Butterfly factorization with error guarantees

Quoc-Tung Le, Rémi Gribonval, Elisa Riccietti, Léon Zheng

BUTTERFLY FACTORIZATION WITH ERROR GUARANTEES*

QUOC-TUNG LE[†,1], LÉON ZHENG[†,1,2], ELISA RICCIETTI [1], RÉMI GRIBONVAL [3]

Abstract. In this paper, we investigate the butterfly factorization problem, i.e., the problem of approximating a matrix by a product of sparse and structured factors. We propose a new formal mathematical description of such factors, that encompasses many different variations of butterfly factorization with different choices of the prescribed sparsity patterns. Among these supports, we identify those that ensure that the factorization problem admits an optimum, thanks to a new property called "chainability". For those supports we propose a new butterfly algorithm that yields an approximate solution to the butterfly factorization problem and that is supported by stronger theoretical guarantees than existing factorization methods. Specifically, we show that the ratio of the approximation error by the minimum value is bounded by a constant, independent of the target matrix.

1. Introduction. Algorithms for the rapid evaluation of linear operators are important tools in many domains like scientific computing, signal processing, and machine learning. In such applications, where a very large number of parameters is involved, the *direct* computation of the matrix-vector multiplication hardly scales due to its quadratic complexity in the matrix size. Many existing works therefore rely on analytical or algebraic assumptions on the considered matrix to approximate the evaluation of matrix-vector multiplication with a subquadratic complexity. Examples of such structures include low-rank matrices, hierarchical matrices [15], fast multipole methods [11], etc.

Among these different structures, previous work has identified another class of matrices that can be compressed for accelerating matrix multiplication. It is the class of so-called *butterfly* matrices [31, 33, 2], and includes many matrices appearing in scientific computing problems, like kernel matrices associated with special function transforms [33, 42] or Fourier integral operators [2, 7, 24]. Such matrices satisfy a certain low-rank property, named the *complementary low-rank* property [23]: it has been shown that if specific submatrices of a target matrix $\mathbf{A}$ of size $n \times n$ are numerically low-rank, then $\mathbf{A}$ can be compressed by successive hierarchical low-rank approximations of these submatrices, and that as a result it can be approximated by a sparse factorization

$$\hat{\mathbf{A}} = \mathbf{X}_1 \dots \mathbf{X}_L$$

with $L = \mathcal{O}(\log n)$ factors $\mathbf{X}_\ell$ having at most $\mathcal{O}(n)$ nonzero entries for each $\ell \in [\![L]\!] := \{1, \dots, L\}$. This sparse factorization, called in general *butterfly factorization*, would then yield a fast algorithm for the approximate evaluation of the matrix-vector multiplication by $\mathbf{A}$, in $\mathcal{O}(n \log n)$ complexity.

*Submitted to the editors DATE. †Equal contribution, the first two co-authors are listed alphabetically. [1]ENS de Lyon, CNRS, Inria, Université Claude Bernard Lyon 1, LIP, UMR 5668, 69342, Lyon cedex 07, France; [2]valeo.ai, Paris, France; [3]Inria, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP, UMR 5668, 69342, Lyon cedex 07, France (quoc-tung.le@tse-fr.eu, leonzheng314@gmail.com, elisa.riccietti@ens-lyon.fr, remi.gribonval@inria.fr). This work has been finalised while Tung Le was a postdoctoral researcher at Toulouse School of Economics, France and Léon Zheng a researcher at Huawei research center, Paris, France.

Funding: This project was supported in part by the AllegroAssai ANR project ANR-19-CHIA-0009, by the CIFRE fellowship N°2020/1643, and by the SHARP ANR project ANR-23-PEIA-0008 funded in the context of the France 2030 program. Tung Le was supported by AI Interdisciplinary Institute ANITI funding, through the French "Investments for the Future – PIA3" program under the grant agreement ANR-19-PI3A0004, and Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant numbers FA8655-22-1-7012.

An alternative definition of the butterfly factorization refers to a sparse matrix factorization with *specific* constraints on the sparse factors. According to [5, 6, 20, 44, 4, 27], a matrix $\mathbf{A}$ admits a certain butterfly factorization if, up to some row and column permutations, it can be factorized into a certain number of factors $\mathbf{X}_1, \dots, \mathbf{X}_L$ for a prescribed number $L \geq 2$, such that each factor $\mathbf{X}_\ell$ for $\ell \in [\![L]\!]$ satisfies a so-called *fixed-support* constraint, i.e., the support of $\mathbf{X}_\ell$, denoted $\mathtt{supp}(\mathbf{X}_\ell)$, is included in the support of a prescribed binary matrix $\mathbf{S}_\ell$. The different existing butterfly factorizations only vary by their number of factors L, and their choice of binary matrices $\mathbf{S}_1, \dots, \mathbf{S}_L$. Let us give some examples of such factorizations.

1. **Square dyadic butterfly factorization** [5, 6, 20, 44]. It is defined for matrices of size $n \times n$ where n is a power of two. The number of factors is $L := \log_2 n$. For $\ell \in [\![L]\!]$, the factor $\mathbf{X}_\ell$ is of size $n \times n$, and satisfies the support constraint $\mathtt{supp}(\mathbf{X}_\ell) \subseteq \mathtt{supp}(\mathbf{S}_\ell)$, where

$$\forall \ell \in [\![L]\!], \quad \mathbf{S}_\ell := \mathbf{I}_{2^{\ell-1}} \otimes \mathbf{1}_{2\times 2} \otimes \mathbf{I}_{n/2^\ell}.$$

   Here, $\mathbf{I}_n$ denotes the identity matrix of size n, $\mathbf{1}_{p\times q}$ denotes the matrix of size $p \times q$ full of ones, and $\otimes$ denotes the Kronecker product. This butterfly factorization appears in many structured linear maps commonly used in machine learning and signal processing, like the Hadamard matrix, or the discrete Fourier transform (DFT) matrix (up to the bit-reversal permutation of column indices). Other structured matrices like circulant matrix, Toeplitz matrix or Fastfood transform [41] can be written as a product of matrices admitting such a butterfly factorization, up to matrix transposition [6]. This factorization is also used to design structured random orthogonal matrices [36], and for quadrature rules on the hypersphere [32].
2. **Monarch factorization** [4]. A Monarch factorization parameterized by two integers p, q decomposes a matrix $\mathbf{A}$ of size $m \times n$ into $L := 2$ factors $\mathbf{X}_1$, $\mathbf{X}_2$ such that $\mathtt{supp}(\mathbf{X}_\ell) \subseteq \mathtt{supp}(\mathbf{S}_\ell)$ for $\ell = 1, 2$ where

$$\mathbf{S}_1 := \mathbf{1}_{p\times q} \otimes \mathbf{I}_{\frac{m}{p}}, \quad \mathbf{S}_2 := \mathbf{I}_q \otimes \mathbf{1}_{\frac{m}{p}\times\frac{n}{q}}.$$

   Here, we assume that p, q divides m, n respectively. The DFT matrix of size $n \times n$ admits such a factorization for $p = q$, up to a column permutation. Indeed, according to the Cooley-Tukey algorithm, computing the discrete Fourier transform of size n is equivalent to performing p discrete Fourier transforms of size n/p first, and then n/p discrete Fourier transforms of size p, see, e.g., equations (14) and (21) in [9].
3. **Deformable butterfly factorization** [27]. Previous conventional butterfly factorizations can be generalized as follows. Given an integer $L \geq 2$, a matrix $\mathbf{A}$ admits a deformable butterfly factorization parameterized by a list of tuples $(p_\ell, q_\ell, r_\ell, s_\ell, t_\ell)_{\ell=1}^L$ if $\mathbf{A} = \mathbf{X}_1 \dots \mathbf{X}_L$ where each factor $\mathbf{X}_\ell$ for $\ell \in [\![L]\!]$ is of size $p_\ell \times q_\ell$ and has a support included in $\mathtt{supp}(\mathbf{S}_\ell)$, defined as:

$$\forall \ell \in [\![L]\!], \quad \mathbf{S}_\ell := \mathbf{I}_{\frac{p_\ell}{r_\ell t_\ell}} \otimes \mathbf{1}_{r_\ell \times s_\ell} \otimes \mathbf{I}_{t_\ell}.$$

   Here, it is assumed that $\frac{p_\ell}{r_\ell t_\ell} = \frac{q_\ell}{s_\ell t_\ell}$ is an integer, for each $\ell \in [\![L]\!]$.

In all these examples the fixed-support constraint on each factor $\mathbf{X}$ takes the form $\mathtt{supp}(\mathbf{X}) \subseteq \mathtt{supp}(\mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d)$ for some integer parameters (a, b, c, d). Figure 1 illustrates the sparsity pattern $\mathbf{S}_{\boldsymbol{\pi}} := \mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d$ of a factor associated with the

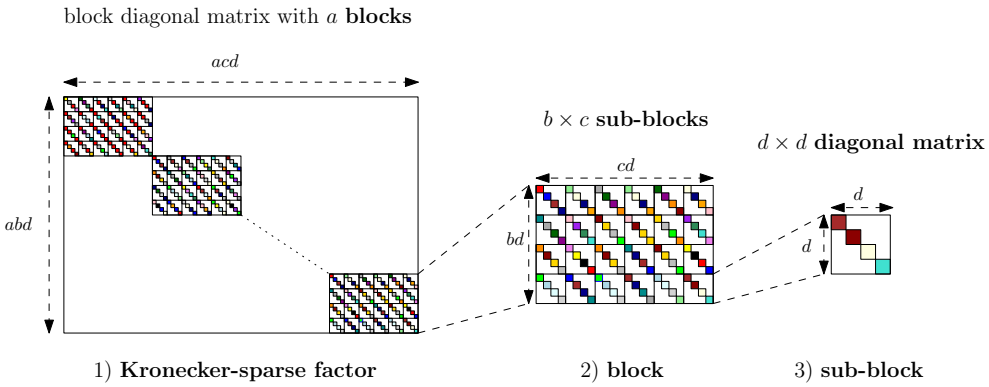


Fig. 1: Illustration of the support of a factor with pattern $\boldsymbol{\pi} = (a, b, c, d)$. The colored squares indicate the indices belonging to the support. The sub-figures (1), (2), (3) illustrate respectively the concepts of factor, block and sub-block.

tuple $\boldsymbol{\pi} = (a, b, c, d)$, that we call a *pattern*. One of the main benefits of choosing such fixed-support constraints *instead of an arbitrary sparse support* is its *block structure* that enables efficient implementation on specific hardware like Intelligence Processing Unit (IPU) [39] or GPU [5, 4, 13], with practical speed-up for matrix multiplication.

Such butterfly factorizations have been used in some machine learning applications. In line with recent works [5, 6, 4, 27], the parameterization (1.2) can be used to construct a generic representation for structured matrices that is not only expressive, but also differentiable and thus compatible with machine learning pipelines involving gradient-based optimization of parameters given training samples. The expected benefits then range from a more compressed storage and better generalization properties (thanks to the reduced number of parameters) to possibly faster implementations. For instance:

- The square dyadic butterfly factorization was used to replace hand-crafted structures in speech processing models or channel shuffling in certain convolutional neural networks, or to learn a latent permutation [6].
- The Monarch parameterization [4] of certain weight matrices in transformers for vision or language tasks led to speed-ups of training and inference time.
- Certain choices of deformable butterfly parameterizations [27] of kernel weights in convolutional layers, for vision tasks, led to similar performance as the original convolutional neural network with fewer parameters.

1.1. Problem formulation and contributions. This paper focuses on the problem of approximating a target matrix $\mathbf{A}$ by a product of structured sparse factors associated with a given *architecture* $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$:

$$E^{\boldsymbol{\beta}}(\mathbf{A}) := \inf_{(\mathbf{X}_\ell)_{\ell=1}^L} \|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F^2 = \inf_{\mathbf{B}} \|\mathbf{A} - \mathbf{B}\|_F^2, \tag{1.1}$$

where $\mathbf{B}$ is a butterfly matrix (cf. (1.2) below and Definition 4.4), each $\mathbf{X}_\ell$ is a factor with sparsity pattern prescribed by $\boldsymbol{\pi}_\ell$, and $\|\cdot\|_F$ is the Frobenius norm. We will call these factors "Kronecker-sparse factor", due to the Kronecker-structure of their sparsity pattern. Several methods have been proposed to address this butterfly factorization problem, but we argue that they either lack guarantees of success, or only have partial guarantees. We fix this issue here by introducing a new butterfly algorithm endowed

Table 1: Existing architectures $\boldsymbol{\beta}$ in the literature. ($\star$) Note that [27] did not explicitly state constraints on $(a_\ell, b_\ell, c_\ell, d_\ell)_{\ell=1}^L$ for deformable butterfly factorization, because they use an alternative description of the sparsity patterns of the factors. See Example 4.11 for more details.

Architectures	L	**Size**	**Values of** $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$	**Chainable?**
Low-rank matrix	2	$m \times n$	$(1, m, r, 1)$, $(1, r, n, 1)$	Yes
Monarch [4]	2	$m \times n$	$(1, p, q, m/p)$, $(q, m/p, n/q, 1)$	Yes
Square dyadic butterfly [5]	any	$2^L \times 2^L$	$(2^{\ell-1}, 2, 2, 2^{L-\ell})_{\ell=1}^L$	Yes
($\star$) Deformable butterfly [27]	any	$m \times n$	$(a_\ell, b_\ell, c_\ell, d_\ell)_{\ell=1}^L$	Yes
Kaleidoscope [6]	even	$2^{L/2} \times 2^{L/2}$	$\boldsymbol{\pi}_\ell = \begin{cases} (2^{\ell-1}, 2, 2, 2^{L/2-\ell}) & \text{if } \ell \leq L/2 \\ (2^{L-\ell}, 2, 2, 2^{\ell-L/2-1}) & \text{if } \ell > L/2 \end{cases}$	No

with theoretical guarantees.

More precisely, the main contributions of this paper are:

1. To introduce, via the definition of a Kronecker-sparse factor, a formal mathematical description of the "deformable butterfly factors" of [27]. While we owe [27] the original idea of extending previous butterfly factorizations, the mathematical formulation of the prescribed supports as Kronecker products is a novelty that allows a theoretical study of the corresponding butterfly factorization, as done in this paper. Moreover, our parameterization uses 4 parameters and removes the redundancy in the original 5-parameter description of deformable butterfly factors of [27]. Table 1 summarizes the main characteristics of existing butterfly architectures covered by our framework.
2. To define the *chainability* of an architecture $\boldsymbol{\beta}$ (Definition 4.10), which is basically a "stability" property that ensures that a product of Kronecker-sparse factors is still a Kronecker-sparse factor. We prove that Problem (1.1) admits an optimum when $\boldsymbol{\beta}$ is chainable (Theorem 7.8).
3. To characterize *analytically* the set of butterfly matrices with architecture $\boldsymbol{\beta}$,

$$\mathcal{B}^{\boldsymbol{\beta}} := \{\mathbf{X}_1 \dots \mathbf{X}_L \,|\, \mathtt{supp}(\mathbf{X}_\ell) \subseteq \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_\ell}) \ \forall \ell \in [\![L]\!]\}\,, \tag{1.2}$$

   for a chainable $\boldsymbol{\beta}$, in terms of low-rank properties of certain submatrices of $\mathbf{A}$ (Corollary 7.7) that are equivalent to a generalization of the complementary low-rank property (Definition F.2 and Corollary F.4).
4. To define the *redundancy* of a chainable architecture (Definition 4.15). Intuitively, a chainable architecture $\boldsymbol{\beta}$ is redundant if one can replace it with a "compressed" (non-redundant) one $\boldsymbol{\beta}'$ such that $\mathcal{B}^{\boldsymbol{\beta}} = \mathcal{B}^{\boldsymbol{\beta}'}$ (Proposition 4.21). Thus, from the perspective of accelerating linear operators, redundant architectures have no practical interest.
5. To propose a new butterfly algorithm (Algorithm 6.1) able to provide an approximate solution to Problem (1.1) *for non-redundant chainable architectures*. Compared to previous similar algorithms, this algorithm introduces a new orthogonalization step that is key to obtain approximation guarantees. The algorithm can be easily extended to redundant chainable architectures, with the same theoretical guarantee (see Remark 6.4).
6. To prove that, for a chainable $\boldsymbol{\beta}$, Algorithm 6.1 outputs butterfly factors

Table 2: The approximation ratio $C_{\boldsymbol{\beta}}$ (see Equation (1.3)) of Algorithm 6.1 with a selection of chainable architectures $\boldsymbol{\beta}$ from Table 1.

Parameterization	**Size**	$L = \lvert\boldsymbol{\beta}\rvert$	$C_{\boldsymbol{\beta}}$ **in** (7.4) **- Theorem 7.4**	$C_{\boldsymbol{\beta}}$ **in** (7.5) **- Theorem 7.4**
Low rank matrix	$m \times n$	2	1	1
Monarch [4]	$m \times n$	2	1	1
Square dyadic butterfly [5]	$n \times n$	$\log n$	$\log n - 1$	$\sqrt{\log n - 1}$

$(\hat{\mathbf{X}}_\ell)_{\ell=1}^L$ such that

$$\|\mathbf{A} - \hat{\mathbf{X}}_1 \dots \hat{\mathbf{X}}_L\|_F \leq C_{\boldsymbol{\beta}} \cdot \inf_{(\mathbf{X}_\ell)_{\ell=1}^L} \|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F, \tag{1.3}$$

where $C_{\boldsymbol{\beta}} \geq 1$ depends *only* on $\boldsymbol{\beta}$ (Theorem 7.4), see Table 2 for examples. To the best of our knowledge, this is the first time such a bound is established for a butterfly approximation algorithm.

1.2. Outline. Section 2 discusses related work. Section 3 introduces some preliminaries on two-factor matrix factorization with fixed-support constraints. This is also where we setup our general notations. Section 4 formalizes the definition of deformable butterfly factorization associated with $\boldsymbol{\beta}$, and introduces the *chainability* and *non-redundancy* conditions for an architecture $\boldsymbol{\beta}$, that will be at the core of the proof of error guarantees on our proposed butterfly algorithm. Section 5 extends an existing hierarchical algorithm, currently expressed only for dyadic butterfly factorization, to any chainable $\boldsymbol{\beta}$. For non-redundant chainable $\boldsymbol{\beta}$, Section 6 introduces novel orthonormalization operations in the proposed butterfly algorithm. This allows us to establish in Section 7 our main results on the control of the approximation error and the low-rank characterization of butterfly matrices associated with chainable $\boldsymbol{\beta}$. Section 8 proposes some numerical experiments about the proposed butterfly algorithm. The most technical proofs are deferred to the appendices.

2. Related work. Several methods have been proposed to address the butterfly factorization problem (1.1), but we argue that they either lack guarantees of success, or only have partial guarantees.

First-order methods. Optimization methods based on gradient descent [5] or alternating least squares [27] are not suitable for Problem (1.1) and lack guarantees of success, because of the non-convexity of the objective function. In fact, the problem of approximating a given matrix by the product of factors with fixed-support constraints, as it is the case for (1.1), is generally NP-hard and might even lead to numerical instability even for $L = 2$ factors, as shown in [18]. In contrast, we show that the minimum of (1.1) always exists for chainable $\boldsymbol{\beta}$.

Hierarchical approach for butterfly factorization. For the specific choice of $\boldsymbol{\beta}$ corresponding to a square dyadic butterfly factorization, i.e., with the architecture $\boldsymbol{\beta} = (2^{\ell-1}, 2, 2, 2^{L-\ell})_{\ell=1}^L$, there exists an efficient hierarchical algorithm for Problem (1.1), endowed with *exact* recovery guarantees [20, 44]. The hierarchical approach performs successive two-factor matrix factorizations, until the desired number of factors L is obtained. In the case of square dyadic butterfly factorization, it is shown that each two-factor matrix factorization in the hierarchical procedure can be solved optimally by computing the best rank-one approximation of some specific submatrices [18], which

leads to an overall $\mathcal{O}(n^2)$ complexity for approximating a matrix of size $n \times n$ with the hierarchical procedure. In fact, the hierarchical algorithm in [20, 44] can be seen as a variation of previous butterfly algorithms [23], with the novelty that it works for *any* hierarchical order under which successive two-layers matrix factorizations are performed, whereas existing butterfly algorithms [31, 33, 2] were only focusing on some *specific* hierarchical orders [29]. However, the question of controlling the approximation error of the algorithm was left as an open question in [44]. Moreover, it was not known in the literature if the hierarchical algorithm [44, 20] could be extended to architectures $\boldsymbol{\beta}$ beyond the square dyadic one. Both questions are answered positively here.

Butterfly algorithms and the complementary low-rank property. Butterfly algorithms [30, 31, 33, 2, 23, 24, 29] look for an approximation of a target matrix $\mathbf{A}$ by a sparse factorization $\hat{\mathbf{A}} = \mathbf{X}_1 \dots \mathbf{X}_L$, assuming that $\mathbf{A}$ satisfies the so-called *complementary low-rank property*, formally introduced in [23]. This low-rank property assumes that the rank of certain submatrices of $\mathbf{A}$ restricted to some specific blocks is numerically low and that these blocks satisfy some conditions described by a hierarchical partitioning of the row and column indices, using the notion of *cluster tree* [15]. Then, the butterfly algorithm leverages this low-rank property to approximate the target matrix by a data-sparse representation, by performing successive low-rank approximation of specific submatrices. The literature in numerical analysis describes many linear operators associated with matrices satisfying the complementary low-rank property, such as kernel matrices encountered in electromagnetic or acoustic scattering problems [30, 31, 14], special function transforms [34], spherical harmonic transforms [40] or Fourier integral operators [2, 42, 24, 22, 25].

The formal definition of the complementary low-rank property currently given in the literature only considers cluster trees that are dyadic [23] or quadtrees [25]. In this work, we give a more general definition of the complementary low-rank property that considers arbitrary cluster trees. To the best of our knowledge, this allows us to give the first formal characterization of the set of matrices admitting a (deformable) butterfly factorization associated with an architecture $\boldsymbol{\beta}$, as defined in (1.2), using this extended definition of the complementary low-rank property. In particular, this shows that the definition in (1.2) is more general than the previous definitions of the complementary low-rank property that were restricted to dyadic trees or quadtrees [23, 25].

Existing error bounds for butterfly algorithms. Several existing butterfly algorithms [33, 2, 23, 24] are guaranteed to provide an approximation error $\|\mathbf{A} - \hat{\mathbf{A}}\|_F$ equal to zero, when $\mathbf{A}$ satisfies *exactly* the complementary low-rank property, i.e., the best low-rank approximation errors of the submatrices described by the property are *exactly* zero [33, 23]. However, when these submatrices are only approximately low-rank (with a positive best low-rank approximation error), existing butterfly factorization algorithms *are not* guaranteed to provide an approximation $\hat{\mathbf{A}}$ with the *best* approximation error. To the best of our knowledge, the only existing error bound in the literature is based on a butterfly algorithm that performs successive low-rank approximation of blocks $\mathbf{M}$ [29]. However, this bound does not compare the approximation error $\|\mathbf{A} - \hat{\mathbf{A}}\|_F$ to the *best* approximation error, that is, the minimal error $\|\mathbf{A} - \mathbf{A}^*\|_F$ with $\mathbf{A}^*$ satisfying *exactly* the complementary low-rank property. Moreover, in contrast to our algorithm, the algorithm proposed in [29] is not designed for butterfly factorization problems with a *fixed* architecture. In [29] the architecture is the result of the stopping criterion that imposes a given accuracy on the low-rank approximations of the blocks. We discuss this further in Subsection 7.6. In this paper, we thus propose the first

error bound for butterfly factorization that compares the approximation error to the minimal approximation error, cf. (1.3).

3. Preliminaries. Following the hierarchical approach [21, 20, 43], our analysis of the butterfly factorization problem (1.1) with *multiple* factors in general ($L \geq 2$) relies on the analysis of the simplest setting with only $L = 2$ factors. This setting is studied in [18] and after setting up our general notations we recall some important results that will be used in the rest of the paper.

3.1. Notations. The set $[\![a, b]\!]$ is the set of integers $\{a, a+1, \ldots, b\}$ for $a \leq b$, and $[\![a]\!] := [\![1, a]\!]$. The notation $a \mid b$ means that a divides b. $A \times B$ is the Cartesian product of two sets A and B. $|A|$ is the cardinal of a set A. By abuse of notation, for any matrix $\mathbf{X}$ and any binary matrix $\mathbf{S}$, the support constraint $\mathtt{supp}(\mathbf{X}) \subseteq \mathtt{supp}(\mathbf{S})$ is simply written as $\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{S}$. $\mathbf{X}[i,:]$ and $\mathbf{X}[:,j]$ are the i-th row and the j-th column of $\mathbf{X}$, respectively. $\mathbf{X}[i,j]$ is the entry of $\mathbf{X}$ at the i-th row and j-th column. $\mathbf{X}[I,:]$ and $\mathbf{X}[:,J]$ are the submatrices of $\mathbf{X}$ restricted to a subset of row indices I and a subset of column indices J, respectively. $\mathbf{X}[I,J]$ is the submatrix of $\mathbf{X}$ restricted to both I and J. $\mathbf{X}^\top$ is the transposed matrix of $\mathbf{X}$. $\mathbf{0}_{m\times n}$ (resp. $\mathbf{1}_{m\times n}$) is the $m \times n$ matrix full of zeros (resp. of ones). The indicator (column) vector of a subset $R \subseteq [\![m]\!]$ is denoted $\mathbf{1}_R$. The rank of a matrix $\mathbf{M}$ is denoted $\mathtt{rank}(\mathbf{M})$. Finally, for any matrix $\mathbf{X}$, we denote $\varepsilon_r^2(\mathbf{X}) := \min_{\mathbf{Y}:\ \mathtt{rank}(\mathbf{Y})\leq r} \|\mathbf{X}-\mathbf{Y}\|_F^2$, and $\mathtt{rankproj}_r(\mathbf{X})$ is defined as the collection of all $\mathbf{Y}$ of rank at most r achieving the minimum. All these matrices have the same Frobenius norm, denoted by $\|\mathtt{rankproj}_r(\mathbf{X})\|_F$.

3.2. Two-factor, fixed-support matrix factorization. Given two binary matrices $\mathbf{L}, \mathbf{R}$, the problem of *fixed-support matrix factorization* (FSMF) with two factors is formulated as:

$$\inf_{(\mathbf{X},\mathbf{Y})} \quad \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_F^2, \text{ with } \mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}, \mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}. \tag{3.1}$$

While Problem (3.1) is NP-hard[1] in general [18, Theorem 2.4], it becomes tractable under certain conditions on $(\mathbf{L}, \mathbf{R})$. To describe one of these conditions, we recall the following definitions.

