress is performed by means of a truncated Rank Revealing QR (RRQR) factorization (in our case, a QR factorization with column pivoting which is stopped when the diagonal coefficient of R falls below the prescribed threshold) which, as explained in Subsection 7.1, will be used in our numerical experiments. This assumption does not hold for Singular Value Decompostion (SVD) for which the Compress cost is $O(b^3)$; however, this would not change the final complexity of this standard variant, as the complexity of the Compress step would then be

step	type	cost	number	$\mathcal{C}_{step}(b,p)$	$\mathcal{C}_{step}(m,x)$
Factor	FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Solve	FR-FR	$O(b^3)$	$O(p^2)$	$O(p^2b^3)$	$O(m^{2+x})$
Compress	LR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
Inner Product	LR-LR	$O(br^2)$	$O(p^3)$	$O(p^3br^2)$	$O(m^{3-2x}r^2)$
	LR-FR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
	FR-FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Outer Product	LR	$O(b^2r)$	$O(p^3)$	$O(p^3b^2r)$	$O(m^{3-x}r)$

Table 1

Main operations for the BLR (standard UFSC variant) factorization of a dense matrix of order m, with blocks of size b, and low-rank blocks of rank at most r. We note $p = m/b$. type: type of the block(s) on which the operation is performed. cost: cost of performing the operation once. number: number of times the operation is performed. $\mathcal{C}_{step}(b,p)$: obtained by multiplying the cost and number columns (equation (19)). $\mathcal{C}_{step}(m,x)$: obtained with the assumption that $b = O(m^x)$ (and thus $p = O(m^{1-x})$), for some $x \in [0,1]$.

of the same order as that of the Solve step.

We can then use (18) to compute the cost of the factorization. The boundedness of $N^5_{na} = O(1)$ ensures that only a constant number of blocks on each line are full-rank. From that we derive the fourth column of Table 1, which counts the number of blocks on which the step is performed.

The BLR factorization cost of each step is then equal to

$$\mathcal{C}_{step}(b,p) = cost_{step} * number_{step} \tag{19}$$

and is reported in the fifth column of Table 1. Then, we assume the block size b is of order $O(m^x)$, where x is a real value in $[0,1]$, and thus the number of blocks p per row and/or column is of order $O(m^{1-x})$. Then by substituting b and p by their value, we compute $\mathcal{C}_{step}(m,x)$ in the last column.

We can then compute the total flop complexity of the dense BLR factorization as the sum of the cost of all steps:

$$\mathcal{C}(m,x) = O(rm^{3-x} + m^{2+x}) \tag{20}$$

Similarly, the factor size complexity of a dense BLR matrix can be computed as

$$O(N_{LR} * br + N_{FR} * b^2) = O(p^2br + N^5_{na}pb^2) = O(p^2br + pb^2) \tag{21}$$

where $N_{LR} = O(p^2)$ and $N_{FR} = O(p)$ are the number of low-rank and full-rank blocks in the matrix, respectively. Thus, the factor size complexity is:

$$\mathcal{M}(m,x) = O(rm^{2-x} + m^{1+x}) \tag{22}$$

It then remains to compute the optimal x^* which minimizes the complexity. We consider a general rank bound $r = O(m^\alpha)$, with $\alpha \in [0,1]$. Equations (20) and (22) become

$$\mathcal{C}(m,x) = O(m^{3+\alpha-x} + m^{2+x}) \tag{23}$$

$$\mathcal{M}(m,x) = O(m^{2+\alpha-x} + m^{1+x}) \tag{24}$$

respectively. Then, the optimal x^* is given by

$$x^* = \frac{1+\alpha}{2} \tag{25}$$

which leads to optimal complexities

$$\mathcal{C}(m) = \mathcal{C}(m,x^*) = O(m^{2.5+\alpha/2}) \tag{26}$$

$$\mathcal{M}(m) = \mathcal{M}(m,x^*) = O(m^{1.5+\alpha/2}) \tag{27}$$

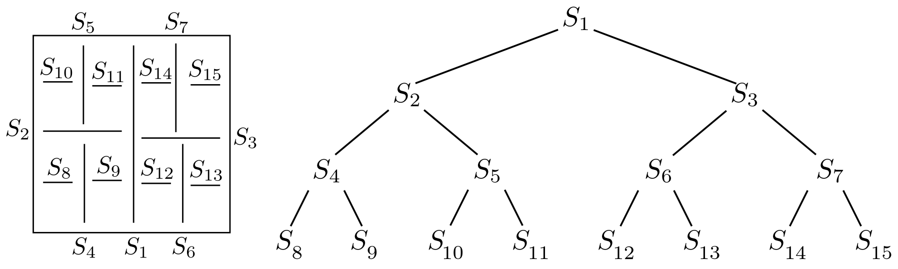


FIG. 6. *A general nested dissection and its corresponding separator tree.*

It is remarkable that the value of x^* is the same for both the flop and factor size complexities, i.e. that both complexities are minimized by the same x. This was not guaranteed, and is a desirable property as we do not need to choose which complexity to minimize at the expense of the other.

In particular, the case $r = O(1)$ leads to complexities in $O(m^{2.5})$ for flops and $O(m^{1.5})$ for factor size, while the case $r = O(\sqrt{m})$ leads to $O(m^{2.75})$ for flops and $O(m^{1.75})$ for factor size. The link between dense and sparse rank bounds will be made in Section 5.2.

Note that the fully-structured BLR factorization (when A is available under compressed form at no cost, i.e. when the Compress step does not need to be performed) has the same complexity as the non-fully-structured factorization, since the Compress is asymptotically negligible with respect to the Solve step. This is not the case in the hierarchical case, where the construction of the compressed matrix, whose cost is in $O(m^2 r)$ [25], becomes the bottleneck when it has to be performed.

5. From dense to sparse BLR complexity.

We first describe in Subsection 5.1 how the BLR clustering is computed in the context of the multifrontal method and the relation between frontal matrices and BLR approximants of the Schur complements of the stiffness matrix A. Then, we extend in Subsection 5.2 the computation of the factorization complexity to the sparse multifrontal case.

5.1. BLR clustering and BLR approximants of frontal matrices.

In a multifrontal context, the computation of the BLR clustering strongly depends on how the assembly tree is built, since the separator ordering adds some constraints to the BLR partitioning (specifically, variables from different separators cannot be regrouped in the same BLR cluster).

We will assume that the assembly tree is built by means of a nested dissection [21]; this better suits the context of our work and allows for an easier understanding of how low-rank approximation techniques can be used within sparse multifrontal solvers. Nested dissection divides the adjacency graph into two *domain* subgraphs separated by a third *separator* subgraph (S_1 in Figure 6). The process is then recursively applied to the two domain subgraphs until the domain subgraphs become too small to be subdivided again. This generates a separator tree, as illustrated in Figure 6.

Each frontal matrix is associated with a separator in the tree. The fully-summed variables of a frontal matrix match the variables of the separator. The non fully-summed variables of a front form a *border* of the separator's subtree and correspond to pieces of *ancestor* separators found higher in the separator tree.

Thanks to the bottom-up traversal of the assembly tree, the rows and columns of the fully-summed variables of a frontal matrix associated to a separator S thus belong to the Schur complement of the variables of the two domain subgraphs separated by S. From this and the existence of low-rank approximants of the Schur complements of the stiffness matrix A, which was established in Section 3, it results that the fronts can be approximated by BLR matrices. Algorithm 1 can easily be adapted to the partial factorization of the fronts.

The admissibility condition ($\mathcal{H}$ or BLR) requires geometric information to compute the diameter and distances. To remain in a purely algebraic context, we use the adjacency graph of the matrix A instead. The BLR clustering is computed with a k-way partitioning of each separator subgraph. A detailed description can be found in Amestoy et al [1]. An example of BLR clus-

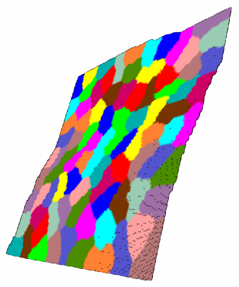


FIG. 7. *BLR clustering obtained on the root separator of a Poisson 128^3 problem*

tering obtained with this method is shown in Figure 7. In particular, we note that the clustering respects the BLR-admissibility condition (Adm_{BLR}).

5.2. Computation of the sparse BLR multifrontal complexity. The BLR multifrontal complexity in the sparse case can be directly derived from the dense complexity.

The flop and factor size complexities $\mathcal{C}_{MF}(N)$ and $\mathcal{M}_{MF}(N)$ of the BLR multifrontal factorization are reported in Table 2. We assume a nested dissection ordering [21] (with separators in cross shape).