Definition 3.1 (Rank-one contribution supports [18, 44]). *The* rank-one contribution supports *of two binary matrices* $\mathbf{L} \in \{0,1\}^{m\times r}, \mathbf{R} \in \{0,1\}^{r\times n}$ *is the tuple* $\varphi(\mathbf{L}, \mathbf{R})$ *of* r *binary matrices defined by:*

$$\varphi(\mathbf{L}, \mathbf{R}) := (\mathbf{U}_i)_{i=1}^r, \quad \textit{where} \quad \mathbf{U}_i := \mathbf{L}[:,i]\mathbf{R}[i,:] \in \{0,1\}^{m\times n}.$$

Figure 2 illustrates the notion of *rank-one contribution supports* in Definition 3.1.

Remark 3.2. The binary matrix $\mathbf{L}[:,i]\mathbf{R}[i,:]$ for $i \in [\![r]\!]$ encodes the *support constraint* of $\mathbf{X}[:,i]\mathbf{Y}[i,:]$ for each $(\mathbf{X}, \mathbf{Y})$ such that $\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}$, $\mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}$.

The rank-one supports $(\mathbf{U}_i)_{i=1}^r$ defines an equivalence relation and its induced equivalence classes on the set of indices $[\![r]\!]$, as illustrated in Figure 2.

Definition 3.3 (Equivalence classes of rank-one supports, representative rank-one supports [18]). *Given* $\mathbf{L} \in \{0,1\}^{m\times r}, \mathbf{R} \in \{0,1\}^{r\times n}$, *denoting* $(\mathbf{U}_i)_{i=1}^r = \varphi(\mathbf{L}, \mathbf{R})$, *define an equivalence relation on the index set* $[\![r]\!]$ *of the rows of* $\mathbf{L}$ */ columns of* $\mathbf{R}$ *as:*

$$i \sim j \iff \mathbf{U}_i = \mathbf{U}_j.$$

[1] and does not always admit an optimum: the infimum may not be achieved [18, Remark A.1].

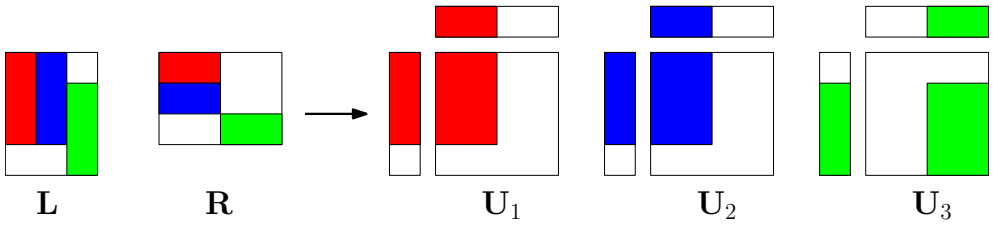


Fig. 2: An example of support constraints $(\mathbf{L}, \mathbf{R})$ and the supports of the corresponding rank-one contributions. Colored parts indicate indices inside the support constraints $\mathbf{L}, \mathbf{R}$ and $\mathbf{U}_i$ for $i \in [\![3]\!]$. $\{1, 2\}$ and $\{3\}$ are the two equivalence classes (Definition 3.3).

This yields a partition of the index set $[\![r]\!]$ into equivalence classes, denoted $\mathcal{P}(\mathbf{L}, \mathbf{R})$. For each $P \in \mathcal{P}(\mathbf{L}, \mathbf{R})$, denote $\mathbf{U}_P$ a representative rank-one support, $R_P \subseteq [\![m]\!]$ and $C_P \subseteq [\![n]\!]$ the supports of rows and columns in $\mathbf{U}_P$, respectively, i.e., $\mathtt{supp}(\mathbf{U}_P) = R_P \times C_P$, and denote $|P|$ the cardinal of the equivalence class P.

We now recall a sufficient condition on the binary support matrices $(\mathbf{L}, \mathbf{R})$ for which corresponding instances of Problem (3.1) can be solved in polynomial time via Algorithm 3.1.

THEOREM 3.4 (Tractable support constraints of Problem (3.1) [18, Theorem 3.3]). *If all components $\mathbf{U}_i$ of $\varphi(\mathbf{L}, \mathbf{R})$ are pairwise disjoint or identical, then Algorithm 3.1 yields an optimal solution of Problem (3.1), and the infimum of Problem (3.1) is :*

$$\inf_{\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}, \mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}} \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_F^2 = \sum_{P \in \mathcal{P}(\mathbf{L}, \mathbf{R})} \min_{\mathbf{B}, \mathtt{rank}(\mathbf{B}) \leq |P|} \|\mathbf{A}[R_P, C_P] - \mathbf{B}\|_F^2 + c, \tag{3.2}$$

where[2] $c := \sum_{(i,j) \notin \mathtt{supp}(\mathbf{L}\mathbf{R})} \mathbf{A}[i, j]^2$ *is a constant depending only on* $(\mathbf{A}, \mathbf{L}, \mathbf{R})$.

Equation (3.2) was not proved in [18], so we provide a complete proof of Theorem 3.4 in Appendix A. The main idea is the following.

Sketch of proof. For $(\mathbf{X}, \mathbf{Y})$ such that $\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}$ and $\mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}$:

$$\|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_F^2 = \sum_{P \in \mathcal{P}(\mathbf{L}, \mathbf{R})} \|\mathbf{A}[R_P, C_P] - \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2 + c, \tag{3.3}$$

because the fact that the components $\mathbf{U}_i$ of $\varphi(\mathbf{L}, \mathbf{R})$ are pairwise disjoint or identical implies that the blocks of indices $R_P \times C_P$ are pairwise disjoint. Thus, minimizing the left-hand-side is equivalent to minimizing each summand in the right-hand side, which is equivalent to finding the best rank-$|P|$ approximation of the matrix $\mathbf{A}[R_P, C_P]$ for each $P \in \mathcal{P}(\mathbf{L}, \mathbf{R})$. ∎

Remark 3.5. Best low-rank approximation in line 3 of Algorithm 3.1 can be computed via truncated singular value decomposition (SVD). Note that the definition of $\hat{\mathbf{H}}, \hat{\mathbf{K}}$ in this line is not unique, because, for instance, the product $\hat{\mathbf{H}}\hat{\mathbf{K}}$ is invariant to some rescaling of columns and rows.

[2] Note that $\mathbf{L}\mathbf{R}$ is a product of two binary matrices.

Algorithm 3.1 Two-factor fixed-support matrix factorization

Require: $\mathbf{A} \in \mathbb{C}^{m\times n}$, $\mathbf{L} \in \{0,1\}^{m\times r}$, $\mathbf{R} \in \{0,1\}^{r\times n}$
Ensure: $(\mathbf{X}, \mathbf{Y})$ such that $\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}$, $\mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}$
1: $(\mathbf{X}, \mathbf{Y}) \leftarrow (\mathbf{0}_{m\times r}, \mathbf{0}_{r\times n})$
2: **for** $P \in \mathcal{P}(\mathbf{L}, \mathbf{R})$ (cf. Definition 3.3) **do**
3: $(\mathbf{X}[R_P, P], \mathbf{Y}[P, C_P]) \leftarrow (\hat{\mathbf{H}}, \hat{\mathbf{K}}) \in \underset{\substack{\mathbf{H}\in\mathbb{C}^{|R_P|\times|P|}\\ \mathbf{K}\in\mathbb{C}^{|P|\times|C_P|}}}{\arg\min} \|\mathbf{A}[R_P, C_P] - \mathbf{HK}\|_F$
4: **end for**
5: **return** $(\mathbf{X}, \mathbf{Y})$

4. Deformable butterfly factorization. This section presents a mathematical formulation of the deformable butterfly factorization [27] associated with a sequence of patterns $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ called an architecture. We then introduce the notions of *chainability* and *non-redundancy* of an architecture, that are crucial conditions for constructing a butterfly algorithm for Problem (1.1) with error guarantees.

4.1. A mathematical formulation for Kronecker-sparse factors. Many butterfly factorizations [5, 6, 20, 44, 4, 27] take the form $\mathbf{A} = \mathbf{X}_1 \dots \mathbf{X}_L$ with $\mathtt{supp}(\mathbf{X}_\ell) \subseteq \mathbf{I}_{a_\ell} \otimes \mathbf{1}_{b_\ell\times c_\ell} \otimes \mathbf{I}_{d_\ell}$ for $\ell \in [\![L]\!]$, for some parameters $(a_\ell, b_\ell, c_\ell, d_\ell)_{\ell=1}^L$, cf. Section 1. We therefore introduce the following definition.

Definition 4.1 (Kronecker-sparse factors and their sparsity patterns). *For $a, b, c, d \in \mathbb{N}$, a* Kronecker-sparse factor *of pattern $\boldsymbol{\pi} := (a, b, c, d)$ (or $\boldsymbol{\pi}$-factor) is a matrix in $\mathbb{R}^{m\times n}$ or $\mathbb{C}^{m\times n}$, where $m := abd$, $n := acd$, such that its support is included in $\mathbf{S}_{\boldsymbol{\pi}} := \mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d \in \{0,1\}^{m\times n}$. The tuple $\boldsymbol{\pi}$ will be called an* elementary Kronecker-sparse pattern*, or simply a* pattern*. The set of all $\boldsymbol{\pi}$-factors is denoted by $\Sigma^{\boldsymbol{\pi}}$.*

Figure 1 illustrates the support of a $\boldsymbol{\pi}$-factor, for a given pattern $\boldsymbol{\pi} = (a, b, c, d)$. A $\boldsymbol{\pi}$-factor matrix is block diagonal with a blocks in total. By definition, each block in the diagonal has support included in $\mathbf{1}_{b\times c} \otimes \mathbf{I}_d$. Thus, each block is a *block matrix* of size $b \times c$, where each *sub-block* is a diagonal matrix of size $d \times d$.

Example 4.2. *The following matrices are $\boldsymbol{\pi}$-factors for certain choices of $\boldsymbol{\pi}$.*
1. **Dense matrix***: Any matrix of size $m \times n$ is a $(1, m, n, 1)$-factor.*
2. **Diagonal matrix***: Any diagonal matrix of size $m \times m$ is either a $(m, 1, 1, 1)$-factor or $(1, 1, 1, m)$-factor.*
3. **Factors in a square dyadic butterfly factorization** *[5, 6, 20, 44]: the pattern of the ℓ-th factor is $\boldsymbol{\pi}_\ell = (2^{\ell-1}, 2, 2, 2^{L-\ell})$ for $\ell \in [\![L]\!]$.*
4. **Factors in a Monarch factorization** *[4]: the patterns of the two factors are $\boldsymbol{\pi}_1 = (1, p, q, m/p)$, $\boldsymbol{\pi}_2 = (q, m/p, n/q, 1)$ for p, q such that $p \mid m$ and $r \mid n$.*

Lemma 4.3 (Sparsity level of a $\boldsymbol{\pi}$-factor). *For $\boldsymbol{\pi} = (a, b, c, d)$, the number of nonzero entries of a $\boldsymbol{\pi}$-factor of size $m \times n$ is at most $\|\boldsymbol{\pi}\|_0 := abcd = mc = nb$.*

Proof. The cardinal of $\mathtt{supp}(\mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d)$ is $abcd = mc = nb = \frac{mn}{ad}$. ∎

A $\boldsymbol{\pi}$-factor is sparse if it has few nonzero entries compared to its size, i.e., if $\|\boldsymbol{\pi}\|_0 \ll mn$, or equivalently if $ad \gg \mathcal{O}(1)$. Given a number of factors $L \geq 1$, a sequence of patterns $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ parameterizes the set

$$\Sigma^{\boldsymbol{\beta}} := \Sigma^{\boldsymbol{\pi}_1} \times \ldots \times \Sigma^{\boldsymbol{\pi}_L} \tag{4.1}$$

of L-tuples of $\boldsymbol{\pi}_\ell$-factors, $\ell = 1, \dots, L$. Since we are interested in matrix products $\mathbf{X}_1 \dots \mathbf{X}_L$ for $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$, we will only consider sequences of patterns $\boldsymbol{\beta}$ such that the size of $\mathbf{X}_\ell \in \Sigma^{\boldsymbol{\pi}_\ell}$ and $\mathbf{X}_{\ell+1} \in \Sigma^{\boldsymbol{\pi}_{\ell+1}}$ are compatible for computing the matrix product $\mathbf{X}_\ell \mathbf{X}_{\ell+1}$, for each $\ell \in [\![L-1]\!]$. In other words, we require that the sequence of patterns $\boldsymbol{\beta}$ satisfies:

$$\forall \ell \in [\![L-1]\!], \quad \underbrace{a_\ell c_\ell d_\ell}_{n_\ell} = \underbrace{a_{\ell+1} b_{\ell+1} d_{\ell+1}}_{m_{\ell+1}}. \tag{4.2}$$

Therefore, under assumption (4.2), a sequence $\boldsymbol{\beta}$ can describe a factorization of the type $\mathbf{A} = \mathbf{X}_1 \dots \mathbf{X}_L$ such that $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$. We introduce the following terminology for such a sequence.

Definition 4.4 (Butterfly architecture and butterfly matrices). *A sequence of patterns $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ is called a* (deformable) butterfly architecture, or simply an *architecture*, when it satisfies (4.2). *By analogy with deep networks, the number of factors is called the depth of the chain and denoted by $|\boldsymbol{\beta}| := L$ and, using the notation $\|\boldsymbol{\pi}\|_0$ from Lemma* 4.3*, the number of parameters is denoted by*

$$\|\boldsymbol{\beta}\|_0 := \sum_{\ell=1}^{L} \|\boldsymbol{\pi}_\ell\|_0.$$

For any architecture $\boldsymbol{\beta}$, $\mathcal{B}^{\boldsymbol{\beta}}$ is the set of *(deformable) butterfly matrices* associated with $\boldsymbol{\beta}$, as defined in (1.2). We also say that any $\mathbf{A} \in \mathcal{B}^{\boldsymbol{\beta}}$ admits an *exact (deformable) butterfly factorization* associated with the architecture $\boldsymbol{\beta}$. Table 1 describes existing architectures fitting our framework.

The rest of this section introduces two important properties of an architecture $\boldsymbol{\beta}$:

- *Chainability* will be shown (Theorem 7.8) to ensure the existence of an optimum in (1.1), so that we can replace "inf" by "min" in (1.1). We also show that, for any chainable architecture, one can exploit a hierarchical algorithm (Algorithm 5.1) that extends an algorithm from [20, 43] to compute an approximate solution to Problem (1.1).
- *Non-redundancy* is an additional property satisfied by some chainable architectures $\boldsymbol{\beta}$, that allows us to insert orthonormalization steps in the hierarchical algorithm, in order to control the approximation error for Problem (1.1) in the sense of (1.3). Non-redundancy plays the role of an intermediate tool to design and analyze our algorithms. However, it should not be treated as an additional hypothesis, because we do propose a factorization method (cf. Remark 6.4), endowed with error guarantees, for *any chainable architecture, whether redundant or not.*

Both conditions are first defined for the most basic architectures $\boldsymbol{\beta}$ of depth $|\boldsymbol{\beta}| = 2$, before being generalized to architectures $\boldsymbol{\beta}$ of arbitrary depth $L \geq 2$.

4.2. Chainability. We start by defining this condition in the case of architectures of depth $L = 2$. This definition is primarily introduced to ensure a key "stability" property given next in Proposition 4.7, which will have many nice consequences.

Definition 4.5 (Chainable pair of patterns, operator $*$ on patterns). *Two patterns $\boldsymbol{\pi}_1 := (a_1, b_1, c_1, d_1)$ and $\boldsymbol{\pi}_2 := (a_2, b_2, c_2, d_2)$ are* chainable *if:*

1. $\frac{a_1 c_1}{a_2} = \frac{b_2 d_2}{d_1}$ *and this quantity*[3]*, denoted $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, is an integer;*

[3]As we will see, it plays the role of a rank, hence the choice of r to denote it.

2. $a_1 \mid a_2$ *and* $d_2 \mid d_1$.

We also say that the pair $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *is chainable. Observe that we always have* $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = c_1(a_1/a_2) = b_2(d_2/d_1) \leq \min(b_2, c_1)$, *and* $a_1c_1d_1 = a_2b_2d_2$. *We define the operator* $*$ *on the set of chainable pairs of patterns as follows: if* $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *is chainable, then*

$$\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2 := \left(a_1, \frac{b_1d_1}{d_2}, \frac{a_2c_2}{a_1}, d_2\right) \in \mathbb{N}^4. \tag{4.3}$$

Note that even though the definition of $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ involves the quotient a_1/a_2 (and d_2/d_1), assumption 2 in Definition 4.5 is indeed that a_1 divides a_2 (and d_2 divides d_1).

Remark 4.6. The order $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ in the definition matters, *i.e.*, this property is not symmetric: the chainability of $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ does not imply that of $(\boldsymbol{\pi}_2, \boldsymbol{\pi}_1)$. Moreover, by the first condition of Definition 4.5, a chainable pair is indeed an architecture in the sense of Definition 4.4.

Definition 4.5 comes with the following two key results.

PROPOSITION 4.7. *If* $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *is chainable, then:*

$$\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2} = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)\mathbf{S}_{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}. \tag{4.4}$$

The proof is deferred to Appendix B.1. The equality (4.4) was proved in [44, Lemma 3.4] for the choice $\boldsymbol{\pi}_1 = (2^{\ell-1}, 2, 2, 2^{L-\ell})$ and $\boldsymbol{\pi}_2 = (2^\ell, 2, 2, 2^{L-\ell-1})$, for any integer $L \geq 2$ and $\ell \in [\![L-1]\!]$. Proposition 4.7 extends (4.4) to *all* chainable pairs $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$.

Chainability and Definition 4.1 imply that $\forall (\mathbf{X}_1, \mathbf{X}_2) \in \Sigma^{\boldsymbol{\pi}_1} \times \Sigma^{\boldsymbol{\pi}_2}$, $\mathbf{X}_1\mathbf{X}_2 \in \Sigma^{(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2)}$, i.e., a product of Kronecker-sparse factors with chainable patterns $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ is still a Kronecker-sparse factor, with pattern $\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2$. Moreover, the matrix supports corresponding to a pair of chainable patterns also satisfy useful many interesting properties related to Definition 3.3 and Theorem 3.4, as shown in the following result proved in Appendix B.2:

LEMMA 4.8. *If* $\boldsymbol{\beta} := (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *is chainable then (with the notations of Definition* 3.3*) for each* $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ *we have*

1. $R_P = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1}[:, i])$ *and* $C_P = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_2}[i, :])$ *for every* $i \in P$.
2. *The sets* $R_P \times C_P$, $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ *are pairwise disjoint.*
3. $|P| = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, $|R_P| = b_1$ *and* $|C_P| = c_2$ *(with* $\boldsymbol{\pi}_i = (a_i, b_i, c_i, d_i)$*).*
4. $\mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}) = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2}) = \cup_{P \in \mathcal{P}} R_P \times C_P$.

LEMMA 4.9 (Associativity of $*$). *If* $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *and* $(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ *are chainable, then*

1. $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3)$ *and* $(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ *are chainable;*
2. $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *and* $r(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$*;*
3. $\boldsymbol{\pi}_1 * (\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = (\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2) * \boldsymbol{\pi}_3$.

The proof of Lemma 4.9 is deferred to Appendix B.3. We can now extend the definition of chainability to a general architecture $\boldsymbol{\beta}$ of arbitrary depth $L \geq 1$.

DEFINITION 4.10 (Chainable architecture). *An architecture* $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$, $L \geq 2$, *is* chainable *if* $\boldsymbol{\pi}_\ell$ *and* $\boldsymbol{\pi}_{\ell+1}$ *are chainable for each* $\ell \in [\![L-1]\!]$ *in the sense of Definition* 4.5. *We then denote* $\mathbf{r}(\boldsymbol{\beta}) = (r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}))_{\ell=1}^{L-1} \in \mathbb{N}^{L-1}$. *By convention any architecture of depth* $L = 1$ *is also chainable.*

EXAMPLE 4.11. *One can check that the square dyadic butterfly architecture (resp. the Monarch architecture), cf. Example* 4.2, *are chainable, with* $\mathbf{r}(\boldsymbol{\beta}) = (1, \ldots, 1)$ *(resp.* $\mathbf{r}(\boldsymbol{\beta}) = (1)$*). They are particular cases of the* 5*-parameter deformable butterfly architecture of [27], which is chainable with* $\mathbf{r}(\boldsymbol{\beta}) = (1, \ldots, 1)$. *In contrast, the*

Kaleidoscope architecture of depth $2L$ *with* $L \geq 2$ *of Table* 1 *is* not *chainable, because for* $\ell = L+1$ *we have* $\boldsymbol{\pi}_\ell = (2^{L-1}, 2, 2, 1)$, $\boldsymbol{\pi}_{\ell+1} = (2^{L-2}, 2, 2, 2)$, *and this pair is not chainable since* 2^{L-1} *does not divide* 2^{L-2}.

We state in the following some useful properties of chainable architectures.

LEMMA 4.12. *If* $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ *with* $L \geq 2$ *is chainable then* $\mathcal{B}^{\boldsymbol{\beta}} \subseteq \Sigma^{(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L)}$, *with*

$$\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = \left(a_1, \frac{b_1 d_1}{d_L}, \frac{a_L c_L}{a_1}, d_L\right). \tag{4.5}$$

Partial proof. Proposition 4.7 yields $\mathcal{B}^{\boldsymbol{\beta}} \subseteq \Sigma^{(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L)}$ when $L = 2$. This extends to any $L \geq 2$ by an induction. We prove (4.5) in Appendix B.4. ∎

Remark 4.13. As a consequence of this lemma, if the first pattern $\boldsymbol{\pi}_1$ of a chainable architecture $\boldsymbol{\beta}$ satisfies $a_1 > 1$ then all matrices in $\mathcal{B}^{\boldsymbol{\beta}}$ have a support included in $\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L}$, which has zeroes outside its main block diagonal structure (see Figure 1). A similar remark holds when $d_L > 1$, and in both cases we conclude that $\mathcal{B}^{\boldsymbol{\beta}}$ does not contain any dense matrix where all entries are nonzero. In contrast, when $a_1 = d_L = 1$, it is known for specific architectures that some dense matrices do belong to $\mathcal{B}^{\boldsymbol{\beta}}$. This is notably the case when $\boldsymbol{\beta}$ is the square dyadic butterfly architecture or the Monarch architecture (see Example 4.2): then we have $\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = (a_1, m, n, d_L) = (1, m, n, 1)$ for some integers m, n, and indeed the Hadamard (or the DFT matrix, up to bit-reversal permutation of its columns, cf. [5]) is a dense matrix belonging to $\mathcal{B}^{\boldsymbol{\beta}}$.

Next we state an essential property of chainable architectures. It builds on and extends Lemma 4.9, and corresponds to a form of *stability* under pattern multiplication that will serve as a cornerstone to support the introduction of hierarchical algorithms.

LEMMA 4.14. *If* $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ *is chainable then for each* $1 \leq q \leq s < t \leq L$, *the patterns* $(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s)$ *and* $(\boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t)$ *are well-defined and chainable with* $r(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t) = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$.

The proof is deferred to Appendix B.5.

4.3. Non-redundancy. A first version of our proposed hierarchical factorization algorithm (expressed recursively in Algorithm 5.1) will be applicable to any chainable architecture $\boldsymbol{\beta}$. However, establishing approximation guarantees as in Equation (1.3) will require a variant of this algorithm (Algorithm 6.1) involving certain orthonormalization steps, which are only well-defined if the architecture $\boldsymbol{\beta}$ satisfies an additional *non-redundancy* condition. Fortunately, any redundant architecture $\boldsymbol{\beta}$ can be transformed (Proposition 4.21) into an expressively equivalent architecture $\boldsymbol{\beta}'$ (*i.e.*, $\mathcal{B}^{\boldsymbol{\beta}'} = \mathcal{B}^{\boldsymbol{\beta}}$) with reduced number of parameters ($\|\boldsymbol{\beta}'\|_0 \leq \|\boldsymbol{\beta}'\|_0$) thanks to Algorithm 4.1 below. This will be instrumental in introducing the proving approximation guarantees of the final butterfly algorithm Algorithm 6.1 applicable to *any (redundant or not) chainable architecture*.

To define redundancy of an architecture we begin by considering elementary pairs.

DEFINITION 4.15 (Redundant architecture). *A chainable pair of patterns* $\boldsymbol{\pi}_1 = (a_1, b_1, c_1, d_1)$ *and* $\boldsymbol{\pi}_2 = (a_2, b_2, c_2, d_2)$ *is* redundant *if* $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) \geq \min(b_1, c_2)$ *(i.e., if* $a_1 c_1 \geq a_2 c_2$ *or* $b_2 d_2 \geq b_1 d_1$*). A chainable architecture* $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$, $L = |\boldsymbol{\beta}| \geq 1$, *is* redundant *if there exists* $\ell \in [\![L-1]\!]$ *such that* $(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1})$ *is redundant. Observe that by definition, any chainable architecture with* $|\boldsymbol{\beta}| = 1$ *is non-redundant.*

Remark 4.16. By Definition 4.5 we always have $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) \leq \min(b_2, c_1)$ for a chainable pair $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, hence a redundant one satisfies $\min(b_1, c_2) \leq r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) \leq \min(b_2, c_1)$. A *non-redundant* one satisfies $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) \leq \min(b_1 - 1, c_2 - 1, b_2, c_1)$.

LEMMA 4.17. *If $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ is chainable and non-redundant then, for any $1 \leq r \leq s < t \leq L$, the pair $(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t)$ is chainable and non-redundant.*

The proof is deferred to Appendix B.6.

EXAMPLE 4.18. *The architecture $\boldsymbol{\beta} := (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) := ((1, m, r, 1), (1, r, n, 1))$ is chainable, with $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = r$. The set $\mathcal{B}^{\boldsymbol{\beta}}$ is the set of $m \times n$ matrices of rank at most r. $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ is redundant if $r \geq \min(m, n)$. We observe that on this example redundancy corresponds to the case where $\mathcal{B}^{\boldsymbol{\beta}}$ is the set of* all *$m \times n$ matrices.*

A (chainable and) redundant architecture is as expressive as a smaller chainable architecture with less parameters. This is first proved for pairs, i.e., $\boldsymbol{\beta} := (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$.