At each level ℓ of the separators tree, we need to factorize $(2^d)^\ell$ fronts of order $O((\frac{N}{2^\ell})^{d-1})$, for ℓ ranging from 0 to $L = \log_2(N)$. Therefore, the flop complexity $\mathcal{C}_{MF}(N)$ to factorize a sparse matrix of order N^d is

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} \mathcal{C}_\ell(N) = \sum_{\ell=0}^{L} (2^d)^\ell \mathcal{C}((\frac{N}{2^\ell})^{d-1}, x_\ell^*) \tag{28}$$

where $\mathcal{C}_\ell(N)$ is the cost of factorizing all the fronts on the ℓ-th level, i.e. $\mathcal{C}_\ell(N) = (2^d)^\ell \mathcal{C}(m_\ell, x_\ell^*)$ with $m_\ell = (\frac{N}{2^\ell})^{d-1}$. x_ℓ^* is the optimal choice of x_ℓ at the ℓ-th level. Using the dense complexity equation (20), we compute $\mathcal{C}_\ell(m_\ell, x_\ell^*)$ and report the corresponding value of $\mathcal{C}_\ell(N)$ in Table 2 for two cases, $r = O(1)$ and $r = O(N)$ (third and sixth columns, respectively).

The case $r = O(1)$ is equivalent to $\forall \ell, r = O(m_\ell^{\alpha_\ell})$, with $\forall \ell, \alpha_\ell = 0$ and thus (25) yields

$$\forall \ell,\ x_\ell^* = x^* = \frac{1}{2} \tag{29}$$

Then, $\mathcal{C}_{MF}(N)$ can easily be computed as a geometric series of common ratio $2^{(5-3d)/2}$, and is given in the fourth column.

The case $r = O(N)$ is slightly more complex because α_ℓ, and thus x_ℓ^*, varies with the level ℓ. However, $\mathcal{C}_{MF}(N)$ remains a geometric series and its value is given in the last column. The details of the computation are provided in the appendix for the sake of readability.

Using (22), we similarly compute the factor size complexity:

$$\mathcal{M}_{MF}(N) = \sum_{\ell=0}^{L} \mathcal{M}_\ell(N) = \sum_{\ell=0}^{L} (2^d)^\ell \mathcal{M}((\frac{N}{2^\ell})^{d-1}, x_\ell^*) \tag{30}$$

and report the results in Table 2.

6. BLR variants and their complexity. We first describe in Subsection 6.1 and Algorithm 2 two BLR variants, LUAR and CUFS, whose aim is to further reduce the number of operations of the standard BLR factorization (Algorithm 1). We then compute their complexity in Subsection 6.2.

		$r = O(1)$			$r = O(N)$	
d	x^*	$\mathcal{C}_\ell(N)$	$\mathcal{C}_{MF}(N)$	x^*_ℓ	$\mathcal{C}_\ell(N)$	$\mathcal{C}_{MF}(N)$
2D	1/2	$O(2^{-\ell/2}N^{2.5})$	$O(N^{2.5})$	FR	FR	$O(N^3)$
3D	1/2	$O(2^{-2\ell}N^5)$	$O(N^5)$	$\frac{3L-2\ell}{4L-4\ell}$	$O(2^{-2\ell}N^{5.5})$	$O(N^{5.5})$
d	x^*	$\mathcal{M}_\ell(N)$	$\mathcal{M}_{MF}(N)$	x^*_ℓ	$\mathcal{M}_\ell(N)$	$\mathcal{M}_{MF}(N)$
2D	1/2	$O(2^{\ell/2}N^{1.5})$	$O(N^2)$	FR	FR	$O(N^2 \log N)$
3D	1/2	$O(N^3)$	$O(N^3 \log N)$	$\frac{3L-2\ell}{4L-4\ell}$	$O(N^{3.5})$	$O(N^{3.5} \log N)$

TABLE 2

Flop and factor size complexity of the BLR (standard UFSC variant) multifrontal factorization of a sparse matrix of order N^d. d: dimension. $\mathcal{C}_\ell(N)/\mathcal{M}_\ell(N)$: flop/factor size complexity at level $\ell < L$ in the separator tree, computed using the dense complexity equations (20) and (22). The case $\ell = L$ does not modify the overall complexity and can be ignored (see Appendix). $\mathcal{C}_{MF}(N)/\mathcal{M}_{MF}(N)$: total multifrontal flop/factor size complexity, computed using Equations (28) and (30). When a column indicates "FR", this means that our bounds are not good enough to compute a meaningful low-rank complexity.

6.1. BLR variants: LUAR and CUFS. The first variant of Algorithm 1, is referred to as *Low-rank Updates Accumulation and Recompression* (LUAR). It consists in accumulating the update matrices $\widetilde{C}_{ik}^{(j)}$ together, as shown on line 10 of Algorithm 2:

$$\widetilde{C}_{ik}^{(acc)} := \widetilde{C}_{ik}^{(acc)} \;+\; \widetilde{C}_{ik}^{(j)}$$

Note that in the previous equation, the + sign denotes a low-rank sum operation. Specifically, if we note $A = C_{ik}^{(acc)}$ and $B = C_{ik}^{(j)}$, then

$$\widetilde{B} = \widetilde{C}_{ik}^{(j)} = X_{ij}(Y_{ij}^T D_{jj} Y_{kj}) X_{kj}^T = X_B C_B Y_B^T$$

with $X_B = X_{ij}$, $C_B = Y_{ij}^T D_{jj} Y_{kj}$, and $Y_B = X_{kj}$. Similarly, $\widetilde{A} = \widetilde{C}_{ik}^{(acc)} = X_A C_A Y_A^T$. Then the low-rank sum operation is defined by:

$$\widetilde{A} \;+\; \widetilde{B} = X_A C_A Y_A^T \;+\; X_B C_B Y_B^T = (X_A \quad X_B)\begin{pmatrix} C_A & \\ & C_B \end{pmatrix}(Y_A \quad Y_B)^T = X_S C_S Y_S^T = \widetilde{S}$$

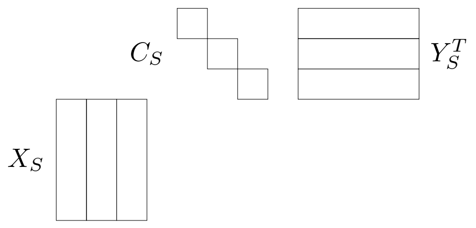
where $\widetilde{S}$ is a low-rank approximant of $S = A + B$.

This technique allows for additional compression, as the accumulated updates $\widetilde{C}_{ik}^{(acc)}$ can be recompressed (as shown on line 12) before the Outer Product. A visual representation is given in Figure 8. To compute the cost of the Recompress, we need to bound the rank of the accumulators X_S and Y_S. If the Compress is done with an SVD or RRQR operation, then each accumulated update in X_S and Y_S is an orthonormal basis of an admissible block. Thus, the accumulator is a basis of a superblock which is itself admissible (because the union of admissible blocks remains admissible) and thus the rank of the accumulator is bounded by r.

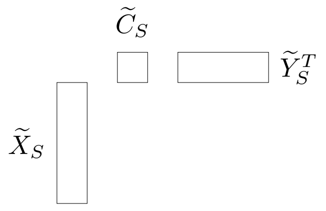
Thus, by recompressing the accumulators, we only need to do one Outer Product (of size r) per block, instead of $O(p)$ (one for each update matrix). This leads to a substantial theoretical improvement, as it lowers the cost of the Outer Product from $O(b^2r) * O(p^3) = O(p^3b^2r)$ to $O(b^2r) * O(p^2) = O(p^2b^2r)$ (see Table 3, column $\mathcal{C}_{step}(b,p)$), even though the recompressions of the accumulated updates (Recompress operation) introduce an overhead cost, equal to $O(pbr^2) * O(p^2) = O(p^3br^2)$.

Next, let us present the so-called CUFS variant. In this variant, we perform the Compress before the Solve. The Solve step can thus be performed on low-rank blocks (as shown on line 17) and its cost is thus reduced to $O(p^2b^2r + pb^3)$ (as reported in Table 3, column $\mathcal{C}_{step}(b,p)$).

Furthermore, we can combine the LUAR and CUFS variants. Since we perform the Solve in low-rank, we don't need to decompress the update matrices of the low-rank off-diagonal blocks. Thus, we can further reduce the cost of the factorization by keeping the recompressed accumulated



(a) Accumulated updates before recompression



(b) Accumulated updates after recompression

FIG. 8. *Low-rank Updates Accumulation*

Algorithm 2 Dense BLR LDL^T (left-looking) factorization: CUFS+LUAR variant.

```
{Input: a $p \times p$ block matrix $F$ of order $m$; $F = [F_{ij}]_{i=1:p,j=1:p}$}
for $k = 1$ to $p$ do
    for $i = k + 1$ to $p$ do
        Compress: $F_{ik} \approx \widetilde{F}_{ik} = X_{ik}Y_{ik}^T$
    end for
    for $i = k$ to $p$ do
        Update $\widetilde{F}_{ik}$:
        for $j = 1$ to $k - 1$ do
            Inner Product: $\widetilde{C}_{ik}^{(j)} \leftarrow X_{ij}(Y_{ij}^T D_{jj} Y_{kj}) X_{kj}^T$
            Accumulate update: $\widetilde{C}_{ik}^{(acc)} \leftarrow \widetilde{C}_{ik}^{(acc)} + \widetilde{C}_{ik}^{(j)}$
        end for
        $\widetilde{C}_{ik}^{(acc)} \leftarrow$ Recompress($\widetilde{C}_{ik}^{(acc)}$)
        $\widetilde{F}_{ik} \leftarrow \widetilde{F}_{ik} - \widetilde{C}_{ik}^{(acc)}$
    end for
    Factor: $F_{kk} = L_{kk}D_{kk}L_{kk}^T$
    for $i = k + 1$ to $p$ do
        Solve: $\widetilde{F}_{ik} \leftarrow \widetilde{F}_{ik} L_{kk}^{-T} D_{kk}^{-1} = X_{ik}(Y_{ik}^T L_{kk}^{-T} D_{kk}^{-1})$
    end for
end for
```

updates $\widetilde{C}_{ik}^{(acc)}$ as the low-rank representation of the block F_{ik}, and thus suppress the Outer Product step.

Note that the CUFS strategy requires an extension of pivoting strategies, because off-diagonal blocks are now in compressed form. Although one can simply pivot inside diagonal blocks and restrict the pivot stability check within diagonal blocks, a more promising approach would be to extend the pivoting strategy in order to take into account the low-rank off-diagonal blocks, as briefly mentioned in Section 8.2.

6.2. Complexity of the BLR variants. The computation of the complexity of the BLR variants is very similar to that of the standard version, computed in Section 4. We provide the equivalent of Tables 1 and 2 for the BLR variants in Tables 3 and 4, respectively.

In Table 3, we report the cost of each step of the factorization. These costs have been explained in the previous subsection. For UFSC+LUAR, the steps whose cost has changed with respect to the UFSC variant have been grayed out; similarly, for CUFS+LUAR, the steps whose cost has changed with respect to UFSC+LUAR have been grayed out.

By summing the cost of all steps, we obtain the flop complexity of the dense factorization. In the UFSC+LUAR variant, it is given by:

$$\mathcal{C}(m, x) = O(r^2 m^{3-2x} + m^{2+x}) \tag{31}$$

Compared to (20), the low-rank term of the complexity has thus been reduced from $O(rm^{3-x})$

UFSC+LUAR variant					
step	type	cost	number	$\mathcal{C}_{step}(b,p)$	$\mathcal{C}_{step}(m,x)$
Factor	FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Solve	FR-FR	$O(b^3)$	$O(p^2)$	$O(p^2b^3)$	$O(m^{2+x})$
Compress	LR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
Inner Product	LR-LR	$O(br^2)$	$O(p^3)$	$O(p^3br^2)$	$O(m^{3-2x}r^2)$
	LR-FR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
	FR-FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Recompress	LR	$O(bpr^2)$	$O(p^2)$	$O(p^3br^2)$	$O(m^{3-2x}r^2)$
Outer Product	LR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
CUFS+LUAR variant					
step	type	cost	number	$\mathcal{C}_{step}(b,p)$	$\mathcal{C}_{step}(m,x)$
Factor	FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Solve	FR-FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
	LR-FR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
Compress	LR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
Inner Product	LR-LR	$O(br^2)$	$O(p^3)$	$O(p^3br^2)$	$O(m^{3-2x}r^2)$
	LR-FR	$O(b^2r)$	$O(p^2)$	$O(p^2b^2r)$	$O(m^2r)$
	FR-FR	$O(b^3)$	$O(p)$	$O(pb^3)$	$O(m^{1+2x})$
Recompress	LR	$O(bpr^2)$	$O(p^2)$	$O(p^3br^2)$	$O(m^{3-2x}r^2)$
Outer Product	LR	—	—	—	—

TABLE 3

Main operations for the factorization of a dense matrix of order m, with blocks of size b, and low-rank blocks of rank at most r. We note $p = m/b$. type: type of the block(s) on which the operation is performed. cost: cost of performing the operation once. number: number of times the operation is performed. $\mathcal{C}_{step}(b,p)$: obtained by multiplying the cost and number columns (equation (19)). $\mathcal{C}_{step}(m,x)$: obtained with the assumption that $b = O(m^x)$ (and thus $p = O(m^{1-x})$), for some $x \in [0,1]$.

to $O(r^2m^{3-2x})$ thanks to the recompression of the accumulated updates. The full-rank term $O(m^{2+x})$ remains the same. By recomputing the value of x^*, we achieve flop complexity gains: $\mathcal{C}(m)$ becomes of order $O(m^{2+(2\alpha+1)/3})$ for $r = O(m^\alpha)$, which yields in particular $O(m^{2.33})$ for $r = O(1)$ and $O(m^{2.67})$ for $r = O(\sqrt{m})$.

In the same way, the flop complexity for the dense factorization with the CUFS+LUAR variant is given by:

$$\mathcal{C}(m,x) = O(r^2m^{3-2x} + m^{1+2x}) \tag{32}$$

This time, the full-rank term has been reduced from $O(m^{2+x})$ to $O(m^{1+2x})$. By recomputing x^*, we achieve further flop complexity gains: $\mathcal{C}(m)$ becomes of order $O(m^{2+\alpha})$ for $r = O(m^\alpha)$, which yields in particular $O(m^2)$ for $r = O(1)$ and $O(m^{2.5})$ for $r = O(\sqrt{m})$.

Note that for the CUFS+LUAR variant, the Compress step has become asymptotically dominant and thus the assumption that its cost is $O(b^2r)$ is now necessary to obtain the complexity reported in Equation (32).

Also note that the factor size complexity is not affected by the BLR variant used.

The sparse flop complexities are derived from the dense ones in the same way as they are for the standard UFSC variant. The results are reported in Table 4.

A summary of the sparse complexities for all BLR variants, as well as the full-rank and $\mathcal{H}$ complexities, is given in Tables 5 and 6, in the case $r = O(1)$ and $r = O(N)$, respectively.

7. Numerical experiments. In this section we compare the experimental complexity of the full-rank solver with each of the BLR variants previously presented (UFSC, UFSC+LUAR, CUFS+LUAR). We also discuss the choice of two parameters, the low-rank threshold ε and the block size b, and their impact on the complexity.

		$r = O(1)$			$r = O(N)$	
d	x^*	$\mathcal{C}_\ell(N)$	$\mathcal{C}_{MF}(N)$	x^*_ℓ	$\mathcal{C}_\ell(N)$	$\mathcal{C}_{MF}(N)$
			UFSC+LUAR			
2D	1/3	$O(2^{-\ell/3}N^{2.33})$	$O(N^{2.33})$	FR	FR	$O(N^3)$
3D	1/3	$O(2^{-5\ell/3}N^{4.67})$	$O(N^{4.67})$	$\frac{2L-\ell}{3L-3\ell}$	$O(2^{-5\ell/3}N^{5.33})$	$O(N^{5.33})$
			CUFS+LUAR			
2D	1/2	$O(N^2)$	$O(N^2 \log N)$	FR	FR	$O(N^3)$
3D	1/2	$O(2^{-\ell}N^4)$	$O(N^4)$	$\frac{3L-2\ell}{4L-4\ell}$	$O(2^{-\ell}N^5)$	$O(N^5)$

TABLE 4

Flop and factor size complexity of the BLR multifrontal factorization of a sparse matrix of order N^d. d: dimension. $\mathcal{C}_\ell(N)$: flop complexity at level $\ell < L$ in the separator tree, computed using the dense complexity equations (31) and (32) for the UFSC+LUAR and CUFS+LUAR variants, respectively. The case $l = L$ does not modify the overall complexity and can be ignored (see Appendix). $\mathcal{C}_{MF}(N)$: total multifrontal flop complexity, computed using Equation (28). When a column indicates "FR", this means that our bounds are not good enough to compute a meaningful low-rank complexity.

	operations		factor size	
	2D	3D	2D	3D
FR	$O(n^{1.5})$	$O(n^2)$	$O(n \log n)$	$O(n^{1.33})$
BLR UFSC	$O(n^{1.25})$	$O(n^{1.67})$	$O(n)$	$O(n \log n)$
BLR UFSC+LUAR	$O(n^{1.17})$	$O(n^{1.56})$	$O(n)$	$O(n \log n)$
BLR CUFS+LUAR	$O(n \log n)$	$O(n^{1.33})$	$O(n)$	$O(n \log n)$
$\mathcal{H}$	$O(n \log n)$	$O(n^{1.33})$	$O(n)$	$O(n)$
$\mathcal{H}$ (fully-structured)	$O(n)$	$O(n)$	$O(n)$	$O(n)$

TABLE 5

Flop and factor size complexities of the BLR multifrontal factorization of a system of n unknowns, considering the case $r = O(1)$ and an optimal choice of b. In the fully-structured case, the original matrix is assumed to be already compressed. In the non-fully-structured case, the cost of compressing the original matrix is included. We remind that the complexity of the BLR fully-structured factorization is the same as that of the non-fully-structured one.