In order to do this we need the following result, characterizing precisely the set of matrices $\mathcal{B}^{\boldsymbol{\beta}} = \{\mathbf{X}_1 \mathbf{X}_2 \,|\, \mathbf{X}_i \in \Sigma^{\boldsymbol{\pi}_i}, i \in [\![2]\!]\}$ as the set of matrices having a support included in $\mathbf{S}_{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}$ and with selected low-rank blocks. It is proved in Appendix B.7.

LEMMA 4.19. *Let $\boldsymbol{\beta} := (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ be chainable, and (with the notations of Definition 3.3) consider the following set of matrices of size equal to those in $\mathcal{B}^{\boldsymbol{\beta}}$:*

$$\mathcal{A}^{\boldsymbol{\beta}} := \{\mathbf{A} : \mathtt{rank}\big(\mathbf{A}[R_P, C_P]\big) \leq r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2),\ \forall P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})\}. \tag{4.6}$$

We have

$$\mathcal{B}^{\boldsymbol{\beta}} = \Sigma^{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2} \cap \mathcal{A}^{\boldsymbol{\beta}}. \tag{4.7}$$

LEMMA 4.20. *Consider a chainable pair $\boldsymbol{\beta} = (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. If $\boldsymbol{\beta}$ is redundant, then the single-factor architecture $\boldsymbol{\beta}' = (\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2)$ satisfies :*

1. $\mathcal{B}^{\boldsymbol{\beta}} = \Sigma^{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2} = \mathcal{B}^{\boldsymbol{\beta}'}$.
2. $\|\boldsymbol{\beta}'\|_0 = \|\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2\|_0 < \|\boldsymbol{\pi}_1\|_0 + \|\boldsymbol{\pi}_2\|_0 = \|\boldsymbol{\beta}\|_0$.

Proof. By Lemma 4.19 we have $\mathcal{B}^{\boldsymbol{\beta}} = \Sigma^{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2} \cap \mathcal{A}^{\boldsymbol{\beta}}$ and by Lemma 4.8 we have $|R_P| = b_1, |C_P| = c_2$ for each $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$. The first claim follows from the fact that $\mathcal{A}^{\boldsymbol{\beta}}$ is the set of all matrices of appropriate size: indeed for any such matrix $\mathbf{A}$, the block $\mathbf{A}[R_P, C_P]$ is of size $b_1 \times c_2$ hence its rank is at most $\min(b_1, c_2)$ which is smaller than or equal to $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ since $\boldsymbol{\beta}$ is redundant. By definition of $\mathcal{A}^{\boldsymbol{\beta}}$ this shows that $\mathbf{A} \in \mathcal{A}^{\boldsymbol{\beta}}$. For the second claim, by Definition 4.4 of $\|\boldsymbol{\beta}\|_0$ and $\|\boldsymbol{\beta}'\|_0$, we only need to prove the strict inequality. Since $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ is (chainable and) redundant, we have either $a_1 c_1 \geq a_2 c_2$ or $b_2 d_2 \geq b_1 d_1$, hence by Lemma 4.3 and Equation (4.3) we obtain $\|\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2\|_0 = a_2 c_2 b_1 d_1 < a_1 c_1 b_1 d_1 + a_2 c_2 b_2 d_2 = \|\boldsymbol{\pi}_1\|_0 + \|\boldsymbol{\pi}_2\|_0$. ∎

Lemma 4.20 serves as a basis to define Algorithm 4.1, which replaces any chainable (and possibly redundant) architecture by a "smaller" non-redundant one.

PROPOSITION 4.21. *For any chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$, Algorithm 4.1 stops in finitely many iterations and returns an architecture $\boldsymbol{\beta}'$ such that:*

1. *$\boldsymbol{\beta}'$ is chainable and non-redundant, and either a single factor architecture $\boldsymbol{\beta}' = (\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L)$, or a multi-factor one $\boldsymbol{\beta}' = (\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{\ell_1}, \boldsymbol{\pi}_{\ell_1+1} * \ldots * \boldsymbol{\pi}_{\ell_2}, \ldots, \boldsymbol{\pi}_{\ell_p+1} * \ldots * \boldsymbol{\pi}_L)$ for some indices $1 \leq \ell_1 < \ldots < \ell_p < L$ with $p \in [\![1, L-1]\!]$;*
2. *$\mathcal{B}^{\boldsymbol{\beta}'} = \mathcal{B}^{\boldsymbol{\beta}}$;*
3. *$\|\boldsymbol{\beta}'\|_0 \leq \|\boldsymbol{\beta}\|_0$.*

Algorithm 4.1 Architecture redundancy removal algorithm

Require: A chainable $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$
Ensure: A chainable and non-redundant $\boldsymbol{\beta}' = (\boldsymbol{\pi}'_\ell)_{\ell=1}^{L'}$ $(1 \leq L' \leq L)$
1: $\boldsymbol{\beta}' \leftarrow \boldsymbol{\beta}$.
2: **while** $\boldsymbol{\beta}'$ is redundant (cf. Definition 4.15) **do**
3: $\quad (\boldsymbol{\pi}'_\ell)_{\ell=1}^{L'} \leftarrow \boldsymbol{\beta}'$
4: $\quad \ell \leftarrow$ an integer ℓ such that $(\boldsymbol{\pi}'_\ell, \boldsymbol{\pi}'_{\ell+1})$ is redundant (cf. Definition 4.15)
5: $\quad \boldsymbol{\beta}' \leftarrow (\boldsymbol{\pi}'_1, \ldots, \boldsymbol{\pi}'_{\ell-1}, \boldsymbol{\pi}'_\ell * \boldsymbol{\pi}'_{\ell+1}, \boldsymbol{\pi}'_{\ell+2}, \ldots, \boldsymbol{\pi}'_{L'})$
6: **end while**
7: **return** $\boldsymbol{\beta}'$

Proof. Algorithm 4.1 terminates since $|\boldsymbol{\beta}'|$ decreases at each iteration. At each iteration, the updated $\boldsymbol{\beta}'$ is obtained by replacing a chainable redundant pair $(\boldsymbol{\pi}'_\ell, \boldsymbol{\pi}'_{\ell+1})$ by a single pattern $(\boldsymbol{\pi}'_\ell * \boldsymbol{\pi}'_{\ell+1})$. The architecture $\boldsymbol{\beta}'$ remains chainable by Lemma 4.14 and by chainability of $\boldsymbol{\beta}$, hence the algorithm can continue with no error. Due to the condition of the "while" loop, the returned $\boldsymbol{\beta}'$ is either non-redundant with $|\boldsymbol{\beta}'| > 1$, or $|\boldsymbol{\beta}'| = 1$ in which case it is in fact also non-redundant by Definition 4.15. This yields the first condition (a formal proof of the final form of $\boldsymbol{\beta}'$ can be done by an easy but tedious induction left to the reader). Moreover, a straightforward consequence of Lemma 4.20 is that the update of $\boldsymbol{\beta}'$ in line 5 does not change $\mathcal{B}^{\boldsymbol{\beta}'}$, and it strictly decreases $\|\boldsymbol{\beta}'\|_0$ if the condition of the "while" loop is met at least once (otherwise the algorithm outputs $\boldsymbol{\beta}' = \boldsymbol{\beta}$). This yields the two other properties. ∎

In particular, Algorithm 4.1 applied to a redundant architecture $\boldsymbol{\beta}$ in Example 4.18 returns $\boldsymbol{\beta}' = ((1, m, r, 1) * (1, r, n, 1)) = ((1, m, n, 1))$.

4.4. Constructing a chainable architecture for a target matrix size. It is natural to wonder what (non-redundant) chainable architectures $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$ allow to implement dense matrices of a prescribed size. This is the object of the next lemma, which is proved in Appendix B.8.

LEMMA 4.22. *Consider integers $m, n, L \geq 2$. If $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$ is a chainable architecture such that $\mathcal{B}^{\boldsymbol{\beta}}$ is made of $m \times n$ matrices and contains at least one dense matrix, then there exists a factorization of m (resp. of n) into L integers q_ℓ (resp. p_ℓ), $1 \leq \ell \leq L$, and a sequence of $L-1$ integers r_ℓ, $1 \leq \ell \leq L-1$ such that: with the convention $r_0 = r_L = 1$, for each $1 \leq \ell \leq L$ we have $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$ with*

$$a_\ell = \prod_{j=1}^{\ell-1} p_\ell, \quad \text{and } d_\ell = \prod_{j=\ell+1}^{L} q_\ell, \tag{4.8}$$

$$b_\ell = q_\ell r_{\ell-1}, \quad \text{and } c_\ell = p_\ell r_\ell. \tag{4.9}$$

Vice-versa, any architecture defined as above with integers such that $n = \prod_{\ell=1}^{L} p_\ell$ and $m = \prod_{\ell=1}^{L} q_\ell$ is chainable, and the set $\mathcal{B}^{\boldsymbol{\beta}} \subseteq \mathbb{R}^{m \times n}$ contains at least one dense matrix.

The architecture $\boldsymbol{\beta}$ is non-redundant if, and only if, $r_1 < q_1$, $r_{L-1} < p_L$, and

$$\frac{1}{p_\ell} < \frac{r_\ell}{r_{\ell-1}} < q_\ell, \ 2 \leq \ell \leq L-1. \tag{4.10}$$

The proof of the following corollary is straightforward and left to the reader.

COROLLARY 4.23. *Consider integers $m, n \geq 2$ and their integer factorizations into $L \geq 2$ integers p_ℓ, q_ℓ as in Lemma* 4.22.

- *If $p_\ell \geq 2$ and $q_\ell \geq 2$ for every $1 \leq \ell \leq L$, then there exists a choice of integers r_ℓ, $1 \leq \ell \leq L-1$ such that the construction of Lemma* 4.22 *is non-redundant.*
- *If either $q_1 = 1$, $p_L = 1$, or $p_\ell q_\ell = 1$ for some $2 \leq \ell \leq L-1$, then no choice of r_ℓ allows us to obtain a non-redundant architecture.*

As a consequence, given a matrix size $m \times n$, a chainable architecture compatible with this matrix size can be built via the following steps:

1. choosing integer sequences $\boldsymbol{p} := (p_\ell)_{\ell=1}^L$, $\boldsymbol{q} := (q_\ell)_{\ell=1}^L$ that factorize n and m (optionally: with the condition that $p_\ell \geq 2$ and $q_\ell \geq 2$ for every ℓ);
2. choosing an integer sequence $\boldsymbol{r} := (r_\ell)_{\ell=1}^{L-1}$ (optionally: with the condition that $r_1 < q_1$, $r_{L-1} < p_L$, and (4.10) holds with $r_0 = r_L := 1$ by convention);
3. defining of $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$ using (4.8)-(4.9).

Instead of imposing non-redundancy constraints, it is also possible to build a possibly redundant architecture and to exploit the redundancy removal algorithm (Algorithm 4.1).

Remark 4.24. Mathematically oriented readers may notice that for prime m and/or n there are few compatible butterfly architectures. Extending the concepts of this paper to such dimensions would allow to cover known fast transforms for prime dimensions [38]. Nevertheless, typical matrix dimensions in practical applications are composite and lead to many more choices. For instance, the dimensions of weight matrices in the vision transformer architecture [8] are commonly 768, 1024, 1280, 3072, 4096, 5120, which enable many possible choices of chainable architectures for implementing dense matrices, beyond the Monarch architecture [4] that was used specifically to accelerate such neural networks. This will be illustrated in Subsection 8.3.

5. Hierarchical algorithm under the chainability condition. We show how the hierarchical algorithm in [20, 44], initially introduced for specific (square dyadic) architectures, can be directly extended to the case where $\boldsymbol{\beta}$ is *any* chainable architecture. The case where $L = |\boldsymbol{\beta}| = 1$ is trivial since Problem (1.1) is then simply solved by setting $\mathbf{X}_1$ to be a copy of $\mathbf{A}$ where all entries outside of the prescribed support are set to zero. We thus focus on $L \geq 2$ and start with $\boldsymbol{\beta}$ of depth $L = 2$ before considering arbitrary $L \geq 2$.

5.1. Case with $L = 2$ factors. Problem (1.1) with an architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ is simply an instance of Problem (3.1) with $(\mathbf{L}, \mathbf{R}) = (\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$.

LEMMA 5.1. *Consider the pair of supports $(\mathbf{L}, \mathbf{R}) = (\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ associated to any (*not necessarily chainable*) architecture $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. The assumptions of Theorem* 3.4 *hold, hence for* any *architecture $\boldsymbol{\beta}$ of depth $|\boldsymbol{\beta}| = 2$, the two-factor fixed-support matrix factorization algorithm (Algorithm* 3.1*) returns an optimal solution to the corresponding instance of Problem* (1.1).

Proof. As in the proof of Lemma 4.8, the column supports $\{\mathbf{S}_{\boldsymbol{\pi}_1}[:, j]\}_j$ (resp. the row supports $\{\mathbf{S}_{\boldsymbol{\pi}_2}[i, :]\}_i$) are pairwise disjoint or identical. Hence the components $\mathbf{U}_i$ of $\varphi(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ are pairwise disjoint or identical (if their column *and* row supports coincide).

5.2. Case with $L \geq 2$ factors. Consider now Problem (1.1) associated with a *chainable* architecture $\boldsymbol{\beta}$ of depth $L := |\boldsymbol{\beta}| \geq 2$, and a given target matrix $\mathbf{A}$. A first proposition of hierarchical algorithm, introduced in Algorithm 5.1, is a direct adaptation to our framework of previous algorithms [20, 44]. It computes an approximate solution by performing successive two-factor matrix factorization in a certain hierarchical order

Algorithm 5.1 Hierarchical factorization algorithm – recursive version

Require: $\mathbf{A} \in \mathbb{C}^{m\times n}$, chainable $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$, factor-bracketing tree $\mathcal{T}$
Ensure: $\texttt{factors} \in \Sigma^{\boldsymbol{\beta}}$
1: **if** $L = 1$ **then**
2: **return** $(\mathbf{A} \odot \mathbf{S}_{\boldsymbol{\pi}_1})$ ▷{$\odot$ is the Hadamard product.}
3: **end if**
4: $[\![1, s]\!], [\![s+1, L]\!] \leftarrow$ two children of the root $[\![1, L]\!]$ of $\mathcal{T}$
5: $(\mathcal{T}_{\text{left}}, \mathcal{T}_{\text{right}}) \leftarrow$ the corresponding left and right subtrees of $\mathcal{T}$
6: $(\boldsymbol{\pi}_{\text{left}}, \boldsymbol{\pi}_{\text{right}}) \leftarrow (\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_L)$
7: $(\mathbf{X}_{[\![1,s]\!]}, \mathbf{X}_{[\![s+1,L]\!]}) \leftarrow$ Algorithm 3.1$(\mathbf{A}, \mathbf{S}_{\boldsymbol{\pi}_{\text{left}}}, \mathbf{S}_{\boldsymbol{\pi}_{\text{right}}})$
8: $\texttt{left\_factors} \leftarrow$ Algorithm 5.1$(\mathbf{X}_{[\![1,s]\!]}, (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_s), \mathcal{T}_{\text{left}})$
9: $\texttt{right\_factors} \leftarrow$ Algorithm 5.1$(\mathbf{X}_{[\![s+1,L]\!]}, (\boldsymbol{\pi}_{s+1}, \ldots, \boldsymbol{\pi}_L), \mathcal{T}_{\text{right}})$
10: $\texttt{factors} \leftarrow \texttt{left\_factors} \cup \texttt{right\_factors}$
11: **return** factors

that is described by a so-called factor-bracketing tree [44]. Further refinements of the algorithm will later be added to obtain approximation guarantees.

DEFINITION 5.2 (Factor-bracketing tree [44]). *A factor-bracketing tree of $[\![L]\!]$ for a given integer L is a binary tree such that:*

- *each node is an interval $[\![q, t]\!] := \{q, q+1, \ldots, t\}$ for $1 \le q \le t \le L$;*
- *the root is $[\![L]\!]$;*
- *every non-leaf node $[\![q, t]\!]$ for $q < t$ has $[\![q, s]\!]$ as its left child and $[\![s+1, t]\!]$ as its right child, for a certain s such that $q \le s < t$;*
- *a leaf is a singleton $[\![q, q]\!]$ for some $q \in [\![1, L]\!]$.*

Such a tree has exactly $(L-1)$ non-leaf nodes and L leaves.

Before exposing the limitations of Algorithm 5.1 and proposing fixes, let us briefly explain how it works with a focus on its main step in line 7. Consider any factor-bracketing tree $\mathcal{T}$. Algorithm 5.1 computes a matrix $\mathbf{X}_{[\![q,t]\!]} \in \Sigma^{\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t}$ for each node $[\![q, t]\!]$ in a recursive manner. $\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t$ is well-defined for any $1 \le q \le t \le L$ because $\boldsymbol{\beta}$ is chainable. At the root node, we set $\mathbf{X}_{[\![1,L]\!]} := \mathbf{A}$. At each non-leaf node $[\![q, t]\!]$ whose matrix $\mathbf{X}_{[\![q,t]\!]}$ is already computed during the hierarchical procedure, and with children $[\![q, s]\!]$ and $[\![s+1, t]\!]$, at line 7 we compute $(\mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]}) \in \Sigma^{\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s} \times \Sigma^{\boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t}$ that is solution to the following instance of the Fixed Support Matrix Factorization Problem (3.1):

$$\begin{aligned} &\text{Minimize} && \|\mathbf{X}_{[\![q,t]\!]} - \mathbf{X}_{[\![q,s]\!]}\mathbf{X}_{[\![s+1,t]\!]}\|_F \\ &\text{Subject to} && \texttt{supp}(\mathbf{X}_{[\![q,s]\!]}) \subseteq \mathbf{S}_{\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s}, \\ &&& \texttt{supp}(\mathbf{X}_{[\![s+1,t]\!]}) \subseteq \mathbf{S}_{\boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t}. \end{aligned} \tag{5.1}$$

Indeed, by Lemma 5.1, Problem (5.1) is solved by the two-factor fixed support matrix factorization algorithm (Algorithm 3.1), which yields line 7 in Algorithm 5.1. After computing $(\mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]})$, we repeat recursively the procedure on these two matrices independently, as per lines 8 and 9, until we obtain the butterfly factors $(\mathbf{X}_{[\![\ell,\ell]\!]})_{\ell=1}^{L} \in \Sigma^{\boldsymbol{\beta}}$ that yield an approximation $\hat{\mathbf{A}} := \mathbf{X}_{[\![1,1]\!]} \ldots \mathbf{X}_{[\![L,L]\!]} \in \mathcal{B}^{\boldsymbol{\beta}}$ of $\mathbf{A}$. In conclusion, Algorithm 5.1 is a greedy algorithm that seeks the optimal solution at each two-factor matrix factorization problem during the hierarchical procedure.

5.3. Algorithm 5.1 does not satisfy the theoretical guarantee (1.3)**.** However, the control of the approximation error in the form of (1.3) for Algorithm 5.1

in its current form is *impossible*, as illustrated in the following example.

EXAMPLE 5.3. *Consider* $\boldsymbol{\beta} = (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = \big((2^{\ell-1}, 2, 2, 2^{3-\ell})\big)_{\ell=1}^{3}$, *which is the square dyadic architecture of depth* $L = 3$. *Define* $\mathbf{A} := (\mathbf{D}\mathbf{S}_{\boldsymbol{\pi}_1})\mathbf{S}_{\boldsymbol{\pi}_2}\mathbf{S}_{\boldsymbol{\pi}_3}$ *where* $\mathbf{D}$ *is the diagonal matrix with diagonal entries* $(0, 1, 1, 1, 0, 1, 1, 1)$. *Hence,* $\mathbf{A} \in \mathcal{B}^{\boldsymbol{\beta}}$, *meaning that any algorithm with a theoretical guarantee* (1.3) *must output butterfly factors whose product is exactly* $\mathbf{A}$. *However, we claim that this is not the case of Algorithm* 5.1 *with the so-called left-to-right factor-bracketing tree of* $[\![1, 3]\!]$ *(defined as the tree where each left child is a singleton). To see why, let us apply this algorithm to* $\mathbf{A}$.

1. *In the first step, the hierarchical algorithm applies the two-factor fixed support matrix factorization algorithm (Algorithm* 3.1*) with input* $(\mathbf{A}, \mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3})$, *and returns* $(\mathbf{X}_{[\![1,1]\!]}, \mathbf{X}_{[\![2,3]\!]}) \in \Sigma^{\boldsymbol{\pi}_1} \times \Sigma^{\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3}$.
2. *At the second step (which is the last one), Algorithm* 3.1 *is applied to the input* $(\mathbf{X}_{[\![2,3]\!]}, \mathbf{S}_{\boldsymbol{\pi}_2}, \mathbf{S}_{\boldsymbol{\pi}_3})$, *and returns* $(\mathbf{X}_{[\![2,2]\!]}, \mathbf{X}_{[\![3,3]\!]}) \in \Sigma^{\boldsymbol{\pi}_2} \times \Sigma^{\boldsymbol{\pi}_3}$.

By construction, the first and the fifth row of $\mathbf{A}$ *are null, so the first step can return* many possible optimal solutions $(\mathbf{X}_{[\![1,1]\!]}, \mathbf{X}_{[\![2,3]\!]})$ *for the considered instance of Problem* (3.1), *such as* $\mathbf{X}_{[\![1,1]\!]} := \mathbf{D}\mathbf{S}_{\boldsymbol{\pi}_1}$ *and*

$$\mathbf{X}_{[\![2,3]\!]} = \begin{pmatrix} \mathbf{B} & \mathbf{0}_{4\times 4} \\ \mathbf{0}_{4\times 4} & \mathbf{B} \end{pmatrix} \quad \textit{with} \quad \mathbf{B} = \begin{pmatrix} \alpha & \beta & \gamma & \delta \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix},$$

where the scalars $\alpha, \beta, \gamma, \delta$ *can be arbitrary. Then, with the choice* $(\alpha, \beta, \gamma, \delta) = (1, -1, 1, -1)$, *one can check that the second step of the procedure will always output* $\mathbf{X}_{[\![2,2]\!]}$ *and* $\mathbf{X}_{[\![3,3]\!]}$ *such that* $\mathbf{X}_{[\![2,2]\!]}\mathbf{X}_{[\![3,3]\!]} \neq \mathbf{X}_{[\![2,3]\!]}$ *and* $\mathbf{X}_{[\![1,1]\!]}\mathbf{X}_{[\![2,2]\!]}\mathbf{X}_{[\![3,3]\!]} \neq \mathbf{A}$. *In conclusion, this example*[4] *shows that the output of Algorithm* 5.1 *cannot satisfy the theoretical guarantee* (1.3).

The inability to establish an error bound as in (1.3) for Algorithm 5.1 is due to the ambiguity for the choice of optimal factors $(\mathbf{X}_{[\![1,s]\!]}, \mathbf{X}_{[\![s+1,L]\!]})$ returned by the two-factor fixed support matrix factorization algorithm called at line 7 in Algorithm 5.1, cf. Remark 3.5. At each iteration, there are *multiple* optimal pairs of factors, and the choice at line 7 impacts subsequent factorizations in the recursive procedure. To guarantee an error bound of the type (1.3), Section 6 proposes a revision of Algorithm 5.1, where, among all the possible choices, the modified algorithm selects *specific* input matrices at lines 8 and 9.

6. Butterfly algorithm with error guarantees. We now propose a modification of the hierarchical algorithm (Algorithm 5.1) using *orthonormalization operations* that are novel in the context of butterfly factorization. It is based on an unrolled version of Algorithm 5.1 and will be endowed with error guarantees stated and proved in the next section.

6.1. Butterfly algorithm with orthonormalization. The factor-bracketing tree $\mathcal{T}$ in Algorithm 5.1 describes in which order the successive $L-1$ two-factor matrix factorization steps are performed, where $L := |\boldsymbol{\beta}|$. An equivalent way to describe this hierarchical order is to store a permutation $\sigma := (\sigma_\ell)_{\ell=1}^{L-1}$ of $[\![L-1]\!]$, by saving

[4] At first sight, this seems to contradict the so-called exact recovery property [20, 44] of Algorithm 5.1 in the case of square dyadic butterfly factorization. This is not the case, since the statement of these exact recovery results includes a technical assumption excluding matrices with zero columns/rows [44, Theorem 3.10], which is not satisfied by $\mathbf{A}$ here.

each splitting index $s \in [\![L-1]\!]$ that corresponds to the maximum integer in the left child $[\![q,s]\!]$ of each non-leaf node $[\![q,t]\!]$ of $\mathcal{T}$ (cf. Definition 5.2). We can then propose a non-recursive version of Algorithm 5.1, described in Algorithm 6.1, where for any non-empty integer interval we use the shorthand

$$\boldsymbol{\pi}_{[\![p,q]\!]} := \boldsymbol{\pi}_p * \ldots * \boldsymbol{\pi}_q. \tag{6.1}$$

Skipping (for now) lines 11-16, these two algorithms are equivalent when $\mathcal{T}$ and σ match, and thus the new version still suffers from the pitfall highlighted in Example 5.3 regarding error guarantees. This can however be overcome by introducing additional *pseudo-orthonormalization operations* (lines 11-16), involving orthogonalization of certain blocks of the matrix, as explicitly described in Algorithm C.1 in Appendix C.

Algorithm 6.1 Butterfly factorization algorithm – unrolled version (with pseudo-orthonormalization).
NB: removing blue code yields a non-recursive equivalent to Algorithm 5.1, applicable even to *redundant* $\boldsymbol{\beta}$, but not endowed with the guarantees of Theorems 7.2 and 7.3.)

Require: $\mathbf{A} \in \mathbb{C}^{m\times n}$, non-redundant, chainable $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$, permutation $(\sigma_\ell)_{\ell=1}^{L-1}$ of $[\![L-1]\!]$
Ensure: `factor` $\in \Sigma^{\boldsymbol{\beta}}$

1: **if** L $= 1$ **then**
2: **return** $(\mathbf{A} \odot \mathbf{S}_{\boldsymbol{\pi}_1})$ ▷{$\odot$ is the Hadamard product.}
3: **end if**
4: `partition` $\leftarrow (I_1)$ where we denote $I_1 = [\![1, L]\!]$
5: `factors` $\leftarrow (\mathbf{A})$
6: **for** $J = 1, \ldots, L-1$ **do**
7: $(I_j)_{j=1}^{J} \leftarrow$ `partition`
8: $(\mathbf{X}_{I_j})_{j=1}^{J} \leftarrow$ `factors`
9: $s := \sigma_J$
10: $j \leftarrow$ the unique $j \in [\![J]\!]$ such that $I_j := [\![q,t]\!] \ni s$
11: **for** $k = 1, \ldots, j-1$ **do**
12: $(\mathbf{X}_{I_k}, \mathbf{X}_{I_{k+1}}) \leftarrow$ Algorithm C.1$(\boldsymbol{\pi}_{I_k}, \boldsymbol{\pi}_{I_{k+1}}, \mathbf{X}_{I_k}, \mathbf{X}_{I_{k+1}}, \texttt{column})$
13: **end for**
14: **for** $k = J, \ldots, j+1$ **do**
15: $(\mathbf{X}_{I_{k-1}}, \mathbf{X}_{I_k}) \leftarrow$ Algorithm C.1$(\boldsymbol{\pi}_{I_{k-1}}, \boldsymbol{\pi}_{I_k}, \mathbf{X}_{I_{k-1}}, \mathbf{X}_{I_k}, \texttt{row})$
16: **end for**
17: $(\boldsymbol{\pi}_{\text{left}}, \boldsymbol{\pi}_{\text{right}}) \leftarrow (\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t)$
18: $(\mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]}) \leftarrow$ Algorithm 3.1$(\mathbf{X}_{I_j}, \mathbf{S}_{\boldsymbol{\pi}_{\text{left}}}, \mathbf{S}_{\boldsymbol{\pi}_{\text{right}}})$
19: `partition` $\leftarrow (I_1, \ldots, I_{j-1}, [\![q,s]\!], [\![s+1,t]\!], I_{j+1}, \ldots, I_J)$
20: `factors` $\leftarrow (\mathbf{X}_{I_1}, \ldots, \mathbf{X}_{I_{j-1}}, \mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]}, \mathbf{X}_{I_{j+1}}, \ldots, \mathbf{X}_{I_J})$
21: **end for**
22: **return** `factors`

These pseudo-orthonormalization operations in this new butterfly algorithm (Algorithm 6.1) rescale the butterfly factors $(\mathbf{X}_{I_k})_{k=1}^{J}$ without changing their product, in order to make a specific choice of $\mathbf{X}_{[\![q,t]\!]}$ given as input to Algorithm 3.1 at line 18 during subsequent steps of the algorithm, while constructing factors $\mathbf{X}_{I_1}, \ldots, \mathbf{X}_{I_{j-1}}, \mathbf{X}_{I_{j+1}} \ldots, \mathbf{X}_{I_J}$ that are pseudo-orthonormal in the following sense.

Definition 6.1 (Left and right r-unitary factors). *Consider a pattern* $\boldsymbol{\pi} = (a, b, c, d)$ *and* $r \in \mathbb{N}$. *A* $\boldsymbol{\pi}$*-factor* $\mathbf{X}$ *(cf Definition* 4.1*) is left-*r*-unitary (resp. right-*

r-unitary) if $r \mid c$ (resp. $r \mid b$) and for any $\boldsymbol{\pi}'$-factor $\mathbf{Y}$ satisfying $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r$ (resp. $r(\boldsymbol{\pi}', \boldsymbol{\pi}) = r$), we have: $\|\mathbf{XY}\|_F = \|\mathbf{Y}\|_F$ (resp. $\|\mathbf{YX}\|_F = \|\mathbf{Y}\|_F$).

Remark 6.2. The notions of left/right-r-unitary factor introduced in Definition 6.1 are relaxed versions of the usual column/row orthonormality. In particular, a left/right-r-unitary factor is only required to preserve the Frobenius norm of a set of chainable factors upon left/right matrix multiplication. Therefore, a $\boldsymbol{\pi} = (a, b, c, d)$-factor (cf. Definition 4.1) with orthonormal columns (resp. rows) is left-r-unitary (resp. right-r-unitary) for any $r \mid c$ (resp. $r \mid b$). The other implication is not true since $\frac{1}{\sqrt{2}}\mathbf{I}_2 \otimes \mathbf{1}_{2\times 2}$ is a left-1-unitary $(2, 2, 2, 1)$-factor but it is not a column orthonormal matrix. We name our operation *pseudo-orthonormalization* to avoid confusion with the usual orthonormalization operation.

More importantly, left/right-r-unitary notions also share certain properties with column/rows orthonormality such as the stability under matrix multiplication and norm preserving upon both left and right multiplication. These properties will be detailed in Appendix C.

Using Definition 6.1, we can describe the purpose of the pseudo-orthonormalization operation used in the new butterfly algorithm (Algorithm 6.1) as follows:

LEMMA 6.3. *At the J-th iteration of Algorithm* 6.1*, denote*[5] *$I_i = [\![q_i, t_i]\!]$ for any $i \in [\![J]\!]$. After pseudo-orthonormalization operations (cf. line 11-16 - Algorithm* 6.1*), the $\boldsymbol{\pi}_{I_i}$-factor $\mathbf{X}_{I_i}$ for $i = 1, \ldots, j-1$ is left-$r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{t_i+1})$-unitary, and the $\boldsymbol{\pi}_{I_i}$-factor $\mathbf{X}_{I_i}$ for $i = j+1, \ldots, J$ is right-$r(\boldsymbol{\pi}_{q_i-1}, \boldsymbol{\pi}_{q_i})$-unitary, where j is the integer defined in line 10.*

This result plays a key role in deriving a guarantee for Algorithm 6.1 in Section 7. It is proved in Appendix C.2.1.

Remark 6.4. As detailed in Appendix C the orthonormalization operations are well-defined only under the non-redundancy assumption. When the architecture $\boldsymbol{\beta}$ is redundant, by the redundancy removal algorithm (Algorithm 4.1) we can reduce it to a non-redundant architecture $\boldsymbol{\beta}'$ that is expressively equivalent to $\boldsymbol{\beta}$ (i.e, $\mathcal{B}^{\boldsymbol{\beta}'} = \mathcal{B}^{\boldsymbol{\beta}}$) and apply the algorithm to it. This yields an approximation $\mathbf{A} \approx \prod_{\ell=1}^{L'} \mathbf{X}'_\ell$ with $L' := |\boldsymbol{\beta}'|$ and $(\mathbf{X}_{\ell'})_{\ell'=1}^{L'} \in \Sigma^{\boldsymbol{\beta}'}$. By Lemma 4.20 we can then construct $(\mathbf{X}_\ell)_{\ell=1}^{L} \in \Sigma^{\boldsymbol{\beta}}$ that yields an approximation $\mathbf{A} \approx \prod_{\ell=1}^{L} \mathbf{X}_\ell$ with the same approximation error as $\prod_{\ell=1}^{L'} \mathbf{X}'_\ell$. Therefore, in the following we only consider non-redundant chainable architectures.

6.2. Complexity analysis. It is not hard to see that the proposed butterfly algorithm (Algorithm 6.1) has polynomial complexity with respect to the sizes of the butterfly factors and the target matrix, since they only perform a polynomial number of standard matrix operations such as matrix multiplication, QR and SVD decompositions.

THEOREM 6.5 (Complexity analysis). *Consider a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$ where $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$, and a target matrix $\mathbf{A}$ of size $m \times n$. Define*

$$M_{\boldsymbol{\beta}} := \max_{\ell \in [\![L]\!]} a_\ell c_\ell, \quad N_{\boldsymbol{\beta}} = \max_{\ell \in [\![L]\!]} b_\ell d_\ell.$$

When $\boldsymbol{\beta}$ is non-redundant we have $M_{\boldsymbol{\beta}} \leq m$ and $N_{\boldsymbol{\beta}} \leq n$. With the vector $\mathbf{r}(\boldsymbol{\beta})$ of Definition 4.10*, the complexity is bounded by:*

- *$\mathcal{O}(\|\mathbf{r}(\boldsymbol{\beta})\|_1 M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}})$ for Algorithm* 5.1

[5] Observe that by construction $t_i = q_{i+1} - 1$ whenever $i + 1 \in [\![J]\!]$.

- $\mathcal{O}\left((\|\mathbf{r}(\boldsymbol{\beta})\|_1 + |\boldsymbol{\beta}|^2\|\mathbf{r}(\boldsymbol{\beta})\|_\infty)mn\right)$ *for Algorithm* 6.1 *(with a non-redundant* $\boldsymbol{\beta}$*).*

The proof of Theorem 6.5 is in Appendix D. The complexity bounds in Theorem 6.5 are generic for any matrix size m, n, chainable $\boldsymbol{\beta}$ and factor-bracketing tree $\mathcal{T}$ (or equivalent permutation σ). They can be improved for specific $\boldsymbol{\beta}$. For example, in the case of the square dyadic butterfly, [20, 44] showed that the complexity of the hierarchical algorithm (Algorithm 5.1) is $\mathcal{O}(n^2)$ where $n = 2^L$ instead of $\mathcal{O}(\|\mathbf{r}(\boldsymbol{\beta})\|_1 n^2) = \mathcal{O}(n^2 \log n)$. This is optimal in the sense that it already matches the space complexity of the target matrix.

7. Guarantees on the approximation error. One of the main contributions of this paper is to show that the new butterfly algorithm (Algorithm 6.1) outputs an approximate solution to Problem (1.1) that satisfies an error bound of the type (1.3).

7.1. Main results. Our error bounds are based on the following relaxed problem.

DEFINITION 7.1 (First-level factorization). *Given a chainable* $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ *with* $L \geq 2$*, we define for each splitting index* $s \in [\![L-1]\!]$ *the two-factor "split" architecture:*

$$\boldsymbol{\beta}_s := (\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_L).$$

When $L = 2$ *we have* $\boldsymbol{\beta}_1 = \boldsymbol{\beta}$*. For any target matrix* $\mathbf{A}$ *we consider the problem*

$$E^{\boldsymbol{\beta}_s}(\mathbf{A}) := \min_{(\mathbf{X},\mathbf{Y}) \in \Sigma^{\boldsymbol{\beta}_s}} \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_F = \min_{\mathbf{B} \in \mathcal{B}^{\boldsymbol{\beta}_s}} \|\mathbf{A} - \mathbf{B}\|_F. \tag{7.1}$$

The following two theorems are the central theoretical results of this paper. The first one bounds the approximation error of Algorithm 6.1 in the general case where σ can be any permutation. The second one is a tighter bound specific to the case where σ is the identity permutation, corresponding to the so-called *unbalanced tree* of [44].

THEOREM 7.2 (Approximation error, arbitrary permutation σ, Algorithm 6.1). *Let* $\boldsymbol{\beta}$ *be a* non-redundant chainable *architecture of depth* $L \geq 2$*. For any target matrix* $\mathbf{A}$ *and permutation* σ *of* $[\![L-1]\!]$ *with* $L = |\boldsymbol{\beta}|$*, Algorithm* 6.1 *with inputs* $(\mathbf{A}, \boldsymbol{\beta}, \sigma)$ *returns butterfly factors* $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$ *such that*

$$\|\mathbf{A} - \mathbf{X}_1 \ldots \mathbf{X}_L\|_F \leq \sum_{k=1}^{L-1} E^{\boldsymbol{\beta}_{\sigma_k}}(\mathbf{A}). \tag{7.2}$$

THEOREM 7.3 (Approximation error, identity permutation σ, Algorithm 6.1). *Assume that* σ *is either the identity permutation,* $\sigma = (1, \ldots, L-1)$ *or its "converse",* $\sigma = (L-1, \ldots, 1)$*. Under the assumptions and with the notations of Theorem* 7.2*, Algorithm* 6.1 *with inputs* $(\mathbf{A}, \boldsymbol{\beta}, \sigma)$ *returns butterfly factors* $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$ *such that:*

$$\|\mathbf{A} - \mathbf{X}_1 \ldots \mathbf{X}_L\|_F^2 \leq \sum_{k=1}^{L-1} [E^{\boldsymbol{\beta}_k}(\mathbf{A})]^2. \tag{7.3}$$

For $L = 2$ both results yield $\|\mathbf{A} - \mathbf{X}_1\mathbf{X}_2\|_F \leq E^{\boldsymbol{\beta}}(\mathbf{A})$, i.e. the algorithm is optimal.

Before proving these theorems in Subsection 7.5, we state and prove their main consequences: the quasi-optimality of Algorithm 6.1, a "complementary low-rank" characterization of butterfly matrices, and the existence of an optimum for Problem (1.1) when $\boldsymbol{\beta}$ is chainable.

7.2. Quasi-optimality of Algorithm 6.1. The theorems imply that butterfly factors obtained via Algorithm 6.1 satisfy an error bound of the form (1.3).

THEOREM 7.4 (Quasi-optimality of Algorithm 6.1). *Let $\boldsymbol{\beta}$ be* any chainable *architecture of arbitrary depth $L := |\boldsymbol{\beta}| \geq 1$. For any target matrix $\mathbf{A}$, the outputs $(\mathbf{X}_\ell)_{\ell=1}^L$ of Algorithm* 6.1 *with inputs $(\mathbf{A}, \boldsymbol{\beta}, \sigma)$ for arbitrary permutation σ satisfy:*

$$\|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F \leq (\max(L,\, 2) - 1) E^{\boldsymbol{\beta}}(\mathbf{A}). \tag{7.4}$$

When σ is the identity permutation, the outputs also satisfy the finer bound:

$$\|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F \leq \sqrt{\max(L,\, 2) - 1}\, E^{\boldsymbol{\beta}}(\mathbf{A}). \tag{7.5}$$

For $L \in \{1, 2\}$ the output of Algorithm 6.1 is thus indeed optimal. Table 2 summarizes the consequences of Theorem 7.4 for some standard examples of chainable $\boldsymbol{\beta}$. The constant $C_{\boldsymbol{\beta}}$ scales linearly or sub-linearly with respect to $L = |\boldsymbol{\beta}|$, the number of factors. Since most part of the existing architectures have length $\mathcal{O}(\log n)$ with n the size of the matrix, the growth of $C_{\boldsymbol{\beta}}$ is very slow in many practical cases.

A result reminiscent of Theorem 7.4 appears in the quite different context of tensor train decomposition [35, Corollary 2.4]. The proof of Theorem 7.4 has a similar structure and is based on the following lemma. First, we use the fact that the errors in (7.1) lower bound the error in (1.3), by Definition 7.1.

LEMMA 7.5. *If the architecture $\boldsymbol{\beta}$ of depth $L := |\boldsymbol{\beta}|$ is chainable then*

$$\forall s \in [\![L-1]\!], \quad \mathcal{B}^{\boldsymbol{\beta}} \subseteq \mathcal{B}^{\boldsymbol{\beta}_s}. \tag{7.6}$$

Consequently, for any matrix $\mathbf{A}$ the quantity $E^{\boldsymbol{\beta}}(\mathbf{A})$ defined in (1.1) *satisfies:*

$$E^{\boldsymbol{\beta}}(\mathbf{A}) \geq \max_{1 \leq s \leq L-1} E^{\boldsymbol{\beta}_s}(\mathbf{A}). \tag{7.7}$$

Proof. If $\mathbf{B} \in \mathcal{B}^{\boldsymbol{\beta}}$, there exist $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$ such that $\mathbf{B} = \mathbf{X}_1 \dots \mathbf{X}_L$. By Lemma 4.12, $\mathbf{X}_1 \dots \mathbf{X}_s \in \Sigma^{(\boldsymbol{\pi}_1 * \dots * \boldsymbol{\pi}_s)}$ and $\mathbf{X}_{s+1} \dots \mathbf{X}_L \in \Sigma^{(\boldsymbol{\pi}_{s+1} * \dots * \boldsymbol{\pi}_L)}$. ∎

Proof of Theorem 7.4. We start by proving (7.4). We consider two possibilities for the depth $L := |\boldsymbol{\beta}|$ of the non-redundant, chainable architecture $\boldsymbol{\beta}$:

- If $L = 1$: we have $\boldsymbol{\beta} = \{\boldsymbol{\pi}\}$ for some pattern $\boldsymbol{\pi}$. The projection of $\mathbf{A}$ onto $\Sigma^{\boldsymbol{\pi}}$ is simply $\mathbf{A} \odot \mathbf{S}_{\boldsymbol{\pi}} \in \Sigma^{\boldsymbol{\pi}}$, which is exactly the output computed by the algorithm. Hence the obtained factor $\mathbf{X}_1$ satisfies $\|\mathbf{A} - \mathbf{X}_1\|_F = E^{\boldsymbol{\beta}}(\mathbf{A})$.
- If $L \geq 2$: by Lemma 7.5 and Theorem 7.2 $\|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F \leq (L-1) E^{\boldsymbol{\beta}}(\mathbf{A})$.

In both cases, we have $\|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F \leq (\max(L,2) - 1) E^{\boldsymbol{\beta}}(\mathbf{A})$. The proof for (7.5) is similar, the only difference being that: $\|\mathbf{A} - \mathbf{X}_1 \dots \mathbf{X}_L\|_F^2 \leq (\max(L,2) - 1)[E^{\boldsymbol{\beta}}(\mathbf{A})]^2$. The result is proved by taking the square root on both sides. ∎

7.3. Complementary low-rank characterization of butterfly matrices. Another important consequence of Theorem 7.2 is a characterization of matrices admitting an exact butterfly factorization associated with a chainable $\boldsymbol{\beta}$. This allows (when $\boldsymbol{\beta}$ is chainable) to *verify* whether or not a given matrix $\mathbf{A}$ admits exactly a butterfly factorization associated with $\boldsymbol{\beta}$, by checking the rank of a polynomial number of specific submatrices of $\mathbf{A}$. This is feasible using SVDs, and contrasts with the *synthesis* definition of $\mathcal{B}^{\boldsymbol{\beta}}$ given by (1.2), which is a priori harder to verify since it requires checking the *existence* of an exact factorization of $\mathbf{A}$.

DEFINITION 7.6 (Generalized complementary low-rank property). *Consider a chainable architecture $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$. A matrix $\mathbf{A}$ satisfies the* generalized complementary low-rank property *associated with $\boldsymbol{\beta}$ if it satisfies:*

1. $\texttt{supp}(\mathbf{A}) \subseteq \mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L}$;
2. $\texttt{rank}(\mathbf{A}[R_P, C_P]) \leq r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1})$ *for each* $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_\ell}, \mathbf{S}_{\boldsymbol{\pi}_{\ell+1} * \ldots * \boldsymbol{\pi}_L})$ *and* $\ell \in [\![L-1]\!]$ *(with the notations of Definition* 3.3*, Definition* 4.5*).*

We show in Corollary F.4 of Appendix F that this generalized definition indeed coincides with the notion of a complementary low-rank property (Definition F.2) from the literature [23], for every architecture $\boldsymbol{\beta}$ with patterns such that $a_1 = d_L = 1$, i.e., architectures such that $\mathcal{B}^{\boldsymbol{\beta}}$ contains some dense matrices, see Remark 4.13.

The following results show that a matrix admits an exact butterfly factorization associated with $\boldsymbol{\beta}$ if, and only if, it satisfies the associated generalized complementary low-rank property. Note that the complementary low-rank property induced by a chainable butterfly architecture requires the same low-rank constraint for all submatrices at the same level $\ell \in [\![L]\!]$, as opposed to the classical complementary low-rank property (Definition F.2) where these constraints can be different for the submatrices at a same level $\ell \in [\![L]\!]$.

Corollary 7.7 (Characterization of $\mathcal{B}^{\boldsymbol{\beta}}$ for chainable $\boldsymbol{\beta}$). *If* $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ *is chainable with* $L \geq 2$ *then, with the notations of Definition* 7.1 *and Lemma* 4.19*:*

$$\mathcal{B}^{\boldsymbol{\beta}} = \bigcap_{\ell=1}^{L-1} \mathcal{B}^{\boldsymbol{\beta}_\ell} = \Sigma^{(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L)} \cap \bigcap_{\ell=1}^{L-1} \mathcal{A}^{\boldsymbol{\beta}_\ell}. \tag{7.8}$$

Proof. The second equality in (7.8) is a reformulation based on Lemma 4.19, so it only remains to prove the first equality. The inclusion $\mathcal{B}^{\boldsymbol{\beta}} \subseteq \bigcap_{\ell=1}^{L-1} \mathcal{B}^{\boldsymbol{\beta}_\ell}$ is a consequence of Lemma 7.5. We now prove the other inclusion.

First, consider the case of a non-redundant $\boldsymbol{\beta}$. For $\mathbf{A} \in \bigcap_{\ell=1}^{L-1} \mathcal{B}^{\boldsymbol{\beta}_\ell}$, we have $E^{\boldsymbol{\beta}_\ell}(\mathbf{A}) = 0$ for each $\ell \in [\![L-1]\!]$. By Theorem 7.2, Algorithm 6.1 with inputs $(\mathbf{A}, \boldsymbol{\beta}, \sigma)$ and arbitrary permutation σ returns $(\mathbf{X}_\ell)_{\ell=1}^L \in \Sigma^{\boldsymbol{\beta}}$ such that $\|\mathbf{A} - \mathbf{X}_1 \ldots \mathbf{X}_L\|_F \leq \sum_{\ell=1}^{L-1} E^{\boldsymbol{\beta}_\ell}(\mathbf{A}) = 0$. Thus, $\|\mathbf{A} - \mathbf{X}_1 \ldots \mathbf{X}_L\|_F = 0$ and $\mathbf{A} = \mathbf{X}_1 \ldots \mathbf{X}_L \in \mathcal{B}^{\boldsymbol{\beta}}$. This proves $\bigcap_{\ell=1}^{L-1} \mathcal{B}^{\boldsymbol{\beta}_\ell} \subseteq \mathcal{B}^{\boldsymbol{\beta}}$.

For redundant $\boldsymbol{\beta}$, consider $\boldsymbol{\beta}' = (\boldsymbol{\pi}'_\ell)_{\ell=1}^{L'}$ returned by the redundancy removal algorithm (Algorithm 4.1) with input $\boldsymbol{\beta}$. By Proposition 4.21: $\mathcal{B}^{\boldsymbol{\beta}'} = \mathcal{B}^{\boldsymbol{\beta}}$. Moreover, by the same proposition, $\boldsymbol{\beta}'$ is of the form $(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{\ell_1}, \boldsymbol{\pi}_{\ell_1+1} * \ldots * \boldsymbol{\pi}_{\ell_2}, \ldots, \boldsymbol{\pi}_{\ell_p+1} * \ldots * \boldsymbol{\pi}_L)$ for some indices $1 \leq \ell_1 < \ldots < \ell_p < L$ with $p \in [\![L-1]\!]$. Therefore, for any $s \in [\![L'-1]\!]$, there exists $\ell(s) \in [\![L-1]\!]$ such that ${\boldsymbol{\beta}'}_s = \boldsymbol{\beta}_{\ell(s)}$, by associativity of the operator $*$ (Lemma 4.9). Thus,

$$\bigcap_{\ell=1}^{L-1} \mathcal{B}^{\boldsymbol{\beta}_\ell} \subseteq \bigcap_{s=1}^{L'-1} \mathcal{B}^{\boldsymbol{\beta}_{\ell(s)}} = \bigcap_{s=1}^{L'-1} \mathcal{B}^{{\boldsymbol{\beta}'}_s} = \mathcal{B}^{\boldsymbol{\beta}'} = \mathcal{B}^{\boldsymbol{\beta}},$$

where in the first equality we used the result proved above for non-redundant $\boldsymbol{\beta}'$. ∎

7.4. Existence of an optimum. Corollary 7.7 also allows us to prove the existence of optimal solutions for Problem (1.1) when $\boldsymbol{\beta}$ is chainable.

Theorem 7.8 (Existence of optimum in butterfly approximation). *If* $\boldsymbol{\beta}$ *is chainable, then for any target matrix* $\mathbf{A}$*, Problem* (1.1) *admits a minimizer.*

Proof. The set of matrices of rank smaller than a fixed constant is closed, and closed sets are stable under finite intersection, so by the characterization of $\mathcal{B}^{\boldsymbol{\beta}}$ from Corollary 7.7, the set $\mathcal{B}^{\boldsymbol{\beta}}$ is closed. Therefore, Problem (1.1) is equivalent to a projection problem on the non-empty (it contains the zero matrix) closed set $\mathcal{B}^{\boldsymbol{\beta}}$, hence it always admits a minimizer. ∎

The rest of the section is dedicated to the proofs of Theorems 7.2 and 7.3. Readers more interested in numerical aspects of the proposed butterfly algorithms can directly jump to Section 8.

7.5. Proof of Theorems 7.2 and 7.3. Consider an iteration number $J \in [\![L-1]\!]$, and denote $(I_k)_{k=1}^J$ the partition obtained after line 7 and $(\mathbf{X}_{I_k})_{k=1}^J$ the list `factors` obtained *after* the pseudo-orthonormalization operations in lines 11-16, at the J-th iteration of Algorithm 6.1. With $s := \sigma_J$ and j defined in line 10 of Algorithm 6.1 and $[\![q,t]\!] := I_j$, denote

$$\mathbf{X}_{\text{left}}^{(J)} := \mathbf{X}_{I_1} \dots \mathbf{X}_{I_{j-1}}, \qquad \mathbf{X}_{\text{right}}^{(J)} := \mathbf{X}_{I_{j+1}} \dots \mathbf{X}_{I_J}, \tag{7.9}$$

with the convention that $\mathbf{X}_{\text{left}}^{(J)}$ (resp. $\mathbf{X}_{\text{right}}^{(J)}$) is the identity matrix of size $a_1 b_1 d_1$ if $j = 1$ (resp. $a_L c_L d_L$ if $j = J$). We also denote $(\mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]})$ the matrices computed in line 18 of Algorithm 6.1, as well as

$$\mathbf{B}_J := \mathbf{X}_{\text{left}}^{(J)} \mathbf{X}_{[\![q,s]\!]} \mathbf{X}_{[\![s+1,t]\!]} \mathbf{X}_{\text{right}}^{(J)} \tag{7.10}$$

and $R_J := \|\mathbf{A} - \mathbf{B}_J\|_F$. Note that $\mathbf{B}_J$ is the product of butterfly factors in the list `factors` at the end of the iteration J (line 20). In particular, $\mathbf{B}_{L-1} \in \mathcal{B}^{\boldsymbol{\beta}}$ is the product of the butterfly factors returned by the algorithm after $L-1$ iterations. By convention we also define $\mathbf{B}_0 := \mathbf{A}$ and $R_0 := 0$.

Our goal is to control $R_{L-1} = \|\mathbf{A} - \mathbf{B}_{L-1}\|_F$. To this end, it is sufficient to track the evolution of the sequence $(\mathbf{B}_0, \dots, \mathbf{B}_{L-1})$. The following lemma enables a description for the relation between two consecutive matrices $\mathbf{B}_{J-1}$ and $\mathbf{B}_J$, $1 \le J \le L-1$.