7.1. Description of the experimental setting. The BLR standard factorization as well as its variants have been developed and integrated into the general purpose symmetric and unsymmetric sparse multifrontal solver `MUMPS` [4], which was used to run all experiments.

The BLR clustering was computed with a k-way partitioning using `METIS` [29].

All the experiments were performed on one module of a bullx supernode S6130, or MESCA2 node. The MESCA2 is a shared-memory machine equipped with 12 TB of memory and 128 Intel Xeon CPU E7-8867 v3 processors running at 2.50 GHz.

To compute our complexity estimates, we use least-squares estimation to compute the coefficients $\{\beta_i\}_i$ of a regression function f such that $X_{fit} = f(N, \{\beta_i\}_i)$ fits the observed data X_{obs}. We use the following regression function:

$$X_{fit} = e^{\beta_1^*} N^{\beta_2^*} \text{ with } \beta_1^*, \beta_2^* = \arg\min_{\beta_1,\beta_2} \| \log X_{obs} - \beta_1 - \beta_2 \log N\|^2 \quad (33)$$

Test problems. We provide the experimental complexities for two different problems: the Poisson problem and the Helmholtz problem.

For the Poisson problem, the rank bound is in $O(1)$ [13]. For the Helmholtz problem, although there is no rigorous proof of it, it has been shown it is reasonable to assume a rank bound in $O(N)$ [39, 38, 20]. Thus, we will use the Poisson and Helmholtz problems to experimentally validate the complexities computed in Table 5 and 6, respectively.

The Poisson problem generates the symmetric positive definite matrix A from a 7-point discretization. We perform the computations in double-precision arithmetic. We will use a low-

	operations		factor size	
	2D	3D	2D	3D
FR	$O(n^{1.5})$	$O(n^2)$	$O(n \log n)$	$O(n^{1.33})$
BLR UFSC	$O(n^{1.5})$	$O(n^{1.83})$	$O(n \log n)$	$O(n^{1.17} \log n)$
BLR UFSC+LUAR	$O(n^{1.5})$	$O(n^{1.78})$	$O(n \log n)$	$O(n^{1.17} \log n)$
BLR CUFS+LUAR	$O(n^{1.5})$	$O(n^{1.67})$	$O(n \log n)$	$O(n^{1.17} \log n)$
$\mathcal{H}$	$O(n^{1.5})$	$O(n^{1.67})$	$O(n \log n)$	$O(n^{1.17})$
$\mathcal{H}$ (fully-structured)	$O(n)$	$O(n^{1.33})$	$O(n \log n)$	$O(n^{1.17})$

Table 6

Flop and factor size complexities of the BLR multifrontal factorization of a system of n unknowns, considering the case $r = O(N)$, without rank relaxation*, and for an optimal choice of b. In the fully-structured case, the original matrix is assumed to be already compressed. In the non-fully-structured case, the cost of compressing the original matrix is included. We remind that the complexity of the BLR fully-structured factorization is the same as that of the non-fully-structured one.*

rank threshold ε varying from 10^{-14} to 10^{-2} to better understand its influence on the complexity, with no particular application in mind.

The Helmholtz problem builds the matrix A as the complex-valued unsymmetric impedance matrix resulting from the finite-difference discretization of the heterogeneous Helmholtz equation that is the second-order visco-acoustic time-harmonic wave equation for pressure p. The aim is the modeling of visco-acoustic wave propagation in a 3D visco-acoustic homogeneous medium parameterized by wavespeed (4000 m/s), density (1 kg/m^3), and quality factor (10000, no attenuation). The matrix A is built for an infinite medium. This implies that the input grid is augmented with PML absorbing layers. Frequency is fixed and equal to 4 Hz. The grid interval h is computed such that it corresponds to 4 grid point per wavelength. Computations are done in single-precision arithmetic. We will use a low-rank threshold ε varying from 10^{-5} to 10^{-3} because we know these are the values for which the result is meaningful for the application [2].

For both Poisson and Helmholtz, in all the following experiments, the backward error is in good agreement with the low-rank threshold used.

Both the Poisson and Helmholtz problem were discretized using the finite-difference method rather than the finite-elements one, but this is acceptable as both methods are equivalent on equispaced meshes [33].

Compression of the blocks and recompression of the accumulators. To compute the low-rank form of the blocks, we perform a truncated QR factorization with column pivoting (i.e., a truncated version of LAPACK's [8] `_geqp3` routine). The matrix is scaled using an algorithm based on those discussed in [35, 30]. We use an absolute tolerance on the scaled matrix (i.e. we stop the factorization after $|r_{kk}| < \varepsilon$).

Because of our purely algebraic context, we do not know which blocks are admissible and so we assume all off-diagonal blocks are admissible (which is equivalent to using a weak block-admissibility condition). Thus, in our experiments, we try to compress every off-diagonal block. If the prescribed accuracy ε is not achieved after a given number of steps k_{max}, we stop the compression and consider the block to be full-rank. In the following numerical experiments, we have set $k_{max} = b/2$, the rank after which the low-rank representation of a block is not beneficial anymore (in terms of both flops and memory) with respect to the full-rank one.

In Figure 8, it is worth noting that X_S, C_S, and Y_S can all be recompressed. In particular, when the compression is done with truncated QR, due to the form of the Inner Product operation on line 9 of Algorithm 2, both X_S and Y_S are formed of a set of orthonormal matrices, i.e. contain information based on collinearity only, while C_S contains all the norm information. In the following numerical experiments on the LUAR variant, we only recompress C_S, i.e. only exploit the norm information. This allows us to reduce the Recompress overhead cost while still achieving the desired complexity reduction.

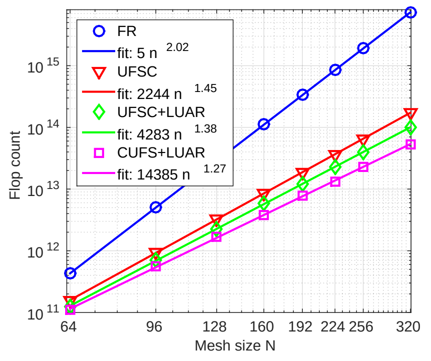
7.2. Experimental complexity of the multifrontal BLR factorization.

7.2.1. Flop complexity of each BLR variant. In Figures 9 and 10, we compare the flop complexity of the full-rank solver with each of the BLR variants previously presented (UFSC, UFSC+LUAR, CUFS+LUAR) for the Poisson problem, and the Helmholtz problem, respectively. The results show that each new variant improves the complexity. Note that we obtain the well-known quadratic complexity of the full-rank version. Results with both geometric nested dissection (Figures 9(a) and 10(a)) and with a purely algebraic ordering computed by `METIS` (Figures 9(b) and 10(b)) are also reported.

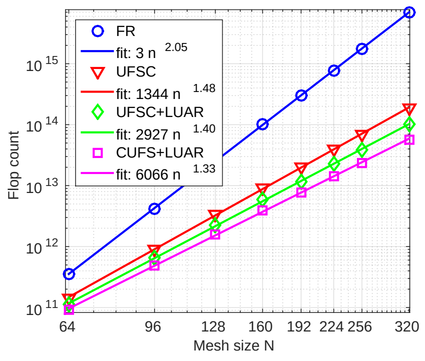
We first analyze the results obtained with geometric nested dissection and compare them with our theoretical results. For Poisson, the standard BLR (UFSC) version achieves a complexity in $O(n^{1.45})$. Moreover, the constant in the big O is equal to 2244, which is quite reasonable, and leads to a substantial improvement of the number of flops performed with respect to the full-rank version. This confirms that the theoretical rank bounds ($N_{na}^4 r_G$) are very pessimistic, as the experimental constants are in fact much smaller. Further compression in the UFSC+LUAR variant lowers the complexity to $O(n^{1.38})$, while the CUFS+LUAR reaches the lowest complexity of the variants, in $O(n^{1.27})$. Although the constants increase with the new variants, they also remain relatively small and they effectively reduce the number of operations with respect to the standard variant, even for the smaller mesh sizes. The same trend is observed for Helmholtz, with complexities in $O(n^{1.84})$ for UFSC, $O(n^{1.79})$ for UFSC+LUAR, and finally $O(n^{1.76})$ for CUFS+LUAR. Thus, the numerical results are in good agreement with the theoretical bounds reported in Tables 5 and 6. The difference between the theoretical and experimental Helmholtz complexity of the CUFS+LUAR variant can be explained by an imperfect block size setting. Indeed, as explained in the appendix, the block size setting is especially critical for this particular variant as for a non-adaptive x_ℓ^* (i.e. $\forall \ell,\ x_\ell^* = x^*$) the $O(n^{1.67})$ bound becomes $O(n^{1.67}\log n)$ (whereas it does not change for the other variants). When taking into account the log factor, the $O(n^{1.76})$ fitting becomes $O(n^{1.69}\log n)$, which is much closer to the theoretical bound.