LEMMA 7.9. *Consider a chainable architecture $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ and a partition of $[\![L]\!]$ into consecutive intervals $\{I_1, \dots, I_J\}$. For each $i \in [\![J]\!]$, let $I_i = [\![q_i, t_i]\!]$, $\mathbf{X}_{I_i}$ be a $(\boldsymbol{\pi}_{q_i} * \dots * \boldsymbol{\pi}_{t_i})$-factor and $\mathbf{B} := \mathbf{X}_{I_1} \dots \mathbf{X}_{I_J}$. Given $j \in [\![J-1]\!]$, if each $\mathbf{X}_{I_i}$ for $i = 1, \dots, j-1$ is left-$r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{t_i+1})$-unitary, and if each $\mathbf{X}_{I_i}$ for $i = j+1, \dots, J$ is right-$r(\boldsymbol{\pi}_{q_i-1}, \boldsymbol{\pi}_{q_i})$-unitary, then for any optimal factorization (solution of (5.1)) $(\mathbf{X}_{[\![q,s]\!]}, \mathbf{X}_{[\![s+1,t]\!]})$ of $\mathbf{X}_{I_j}$ with $q = q_j, t = t_j$, the matrix*

$$\mathbf{B}' := (\mathbf{X}_{I_1} \dots \mathbf{X}_{I_{j-1}}) \mathbf{X}_{[\![q,s]\!]} \mathbf{X}_{[\![s+1,t]\!]} (\mathbf{X}_{I_{j+1}} \dots \mathbf{X}_{I_J})$$

satisfies

$$\mathbf{B}' \in \mathcal{B}^{\boldsymbol{\beta}_s} \quad \text{and} \quad \|\mathbf{B}' - \mathbf{B}\|_F = E^{\boldsymbol{\beta}_s}(\mathbf{B}), \tag{7.11}$$

with $\boldsymbol{\beta}_s$ as in Definition 7.1.

This lemma (proved in Appendix E.1) has a direct corollary (obtained by combining it with Lemma 6.3, which ensures that each factor $\mathbf{X}_{I_i}$ is indeed r-unitary as needed).

LEMMA 7.10. *With the setting of Theorem 7.2, for $J \in [\![L-1]\!]$, the matrix $\mathbf{B}_J$ defined in (7.10) is a projection of $\mathbf{B}_{J-1}$ onto $\mathcal{B}^{\boldsymbol{\beta}_{\sigma_J}}$ (cf. Definition 7.1), i.e.,*

$$\mathbf{B}_J \in \mathcal{B}^{\boldsymbol{\beta}_{\sigma_J}} \quad \text{and} \quad \|\mathbf{B}_{J-1} - \mathbf{B}_J\|_F = E^{\boldsymbol{\beta}_{\sigma_J}}(\mathbf{B}_{J-1}).$$

These two lemmas show the role of the pseudo-orthonormalization, since without this operation and its consequence, i.e., Lemma 6.3, the conclusions of Lemma 7.10 would not hold. Thus, we can describe the sequence $\mathbf{B}_j$ for $j = 1, \dots, L-1$ as a sequence of subsequent projections:

$$\mathbf{B}_0 = \mathbf{A} \overset{\text{Proj}_{\mathcal{B}^{\boldsymbol{\beta}_{\sigma_1}}}}{\longrightarrow} \mathbf{B}_1 \overset{\text{Proj}_{\mathcal{B}^{\boldsymbol{\beta}_{\sigma_2}}}}{\longrightarrow} \dots \overset{\text{Proj}_{\mathcal{B}^{\boldsymbol{\beta}_{\sigma_{L-1}}}}}{\longrightarrow} \mathbf{B}_{L-1} \in \mathcal{B}^{\boldsymbol{\beta}}.$$

where Proj_S is the projection operator onto the set S. We note that since there might be more than one projector, Proj_S is a set-valued mapping.

Moreover, the sequence of architectures $(\boldsymbol{\beta}_s)_{s=1}^{L-1}$ defined in Definition 7.1 possesses another nice property, described in the following lemma:

LEMMA 7.11. *Consider a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^{L}$, a matrix $\mathbf{M}$ of appropriate size, and integers $s, q \in [\![L-1]\!]$. If $\mathbf{N}$ is a projection of $\mathbf{M}$ onto $\mathcal{B}^{\boldsymbol{\beta}_q}$ then*

$$E^{\boldsymbol{\beta}_s}(\mathbf{M}) \geq E^{\boldsymbol{\beta}_s}(\mathbf{N}) \tag{7.12}$$

Lemma 7.9 and Lemma 7.11 are proved in Appendix E.1 and Appendix E.2 respectively. In the following, we admit these results and prove Theorems 7.2 and 7.3.

Proof of Theorems 7.2 *and* 7.3. First, we prove that:

$$E^{\boldsymbol{\beta}_s}(\mathbf{A}) \geq E^{\boldsymbol{\beta}_s}(\mathbf{B}_J), \quad \forall s, J \in [\![L-1]\!]. \tag{7.13}$$

Indeed, for any $J \in [\![L-1]\!]$, by Lemma 7.10, $\mathbf{B}_J$ is a projection of $\mathbf{B}_{J-1}$ onto $\mathcal{B}^{\boldsymbol{\beta}_{\sigma_J}}$. Hence, by Lemma 7.11:

$$E^{\boldsymbol{\beta}_s}(\mathbf{B}_{J-1}) \geq E^{\boldsymbol{\beta}_s}(\mathbf{B}_J), \quad \forall s \in [\![L-1]\!].$$

Since $\mathbf{A} = \mathbf{B}_0$, applying the above inequality recursively yields (7.13).

We now derive the error bound in Theorem 7.2, which is true for an arbitrary permutation σ, as follows.

$$\|\mathbf{A} - \mathbf{B}_{L-1}\| = \|(\mathbf{B}_0 - \mathbf{B}_1) + \ldots + (\mathbf{B}_{L-2} - \mathbf{B}_{L-1})\|_F \leq \sum_{J=1}^{L-1} \|\mathbf{B}_{J-1} - \mathbf{B}_J\|_F$$

and for each $J \in [\![L-1]\!]$ it holds:

$$\|\mathbf{B}_{J-1} - \mathbf{B}_J\|_F \overbrace{=}^{\text{Lemma 7.10}} E^{\boldsymbol{\beta}_{\sigma_J}}(\mathbf{B}_{J-1}) \overbrace{\leq}^{\text{Equation (7.13)}} E^{\boldsymbol{\beta}_{\sigma_J}}(\mathbf{A})$$

and since σ is a permutation of $[\![L-1]\!]$ it holds $\sum_{J=1}^{L-1} E^{\boldsymbol{\beta}_{\sigma_J}}(\mathbf{A}) = \sum_{s=1}^{L-1} E^{\boldsymbol{\beta}_s}(\mathbf{A})$. The proof when σ is the identity or its "converse" is similar: we only replace the first equation by:

$$\|\mathbf{A} - \mathbf{B}_{L-1}\|^2 = \sum_{J=1}^{L-1} \|\mathbf{B}_{J-1} - \mathbf{B}_J\|_F^2$$

Indeed, as proved in Appendix E.3 using an orthogonality argument[6], we have

$$\forall J \in [\![L-1]\!], \quad \forall p \in [\![J, L-1]\!], \quad \langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_p \rangle = 0. \tag{7.14}$$

Hence:

$$\begin{aligned}
\|\mathbf{A} - \mathbf{B}_{L-1}\|_F^2 &= \|(\mathbf{B}_0 - \mathbf{B}_1) + \ldots + (\mathbf{B}_{L-2} - \mathbf{B}_{L-1})\|_F^2 \\
&= \sum_{J=1}^{L-1} \|\mathbf{B}_{J-1} - \mathbf{B}_J\|_F^2 + 2 \sum_{J=1}^{L-1} \sum_{p>J} \langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_{p-1} - \mathbf{B}_p \rangle \\
&\overset{(7.14)}{=} \|\mathbf{B}_0 - \mathbf{B}_1\|_F^2 + \ldots + \|\mathbf{B}_{L-2} - \mathbf{B}_{L-1}\|_F^2.
\end{aligned}$$

∎

[6]This argument has been used in [29] to prove (7.15). We adapt their argument to our context for the self-containedness of our paper.

7.6. Comparison with existing error bounds. The result in the literature that is most related to our proposed error bounds (Theorem 7.2 and Theorem 7.3) is the bound in [29], which takes the form

$$\|\mathbf{A} - \hat{\mathbf{A}}\|_F \leq C_n \epsilon_0 \|\mathbf{A}\|_F, \quad \text{with } C_n = \mathcal{O}(\sqrt{\log n}), \tag{7.15}$$

where $\mathbf{A}$ is an $n \times n$ matrix and ϵ_0 is the maximum relative error $\|\mathbf{M} - \hat{\mathbf{M}}\|_F / \|\mathbf{M}\|_F$ across all blocks $\mathbf{M}$ on which the algorithm performs low-rank approximation, with $\hat{\mathbf{M}}$ a best low-rank approximation of $\mathbf{M}$.

It is noteworthy that these bounds are not immediately comparable because of the difference in problem formulations. First of all, the sparsity patterns of the factors in [29] cannot be expressed by four parameters of a pattern $\boldsymbol{\pi}$. Instead, using the Kronecker supports introduced in this work, the support constraints in [29] can be described using six parameters (a, b, c, d, e, f) as $\mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d \otimes \mathbf{1}_{e\times f}$, a structure also referred to as the "block butterfly" [3]. However, these supports can be transformed into supports of the form considered in Definition 4.1 by multiplying by appropriate permutation matrices $\mathbf{P}, \mathbf{Q}$ to the left and right, respectively, such that $\mathbf{P}(\mathbf{I}_a \otimes \mathbf{1}_{b\times c} \otimes \mathbf{I}_d \otimes \mathbf{1}_{e\times f})\mathbf{Q} = \mathbf{I}_a \otimes \mathbf{1}_{be\times cf} \otimes \mathbf{I}_d$ [13, Appendix C.2]. Thus, the sparsity structures can be regarded as equivalent. But more importantly, in the formulation of the factorization problem, in [29] the architecture is not fixed *a priori* as it is in our case. Instead, given a *fixed* level of relative error ϵ, [29] will find *a posteriori* a sparse architecture associated to some low-rank constraints that are adapted to the considered input matrix, so that the low-rank approximations in the butterfly algorithm satisfy the target error ϵ for the considered input matrix.

Even if the bounds are not directly comparable, we can say that our bound improves with respect to (7.15) because (i) it allows us to compare the approximation error $\|\mathbf{A} - \hat{\mathbf{A}}\|_F$ to the *best* approximation error, that is, the minimal error $\|\mathbf{A} - \mathbf{A}^*\|_F$ with $\mathbf{A}^*$ satisfying *exactly* the complementary low-rank property associated to the prescribed butterfly architecture; (ii) it holds for any factor-bracketing tree (Definition 5.2), while (7.15) is tight to a specific tree; (iii) the constant $C_{\boldsymbol{\beta}}$ is computable *a priori*, while the quantity ϵ_0 in (7.15) can only be determined algorithmically *after* applying the butterfly algorithm on the target matrix $\mathbf{A}$, i.e., when $\hat{\mathbf{A}}$ is available and the error $\|\mathbf{A} - \hat{\mathbf{A}}\|_F$ can be directly computed. Moreover, the algorithm in [29] suffers from the same limitations of Algorithm 5.1 (cf. Example 5.3): ϵ_0 can be *strictly positive* (and arbitrarily close to one) for some target matrices $\mathbf{A}$ *even if* they satisfy exactly the complementary low-rank property.

8. Numerical experiments. We now illustrate the empirical behaviour of the proposed hierarchical algorithm for Problem (1.1). All methods are implemented in Python 3.9.7 using the `PyTorch 2.2.1` package. Our implementation codes are given in [19]. Experiments are conducted on an Apple M3, 2.8 GHz, in float-precision. The following experiments consider the decomposition of real-valued matrices, so we implement Algorithms 3.1, 5.1, and 6.1 with real numbers instead of complex ones.

8.1. Hierarchical algorithm *vs.* existing methods. We consider Problem (1.1) associated with the square dyadic butterfly architecture with $L = 10$ factors. The target matrix $\mathbf{A}$ is the 1024×1024 Hadamard matrix. The compared methods are:

- ***Our butterfly algorithm (Algorithm* 6.1*, with or without pseudo-orthonormalization operations)***: we use the permutation σ of $[\![10]\!]$ corresponding to a balanced factor-bracketing tree [44, 20] (see Figure 7 in Appendix G), i.e., $\sigma = (5, 2, 1, 3, 4, 7, 6, 8, 9)$. Since $\mathbf{A}$ admits an exact factor-

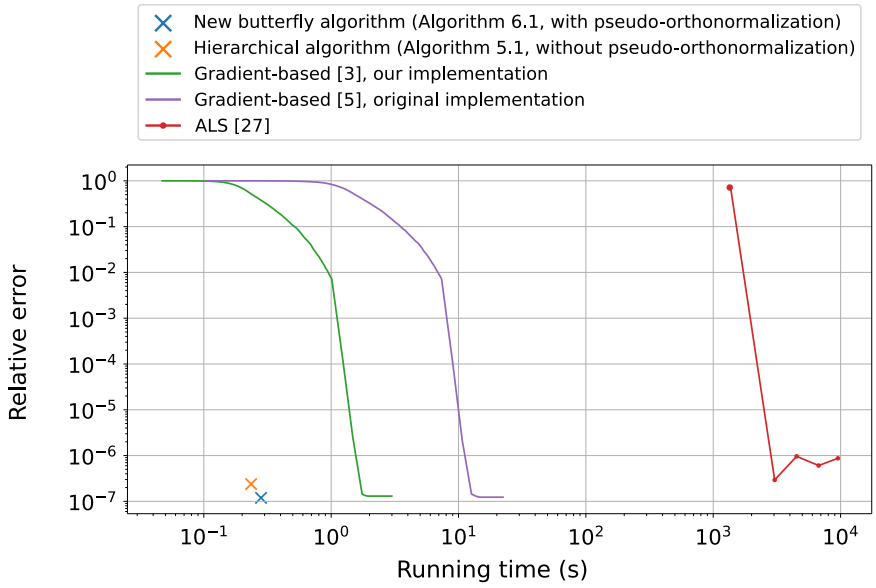


Fig. 3: Relative approximation errors defined as $\|\mathbf{A} - \hat{\mathbf{A}}\|_F / \|\mathbf{A}\|_F$ *vs.* running time of the different algorithms. The target matrix $\mathbf{A}$ is the Hadamard matrix of size 1024×1024, and $\hat{\mathbf{A}}$ is the computed approximation for Problem (1.1) associated with the square dyadic butterfly architecture.

ization, the results (except computation times) are not expected to depend on σ as already documented [44, 20].

In line 3 of the two-factor fixed support matrix factorization algorithm (Algorithm 3.1), the best low-rank approximation is computed via truncated SVD: following [20], we compute the full[7] SVD $\mathbf{UDV}^\top$ of the submatrix $\mathbf{A}[R_P, C_P]$ where the diagonal entries of $\mathbf{D}$ are the singular values in decreasing order, and we set the factors $\mathbf{H} = \mathbf{U}[:, 1]\mathbf{D}^{1/2}$ and $\mathbf{K} = \mathbf{D}^{1/2}\mathbf{V}[:, 1]^\top$ since $r = 1$.

- ***Gradient-based method*** [5]: Using the parameterization $\mathbf{X}_1 \dots \mathbf{X}_L$ for a butterfly matrix in $\mathcal{B}^{\boldsymbol{\beta}}$, this method uses (variants of) gradient descent to optimize all nonzero entries $\mathbf{X}_\ell[i, j]$ for $(i, j) \in \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_\ell})$ and $\ell \in [\![L]\!]$ to minimize (1.1). We use the protocol of [5]: we perform 100 iterations of ADAM[8] [17], followed by 20 iterations of L-BFGS [28][9]. Besides directly benchmarking the implementation from [5], we propose a new implementation of this gradient-based method which is faster than the one of [5]. Please consult our codes [19] for more details.
- ***Alternating least squares (ALS)*** [27]: At each iteration of this iterative algorithm, we optimize the nonzero entries of a given factor $\mathbf{X}_\ell$ for some $\ell \in [\![L]\!]$ while fixing the others, by solving a linear regression problem.

Figure 3 shows that the different methods find an approximate solution nearly up

[7] For the considered matrix dimension this is faster than partial SVD. In higher dimensions, partial SVD can further optimize the algorithm, see the detailed discussion in [44].

[8] The learning rate is set as 0.1, and we choose $(\beta_1, \beta_2) = (0.9, 0.999)$.

[9] L-BFGS terminates when the norm of the gradient is smaller than 10^{-7}.

to machine precision[10], but hierarchical algorithms are several orders of magnitude faster than the gradient-based method [5] and ALS [27]. Using Algorithm 6.1 *without* pseudo-orthonormalization operations is also faster than with these operations. We however show the positive practical impact of pseudo-orthonormalization on noisy problems in the following section.

8.2. To orthonormalize or not to orthonormalize? We now study in practice the impact of the pseudo-orthonormalization operations in the new butterfly algorithm, in terms of running time and approximation error, at different scales of the matrix size n with $n \in \{2^i \mid i = 7, \dots, 13\}$. We consider Problem (1.1) associated with a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_4)$ such that:

- Each factor is of size $(a_\ell b_\ell d_\ell, a_\ell c_\ell d_\ell) = (n, n)$
- $\mathcal{B}^{\boldsymbol{\beta}}$ contains some dense matrices: $\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3 * \boldsymbol{\pi}_4 = (1, n, n, 1)$;
- In the complementary low-rank characterization of $\mathcal{B}^{\boldsymbol{\beta}}$, the rank constraint on the submatrices is $r \geq 2$, i.e., $\mathbf{r}(\boldsymbol{\beta}) = (r, r, r)$. We do not choose $r = 1$ because the pseudo-orthonormalization operations would then be equivalent to some simple rescaling.

Among all of such architectures (that can be characterized and found using Lemma 4.22), we choose the one with the smallest number of parameters, i.e., yielding the smallest $\|\boldsymbol{\beta}\|_0$. The considered target matrix is $\mathbf{A} = \tilde{\mathbf{A}} + \epsilon \frac{\|\tilde{\mathbf{A}}\|_F}{\|\mathbf{E}\|_F} \mathbf{E}$, where $\tilde{\mathbf{A}} := \mathbf{X}_1 \mathbf{X}_2 \mathbf{X}_3 \mathbf{X}_4$, the entries of $\mathbf{X}_\ell \in \Sigma^{\boldsymbol{\pi}_\ell}$ for $\ell \in [\![4]\!]$ are i.i.d. sampled from the uniform distribution in the interval $[0, 1]$, $\mathbf{E}$ is an i.i.d. centered Gaussian matrix with the standard deviation 1, and $\epsilon = 0.1$ is the noise level. The permutation σ for the butterfly algorithm is $\sigma = (2, 1, 3)$, which corresponds to the *balanced* factor-bracketing tree of $[\![4]\!]$.

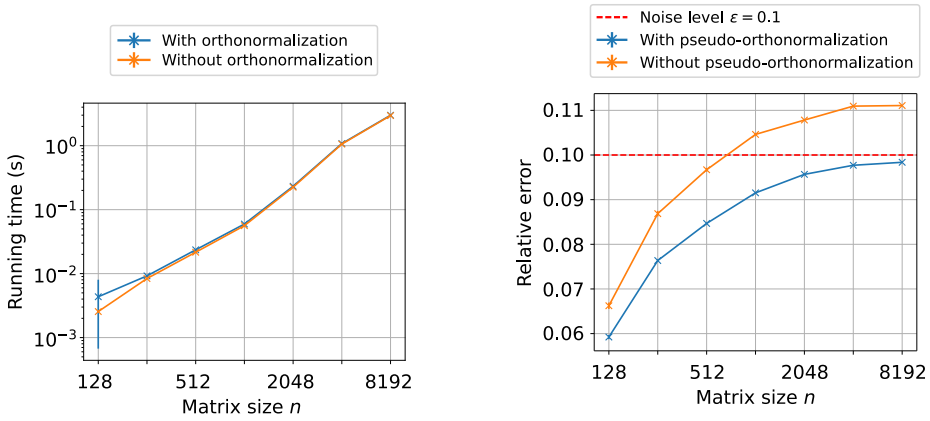
Figure 4a shows that the difference in running time between the new butterfly algorithm (Algorithm 6.1) *with* and *without* pseudo-orthonormalization is negligible, in the regime of large matrix size $n \geq 512$. This means that, asymptotically, the time of orthonormalization operations is not the bottleneck, which is coherent with our complexity analysis given in Theorem 6.5.

In terms of the approximation error, Figure 4b shows that the hierarchical algorithm (Algorithm 6.1) *with* pseudo-orthonormalization returns a smaller (i.e., better) approximation error. Moreover, the relative error with pseudo-orthonormalization error is always smaller than the relative noise level $\epsilon = 0.1$ (cf. Figure 8 for other values of ϵ), which is not the case for the hierarchical algorithm without pseudo-orthonormalization (Algorithm 5.1). In conclusion, besides yielding error guarantees of the form (1.3), the pseudo-orthonormalization operations in our experiments also lead to better approximation in practice.

8.3. Numerical assessment of the bounds In this section, we numerically evaluate the bounds given by Theorem 7.2 and Theorem 7.3. The setting is identical to that of the previous section, except that:

1. We run the new butterfly algorithm (Algorithm 6.1) with multiple permutations associated to various factor-bracketing trees: identity permutation (left-to-right tree, cf. Example 5.3 or [44]), balance permutation (corresponding to the so-called "balanced tree" [44]) and a random one.
2. The number of factors is $L = 5$. The size of the matrix is 4608. The number $4608 = 3^2 \times 2^9$ is a common matrix size appearing when one replaces

[10]Yet, hierarchical algorithms and gradient-based methods are more accurate than ALS by more than one order of magnitude.



(a) Running time *vs.* size n

(b) Relative error $\frac{\|\mathbf{A}-\hat{\mathbf{A}}\|_F}{\|\mathbf{A}\|_F}$ *vs.* size n

Fig. 4: Running time and the relative approximation errors *vs.* the matrix size n, for Algorithm 6.1 (with or without pseudo-orthonormalization), for the instance of Problem (1.1) described in Subsection 8.2 with $L = 4$, $r = 4$. We show mean and standard deviation on the error bars over 10 repetitions of the experiment.

convolutional layers by a product of butterfly factors [27] (the factor 3^2 corresponds to a convolutional kernel of size 3×3). This matrix size is non-dyadic, which partly illustrates the versatility of butterfly factors in replacing linear operators.

3. The experiments are repeated 10 times. The plots in Figure 5 show the average and standard deviation values of the relative error and of the bounds. The standard deviations are not visible due to their small size and the log scale of the plot.

We report the relative errors and the bounds corresponding to each permutation in Figure 5. It is well illustrated by Figure 5 that our theoretical results in Theorem 7.2 and Theorem 7.3 are correct. The bound in Theorem 7.3 (Figure 5-a) is visibly tighter than the one in Theorem 7.2 (Figure 5-b,c) although there is no significant difference among the errors obtained by the three permutations corresponding to different factor bracketing trees (the errors for the three different permutations are almost identical, which makes only one error plot visible).

9. Conclusion We proposed a general definition of (deformable) butterfly architectures, together with a new butterfly algorithm for the problem of (deformable) butterfly factorization (1.1), endowed with new guarantees on the approximation error of the type (1.3), under the condition that the associated architecture $\boldsymbol{\beta}$ satisfies a so-called *chainability* condition. The proposed algorithm involves some novel orthonormalization operations in the context of butterfly factorization. We discuss some perspectives of this work.

Tightness of the error bound. The constants $C_{\boldsymbol{\beta}}$ in Theorem 7.4 scale linearly or even sub-linearly with respect to the depth $L = |\boldsymbol{\beta}|$ of the architecture. Note that the quasi-linear constant C_n in the existing bound (7.15) is not comparable with the constants $C_{\boldsymbol{\beta}}$ in Theorem 7.4, due to the presence of ϵ_0 in the bound (7.15), whereas the

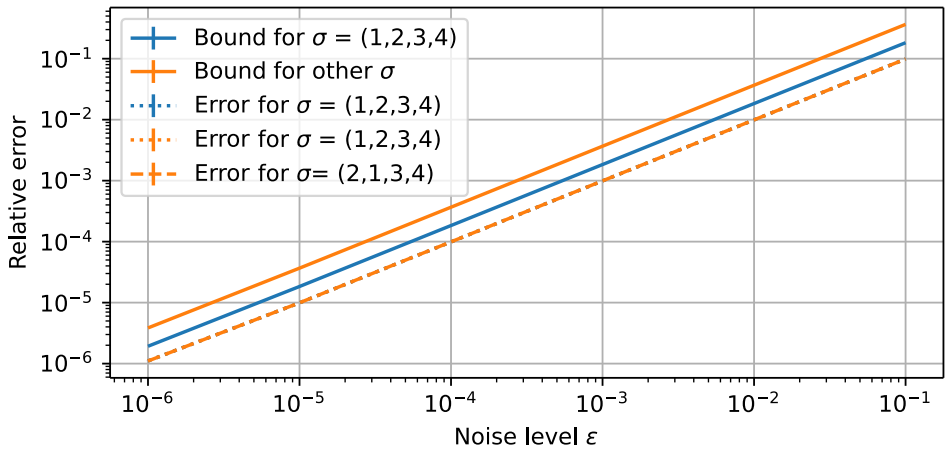


Fig. 5: Bounds and numerical errors of Algorithm 6.1 with different permutations (associated to different factor bracketing trees): identity permutation $(1, 2, 3, 4)$, "balanced" permutation $(2, 1, 3, 4)$, randomized permutation ($(4, 1, 2, 3)$ in this experiment). The bound for the identity permutation is given by Theorem 7.3 while the others are given by Theorem 7.2. Vertical bars show standard deviations over 10 experiments, but they are not visible on the graph because they are small. Note that all error curves (discontinuous lines) are overlapping, so we are seeing only one of them.

constants $C_{\boldsymbol{\beta}}$ in Theorem 7.4 controls the ratio between the approximation error and the minimal error. A natural question is whether the constants $C_{\boldsymbol{\beta}}$ in Theorem 7.4 are tight for an error bound of the type (1.3). If not, can the bounds for the proposed be algorithm be sharpened by a refined theoretical analysis, or is there another algorithm that yields a smaller constant $C_{\boldsymbol{\beta}}$ in the error bound?

Randomized algorithms for low-rank approximation. Algorithms 5.1 and 6.1 need to access all the elements of the target matrix $\mathbf{A} \in \mathbb{C}^{m\times n}$. Thus, the complexity of all algorithms is at least $\mathcal{O}(mn)$. This complexity, however, fails to scale for large m, n (e.g., up to 10^6). Assuming that the target matrix admits a butterfly factorization associated with $\boldsymbol{\beta}$, i.e., $\mathbf{A} \in \mathcal{B}^{\boldsymbol{\beta}}$, is it possible to recover the butterfly factors of $\mathbf{A}$, with a faster algorithm, ideally of complexity $\mathcal{O}(\|\boldsymbol{\beta}\|_0)$? Note that this question was already considered in [23, 22] where randomized algorithms for low-rank approximation [16, 26] are leveraged in the context of butterfly factorization. The question is therefore whether we can still prove some theoretical guarantees of the form (1.3) for butterfly algorithms with such algorithms.

Algorithms beyond the chainability assumption. Although chainability is a sufficient condition for which we can design an algorithm with guarantees on the approximation error, it is natural to ask whether it is also a necessary condition. There exist, in fact, non-chainable architectures for which we can still build an algorithm yielding an error bound (1.3). For instance, this is the case of *arbitrary* architectures of depth $L = 2$ (see Lemma 5.1); or of architectures that satisfy a *transposed* version of Definition 4.10: such architectures can easily be checked to cover some *non-chainable* architectures, and Algorithm 6.1 can be easily adapted to have guarantees similar to Theorems 7.2 and 7.3. Therefore, chainability in the sense of Definition 4.10

is not necessary for theoretical guarantees of the form (1.3). Whether algorithms with performance guarantees can be derived beyond the three above-mentioned cases remains open.

Efficient implementation of butterfly matrix multiplication. While the complexity of matrix-vector multiplication by a $n \times m$ butterfly matrix associated with $\boldsymbol{\beta}$ is theoretically subquadratic if $\|\boldsymbol{\beta}\|_0 = o(mn)$, a practical fast implementation of such a matrix multiplication is not straightforward [13], since dense matrix multiplication algorithms are competitive, e.g., on GPUs. Future work can focus on efficient implementations of butterfly matrix multiplication on different kinds of hardware, in order to harness all of the benefits of the butterfly structure for large-scale applications, like in machine learning for instance.

Butterfly factorization with unknown permutations. In general, it is necessary to take into account row and column permutations in the butterfly factorization problem for more flexibility of the butterfly model $\Sigma^{\boldsymbol{\beta}}$ for an architecture $\boldsymbol{\beta}$. For instance, as mentioned above, the DFT matrix admits a square dyadic butterfly factorization *up to the bit-reversal permutation of column indices*. Therefore, as in [43], the general approximation problem that takes into account row and column permutations is:

$$\inf_{(\mathbf{X}_\ell)_{\ell=1}^{L} \in \Sigma^{\boldsymbol{\beta}}, \mathbf{P}, \mathbf{Q}} \|\mathbf{A} - \mathbf{Q}^\top \mathbf{X}_1 \dots \mathbf{X}_L \mathbf{P}\|_F, \tag{9.1}$$

where $\mathbf{P}$, $\mathbf{Q}$ are unknown permutations part of the optimization problem. Without any further assumption on the target matrix $\mathbf{A}$, solving this approximation problem is conjectured to be difficult. Future work can further study this more general problem, based on the existing heuristic proposed in [43].

Acknowledgement The authors gratefully acknowledge the support of the Centre Blaise Pascal's IT test platform at ENS de Lyon for Machine Learning facilities. The platform operates the SIDUS solution [37] developed by Emmanuel Quemener. This work was partly supported by the ANR, project AllegroAssai ANR-19-CHIA-0009, and by the SHARP ANR project ANR-23-PEIA-0008 in the context of the France 2030 program.

REFERENCES

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004.
[2] E. Candes, L. Demanet, and L. Ying. A fast butterfly algorithm for the computation of fourier integral operators. *Multiscale Modeling & Simulation*, 7(4):1727–1750, 2009.
[3] B. Chen, T. Dao, K. Liang, J. Yang, Z. Song, A. Rudra, and C. Ré. Pixelated butterfly: Simple and efficient sparse training for neural network models. In *International Conference on Learning Representations*, 2022.
[4] T. Dao, B. Chen, N. S. Sohoni, A. D. Desai, M. Poli, J. Grogan, A. Liu, A. Rao, A. Rudra, and C. Ré. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pages 4690–4721. PMLR, 2022.
[5] T. Dao, A. Gu, M. Eichhorn, A. Rudra, and C. Ré. Learning fast algorithms for linear transforms using butterfly factorizations. In *International Conference on Machine Learning*, pages 1517–1527. PMLR, 2019.
[6] T. Dao, N. Sohoni, A. Gu, M. Eichhorn, A. Blonder, M. Leszczynski, A. Rudra, and C. Ré. Kaleidoscope: An efficient, learnable representation for all structured linear maps. In *International Conference on Learning Representations*, 2020.
[7] L. Demanet and L. Ying. Fast wave computation via fourier integral operators. *Mathematics of Computation*, 81(279):1455–1486, 2012.
[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly,

Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[9] P. Duhamel and M. Vetterli. Fast fourier transforms: a tutorial review and a state of the art. *Signal processing*, 19(4):259–299, 1990.

[10] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[11] N. Engheta, W. D. Murphy, V. Rokhlin, and M. S. Vassiliou. The fast multipole method (FMM) for electromagnetic scattering problems. *IEEE Transactions on Antennas and Propagation*, 40(6):634–641, 1992.

[12] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[13] A. Gonon, L. Zheng, P. Carrivain, and Q.-T. Le. Fast inference with kronecker-sparse matrices. *arXiv preprint arXiv:2405.15013*, 2024.

[14] H. Guo, Y. Liu, J. Hu, and E. Michielssen. A butterfly-based direct integral-equation solver using hierarchical lu factorization for analyzing scattering from electrically large conducting objects. *IEEE Transactions on Antennas and Propagation*, 65(9):4742–4750, 2017.

[15] W. Hackbusch. *Hierarchical matrices: Algorithms and analysis*. Springer, 2015.

[16] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[18] Q.-T. Le, E. Riccietti, and R. Gribonval. Spurious valleys, NP-hardness, and tractability of sparse matrix factorization with fixed support. *SIAM Journal on Matrix Analysis and Applications*, 2022.

[19] Q.-T. Le, L. Zheng, R. Gribonval, and E. Riccietti. Code for reproducible research - Butterfly factorization with error guarantees, 2024. https://inria.hal.science/hal-04720713.

[20] Q.-T. Le, L. Zheng, E. Riccietti, and R. Gribonval. Fast learning of fast transforms, with guarantees. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022.

[21] L. Le Magoarou and R. Gribonval. Flexible multilayer sparse approximations of matrices and applications. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):688–700, 2016.

[22] Y. Li and H. Yang. Interpolative butterfly factorization. *SIAM Journal on Scientific Computing*, 39(2):A503–A531, 2017.

[23] Y. Li, H. Yang, E. R. Martin, K. L. Ho, and L. Ying. Butterfly factorization. *Multiscale Modeling & Simulation*, 13(2):714–732, 2015.

[24] Y. Li, H. Yang, and L. Ying. A multiscale butterfly algorithm for multidimensional fourier integral operators. *Multiscale Modeling & Simulation*, 13(2):614–631, 2015.

[25] Y. Li, H. Yang, and L. Ying. Multidimensional butterfly factorization. *Applied and Computational Harmonic Analysis*, 44(3):737–758, 2018.

[26] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

[27] R. Lin, J. Ran, K. H. Chiu, G. Chesi, and N. Wong. Deformable butterfly: A highly structured and sparse linear transform. In *Advances in Neural Information Processing Systems*, volume 34, pages 16145–16157, 2021.

[28] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[29] Y. Liu, X. Xing, H. Guo, E. Michielssen, P. Ghysels, and X. S. Li. Butterfly factorization via randomized matrix-vector multiplications. *SIAM Journal on Scientific Computing*, 43(2):A883–A907, 2021.

[30] E. Michielssen and A. Boag. Multilevel evaluation of electromagnetic fields for the rapid solution of scattering problems. *Microwave and Optical Technology Letters*, 7(17):790–795, 1994.

[31] E. Michielssen and A. Boag. A multilevel matrix decomposition algorithm for analyzing scattering from large structures. *IEEE Transactions on Antennas and Propagation*, 44(8):1086–1093, 1996.

[32] M. Munkhoeva, Y. Kapushev, E. Burnaev, and I. Oseledets. Quadrature-based features for kernel approximation. In *Advances in neural information processing systems*, 2018.

[33] M. O'Neil. *A new class of analysis-based fast transforms*. Yale University, 2007.

[34] M. O'Neil, F. Woolfe, and V. Rokhlin. An algorithm for the rapid evaluation of special function transforms. *Applied and Computational Harmonic Analysis*, 28(2):203–226, 2010.

[35] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–

2317, 2011.

[36] D. S. Parker. *Random butterfly transformations with applications in computational linear algebra*. UCLA Computer Science Department, 1995.

[37] E. Quemener and M. Corvellec. Sidus—the solution for extreme deduplication of an operating system. *Linux Journal*, 2013(235):3, 2013.

[38] C M Rader. Discrete Fourier transforms when the number of data samples is prime. *Proceedings of the IEEE*, 56:1107–1108, 1968.

[39] S.-K. Shekofteh, C. Alles, and H. Fröning. Reducing memory requirements for the ipu using butterfly factorizations. In *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, pages 1255–1263, 2023.

[40] M. Tygert. Fast algorithms for spherical harmonic expansions, III. *Journal of Computational Physics*, 229(18):6181–6192, 2010.

[41] Z. Yang, M. Moczulski, M. Denil, N. De Freitas, A. Smola, L. Song, and Z. Wang. Deep fried convnets. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1476–1483, 2015.

[42] L. Ying. Sparse fourier transform via butterfly algorithm. *SIAM Journal on Scientific Computing*, 31(3):1678–1694, 2009.

[43] L. Zheng, G. Puy, E. Riccietti, P. Pérez, and R. Gribonval. Butterfly factorization by algorithmic identification of rank-one blocks. *arXiv preprint arXiv:2307.00820*, 2023.

[44] L. Zheng, E. Riccietti, and R. Gribonval. Efficient identification of butterfly sparse matrix factorizations. *SIAM Journal on Mathematics of Data Science*, 5(1):22–49, 2023.

Appendix A. Proof for results in Section 3 This section is devoted to the proof of Theorem 3.4. To that end we first introduce the following technical lemma.

LEMMA A.1. *Consider support constraints* $\mathbf{L} \in \{0,1\}^{m\times s}$, $\mathbf{R} \in \{0,1\}^{r\times n}$ *satisfying the conditions of Theorem* 3.4. *With the notations* $R_P \subseteq [\![m]\!]$, $C_P \subseteq [\![n]\!]$, $\mathcal{P}[\mathbf{L},\mathbf{R}]$ *from Definition* 3.3, *for each* $(\mathbf{X},\mathbf{Y})$ *such that* $\mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}, \mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}$, *we have:*

$$\forall P \in \mathcal{P}(\mathbf{L},\mathbf{R}), \quad (\mathbf{XY})[R_P, C_P] = \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]. \tag{A.1}$$

Proof. For any pair of matrices $(\mathbf{X},\mathbf{Y}) \in \mathbb{C}^{m\times r} \times \mathbb{C}^{r\times n}$:

$$\mathbf{XY} = \sum_{i=1}^{r} \mathbf{X}[:,i]\mathbf{Y}[i,:] = \sum_{P\in\mathcal{P}(\mathbf{L},\mathbf{R})} \mathbf{X}[:,P]\mathbf{Y}[P,:]. \tag{A.2}$$

For $P, \tilde{P} \in \mathcal{P}(\mathbf{L},\mathbf{R})$ with $P \neq \tilde{P}$, the components $\mathbf{U}_P$ and $\mathbf{U}_{\tilde{P}}$ of Definition 3.3 are disjoint by assumption on $\mathbf{L},\mathbf{R}$. Since $\mathtt{supp}(\mathbf{X}[:,\tilde{P}]\mathbf{Y}[\tilde{P},:]) \subseteq \mathbf{U}_{\tilde{P}}$, we have $(\mathbf{X}[:,\tilde{P}]\mathbf{Y}[\tilde{P},:])[R_P, C_P] = \mathbf{0}$, and by (A.2) it follows that we have $(\mathbf{XY})[R_P, C_P] = (\mathbf{X}[:,P]\mathbf{Y}[P,:])[R_P, C_P] = \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]$. ∎

The following proof of Theorem 3.4 is mainly taken from [18], but we additionally compute the infimum value of Problem (3.1).

Proof for Theorem 3.4. We use the shorthand $\mathcal{P}$ for $\mathcal{P}(\mathbf{L},\mathbf{R})$, and denote $\Sigma := \{(\mathbf{X},\mathbf{Y}) \,|\, \mathtt{supp}(\mathbf{X}) \subseteq \mathbf{L}, \mathtt{supp}(\mathbf{Y}) \subseteq \mathbf{R}\}$. Recall that $(\mathbf{U}_i)_{i=1}^r := \varphi(\mathbf{X},\mathbf{Y})$ with φ from Definition 3.1. Let $(\mathbf{X},\mathbf{Y}) \in \Sigma$. Then, $\mathtt{supp}(\mathbf{X}[:,P]\mathbf{Y}[P,:]) \subseteq \mathbf{U}_P$ for any $P \in \mathcal{P}$, hence $\mathtt{supp}(\mathbf{XY}) \subseteq \bigcup_{P\in\mathcal{P}} \mathbf{U}_P$, and $(\mathbf{XY}) \odot \overline{\mathbf{U}_\mathcal{P}} = \mathbf{0}$ where we denote $\overline{\mathbf{U}_\mathcal{P}} := ([\![m]\!] \times [\![n]\!]) \setminus \left(\bigcup_{P\in\mathcal{P}} \mathbf{U}_P\right)$. Moreover, by assumption, $\mathbf{U}_P$ and $\mathbf{U}_{\tilde{P}}$ are disjoint for any $P, \tilde{P} \in \mathcal{P}$ such that $P \neq \tilde{P}$, so:

$$\begin{aligned}
\|\mathbf{A} - \mathbf{XY}\|_F^2 &= \left(\sum_{P\in\mathcal{P}} \|(\mathbf{A} - \mathbf{XY}) \odot \mathbf{U}_P\|_F^2\right) + \|(\mathbf{A} - \mathbf{XY}) \odot \overline{\mathbf{U}_\mathcal{P}}\|_F^2 \\
&= \left(\sum_{P\in\mathcal{P}} \|(\mathbf{A} - \mathbf{XY})[R_P, C_P]\|_F^2\right) + \|\mathbf{A} \odot \overline{\mathbf{U}_\mathcal{P}}\|_F^2 \\
&= \left(\sum_{P\in\mathcal{P}} \|\mathbf{A}[R_P, C_P] - (\mathbf{XY})[R_P, C_P]\|_F^2\right) + \|\mathbf{A} \odot \overline{\mathbf{U}_\mathcal{P}}\|_F^2 \\
&\overset{(A.1)}{=} \left(\sum_{P\in\mathcal{P}} \|\mathbf{A}[R_P, C_P] - \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2\right) + \|\mathbf{A} \odot \overline{\mathbf{U}_\mathcal{P}}\|_F^2.
\end{aligned} \tag{A.3}$$

Since $\mathcal{P}$ is a partition, this implies that each term in the sum $\sum_{P\in\mathcal{P}} \|\mathbf{A}[R_P, C_P] - \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|^2$ involves columns of $\mathbf{X}$ and rows of $\mathbf{Y}$ that are not involved in other terms of the sum. Moreover, we remark that for any $P \in \mathcal{P}$, the matrix $\mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]$ is of rank at most $|P|$. In other words, minimizing the right-hand-side of (A.3) with respect to $(\mathbf{X},\mathbf{Y}) \in \Sigma$ is equivalent to minimize each term of the sum for $P \in \mathcal{P}$, which is the problem of finding the best rank-$|P|$ approximation of $\mathbf{A}[R_P, C_P]$. This yields the claimed equation (3.2). ∎

Appendix B. Proof for results in Section 4 and subsection 5.1

B.1. Proof of Proposition 4.7

Proof. Let $r = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. By Definition 4.5, $a_1 \mid a_2$ and $d_2 \mid d_1$, so there are two integers q, s such that $a_2 = qa_1$ and $d_1 = sd_2$. Since $a_1c_1/a_2 = b_2d_2/d_1 = r$ by Definition 4.5, this yields $c_1 = a_2r/a_1 = rq$, $b_2 = d_1r/d_2 = rs$. Thus,

$$\begin{aligned}
\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2} &= (\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 \times rq} \otimes \mathbf{I}_{d_1})\,(\mathbf{I}_{a_2} \otimes \mathbf{1}_{rs \times c_2} \otimes \mathbf{I}_{d_2}) \\
&= (\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 \times q} \otimes \mathbf{1}_{1 \times r} \otimes \mathbf{I}_{d_1})\,(\mathbf{I}_{a_2} \otimes \mathbf{1}_{r \times 1} \otimes \mathbf{1}_{s \times c_2} \otimes \mathbf{I}_{d_2}) \\
&= [(\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 \times q}) \otimes \mathbf{1}_{1 \times r} \otimes \mathbf{I}_{d_1}]\,[\mathbf{I}_{a_2} \otimes \mathbf{1}_{r \times 1} \otimes (\mathbf{1}_{s \times c_2} \otimes \mathbf{I}_{d_2})] \\
&\overset{(\star)}{=} (\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 \times q}) \otimes (\mathbf{1}_{1 \times r}\mathbf{1}_{r \times 1}) \otimes (\mathbf{1}_{s \times c_2} \otimes \mathbf{I}_{d_2}) \\
&= (\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 \times q}) \otimes (r\mathbf{1}_{1 \times 1}) \otimes (\mathbf{1}_{s \times c_2} \otimes \mathbf{I}_{d_2}) \\
&= r\,(\mathbf{I}_{a_1} \otimes \mathbf{1}_{b_1 s \times qc_2} \otimes \mathbf{I}_{d_2}) \\
&= r\mathbf{S}_{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2} \quad \left(\text{because } \frac{b_1 d_1}{d_2} = b_1 s \text{ and } \frac{a_2 c_2}{c_1} = qc_2\right).
\end{aligned}$$

We can use the equality $(\mathbf{A} \otimes \mathbf{C} \otimes \mathbf{E})(\mathbf{B} \otimes \mathbf{D} \otimes \mathbf{F}) = (\mathbf{AB}) \otimes (\mathbf{CD}) \otimes (\mathbf{EF})$ in $(\star)$ because, according to our conditions for chainability, the sizes of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}$ in $(\star)$ make the matrix products $\mathbf{AB}, \mathbf{CD}$ and $\mathbf{EF}$ well-defined. ∎

B.2. Proof of Lemma 4.8 We prove the claims in Lemma 4.8 one by one.

1. Follows by the definition of R_P and C_P.
2. It can be easily verified that the column supports $\{\mathbf{S}_{\boldsymbol{\pi}_1}[:, j]\}_j$ (resp. the row supports $\{\mathbf{S}_{\boldsymbol{\pi}_2}[i, :]\}_i$) are pairwise disjoint or identical, due to the support structure of the form $\mathbf{I}_a \otimes \mathbf{1}_{b \times c} \otimes \mathbf{I}_d$ of a Kronecker-sparse factor. Indeed, the columns (resp. rows) of the Kronecker product $\mathbf{A} \otimes \mathbf{B}$ are equal to the Kronecker product of columns and rows of $\mathbf{A}$ and $\mathbf{B}$ and the matrices $\mathbf{A}, \mathbf{B}$ appearing in our case are either identity matrices or all-one matrices. Thus, for each $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ the sets $R_P \times C_P = \mathbf{S}_{\boldsymbol{\pi}_1}[:, i] \times \mathbf{S}_{\boldsymbol{\pi}_2}[i, :]$ are pairwise disjoint.
3. For any $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$, by Definition 4.1 we have $|R_P| = b_1, |C_P| = c_2$. Setting $r := r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, we have
$$\sum_{P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})} |P|\,\mathbf{U}_P \overset{\text{Def. } 3.3}{=} \sum_{\mathbf{U}_i \in \varphi(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})} \mathbf{U}_i = \mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2} \overset{\text{Prop. } 4.7}{=} r\mathbf{S}_{(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2)},$$
where the second equality simply comes from the rank-one decomposition of matrix multiplication. Since all $\mathbf{U}_P$'s have pairwise disjoint supports (from point 2), we get $|P| = r$ for each $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$.
4. $\mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}) = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2})$ is a consequence of Proposition 4.7. The second equality, $\mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2}) = \cup_{P \in \mathcal{P}} R_P \times C_P$, is a consequence of the rank-one decomposition of the matrix multiplication, i.e., $\mathbf{XY} = \sum_i \mathbf{X}[:, i]\mathbf{Y}[i, :]$, and of the fact that $\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2}$ have non-negative coefficients (thus, the support is equal to the union, and it is not just a subset).

B.3. Proof for Lemma 4.9 Denote $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$ for $\ell \in [\![3]\!]$. Let us show that $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3$ are chainable and $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. Since $(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ is chainable, by definition of $*$ (Definition 4.5), we have:

$$(\tilde{a}, \tilde{b}, \tilde{c}, \tilde{d}) = \tilde{\boldsymbol{\pi}} := \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3 = \left(a_2, \frac{b_2 d_2}{d_3}, \frac{a_3 c_3}{a_2}, d_3\right).$$

We then verify that $(\boldsymbol{\pi}_1, \tilde{\boldsymbol{\pi}})$ satisfies the conditions of Definition 4.5:

1. By chainability of $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, we have $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) := a_1c_1/a_2 = b_2d_2/d_1 \in \mathbb{N}$. Therefore: $a_1c_1/\tilde{a} = a_1c_1/a_2 = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = b_2d_2/d_1 = \tilde{b}\tilde{d}/d_1 \in \mathbb{N}$. This means that $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$.
2. By chainability of $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, we have $a_1 \mid a_2 = \tilde{a}$.
3. By chainability of $(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ (resp. of $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$) we have $\tilde{d} = d_3 \mid d_2 \mid d_1$.

In conclusion, $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3)$ is chainable with $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. Computing $\boldsymbol{\pi}_1 * (\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3)$ explicitly by (4.3) gives $\boldsymbol{\pi}_1 * (\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = \left(a_1, \frac{b_1d_1}{d_3}, \frac{a_3c_3}{a_1}, d_3\right)$. Similarly, we can show that $(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ is also chainable with $r(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$, and we can indeed verify that $(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2) * \boldsymbol{\pi}_3 = \boldsymbol{\pi}_1 * (\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3)$ using (4.3).

B.4. Proof of Lemma 4.12 - explicit formula for $(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L)$ in (4.5) We show that $\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = \left(a_1, \frac{b_1d_1}{d_L}, \frac{a_Lc_L}{a_1}, d_L\right)$, for each chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L = (a_\ell, b_\ell, c_\ell, d_\ell)_{\ell=1}^L$ of depth $L \geq 2$. The proof is an induction on $L \geq 2$. If $L = 2$, the result comes from (4.3). Let $L \geq 2$, and assume that the statement holds for any chainable architecture of depth L. Consider a chainable architecture $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^{L+1} = (a_\ell, b_\ell, c_\ell, d_\ell)_{\ell=1}^{L+1}$ of depth $L+1$. By the induction hypothesis, we have $\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = \left(a_1, \frac{b_1d_1}{d_L}, \frac{a_Lc_L}{a_1}, d_L\right)$. Therefore,

$$\begin{aligned}
\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L * \boldsymbol{\pi}_{L+1} &= \left(a_1, \frac{b_1d_1}{d_L}, \frac{a_Lc_L}{a_1}, d_L\right) * (a_{L+1}, b_{L+1}, c_{L+1}, d_{L+1}) \\
&= \left(a_1, \frac{b_1d_1d_L}{d_Ld_{L+1}}, \frac{a_{L+1}c_{L+1}}{a_1}, d_{L+1}\right) \\
&= \left(a_1, \frac{b_1d_1}{d_{L+1}}, \frac{a_{L+1}c_{L+1}}{a_1}, d_{L+1}\right).
\end{aligned}$$

B.5. Proof for Lemma 4.14 By (4.5), we have:

$$\begin{aligned}
(a, b, c, d) = \boldsymbol{\pi} := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s &= \left(a_q, \frac{b_qd_q}{d_s}, \frac{a_sc_s}{a_q}, d_s\right), \\
(a', b', c', d') = \boldsymbol{\pi}' := \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t &= \left(a_{s+1}, \frac{b_{s+1}d_{s+1}}{d_t}, \frac{a_tc_t}{a_{s+1}}, d_t\right).
\end{aligned} \tag{B.1}$$

We show the chainability of $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ by verifying the conditions of Definition 4.5:

1. By chainability of $(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$, $r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1}) = a_sc_s/a_{s+1} = b_{s+1}d_{s+1}/d_s \in \mathbb{N}$. This means that $ac/a' = a_sc_s/a_{s+1} = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1}) = b_{s+1}d_{s+1}/d_s = b'd'/d$ and $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1}) \in \mathbb{N}$.
2. By chainability of $\boldsymbol{\beta}$, $a_\ell \mid a_{\ell+1}$ for $\ell \in [\![L-1]\!]$, so $a = a_q \mid a_{s+1} = a'$ since $q \leq s$.
3. Similarly $d_{\ell+1} \mid d_\ell$ for all $\ell \in [\![L-1]\!]$, so $d' = d_t \mid d_s = d$

B.6. Proof for Lemma 4.17 The explicit formulas for $(a, b, c, d) = \boldsymbol{\pi} := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s$ and $(a', b', c', d') = \boldsymbol{\pi}' := \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t$ are given in (B.1). By Lemma 4.14, $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ is chainable, with $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$. To show the non-redundancy of $(\boldsymbol{\pi}, \boldsymbol{\pi}')$, it remains to show $r(\boldsymbol{\pi}, \boldsymbol{\pi}') < \min(b, c')$. Let us show $r(\boldsymbol{\pi}, \boldsymbol{\pi}') < b$. Because $\boldsymbol{\beta}$ is not redundant, by Definition 4.15, we have $r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}) < b_\ell$ for any $\ell \in [\![L-1]\!]$. But $r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}) = b_{\ell+1}d_{\ell+1}/d_\ell$ by Definition 4.5. Therefore, $b_{\ell+1}d_{\ell+1} < b_\ell d_\ell$ for any $\ell \in [\![L-1]\!]$. Thus, since $q \leq s$, we have $b_{s+1}d_{s+1} < b_qd_q$. A fortiori, $\frac{b_{s+1}d_{s+1}}{d_s} < \frac{b_qd_q}{d_s}$. But by (B.1), $\frac{b_qd_q}{d_s} = b$ and $\frac{b_{s+1}d_{s+1}}{d_s} = \frac{b'd'}{d} = r(\boldsymbol{\pi}, \boldsymbol{\pi}')$. In conclusion, $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = \frac{b_{s+1}d_{s+1}}{d_s} < \frac{b_qd_q}{d_s} = b$. A similar argument yields $r(\boldsymbol{\pi}, \boldsymbol{\pi}') < c'$. This ends the proof.