We also analyze the influence of the ordering on the complexity. We observe that even though the `METIS` ordering slightly degrades the complexity, results remain close to the geometric nested dissection ordering and still in good agreement with the theoretical bounds. This is a very important property of the BLR factorization as it allows us to remain in a purely algebraic (black box) framework, an essential property for a general purpose solver.

For the remaining experiments, we use the `METIS` ordering.



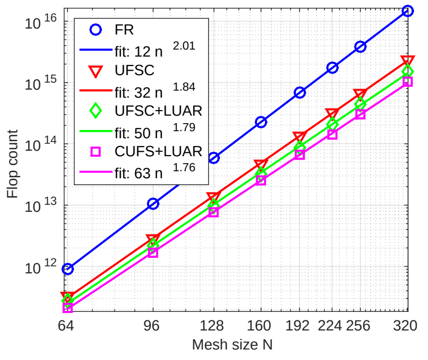
(a) Nested Dissection ordering (geometric)



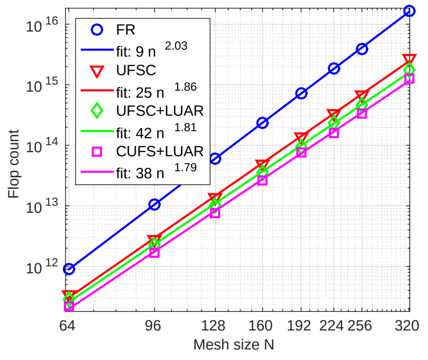
(b) `METIS` ordering (purely algebraic)

FIG. 9. *Flop complexity of each BLR variant (Poisson, $\varepsilon = 10^{-10}$)*

7.2.2. Factor size complexity. To compute the factor size complexity of the BLR solver, we study the evolution of the number of entries in the factors, i.e., the compression rate of L and U. Note that the global compression rate would be even better, because the local matrices that need to be stored during the multifrontal factorization compress more than the factors.



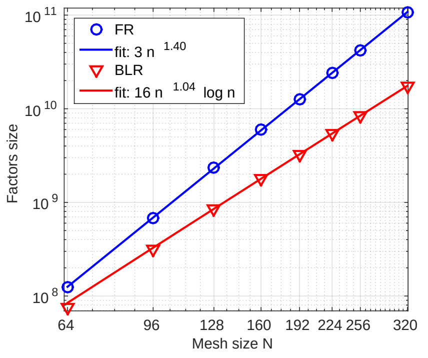
(a) Nested Dissection ordering (geometric)



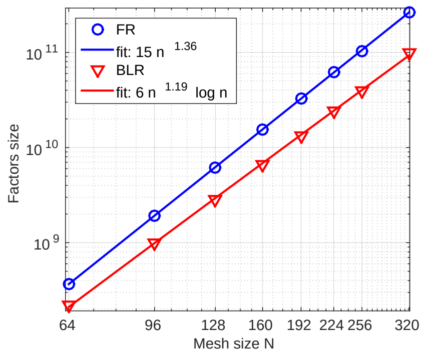
(b) `METIS` ordering (purely algebraic)

FIG. 10. *Flop complexity of each BLR variant (Helmholtz, $\varepsilon = 10^{-4}$)*

In Figure 11, we plot the factor size complexity using the `METIS` ordering for both the Poisson and Helmholtz problems. The different BLR variants do not impact the factor size complexity. Here again, the results are in good agreement with the bounds computed in Tables 5 and 6. The complexity is of order $O(n^{1.04} \log n)$ for Poisson and $O(n^{1.19} \log n)$ for Helmholtz.



(a) Poisson problem ($\varepsilon = 10^{-10}$)



(b) Helmholtz problem ($\varepsilon = 10^{-4}$)

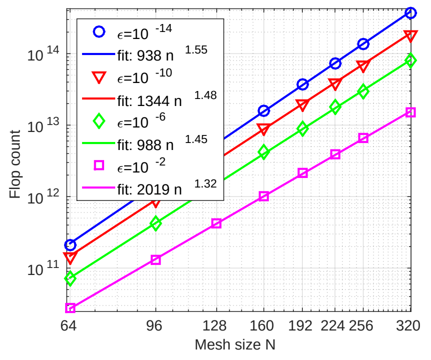
FIG. 11. *Factor size complexity with* `METIS` *ordering*

7.2.3. Low-rank threshold. The theory [13, 10], through the bound r_G, which increases as $|\log \varepsilon|^{d+1}$, states the threshold ε should only play a role in the constant factor of the complexity.

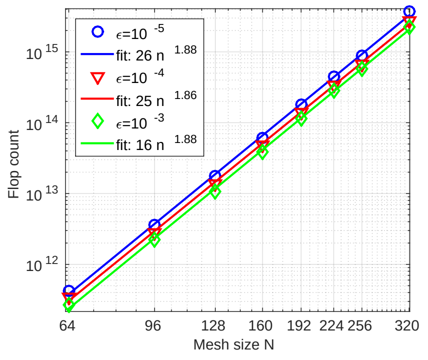
However, that is not exactly what the numerical experiments show. In Figure 12, we compare the complexity for different values of ε. For Helmholtz, the threshold does seem to play a role only in the constant factor, as the complexity exponent remains around $O(n^{1.86})$. However, for Poisson, an interesting trend appears: the complexity gradually lowers from $O(n^{1.55})$ to $O(n^{1.48})$, $O(n^{1.45})$ and $O(n^{1.32})$ with a threshold of 10^{-14}, 10^{-10}, 10^{-6}, and 10^{-2}, respectively.

Our analysis is that the complexity exponent is related to the processing of zero-rank blocks. With absolute tolerance, it is possible for blocks to have a numerical rank equal to zero. However, the bound on the ranks r_G is strictly positive, and thus the theory does not account for zero-rank blocks. This leads to a theoretical sparsity constant c_{sp} equal to $p = m/b$ (i.e. all blocks are considered nonzero-rank), while in fact the actual value of c_{sp} (number of nonzero-rank blocks) may be much less than p.

Clearly, the number of zero-rank blocks increases with the threshold ε and with the mesh size N. What is more interesting, as shown in Table 7, is that the number of zero-rank blocks (N_{ZR}) has a much faster rate of increase with respect to the mesh size than the number of nonzero



(a) Poisson problem



(b) Helmholtz problem

FIG. 12. *Flop complexities of the standard UFSC variant for different thresholds ε*

low-rank blocks (N_{LR}). For example, for $\varepsilon = 10^{-2}$, the number of zero-rank blocks represents 74% of the total for $N = 64$ while it represents 97% for $N = 320$.

		N							
		64	96	128	160	192	224	256	320
$\varepsilon = 10^{-14}$	N_{FR}	40.8	35.5	31.3	30.3	26.4	26.4	23.6	13.4
	N_{LR}	59.2	64.5	68.6	69.6	73.6	73.6	76.4	86.6
	N_{ZR}	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0
$\varepsilon = 10^{-10}$	N_{FR}	21.3	19.1	16.6	17.0	14.6	14.6	12.8	5.8
	N_{LR}	78.6	80.9	83.4	82.9	85.4	85.4	87.1	94.2
	N_{ZR}	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0
$\varepsilon = 10^{-6}$	N_{FR}	2.9	3.2	3.0	3.1	2.5	2.5	2.1	0.6
	N_{LR}	97.0	96.5	96.7	96.4	96.4	95.7	95.3	93.3
	N_{ZR}	0.1	0.3	0.3	0.5	1.0	1.7	2.5	6.1
$\varepsilon = 10^{-2}$	N_{FR}	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	N_{LR}	26.2	17.4	12.2	9.4	7.6	6.4	5.5	3.0
	N_{ZR}	73.8	82.6	87.8	90.6	92.4	93.6	94.5	97.0

TABLE 7
Number of full-rank/low-rank/zero-rank blocks in percentage of the total number of blocks.