B.7. Proof of Lemma 4.19 We first show that $(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ satisfies the condition of Theorem 3.4. As in the proof of point 2 of Lemma 4.8 the column supports $\{\mathbf{S}_{\boldsymbol{\pi}_1}[:,j]\}_j$ (resp. the row supports $\{\mathbf{S}_{\boldsymbol{\pi}_2}[i,:]\}_i$) are pairwise disjoint or identical, hence the components $\mathbf{U}_i$ of $\varphi(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ are pairwise disjoint or identical (if their column *and* row supports coincide).

Therefore, for any matrix $\mathbf{A}$, we have $\mathbf{A} \in \mathcal{B}^{\boldsymbol{\beta}}$ if, and only if:

$$
\begin{aligned}
&\min_{(\mathbf{X},\mathbf{Y})\in\Sigma^{\boldsymbol{\beta}}} \|\mathbf{A}-\mathbf{XY}\|_F^2 = 0 \\
\overset{(3.2)}{\iff} &\sum_{P\in\mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1},\mathbf{S}_{\boldsymbol{\pi}_2})} \min_{\mathbf{B},\mathtt{rank}(\mathbf{B})\le|P|} \|\mathbf{A}[R_P,C_P]-\mathbf{B}\|_F^2 + \sum_{(i,j)\notin \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1}\mathbf{S}_{\boldsymbol{\pi}_2})} \mathbf{A}[i,j]^2 = 0 \\
\overset{(4.4)}{\iff} &\sum_{P\in\mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1},\mathbf{S}_{\boldsymbol{\pi}_2})} \min_{\mathbf{B},\mathtt{rank}(\mathbf{B})\le|P|} \|\mathbf{A}[R_P,C_P]-\mathbf{B}\|_F^2 + \sum_{(i,j)\notin \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1*\boldsymbol{\pi}_2})} \mathbf{A}[i,j]^2 = 0 \\
\iff &\begin{cases} \mathtt{rank}(\mathbf{A}[R_P,C_P]) \le |P|, \quad \forall P\in\mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1},\mathbf{S}_{\boldsymbol{\pi}_2}) \\ \mathbf{A}\in\Sigma^{\boldsymbol{\pi}_1*\boldsymbol{\pi}_2} \end{cases}.
\end{aligned}
$$

This proves (4.7).

B.8. Proof of Lemma 4.22 By Lemma 4.12 we have $\boldsymbol{\pi} := \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = (a_1, b_1d_1/d_L, a_Lc_L/a_1, d_L)$ and $\mathcal{B}^{\boldsymbol{\beta}} \subseteq \Sigma^{\boldsymbol{\pi}}$. Since $\mathcal{B}^{\boldsymbol{\beta}}$ contains some dense matrices, by Definition 4.1 we must have $a_1 = d_L = 1$ hence $\boldsymbol{\pi} = (1, b_1d_1, a_Lc_L, 1)$, and the matrices in $\Sigma^{\boldsymbol{\pi}}$ are of size $b_1d_1 \times a_Lc_L$. As a result $b_1d_1 = m$ and $a_Lc_L = n$, so that $a_L \mid n$, $d_1 \mid m$, and $b_1 = m/d_1$ and $c_L = n/a_L$. Now, by chainability (see Definition 4.5) we also have $a_1 \mid \ldots \mid a_L$ and $d_L \mid \ldots \mid d_1$ and, $a_\ell c_\ell / a_{\ell+1} = b_{\ell+1}d_{\ell+1}/d_\ell = r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}) := r_\ell$ for each $\ell \in [\![L-1]\!]$.

With the convention $a_{L+1} := n$ and $d_0 := m$, the quantities $p_\ell := a_{\ell+1}/a_\ell$ and $q_\ell := d_{\ell-1}/d_\ell$, $1 \le \ell \le L$ are thus integers, and we have $b_1 = m/d_1 = d_0/d_1 = q_1$, $c_L = n/a_L = a_{L+1}/a_L = p_L$. We obtain $m = \prod_1^L p_\ell$, $n = \prod_1^L q_\ell$, and (4.8). For $1 \le \ell \le L-1$ we have

$$
b_{\ell+1} = \frac{d_\ell}{d_{\ell+1}} r_\ell = q_{\ell+1} r_\ell, \quad \text{and } c_\ell = \frac{a_{\ell+1}}{a_\ell} r_\ell = p_\ell r_\ell. \tag{B.2}
$$

By convention $r_0 = r_L = 1$, hence we also have $b_1 = q_1 r_1$ and $c_L = p_L r_L$, so that (4.9) indeed holds. Vice versa it is not difficult to check that given any such integers p_ℓ, q_ℓ, r_ℓ the expressions (4.8)-(4.9) yield a chainable architecture enabling the construction of $\mathbf{1}_{m\times n}$, which is a dense matrix.

We now deal with non-redundancy. By Definition 4.15, each pair $\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}$, $1 \le \ell \le L-1$ is non-redundant if, and only if, $r_\ell := r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1}) < \min(b_\ell, c_{\ell+1}) = \min(q_\ell r_{\ell-1}, p_{\ell+1} r_{\ell+1})$. This reads

$$
\frac{r_\ell}{r_{\ell-1}} < q_\ell \quad \text{and } \frac{r_{\ell+1}}{r_\ell} > \frac{1}{p_{\ell+1}}, \ 1 \le \ell \le L-1
$$

or equivalently (recall that $r_0 = r_L := 1$ by definition): $r_1 < q_1$, $r_{L-1} < p_L$, and (4.10).

Appendix C. Pseudo-orthonormalization operations of Section 6 The goal of this section is to describe the pseudo-orthonormalization operations mentioned in the new butterfly algorithm (Algorithm 6.1), which involve the procedure described in Algorithm C.1.

Algorithm C.1 Column/row-pseudo-orthonormalization

Require: Non-redundant $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, $\mathbf{X} \in \Sigma^{\boldsymbol{\pi}_1}$, $\mathbf{Y} \in \Sigma^{\boldsymbol{\pi}_2}$, $u \in \{\texttt{column}, \texttt{row}\}$
Ensure: $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) \in \Sigma^{\boldsymbol{\pi}_1} \times \Sigma^{\boldsymbol{\pi}_2}$ such that $\tilde{\mathbf{X}}\tilde{\mathbf{Y}} = \mathbf{X}\mathbf{Y}$
1: $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) \leftarrow (\mathbf{0}, \mathbf{0})$
2: **for** $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1}, \mathbf{S}_{\boldsymbol{\pi}_2})$ (cf. Definition 3.3) **do**
3: **if** u is `column` **then**
4: $(\mathbf{Q}, \mathbf{R}) \leftarrow$ QR-decomposition of $\mathbf{X}[R_P, P]$
5: $\tilde{\mathbf{X}}[R_P, P] \leftarrow \mathbf{Q}$
6: $\tilde{\mathbf{Y}}[P, C_P] \leftarrow \mathbf{R}\mathbf{Y}[P, C_P]$
7: **else if** u is `row` **then**
8: $(\mathbf{Q}, \mathbf{R}) \leftarrow$ QR-decomposition of $\mathbf{Y}[P, C_P]^\top$
9: $\tilde{\mathbf{X}}[R_P, P] \leftarrow \mathbf{X}[R_P, P]\mathbf{R}^\top$
10: $\tilde{\mathbf{Y}}[P, C_P] \leftarrow \mathbf{Q}^\top$
11: **end if**
12: **end for**
13: **return** $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}})$

First of all, let us highlight that the orthonormalization operations are well-defined only under the non-redundancy assumption. In Algorithm C.1, the input pair of patterns $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ is assumed to be chainable and non-redundant, thus by Lemma 4.8 and Definition 4.15, we have $|R_P| \leq |P|$ and $|C_P| \leq |P|$, which makes the operations at lines 5 and 10 in Algorithm C.1 well-defined. In Algorithm 6.1, the architecture $\boldsymbol{\beta}$ is assumed to be non-redundant. By Lemma 4.17, this means that the pair $(\boldsymbol{\pi}_{I_k}, \boldsymbol{\pi}_{I_{k+1}})$ at line 12 or the pair $(\boldsymbol{\pi}_{I_{k-1}}, \boldsymbol{\pi}_{I_k})$ at line 15 are chainable and non-redundant. This makes the call to Algorithm C.1 at these lines well-defined.

In the following, we start by providing properties of left/right-r-unitary matrices (cf. Definition 6.1), while the second part of this section is devoted to the proof of Lemma 6.3.

C.1. Properties of left/right-r-unitary factors We introduce properties related to left/right-r-unitary factor from Definition 6.1.

C.1.1. Norm preservation under left and right matrix multiplication

LEMMA C.1. *Consider a chainable $\boldsymbol{\beta} := (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$ and $\mathbf{A}_i \in \Sigma^{\boldsymbol{\pi}_i}$ for $i \in [\![3]\!]$. If $\mathbf{A}_1$ is left-$r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$-unitary and $\mathbf{A}_3$ is right-$r(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3)$-unitary then:*

$$\|\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\|_F = \|\mathbf{A}_2\|_F.$$

Proof. By Lemma 4.9 we have $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, hence

$$\begin{aligned}\|\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\|_F &= \|\mathbf{A}_2\mathbf{A}_3\|_F \quad (\text{since } \mathbf{A}_2\mathbf{A}_3 \in \Sigma^{\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3}, r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)) \\ &= \|\mathbf{A}_2\|_F \quad (\text{by Definition 6.1}).\end{aligned}$$

∎

C.1.2. Stability under matrix multiplication Similarly to classical orthonormal matrices, left/right-r-unitary factors also enjoy a form of stability under matrix multiplication, in the following sense.

LEMMA C.2. *Consider a chainable pair $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ and $\mathbf{A}_i \in \Sigma^{\boldsymbol{\pi}_i}$ for $i \in [\![2]\!]$.*

1. *If $\mathbf{A}_1$ is left-$r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$-unitary and $\mathbf{A}_2$ is left-r-unitary for some integer r, then the product $\mathbf{A}_1\mathbf{A}_2 \in \Sigma^{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}$ is left-r-unitary.*

2. *If* $\mathbf{A}_1$ *is right-*r*-unitary for some integer* r *and* $\mathbf{A}_2$ *is right-*$r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$*-unitary, then the product* $\mathbf{A}_1\mathbf{A}_2 \in \Sigma^{\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2}$ *is right-*r*-unitary.*

Proof. We only prove the first claim. The second one can be dealt with similarly. Denote $\boldsymbol{\pi}_i = (a_i, b_i, c_i, d_i)$ for $i \in [\![2]\!]$. By (4.3), we have:

$$\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2 = \left(a_1, \frac{b_1 d_1}{d_2}, \frac{a_2 c_2}{a_1}, d_2\right).$$

One can verify that $r \mid c_2 \mid a_2 c_2 / a_1$ (since $a_1 \mid a_2$).

Given any pattern $\boldsymbol{\pi}_3$ satisfying $r(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = r$ and any $\boldsymbol{\pi}_3$-factor $\mathbf{A}_3$, we have

$$\|\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\|_F = \|\mathbf{A}_2\mathbf{A}_3\|_F = \|\mathbf{A}_3\|_F,$$

where the first equality comes from the fact that $\mathbf{A}_2\mathbf{A}_3 \in \Sigma^{\boldsymbol{\pi}_2 * \boldsymbol{\pi}_3}$, $\mathbf{A}_1$ is $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$-left-unitary and $r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 * \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ from Lemma 4.9; while the second equality holds because $\mathbf{A}_2$ is left-r-unitary and $r(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = r(\boldsymbol{\pi}_2, \boldsymbol{\pi}_3) = r$ from Lemma 4.9. This concludes the proof. ∎

C.2. Characterization of r-unitary factors and proof of Lemma 6.3 We explicitly characterize left/right-r-unitary factors. This characterization reveals why pseudo-orthonormalization generates left/right-r-unitary factors (i.e., why Lemma 6.3 holds). We first explain why $r \mid c$ is required in Definition 6.1.

LEMMA C.3. *Consider* $\boldsymbol{\pi} := (a, b, c, d)$ *a factor pattern and* $q \in \mathbb{N}$.

- *There exists* $\boldsymbol{\pi}' := (a', b', c', d')$ *such that* $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r$ *if, and only if,* $r \mid c$.
- *There exists* $\boldsymbol{\pi}' := (a', b', c', d')$ *such that* $r(\boldsymbol{\pi}', \boldsymbol{\pi}) = r$ *if, and only if,* $r \mid b$.

Proof. We prove the first claim, the second one is proved similarly. If $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ is chainable, then (by Definition 4.5) $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = ac/a'$ and $a \mid a'$, thus $c = r(a'/a)$ and $r \mid c$. Conversely if $r \mid c$ then $\boldsymbol{\pi}' = (ac/r, r, c, d)$ satisfies the requirements. ∎

Next, we consider the partition $\mathcal{P} = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ of $[\![q]\!]$ (with $q = acd$) and the sets R_P, C_P (for $P \in \mathcal{P}$) from Definition 3.3, with chainable $(\boldsymbol{\pi}, \boldsymbol{\pi}')$. By Lemma 4.8, the sets $R_P \times C_P$ are pairwise disjoint, and for any $\boldsymbol{\pi}$-pattern $\mathbf{X}$ and $\boldsymbol{\pi}'$-pattern $\mathbf{Y}$ we have $\mathtt{supp}(\mathbf{XY}) \subseteq \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}}\mathbf{S}_{\boldsymbol{\pi}'}) = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi} * \boldsymbol{\pi}'}) = \cup_{P \in \mathcal{P}} R_P \times C_P$, so that

$$\|\mathbf{XY}\|_F^2 = \sum_{P \in \mathcal{P}} \|(\mathbf{XY})[R_P, C_P]\|_F^2 \overset{(A.1)}{=} \sum_{P \in \mathcal{P}} \|\mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2. \tag{C.1}$$

This hints that the left-r-unitary of $\boldsymbol{\pi}$-factor $\mathbf{X}$ can be chararacterized via the blocks $\mathbf{X}[R_P, P]$, $P \in \mathcal{P}$. To explicit this characterization we prove that if $\boldsymbol{\pi}'$ satisfies $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r$ then $\mathcal{P} = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ does not depend on $\boldsymbol{\pi}'$. Since it partitions the set $[\![q]\!]$, which indexes the columns of $\mathbf{X}$, we denote it $\mathcal{P}_{\mathtt{col}}(\boldsymbol{\pi}, r)$.

LEMMA C.4. *Consider a pattern* $\boldsymbol{\pi} := (a, b, c, d)$*, an integer* $r \mid c$*, and* $\boldsymbol{\pi}'$ *such that* $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ *is chainable with* $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r$*. The partition* $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ *of Definition* 3.3 *does not depend on* $\boldsymbol{\pi}'$*. We denote it* $\mathcal{P}_{\mathtt{col}}(\boldsymbol{\pi}, r)$*. We have* $|\mathcal{P}| = acd/r = a'd$.

We can similarly define $\mathcal{P}_{\mathtt{row}}(\boldsymbol{\pi}, r)$ when $r \mid b$, with an analog property $\mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}'}, \mathbf{S}_{\boldsymbol{\pi}}) = \mathcal{P}_{\mathtt{row}}(\boldsymbol{\pi}, r)$ whenever $(\boldsymbol{\pi}', \boldsymbol{\pi})$ is chainable with $r(\boldsymbol{\pi}', \boldsymbol{\pi}) = r$. The proof of Lemma C.4 is slightly postponed to immediately state and prove our main characterization.

THEOREM C.5. *Consider a pattern* $\boldsymbol{\pi} := (a, b, c, d)$ *and a natural number* r *such that* $r \mid c$ *(resp.* $r \mid b$*). Let* $\mathbf{X}$ *be a* $\boldsymbol{\pi}$*-factor. The following claims are equivalent:*

1. $\mathbf{X}$ *is left-*r*-unitary (resp. right-*r*-unitary).*

2. *For each* $P \in \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$ *(resp.* $P \in \mathcal{P}_{\texttt{row}}(\boldsymbol{\pi}, r)$*),* $\mathbf{X}[R_P, P]$ *(resp.* $\mathbf{X}[P, C_P]$*) has orthogonal columns (resp. rows) (*R_P, C_P, P *are defined as in Definition* 3.3*).*

Proof of Theorem C.5. We prove the claim for left-r-unitarity (right-r-unitarity is proved similarly). Let $\boldsymbol{\pi}'$ be a pattern satisfying $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r$. Denoting $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$, by Lemma C.4 we have $\mathcal{P} = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$. We exploit (C.1).

1. Assume that $\mathbf{X}$ is left-r-unitary, and fix an arbitrary $P \in \mathcal{P}$. For any $\boldsymbol{\pi}'$-factor $\mathbf{Y}$ such that $\mathbf{Y}[R_{P'}, P'] = \mathbf{0}$ for all $P' \in \mathcal{P}$, $P' \neq P$, by left-r-unitarity of $\mathbf{X}$ we have:

$$\|\mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2 \overset{\text{(C.1)}}{=} \|\mathbf{XY}\|_F^2 = \|\mathbf{Y}\|_F^2 = \|\mathbf{Y}[P, C_P]\|_F^2.$$

   Thus $\mathbf{X}[R_P, P]$ preserves the Frobenius norm of $\mathbf{Y}[P, C_P]$ upon left multiplication. Since this holds for any choice of $\mathbf{Y}[P, C_P]$, the matrix $\mathbf{X}[R_P, P]$ has orthogonal columns. This shows the first implication.
2. Assume that $\mathbf{X}[R_P, P]$ has orthogonal columns for each $P \in \mathcal{P}$. For each $\boldsymbol{\pi}'$-factor $\mathbf{Y}$ and $P \in \mathcal{P}$ we have $\|\mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2 = \|\mathbf{Y}[P, C_P]\|_F^2$, thus,

$$\begin{aligned}\|\mathbf{XY}\|_F^2 \overset{\text{(C.1)}}{=} \sum_{P \in \mathcal{P}} \|\mathbf{X}[R_P, P]\mathbf{Y}[P, C_P]\|_F^2 &= \sum_{P \in \mathcal{P}} \|\mathbf{Y}[P, C_P]\|_F^2 \\ &= \sum_{P \in \mathcal{P}} \|\mathbf{Y}[P, :]\|_F^2 = \|\mathbf{Y}\|_F^2.\end{aligned}$$

   where the first equality of the second row results from the fact that the sub-matrix $\mathbf{Y}[P, :]$ has the form $\begin{bmatrix} \mathbf{Y}[P, C_P] & \mathbf{0} \end{bmatrix}$ up to some column permutations. To see this fact, we remind readers that by definition of $\mathcal{P} = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ (see Definition 3.3) all rows of $\mathbf{S}_{\boldsymbol{\pi}'}[P, :]$ are equal and their support is C_P, and since $\mathbf{Y} \in \Sigma^{\boldsymbol{\pi}'}$ the corresponding rows have support included in C_P . ∎

To prove Lemma C.4 we will actually show that under its assumptions we have $\mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'}) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$ where $\mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$ is specified as follows:

Definition C.6. *Consider a pattern* $\boldsymbol{\pi} := (a, b, c, d)$ *and a natural number* $r \mid c$*. For each pair of integers* $(t, k) \in [\![ac/r]\!] \times [\![d]\!]$ *denote*

$$P_{t,k} := \{k + (t-1)dr + (j-1)d\}_{j \in [\![r]\!]} \subseteq [\![q]\!] \text{ with } q := acd, \tag{C.2}$$

and define $\mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r) := \{P_{t,k}\}_{t,k}$*. An illustration for these sets is given on Figure* 6.

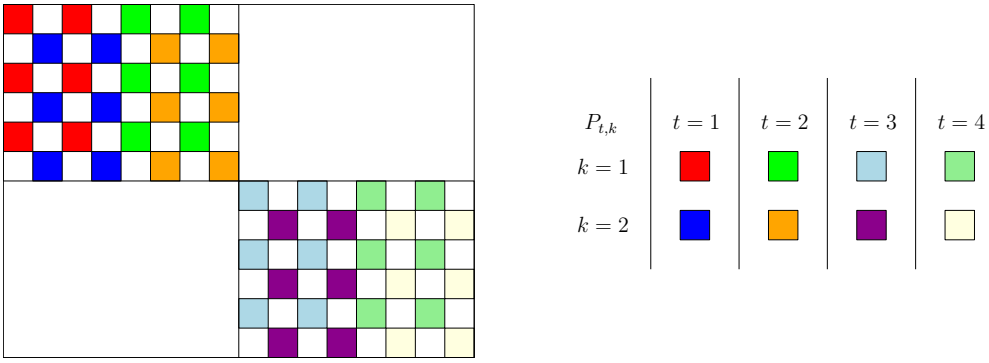


Fig. 6: The partition $\mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$ with $\boldsymbol{\pi} = (2, 3, 4, 2)$ and $r = 2$. Each set $P_{t,k}, (t, k) \in [\![4]\!] \times [\![2]\!]$ gathers the indices of the columns of $\mathbf{S}_{\boldsymbol{\pi}}$ of a given color. See also Figure 1. Same color indicates columns of the same sets $P_{t,k}$ (cf. Definition C.6).

Proof of Lemma C.4. It is easy to check that $\mathcal{P}' := \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$ partitions the index set $[\![q]\!]$, with $q = acd$, into $|\mathcal{P}'| = acd/r$ components $P = P_{t,k} \in \mathcal{P}'$ of equal cardinality $|P| = r$. Since, by Lemma 4.8, the partition $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ of $[\![q]\!]$ has the same property, it is sufficient to prove that, for any $P \in \mathcal{P}$, there exists $(t, k) \in [\![ac/r]\!] \times [\![d]\!]$ such that $P = P_{t,k}$ (such a pair (t, k) is necessarily unique since we consider partitions).

Given an arbitrary $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$, we prove first that there exists a unique $t \in [\![ac/r]\!]$ such that $P \subseteq P_t := \bigcup_{k \in [\![d]\!]} P_{t,k}$. Notice (see also Figure 6) that each P_t is an interval of length dr and since (by chainability of $(\boldsymbol{\pi}, \boldsymbol{\pi}')$) we have $acd = a'b'd'$, $ac/r = a'$, and $dr = b'd'$,

$$P_t = [\![(t-1)dr + 1, tdr]\!] = [\![(t-1)b'd' + 1, tb'd']\!]. \tag{C.3}$$

Further observe that $\{P_t\}_{t \in [\![ac/r]\!]}$ partitions $[\![q]\!] = [\![a'b'd']\!]$ into a' consecutive intervals of length $b'd'$. In other words, P_t indexes the rows of the t-th block of $\mathbf{S}_{\boldsymbol{\pi}'}$, cf. Figure 1. As a result, there exists at least one index t such that $P_t \cap P \neq \emptyset$. We next prove the uniqueness of such a t, which implies $P \subseteq P_t$ as we consider a partition of $[\![q]\!]$. Consider two indices t, t' such that $P_t \cap P \neq \emptyset$ and $P_{t'} \cap P \neq \emptyset$, and $i \in P_t \cap P$, $i' \in P_{t'} \cap P$. By Definition 3.3, since $i, i' \in P$ and $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$, we have $\mathbf{S}_{\boldsymbol{\pi}'}[i, :] = \mathbf{S}_{\boldsymbol{\pi}'}[i', :]$. Since $\mathbf{S}_{\boldsymbol{\pi}'} = \mathbf{I}_{a'} \otimes \mathbf{1}_{b' \times c'} \otimes \mathbf{I}_{d'}$ is block-diagonal with a' blocks of size $b'd' \times c'd'$, the row $\mathbf{S}_{\boldsymbol{\pi}'}[i, :]$ is supported in a subset of the interval $[\![(\ell-1)c'd' + 1, \ell c'd']\!]$ with $\ell \in [\![a']\!]$ the unique integer such that $i \in [\![(\ell-1)b'd' + 1, \ell b'd']\!] = P_\ell$. Since $i \in P_t$ we have $\ell = t$. The same holds for $\mathbf{S}_{\boldsymbol{\pi}'}[i', :]$ with $\ell' = t'$. Since both rows are identical, we deduce that $\ell = \ell'$, hence $t = t'$.

Considering now the unique $t \in [\![ac/r]\!]$ such that $P \subseteq P_t$, since $\{P_{t,k}\}_{k \in [\![d]\!]}$ partitions P_t, there must exist $k \in [\![d]\!]$ such that $P_{t,k} \cap P \neq \emptyset$. Again, we now prove the uniqueness of such a k, which implies that $P \subseteq P_{t,k}$, and since $|P| = q = |P_{t,k}|$, we will obtain $P = P_{t,k}$ as claimed. Consider two indices $k, k' \in [\![d]\!]$ such that $P_{t,k} \cap P \neq \emptyset$ and $P_{t,k'} \cap P \neq \emptyset$, and $i \in P \cap P_{t,k}$ and $i' \in P \cap P_{t,k'}$. By construction (see Definition C.6), we have $i \equiv k \mod d$ and $i \equiv k' \mod d$. To continue we observe that $\{P_{t,k}\}_{k \in [\![d]\!]}$ partitions the interval P_t into d (disjoint) subsets of integers of cardinality r and the elements in such subsets are equally spaced by a distance d. Moreover, as above, since $i, i' \in P$ and $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$, we have $\mathbf{S}_{\boldsymbol{\pi}}[:, i] = \mathbf{S}_{\boldsymbol{\pi}}[:, i']$. By the structure $\mathbf{S}_{\boldsymbol{\pi}} = \mathbf{I}_a \otimes \mathbf{1}_{b \times c} \otimes \mathbf{I}_d$, for each k the columns of $\mathbf{S}_{\boldsymbol{\pi}}$ such that $\texttt{supp}(\mathbf{S}[:, i]) \subseteq P_{t,k}$ share in fact the same support, which is disjoint from the other column supports (see illustration of Figure 6).

Finally, by Definition 4.5 and chainability of $(\boldsymbol{\pi}, \boldsymbol{\pi}')$, we have $r = r(\boldsymbol{\pi}, \boldsymbol{\pi}') = ac/a'$, so that $|\mathcal{P}| = |\mathcal{P}'| = acd/r = a'd$ as claimed. ∎

We conclude the section with a lemma that relates $\mathcal{P}_{\texttt{col}}(\cdot, r)$ for various patterns.