This could suggest that, asymptotically, the major part of the blocks are zero-rank blocks. Then, the number of low-rank blocks in Table 1 would not be $O(p)$ but rather $O(p^\alpha)$, with $\alpha < 1$. For example, for $\varepsilon = 10^{-2}$, the complexity of $O(n^{1.32})$ could be explained by a number of nonzero-rank blocks of order $O(1)$: indeed, if we assume the number of nonzero-rank blocks per row and/or column remains constant, then the dense flop complexity equation (20) is only driven by full-rank operations and becomes:

$$\mathcal{C}(m, x) = O(m^{2+x} + m^2 r) = O(m^{2+x}) \tag{34}$$

since $r \leq b = m^x$. With the optimal $x^* = 0$, the previous equation leads to a dense complexity in $O(m^2)$, which, as shown in Section 4, Table 2, leads to a sparse complexity in $O(n^{1.33})$.

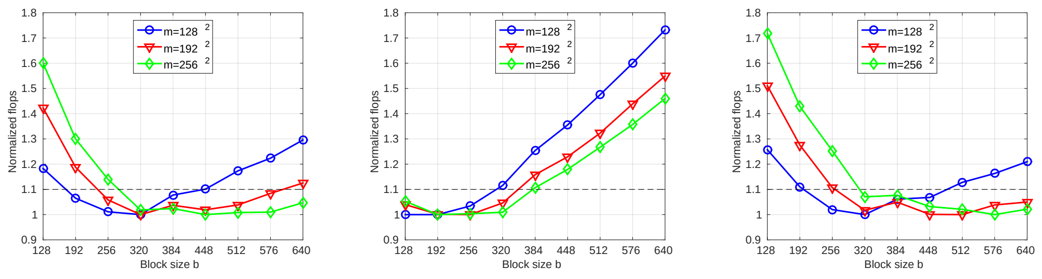
Note that for the sake of conciseness, we present our low-rank threshold analysis on flop complexity and on the standard UFSC variant only. A similar analysis on the other variants and on factor size complexity leads to the same results and conclusions.

7.2.4. Block size. For our theoretical complexity, we have assumed that the block size varies with the front size. Here, we want to show that the complexity of the BLR factorization is not strongly impacted by the choice of the block size, as long as this choice remains reasonable. The choice of a good block size is currently an ongoing research focus; the tuning of the block size for performance is out of the scope of this paper. In Figure 13, we show the number of operations for the BLR factorization of the root node (which is of size $m = N^2$) of the Poisson problem, for block sizes $b \in [128, 640]$. Three trends can be observed.

First, for each matrix, there is a reasonably large range of block sizes around the optimal one that lead to a number of operations that is reasonable with respect to the minimal one. For example, for the UFSC variant and $m = 256^2$, the optimal block size among the ones tested is $b = 448$. However, any block size in the range $[320, 640]$ (all those under the dashed black line) leads to a number of operations at most 10% greater than the minimal one. Thus, we have the flexibility to choose the block size which in turn gives the flexibility to tune the performance of the BLR factorization.

The second trend is that the range of acceptable block sizes (i.e., the block sizes for which the number of operations is not too far from the minimal one) increases with the size of the matrix m. For example, for the UFSC variant, the range of block sizes leading to a number of operations at most 10% greater than the minimal one is $[192, 384]$ for $m = 128^2$, $[256, 576]$ for $m = 192^2$ and $[320, 640]$ for $m = 256^2$. This is expected and in agreement with the theory, at least under the assumption that the block size is of the form $b = O(m^{x^*})$. This shows the importance of having a variable block size during the multifrontal factorization to adapt to the separators' size along the assembly tree, as opposed to fixed block size.

The third trend is observed when comparing the three factorization variants. Compared to the standard UFSC variant (Figure 13(a)), the UFSC+LUAR variant (Figure 13(b)) tends to favor smaller block sizes. This is because the low-rank term of the complexity equation (31) has been further reduced, and thus, in relative, the full-rank term (which increases with the block size) represents a greater part of the computations. This is accounted for by the theory with the optimal value x^* being equal to $1/3$ instead of $1/2$. In turn, compared to the UFSC+LUAR variant, the CUFS+LUAR (Figure 13(c)) benefits from bigger block sizes. This comes from the fact that the full-rank term has been reduced from (31) to (32). This is again consistent with the theory, which leads to $x^* = 1/2$.



(a) Standard UFSC variant (b) UFSC+LUAR variant (c) CUFS+LUAR variant

FIG. 13. *Normalized flops (i.e., the minimal is* 1*) for different block sizes* b *(Poisson problem,* $\varepsilon = 10^{-10}$*). The block sizes under the dashed black line are those for which the number of operations is at most* 10% *greater than the minimal one.*

A similar study on the Helmholtz problem shows very flat curves (i.e., the block size has little effect on the number of operations) as long as the block size remains reasonable.

8. Conclusion.

8.1. Summary. We have computed a bound on the numerical rank of the admissible blocks of BLR matrices arising from discretized elliptic PDEs. This bound is the same as in the hierarchical case (up to a constant), but cannot be obtained by applying directly the theoretical work done on $\mathcal{H}$-matrices. The main idea of the extension to BLR matrices is to identify the blocks that are not low-rank, and to reformulate the admissibility condition of a partition to ensure that they are in negligible number for an admissible partition.

Under this bound assumption, we have computed the theoretical complexity of the BLR multifrontal factorization. The standard version (as defined in Amestoy et al. [1]) can reach a complexity as low as $O(n^{1.67})$ (in 3D, for constant ranks). We have described several variants who further reduce this complexity, down to $O(n^{1.33})$. Our numerical results demonstrate an experimental complexity that is in good agreement with the theoretical bounds. The importance of zero-rank blocks and variable block sizes on the complexity has been identified and analyzed.

8.2. Perspectives. Because the error analysis in [13, 10, 11] is based on block-wise norm estimates, the accuracy of the low-rank approximation depends on the sparsity constant, which in turn, in the BLR case, depends on the size of the problem n. The accuracy of the BLR approximation could thus degrade as n increases. The effect of the BLR approximation on the accuracy of the factorization is out of the scope of this paper and a topic of reseach by itself.

The efficient implementation of the BLR variants, especially in a parallel setting, is a challenging problem and will be discussed in a forthcoming paper. In particular, the following challenges will need to be tackled: first, due to the smaller granularities of the operations involved, it will not be immediate to translate the gains in flops obtained by recompressing the accumulated updates in the LUAR variant, into gains in time. Second, we will need to design efficient strategies, similar to those analyzed in [9], to recompress the accumulated updates that achieve the desired complexity while keeping the overhead cost of the recompressions small. Third, the BLR variants will need to be adapted to a distributed memory setting, where their influence on the pattern of communications will need to be analyzed. Finally, as indicated in Section 6.1, the threshold partial pivoting strategy needs to be extended for the CUFS variant. Assuming that rank revelaling QR is used for off-diagonal block compression, the quality of a candidate pivot could be estimated with respect to the column entries of the R matrices of the low-rank off-diagonal blocks. Strategies close to those suggested in [18] for distributed-memory settings, where off-diagonal blocks are not available locally, could also be applied.

Acknowledgements. This work was performed using HPC resources from CALMIP (Grant 2015 - P0989). We wish to thank CALMIP and BULL for providing access to the MESCA2, and Stéphane Operto for providing the Helmholtz generator. We are grateful to Cleve Ashcraft for discussions that have guided this work. We also thank Julie Anton, Mario Arioli, Pieter Ghysels, Xiaoye S. Li, Sandrine Mouysset, François-Henry Rouet, Daniel Ruiz, Clément Weisbecker, and Jianlin Xia for various discussions. We are also grateful to the editor Per-Gunnar Martinsson, and two anonymous referees for their insightful and useful remarks. This works was partially supported by ANR-11-LABX-0040-CIMI within the program ANR-11-IDEX-0002-02.

REFERENCES

[1] P. R. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker, *Improving multifrontal methods by means of block low-rank representations*, SIAM Journal on Scientific Computing, 37 (2015), pp. A1451–A1474.

[2] P. R. Amestoy, R. Brossier, A. Buttari, J.-Y. L'Excellent, T. Mary, L. Métivier, A. Miniussi, and S. Operto, *Fast 3D frequency-domain full waveform inversion with a parallel Block Low-Rank multifrontal direct solver: application to OBC data from the North Sea*, Geophysics, 81 (2016), pp. R363 – R383.

[3] P. R. Amestoy, A. Buttari, I. S. Duff, A. Guermouche, J.-Y. L'Excellent, and B. Uçar, *The multifrontal method*, in Encyclopedia of Parallel Computing, D. Padua, ed., Springer, 2011, pp. 1209–1216.

[4] ———, *MUMPS*, in Encyclopedia of Parallel Computing, D. Padua, ed., Springer, 2011, pp. 1232–1238.

[5] P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary, *Complexity and performance of the Block Low-Rank multifrontal factorization*, in SIAM Conference on Parallel Processing (SIAM PP16), Paris, France, April 2016.

[6] A. Aminfar, S. Ambikasaran, and E. Darve, *A fast block low-rank dense solver with applications to finite-element matrices*, CoRR, abs/1403.5337 (2014).

[7] A. Aminfar and E. Darve, *A fast sparse solver for finite-element matrices*, CoRR, abs/1410.2697 (2014).

[8] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, SIAM Press, Philadelphia, PA, third ed., 1995.

[9] J. Anton, C. Ashcraft, and C. Weisbecker, *A Block Low-Rank multithreaded factorization for dense BEM operators*, in SIAM Conference on Parallel Processing (SIAM PP16), Paris, France, April 2016.

[10] M. Bebendorf, *Efficient inversion of Galerkin matrices of general second-order elliptic differential operators with nonsmooth coefficients*, Math. Comp., 74 (2005), pp. 1179–1199.

[11] M. Bebendorf, *Why finite element discretizations can be factored by triangular hierarchical matrices*, SIAM Journal on Numerical Analysis, 45 (2007), p. 1472.

[12] M. Bebendorf, *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, vol. 63 of Lecture Notes in Computational Science and Engineering (LNCSE), Springer-Verlag, 2008. ISBN 978-3-540-77146-3.

[13] M. Bebendorf and W. Hackbusch, *Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients*, Numerische Mathematik, 95 (2003), pp. 1–28.

[14] S. Börm, L. Grasedyck, and W. Hackbusch, *Introduction to hierarchical matrices with applications*, Engineering analysis with boundary elements, 27 (2003), pp. 405–422.

[15] S. Chandrasekaran, M. Gu, and T. Pals, *A fast ULV decomposition solver for hierarchically semiseparable representations*, SIAM Journal on Matrix Analysis and Applications, 28 (2006), pp. 603–622.

[16] H. Cheng, Z. Gimbutas, P. G. Martinsson, and V. Rokhlin, *On the compression of low rank matrices*, SIAM Journal on Scientific Computing, 26 (2005), pp. 1389–1404.

[17] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Oxford University Press, London, 1986.

[18] I. S. Duff and S. Pralet, *Towards stable mixed pivoting strategies for the sequential and parallel solution of sparse symmetric indefinite systems*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 1007–1024.

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

[20] B. Engquist and L. Ying, *Sweeping preconditioner for the helmholtz equation: Hierarchical matrix representation*, Communications on Pure and Applied Mathematics, 64 (2011), pp. 697–735.

[21] J. A. George, *Nested dissection of a regular finite-element mesh*, SIAM Journal on Numerical Analysis, 10 (1973), pp. 345–363.

[22] P. Ghysels, X. S. Li, F.-H. Rouet, S. Williams, and A. Napov, *An efficient multi-core implementation of a novel HSS-structured multifrontal solver using randomized sampling*, SIAM Journal on Scientific Computing, (2016). To appear.

[23] A. Gillman, *Fast direct solvers for elliptic partial differential equations*, PhD thesis, University of Colorado, 2011.

[24] A. Gillman, P. Young, and P.-G. Martinsson, *A direct solver with $\mathcal{O}(N)$ complexity for integral equations on one-dimensional domains*, Frontiers of Mathematics in China, 7 (2012), pp. 217–247.

[25] L. Grasedyck and W. Hackbusch, *Construction and arithmetics of h-matrices*, Computing, 70 (2003), pp. 295–334.

[26] W. Hackbusch, *A sparse matrix arithmetic based on h-matrices. part I: introduction to h-matrices*, Computing, 62 (1999), pp. 89–108.

[27] W. Hackbusch, B. N. Khoromskij, and R. Kriemann, *Hierarchical matrices based on a weak admissibility criterion*, Computing, 73 (2004), pp. 207–243.

[28] K. L. Ho and L. Ying, *Hierarchical interpolative factorization for elliptic operators: differential equations*, ArXiv e-prints, (2013).

[29] G. Karypis and V. Kumar, MeTiS *– A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices – Version 4.0*, University of Minnesota, Sept. 1998.

[30] P. A. Knight, D. Ruiz, and B. Uçar, *A symmetry preserving algorithm for matrix scaling*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 931–955.

[31] J. W. H. Liu, *The role of elimination trees in sparse factorization*, SIAM Journal on Matrix Analysis and Applications, 11 (1990), pp. 134–172.

[32] ———, *The multifrontal method for sparse matrix solution: Theory and Practice*, SIAM Review, 34 (1992), pp. 82–109.

[33] J. Peiró and S. Sherwin, *Finite difference, finite element and finite volume methods for partial differential equations*, in Handbook of materials modeling, Springer, 2005, pp. 2415–2446.

[34] H. Pouransari, P. Coulier, and E. Darve, *Fast hierarchical solvers for sparse matrices using low-rank approximation*, ArXiv e-prints, (2015).

[35] D. Ruiz, *A scaling algorithm to equilibrate both rows and columns norms in matrices*, Tech. Rep. RT/APO/01/4, ENSEEIHT-IRIT, 2001. Also appeared as RAL report RAL-TR-2001-034.

[36] R. Schreiber, *A new implementation of sparse Gaussian elimination*, ACM Transactions on Mathematical Software, 8 (1982), pp. 256–276.

[37] D. A. Sushnikova and I. V. Oseledets, *"Compress and eliminate" solver for symmetric positive definite sparse matrices*, ArXiv e-prints, (2016).

[38] S. Wang, X. S. Li, F.-H. Rouet, J. Xia, and M. V. De Hoop, *A parallel geometric multifrontal solver using hierarchically semiseparable structure*, ACM Trans. Math. Softw., 42 (2016), pp. 21:1–21:21.

[39] J. Xia, *Efficient structured multifrontal factorization for general large sparse matrices*, SIAM Journal on Scientific Computing, 35 (2013), pp. A832–A860.

[40] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Superfast multifrontal method for large structured linear systems of equations*, SIAM Journal on Matrix Analysis and Applications, 31 (2009), pp. 1382–1411.

[41] ———, *Fast algorithms for hierarchically semiseparable matrices*, Numer. Linear Algebra with Appl., 17 (2010), pp. 953–976.

Appendix: BLR approximants and proofs.

BLR approximant of B. The construction of $\widetilde{B}$ is the same for a BLR or an $\mathcal{H}$-partitioning, and we can thus rely on the work of Hackbusch and Bebendorf [13].

The main idea behind this construction is to exploit the decay property of Green functions. As shown in Hackbusch and Bebendorf ([13], Theorem 3.4), for any admissible block $\sigma \times \tau \in \mathcal{B}_A$, $B_{\sigma\times\tau}$ can be approximated by a low-rank matrix $B^{\varepsilon}_{\sigma\times\tau}$ of numerical rank less than r_G.

Therefore, we construct $\widetilde{B} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_G, N_{na})$ as follows:

$$\forall \sigma \times \tau \in \mathfrak{S}(\mathcal{I})^2,\ \widetilde{B}_{\sigma\times\tau} = \begin{cases} B^{\varepsilon}_{\sigma\times\tau} & \text{if } \sigma\times\tau \in \mathcal{B}_A \\ B_{\sigma\times\tau} & \text{otherwise} \end{cases} \tag{35}$$

BLR approximant of M^{-1}. The construction of $\widetilde{M}^{-1}$ is also very similar to the one in Hackbusch & Bebendorf [13]. The main idea is that the inverse mass matrix asymptotically tends towards a block-diagonal matrix. More precisely, it is shown that, for any block $\sigma \times \tau \in \mathfrak{S}(\mathcal{I})^2$,

$$\|M^{-1}_{\sigma\times\tau}\| \le O(\sqrt{\#\sigma\#\tau}\ c^{\sqrt[2d]{\#\sigma\#\tau}\ \operatorname{dist}(X_\sigma, X_\tau)})\|M^{-1}\|$$

where $c < 1$ ([13], Lemma 4.2). Therefore, $\|M^{-1}_{\sigma\times\tau}\|$ tends towards zero when $\#\sigma, \#\tau$ tend towards infinity (which is the case for a non-constant block size b), as long as $\operatorname{dist}(X_\sigma, X_\tau) > 0$, i.e., as long as $\sigma \times \tau \in \mathcal{B}_A$.

Therefore, we construct $\widetilde{M}^{-1} \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), 0, N_{na})$ as follows:

$$\forall \sigma \times \tau \in \mathfrak{S}(\mathcal{I})^2,\ \widetilde{M}^{-1}_{\sigma\times\tau} = \begin{cases} 0 & \text{if } \sigma\times\tau \in \mathcal{B}_A \\ M^{-1}_{\sigma\times\tau} & \text{otherwise} \end{cases} \tag{36}$$

Proof of Theorem 3.

Proof. Let $A \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_A, q_A)$ and $B \in \mathcal{BLR}(\mathfrak{S}(\mathcal{I}), r_B, q_B)$ be two $c_{sp} \times c_{sp}$ BLR matrices and let $P = AB$ be their product.

For all $i, j \in [1, c_{sp}]$, we note A_{ij}, B_{ij}, and P_{ij} the (i, j)-th subblock of matrix A, B, and P, respectively. We also note $\operatorname{Rk}(X)$ the numerical rank of a matrix X at accuracy ε.

We define

$$q_A(i) = \{k \in [1, c_{sp}]; A_{ik} \notin \mathcal{B}_A\}$$
$$q_B(j) = \{k \in [1, c_{sp}]; B_{kj} \notin \mathcal{B}_A\}$$

and thus $q_A = \max_{i\in[1,c_{sp}]} \#q_A(i)$ and $q_B = \max_{j\in[1,c_{sp}]} \#q_B(j)$.

Then, for any $i, j \in [1, c_{sp}]$, it holds:

$$P_{ij} = \sum_{k=1}^{c_{sp}} A_{ik}B_{kj} = \overbrace{\sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\in\mathcal{B}_A}} A_{ik}B_{kj}}^{S_1} + \overbrace{\sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\notin\mathcal{B}_A}} A_{ik}B_{kj}}^{S_2} + \overbrace{\sum_{\substack{A_{ik}\notin\mathcal{B}_A \\ B_{kj}\in\mathcal{B}_A}} A_{ik}B_{kj}}^{S_3} + \overbrace{\sum_{\substack{A_{ik}\notin\mathcal{B}_A \\ B_{kj}\notin\mathcal{B}_A}} A_{ik}B_{kj}}^{S_4}$$

We then seek a bound on the rank of each of the four terms S_1 to S_4.

1.
$$\begin{aligned}\operatorname{Rk}(S_1) &\le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\in\mathcal{B}_A}} \operatorname{Rk}(A_{ik}B_{kj}) \le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\in\mathcal{B}_A}} \min(\operatorname{Rk}(A_{ik}), \operatorname{Rk}(B_{kj})) \\ &\le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\in\mathcal{B}_A}} \min(r_A, r_B) \le c_{sp}\min(r_A, r_B)\end{aligned}$$

2.
$$\begin{aligned}\operatorname{Rk}(S_2) &\le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\notin\mathcal{B}_A}} \operatorname{Rk}(A_{ik}B_{kj}) \le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\notin\mathcal{B}_A}} \min(\operatorname{Rk}(A_{ik}), \operatorname{Rk}(B_{kj})) \\ &\le \sum_{\substack{A_{ik}\in\mathcal{B}_A \\ B_{kj}\notin\mathcal{B}_A}} r_A \le \#q_B(j)\, r_A \le q_B r_A\end{aligned}$$

3.

$$\begin{aligned}\mathrm{Rk}\,(S_3) &\leq \sum_{\substack{A_{ik}\notin\mathcal{B}_A\\B_{kj}\in\mathcal{B}_A}} \mathrm{Rk}\,(A_{ik}B_{kj}) \leq \sum_{\substack{A_{ik}\notin\mathcal{B}_A\\B_{kj}\in\mathcal{B}_A}} \min(\mathrm{Rk}\,(A_{ik})\,, \mathrm{Rk}\,(B_{kj}))\\ &\leq \sum_{\substack{A_{ik}\notin\mathcal{B}_A\\B_{kj}\in\mathcal{B}_A}} r_B \leq \#q_A(i)\; r_B \leq q_A r_B\end{aligned}$$

4. It holds that

$$\forall i \in [1, c_{sp}], \quad \#\{j \in [1, c_{sp}]; q_A(i) \cap q_B(j) \neq \emptyset\} \leq q_A q_B \tag{37}$$
$$\forall j \in [1, c_{sp}], \quad \#\{i \in [1, c_{sp}]; q_A(i) \cap q_B(j) \neq \emptyset\} \leq q_A q_B \tag{38}$$

(37) states that the number of non-admissible blocks on any row of P is bounded by $q_A q_B$, while (38) states that the number of non-admissible blocks on any column of P is also bounded by $q_A q_B$. Thus, putting (37) and (38) together, we have $q_P = q_A q_B$.

Proof of (37). Let $i \in [1, c_{sp}]$. For all $k \in q_A(i)$, it holds

$$\#\{j \in [1, c_{sp}]; B_{kj} \notin \mathcal{B}_A\} \leq q_B$$

and since $\#q_A(i) \leq q_A$, we conclude

$$\#\{j \in [1, c_{sp}]; P_{ij} \notin \mathcal{B}_A\} \leq q_A q_B$$

The proof of (38) works in the same way. □

Therefore, $S_4 = 0$ and thus $\mathrm{Rk}\,(S_4) = 0$, except for $q_P = q_A q_B$ blocks whose rank is not bounded. For the rest, their rank is thus bounded by

$$r_P = c_{sp} \min(r_A, r_B) + q_B r_A + q_A r_B$$ □

Computation of the multifrontal complexity. We consider here the UFSC variant in the 3D case. We recall Equation (28):

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} \mathcal{C}_\ell(N) = \sum_{\ell=0}^{L} (2^d)^\ell \mathcal{C}((\frac{N}{2^\ell})^{d-1}, x_\ell^*)$$

Using Equation (26) leads to

$$\mathcal{C}_\ell(N) = 2^{-\ell(2+\alpha)} N^{5+\alpha} \tag{39}$$

The case $r = O(1)$ is equivalent to $\forall \ell, r = O(m_\ell^{\alpha_\ell})$, with $\forall \ell, \alpha_\ell = 0$ and thus (25) yields $\forall \ell,\ x_\ell^* = x^* = \frac{1}{2}$. Then, (39) leads to

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} 2^{-2\ell} N^5 \tag{40}$$

which is a geometric series of common ratio $Q = 2^{-2} < 1$, and thus we obtain

$$\mathcal{C}_{MF}(N) = \frac{1 - Q^{L+1}}{1 - Q} N^5 = O(N^5) \tag{41}$$

However, if $r = O(N)$, then r is of the form $r = O(m_\ell^{\alpha_\ell})$, where α_ℓ varies with the level and thus $x_\ell^* = (1 + \alpha_\ell)/2$ also varies with the level. Specifically, it holds

$$\alpha_\ell = \begin{cases} \frac{L}{2(L-\ell)} & \text{if } \ell \neq L \\ 0 & \text{otherwise} \end{cases} \tag{42}$$

In the case $\ell = L$, (39) yields $\mathcal{C}_L(N) = O(N^3)$, which is negligible and thus ignored. For the rest of the levels, it holds

$$x_\ell^* = \frac{3L - 2\ell}{4(L-\ell)} \tag{43}$$

Then, noting $\beta = \frac{\alpha_\ell}{L} = \frac{1}{2(L-\ell)}$, and since $2^L = N$, (39) leads to

$$\mathcal{C}_\ell(N) = 2^{-2\ell} 2^{-L\ell\beta} N^{5+L\beta} = 2^{-2\ell} N^{5+(L-\ell)\beta} = 2^{-2\ell} N^{5.5} \tag{44}$$

and thus $\mathcal{C}_{MF}(N)$ is again a geometric series and is thus of order $O(N^{5.5})$

Note that in the $r = O(N)$ case with a non-adaptive $x_\ell^* = x_0^* = 3/4$, (31) yields

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} 2^{-\ell/2} N^{5.5} \tag{45}$$

and thus the same complexity exponent is achieved (i.e. an adaptive x_ℓ^* only improves the constant).

For the CUFS+LUAR variant, (39) becomes instead

$$\mathcal{C}_\ell(N) = 2^{-\ell(1+2\alpha)} N^{4+2\alpha} \tag{46}$$

In the $r = O(1)(\alpha = 0)$ case, this leads to

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} 2^{-\ell} N^4 = O(N^4) \tag{47}$$

In the $r = O(N)$ case, with an adaptive x_ℓ^*,

$$\mathcal{C}_\ell(N) = 2^{-\ell} 2^{-2L\ell\beta} N^{4+2L\beta} = 2^{-\ell} N^{4+2(L-\ell)\beta} = 2^{-\ell} N^5 \tag{48}$$

and thus $\mathcal{C}_{MF}(N) = O(N^5)$. However, for a non-adaptive $x_\ell^* = x_0^* = 3/4$, (32) yields

$$\mathcal{C}_{MF}(N) = \sum_{\ell=0}^{L} 2^0 N^5 = O(N^5 \log N) \tag{49}$$

Thus, for the CUFS+LUAR variant, a non-adaptive x_ℓ^* introduces a log factor.

The computation of the factor size complexity and the flop complexity of the UFSC+LUAR variants, and the computations in the 2D case are similar and left to the reader.