LEMMA C.7. *Consider a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ and a natural number $r \mid c_L$ where $\boldsymbol{\pi}_L = (a_L, b_L, c_L, d_L)$. Then $r \mid c$ where $\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = (a, b, c, d)$, and $\mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_L, r)$.*

Proof. We prove the result for $L = 2$. The general case follows by induction. Consider $\boldsymbol{\pi}_i = (a_i, b_i, c_i, d_i)$, with $i \in [\![2]\!]$, and an integer $r \mid c_2$. By definition of the $*$ operator in Equation (4.3) we have

$$(a, b, c, d) := \boldsymbol{\pi}_1 * \boldsymbol{\pi}_2 = \left(a_1, \frac{b_1 d_1}{d_2}, \frac{a_2 c_2}{a_1}, d_2\right)$$

Since $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ are chainable, we have $a_1 \mid a_2$. Since $r \mid c_2$, we have $r \mid (a_2/a_1)c_2$, i.e., $r \mid c$ as claimed. A direct calculation then shows that $\mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_2, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_1 * \boldsymbol{\pi}_2, r)$,

since for each $t \in [\![ac/r]\!] = [\![a_2c_2/r]\!]$, $k \in [\![d]\!] = [\![d_2]\!]$, (C.2) yields the very same set $P_{t,k}$ (as $dr = d_2 r$ and $d = d_2$). ∎

C.2.1. Proof of Lemma 6.3 This is a direct corollary of Lemma C.8 below.

With the notations of Lemma 6.3, the constant r involved in Lemma C.8 applied to $\boldsymbol{\pi} = \boldsymbol{\pi}_{I_i}$ and $\boldsymbol{\pi}' = \boldsymbol{\pi}_{I_{i+1}}$ for $i = 1, \dots, j-1$ is indeed equal to $r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{t_i+1})$:

1. The intervals $I_i := [\![q_i, t_i]\!]$ in `partition` are a sorted partition of $[\![1, L]\!]$. Therefore, $t_i + 1 = q_{i+1}$ for each i.
2. By Lemma 4.14, $r(\boldsymbol{\pi}_{I_i}, \boldsymbol{\pi}_{I_{i+1}}) = r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{q_{i+1}}) = r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{t_i+1})$.

LEMMA C.8. *Consider a non-redundant chainable pair $(\boldsymbol{\pi}, \boldsymbol{\pi}')$, and $(\mathbf{X}, \mathbf{Y}) \in \Sigma^{\boldsymbol{\pi}} \times \Sigma^{\boldsymbol{\pi}'}$. Denote $r := r(\boldsymbol{\pi}, \boldsymbol{\pi}')$. Algorithm* C.1 *with input $u \in \{\texttt{column}, \texttt{row}\}$ returns $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) \in \Sigma^{\boldsymbol{\pi}} \times \Sigma^{\boldsymbol{\pi}'}$ such that $\tilde{\mathbf{X}}\tilde{\mathbf{Y}} = \mathbf{XY}$ and:*

- *if $u = \texttt{column}$ then $\tilde{\mathbf{X}}$ is left-r-unitary;*
- *otherwise if $u = \texttt{row}$ then $\tilde{\mathbf{Y}}$ is right-r-unitary.*

Proof. We only prove the first point, as the second point can be addressed similarly. Denoting $\mathcal{P} = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$, by Lemma 4.8, for any $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) \in \Sigma^{\boldsymbol{\pi}} \times \Sigma^{\boldsymbol{\pi}'}$, $\texttt{supp}(\mathbf{XY})$ and $\texttt{supp}(\tilde{\mathbf{X}}\tilde{\mathbf{Y}})$ are both included in $\bigcup_{P \in \mathcal{P}} R_P \times C_P$ where $\{R_P \times C_P\}_{P \in \mathcal{P}}$ are pairwise disjoint. As a result, we have:

$$\begin{aligned} \mathbf{XY} = \tilde{\mathbf{X}}\tilde{\mathbf{Y}} &\iff \forall P \in \mathcal{P}, \quad (\mathbf{XY})[R_P, C_P] = (\tilde{\mathbf{X}}\tilde{\mathbf{Y}})[R_P, C_P] \\ &\iff \forall P \in \mathcal{P}, \quad (\mathbf{XY})[R_P, C_P] = \tilde{\mathbf{X}}[R_P, P]\tilde{\mathbf{Y}}[P, C_P], \end{aligned}$$

where the second equivalence comes by Lemma A.1 and by chainability of $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. Moreover, by Theorem C.5, $\tilde{\mathbf{X}} \in \Sigma^{\boldsymbol{\pi}}$ is left-r-unitary if, and only if, $\tilde{\mathbf{X}}[R_P, P]$ has orthonormal columns for each $P \in \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r) = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'}) = \mathcal{P}$.

The output $(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}})$ of Algorithm C.1 is built via the QR-decomposition of $\mathbf{X}[R_P, P] = \mathbf{QR}$, and setting $\tilde{\mathbf{X}}[R_P, P] = \mathbf{Q}$, $\tilde{\mathbf{Y}}[P, C_P] = \mathbf{RY}[P, C_P]$, which is possible because $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ is assumed to be non-redundant. Indeed, to enable $\mathbf{Q} \in \mathbb{R}^{|R_P| \times |P|}$ to have orthonormal columns, we must have $|R_P| \geq |P|$, or equivalently, $b \geq q$ (assuming that $\boldsymbol{\pi} = (a, b, c, d)$, see Lemma 4.8). This is the non-redundancy criterion (cf. Definition 4.15). Thus, $\tilde{\mathbf{X}}[R_P, P]$ has orthonormal columns and $\tilde{\mathbf{X}}[R_P, P]\tilde{\mathbf{Y}}[P, C_P] = \mathbf{QRY}[P, C_P] = \mathbf{X}[R_P, P]\mathbf{Y}[P, C_P] = (\mathbf{XY})[R_P, C_P]$. ∎

C.2.2. Further useful technical properties of r-unitary factors

LEMMA C.9. *Let $\mathbf{X}$ be a left-r-unitary $\boldsymbol{\pi}$-factor with $\boldsymbol{\pi} = (a, b, c, d)$. For any $t \in [\![ac/r]\!]$, the submatrix $\mathbf{X}[:, P_t]$ with P_t as in* (C.3) *has orthonormal columns.*

Proof. From the proof of Lemma C.4 the sets $(P_{t,k})_{k \in [\![d]\!]}$ from Definition C.6 partition P_t into d subsets of integers of cardinality r and the elements of each subsets are equally spaced by a distance d. We first show that the columns of $\mathbf{X}[:, P_t]$ coming from distinct blocks $P := P_{t,k}$, $P' := P_{t,k'}$ (where $1 \leq k \neq k' \leq d$) are mutually orthogonal, as they have disjoint supports. Due to the structure of $\mathbf{S}_{\boldsymbol{\pi}} = \mathbf{I}_a \otimes \mathbf{1}_{b \times c} \otimes \mathbf{I}_d$, if two columns share the same support, the difference of their indices is divisible by d. Otherwise, their supports are disjoint. Thus, P and P' have disjoint column supports.

To conclude we show that each submatrix $\mathbf{X}[:, P_{t,k}]$ has orthonormal columns. Indeed, by Definition C.6, we have $P := P_{t,k} \in \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}, r)$, hence by Theorem C.5 the block $\mathbf{X}[R_P, P]$ has orthonormal columns. Since R_P is the support of the columns of $\mathbf{S}_{\boldsymbol{\pi}}[:, P]$, and as $\texttt{supp}(\mathbf{X}) \subseteq \texttt{supp}(\mathbf{S}_{\boldsymbol{\pi}})$, the submatrix $\mathbf{X}[:, P]$ is zero outside of the block $\mathbf{X}[R_P, P]$ hence it also has orthonormal columns.

Remark C.10. Although the columns of $\mathbf{X}[:, P_t]$ are orthonormal and the sets P_t are pairwise disjoint, it does not imply that $\mathbf{X}$ has orthonormal columns. For example, consider the sets P_1 (indices of columns in red and blue) and P_2 (columns in green and orange) illustrated in Figure 6. Even though these two sets are disjoint the supports of their columns are not. Thus, the matrix can be left-2-unitary without having orthonormal columns. This example also explains our earlier Remark 6.2.

COROLLARY C.11. *Consider a chainable pair of patterns* $(\boldsymbol{\pi}, \boldsymbol{\pi}')$. *If* $\mathbf{X} \in \Sigma^{\boldsymbol{\pi}}$ *is left-*$r(\boldsymbol{\pi}, \boldsymbol{\pi}')$*-unitary, then for any column index* i *of* $\mathbf{S}_{\boldsymbol{\pi}'}$ *the submatrix* $\mathbf{X}[:, T_i]$, *where* $T_i = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}'}[:, i])$, *has orthonormal columns.*

Proof. Denote $\boldsymbol{\pi} = (a, b, c, d)$, $\boldsymbol{\pi}' = (a', b', c', d')$, $r = r(\boldsymbol{\pi}, \boldsymbol{\pi}')$. Since $\mathbf{S}_{\boldsymbol{\pi}'} = \mathbf{I}_{a'} \otimes \mathbf{1}_{b' \times c'} \otimes \mathbf{I}_{d'}$ is block-diagonal with a' blocks of size $b'd' \times c'd'$, the set T_i is a subset of the interval $P_t = [\![(t-1)b'd' + 1, tb'd']\!]$ with $t \in [\![a']\!]$ the unique integer such that $i \in [\![(t-1)c'd', tc'd']\!]$. By chainability of $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ we have $b'd' = dr$. By Lemma C.9, $\mathbf{X}[:, P_t]$ has orthonormal columns, and therefore so does $\mathbf{X}[:, T_i]$. ∎

The following corollary will be handy to prove (E.9), in the proof of Lemma E.1.

COROLLARY C.12. *Consider a chainable architecture of* $\boldsymbol{\beta} = (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_L)$. *Denote*

$$\boldsymbol{\pi}'_1 := \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{q-1}, \quad \boldsymbol{\pi}'_2 := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t, \quad \boldsymbol{\pi}'_3 := \boldsymbol{\pi}_{t+1} * \ldots * \boldsymbol{\pi}_L.$$

If $\mathbf{X} \in \Sigma^{\boldsymbol{\pi}'_1}$ *and* $\mathbf{Z} \in \Sigma^{\boldsymbol{\pi}'_3}$ *are respectively left-*$r(\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2)$*-unitary and right-*$r(\boldsymbol{\pi}'_2, \boldsymbol{\pi}'_3)$*-unitary, then* $\mathbf{X}[:, R]$ *(resp.* $\mathbf{Z}[C, :]$*) has orthonormal columns (resp. rows) for any column (resp. row) support* R *(resp.* C*) of* $\mathbf{S}_{\boldsymbol{\pi}'_2}$, *i.e., whenever* $R = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_2}[:, i])$ *(resp.* $C = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_2}[i, :])$*) for some row (resp. column) index* i.

Proof. We consider the case of columns of $\mathbf{X}[:, R]$. The other case can be dealt with similarly. The proof follows from Corollary C.11 because:

1. By Lemma 4.14, $(\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2)$ is chainable with $r(\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2) = r(\boldsymbol{\pi}_{q-1}, \boldsymbol{\pi}_q)$;
2. $\mathbf{X}$ is a left-$r(\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2)$-unitary $\boldsymbol{\pi}'_1$-factor;
3. $R = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_2}[:, i])$ for some column index i. ∎

Appendix D. Complexity of hierarchical algorithms - proof of Theorem 6.5 We analyze the complexity of each of the components involved in the proposed hierarchical algorithms.

D.1. Complexity of Algorithm 3.1 This algorithm essentially performs several low-rank approximations that are typically computed using truncated SVD.

LEMMA D.1. *Consider* $(\mathbf{L}, \mathbf{R})$ *satisfying the condition of Theorem* 3.4. *For any matrix* $\mathbf{A}$, *the complexity of the two-factor fixed support matrix factorization algorithm (Algorithm* 3.1*) with input* $\mathbf{A}, \mathbf{L}, \mathbf{R}$ *is*

$$\mathcal{O}\left(\sum_{P \in \mathcal{P}(\mathbf{L}, \mathbf{R})} |P||R_P||C_P| \right).$$

Proof. The algorithm performs the best rank-$|P|$ approximation of a submatrix of size $|R_P| \times |C_P|$ for each $P \in \mathcal{P}(\mathbf{L}, \mathbf{R})$ and the complexity of the truncated SVD at order k for an $m \times n$ matrix is $\mathcal{O}(kmn)$ [16]. ∎

We apply this complexity analysis to the case where $(\mathbf{L}, \mathbf{R})$ are butterfly supports corresponding to a chainable pair of patterns.

LEMMA D.2. *Consider chainable patterns* $\boldsymbol{\pi} = (a, b, c, d)$, $\boldsymbol{\pi}' = (a', b', c', d')$. *Denoting* $r := r(\boldsymbol{\pi}, \boldsymbol{\pi}')$, *the complexity* $C(\boldsymbol{\pi}, \boldsymbol{\pi}')$ *of the two-factor fixed support matrix factorization algorithm (Algorithm* 3.1*) with input* $\mathbf{A}, \mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'}$ *is*

$$C(\boldsymbol{\pi}, \boldsymbol{\pi}') = \mathcal{O}(ra'bc'd). \tag{D.1}$$

Proof. By Lemma 4.8, for each $P \in \mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ we have $|P| = r$, $|R_P| = b$ and $|C_P| = c'$. By Lemma C.4 we have $|\mathcal{P}| = a'd$. By Lemma D.1 it follows that $C(\boldsymbol{\pi}, \boldsymbol{\pi}') = \mathcal{O}(a'drbc')$.

Remark D.3. In practice, best low-rank approximations in Algorithm 3.1 via truncated SVDs can be computed in parallel. For arbitrary $\mathbf{L}, \mathbf{R}$ as in Lemma D.1 this can decrease the complexity down to $\mathcal{O}(\max_{P \in \mathcal{P}(\mathbf{L},\mathbf{R})} |P||R_P||C_P|)$ when parallelizing accross $|\mathcal{P}(\mathbf{L}, \mathbf{R})|$ processes. For butterfly factors as in Lemma D.2, this decreases the complexity down to $\mathcal{O}(rbc')$.

D.2. Complexity of Algorithm 5.1 This algorithm is based on Algorithm 3.1. We prove the first point of Theorem 6.5 in the following lemma.

LEMMA D.4. *Consider a non-redundant chainable architecture* $\boldsymbol{\beta}$ *and a matrix* $\mathbf{A}$ *of size* $m \times n$. *With notations of Theorem* 6.5, *the complexity of the hierarchical algorithm (Algorithm* 5.1*) with inputs* $\boldsymbol{\beta}$, $\mathbf{A}$ *and any factor-bracketing tree* $\mathcal{T}$ *is at most:*

- $\mathcal{O}(\|\mathbf{r}(\boldsymbol{\beta})\|_1 M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}})$ *in the general case;*
- $\mathcal{O}(\|\mathbf{r}(\boldsymbol{\beta})\|_1 mn)$ *if* $\boldsymbol{\beta}$ *is non-redundant.*

Proof. Since Algorithm 5.1 performs $(L-1)$ factorizations of the form of Problem (5.1) using the two-factor fixed-support matrix factorization algorithm (Algorithm 3.1), its complexity is equal to the sum of the complexity of each of these $(L-1)$ factorizations.

Fix $1 \leq q \leq s < t \leq L$. By Lemma 4.14, $r_s := r(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t) = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$. By (4.5) we have

$$\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s = \left(a_q, \frac{b_q d_q}{d_s}, \frac{a_s c_s}{a_q}, d_s\right), \ \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t = \left(a_{s+1}, \frac{b_{s+1} d_{s+1}}{d_t}, \frac{a_t c_t}{a_{s+1}}, d_t\right),$$

hence, Lemma D.2 yields $C(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t) = \mathcal{O}(r_s a_t c_t b_q d_q)$, which is upper bounded by $\mathcal{O}(r_s M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}})$, by definition of $M_{\boldsymbol{\beta}} := \max_\ell a_\ell c_\ell, N_{\boldsymbol{\beta}} := \max_\ell b_\ell d_\ell$. Therefore, the overall complexity of Algorithm 5.1 is upper bounded by:

$$\sum_{s=1}^{L-1} \mathcal{O}(r_s M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}}) = \mathcal{O}(\|\mathbf{r}(\boldsymbol{\beta})\|_1 M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}}). \tag{D.2}$$

When $\boldsymbol{\beta}$ is non-redundant, by Definition 4.15 we have $a_\ell c_\ell < a_{\ell+1} c_{\ell+1}$ and $b_\ell d_\ell > b_{\ell+1} d_{\ell+1}$ for all $\ell \in [\![L-1]\!]$. Since $m = a_1 b_1 d_1$ and $n = a_L c_L d_L$ (cf. Lemma 4.12 and Definition 4.1) we obtain $M_{\boldsymbol{\beta}} = \max_\ell a_\ell c_\ell < a_L c_L = n/d_L \leq n$ and $N_{\boldsymbol{\beta}} = \max_\ell b_\ell d_\ell < b_1 d_1 = m/a_1 \leq m$. ∎

D.3. Complexity of Algorithm C.1 We now analyze the complexity of the construction of orthonormal butterfly factors in Algorithm C.1.

LEMMA D.5. *The complexity of column/row-pseudo-orthonormalization (Algorithm* C.1*) with input* $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \mathbf{X}, \mathbf{Y}, u)$ *for any* $u \in \{\texttt{column}, \texttt{row}\}$ *and any non-redundant chainable pair* $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ *is*

$$\mathcal{O}(r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)(\|\boldsymbol{\pi}_1\|_0 + \|\boldsymbol{\pi}_2\|_0)),$$

with the notation $\|\boldsymbol{\pi}\|_0$ *from Lemma* 4.3.

Proof. We only consider the case $u = \texttt{column}$, since the other case can be dealt with similarly. Denote $r = r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ and $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$ for $\ell \in [\![2]\!]$. By Lemma 4.8 for each $P \in \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ we have $|P| = r$, $|R_P| = b_1$, $|C_P| = c_2$, and we have $r \leq \min(b_1, c_2)$ (cf. Definition 4.5). Thus, at each iteration of Algorithm C.1:

- the complexity of line 4 is $\mathcal{O}(|R_P||P|^2) = \mathcal{O}(b_1 r^2)$, since the complexity of QR-decomposition of an $m \times n$ matrix is $\mathcal{O}(\min(m,n)mn)$ [12, Section 5.2];
- the complexity of line 5 is $\mathcal{O}(|R_P||P|) = \mathcal{O}(b_1 r)$;
- the complexity of line 6 is $\mathcal{O}(|P|^2|C_P|) = \mathcal{O}(c_2 r^2)$.

Overall, the complexity of an iteration is $\mathcal{O}(r^2(b_1+c_2))$. There are $a_1c_1d_1/r = a_2b_2d_2/r$ equivalence classes (by Lemma C.4 and chainability, that implies $a_1c_1d_1 = a_2b_2d_2$, see Definition 4.5), and by Lemma 4.3 $a_\ell b_\ell c_\ell d_\ell = \|\boldsymbol{\pi}_\ell\|_0$, hence the overall complexity is

$$\begin{aligned}\mathcal{O}\left(\frac{a_1c_1d_1}{r}r^2b_1 + \frac{a_2b_2d_2}{r}r^2c_2\right) &= \mathcal{O}(r(a_1b_1c_1d_1 + a_2b_2c_2d_2)) \\ &= \mathcal{O}(r(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)(\|\boldsymbol{\pi}_1\|_0 + \|\boldsymbol{\pi}_2\|_0)).\end{aligned}$$

∎

D.4. Complexity of Algorithm 6.1 We can now prove the second point of Theorem 6.5, where we assume that $\boldsymbol{\beta}$ is not redundant so that $M_{\boldsymbol{\beta}} \leq m$ and $N_{\boldsymbol{\beta}} \leq n$. By Lemma D.5 the complexity for each call to Algorithm C.1 in lines 12 and 15 of the new butterfly algorithm (Algorithm 6.1) is given by $\mathcal{O}(r(\boldsymbol{\pi}_{I_{k-1}}, \boldsymbol{\pi}_{I_k})(\|\boldsymbol{\pi}_{I_{k-1}}\|_0 + \|\boldsymbol{\pi}_{I_k}\|_0))$ where $\boldsymbol{\pi}_I = \boldsymbol{\pi}_q * \ldots \boldsymbol{\pi}_s$ if $I := [\![q, s]\!]$. By Lemma 4.14, $r(\boldsymbol{\pi}_{I_{k-1}}, \boldsymbol{\pi}_{I_k}) = r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$ for some $1 \leq s \leq L-1$, thus is bounded by $\|r(\boldsymbol{\beta})\|_\infty$ (cf. Definition 4.10). Moreover, $\boldsymbol{\pi}_{I_k}$ (resp. $\boldsymbol{\pi}_{I_{k-1}}$) has the form $(a_q, \frac{b_q d_q}{d_s}, \frac{a_s c_s}{a_q}, d_s)$ for some $1 \leq q \leq s \leq L$ (cf. Lemma 4.12). Thus, $\|\boldsymbol{\pi}_{I_k}\|_0$ (resp. $\|\boldsymbol{\pi}_{I_{k-1}}\|_0$) is bounded (recall the notation $\|\boldsymbol{\pi}\|_0$ from Lemma 4.3) by $\max_{q,s} b_q d_q a_s c_s \leq M_{\boldsymbol{\beta}} N_{\boldsymbol{\beta}} \leq mn$. In conclusion, the complexity of each call to Algorithm C.1 is at most $\mathcal{O}(mn\|\mathbf{r}(\boldsymbol{\beta})\|_\infty)$. At the J-th iteration for some $J \in [\![L-1]\!]$, there are at most $(J-1)$ calls to Algorithm C.1, so the total complexity for the orthonormalization operations accross all the $|\boldsymbol{\beta}| - 1$ iterations is at most $\mathcal{O}(|\boldsymbol{\beta}|^2 mn\|\mathbf{r}(\boldsymbol{\beta})\|_\infty)$. Therefore, the complexity of Algorithm 6.1 is given by

$$\mathcal{O}\left((|\boldsymbol{\beta}|^2\|\mathbf{r}(\boldsymbol{\beta})\|_\infty + \|\mathbf{r}(\boldsymbol{\beta})\|_1)mn\right).$$

Appendix E. Proof of the results of Section 7

E.1. Proof of Lemma 7.9 First, we prove that $\mathbf{B}' \in \mathcal{B}^{\boldsymbol{\beta}_s}$. By definition

$$\mathbf{B}' = \underbrace{\left(\mathbf{X}_{P_1} \ldots \mathbf{X}_{P_{j-1}} \mathbf{X}_{[\![q,s]\!]}\right)}_{\mathbf{B}'_{(\text{left})}} \underbrace{\left(\mathbf{X}_{[\![s+1,t]\!]} \mathbf{X}_{P_{j+1}} \ldots \mathbf{X}_{P_J}\right)}_{\mathbf{B}'_{(\text{right})}}$$

and since by assumption $\mathbf{X}_{P_i} \in \Sigma^{\boldsymbol{\pi}_{q_i} * \cdots * \boldsymbol{\pi}_{t_i}}$ for every i, and $\boldsymbol{\beta}$ is chainable, a recursive application of Proposition 4.7 yields that $\mathbf{B}'_{(\text{left})} \in \Sigma^{\boldsymbol{\pi}_1 * \cdots * \boldsymbol{\pi}_s}$, $\mathbf{B}'_{(\text{right})} \in \Sigma^{\boldsymbol{\pi}_{s+1} * \cdots * \boldsymbol{\pi}_L}$. Hence, $\mathbf{B}' \in \mathcal{B}^{\boldsymbol{\beta}_s}$.

We now prove Equation (7.11). By Lemma 4.14 and chainability of $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$, the patterns $\boldsymbol{\pi}'_1 := \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{q-1}$, $\boldsymbol{\pi}'_2 := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t$ and $\boldsymbol{\pi}'_3 := \boldsymbol{\pi}_{t+1} * \ldots * \boldsymbol{\pi}_L$ are well-defined, and the two-factor architectures $\boldsymbol{\beta}_{\texttt{left}} := (\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2)$ and $\boldsymbol{\beta}_{\texttt{right}} := (\boldsymbol{\pi}'_2, \boldsymbol{\pi}'_3)$ are both chainable with

$$r(\boldsymbol{\pi}'_1, \boldsymbol{\pi}'_2) = r(\boldsymbol{\pi}_{q-1}, \boldsymbol{\pi}_q) \text{ and } r(\boldsymbol{\pi}'_2, \boldsymbol{\pi}'_3) = r(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1}).$$

Moreover, by assumption, for each $i \in [\![j-1]\!]$ the factor $\mathbf{X}_{P_i}$ is left-$r(\boldsymbol{\pi}_{t_i}, \boldsymbol{\pi}_{t_i+1})$-unitary hence, using the notations $\mathbf{X}^{(J)}_{\text{left}}$, $\mathbf{X}^{(J)}_{\text{right}}$ of (7.9), multiple applications of Lemma C.2 yield that $\mathbf{X}^{(J)}_{\text{left}} \in \Sigma^{\boldsymbol{\pi}'_1}$ is left-$r(\boldsymbol{\pi}_{r-1}, \boldsymbol{\pi}_r)$-unitary if $r \geq 2$ (if $r = 1$ then by convention $\mathbf{X}^{(J)}_{\text{left}}$ is the identity). Similarly, $\mathbf{X}^{(J)}_{\text{right}} \in \Sigma^{\boldsymbol{\pi}'_3}$ is right-$r(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1})$-unitary when $t \leq L-1$ (and the identity when $t = L$). Finally, by Proposition 4.7, both $\mathbf{X}_{[\![q,t]\!]}$ and $\mathbf{X}_{[\![q,s]\!]}\mathbf{X}_{[\![s+1,t]\!]}$ (and therefore their difference) belong to $\Sigma^{(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t)} = \Sigma^{\boldsymbol{\pi}'_2}$, and denoting $\tilde{\boldsymbol{\pi}}_1 := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_{s-1}$ and $\tilde{\boldsymbol{\pi}}_2 = \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t$, since $\mathbf{X}_{[\![q,s]\!]}$ and $\mathbf{X}_{[\![s+1,t]\!]}$ solve (5.1) we obtain

$$\begin{aligned}
\|\mathbf{B}' - \mathbf{B}\|_F &= \|\mathbf{X}^{(J)}_{\text{left}}(\mathbf{X}_{[\![q,t]\!]} - \mathbf{X}_{[\![q,s]\!]}\mathbf{X}_{[\![s+1,t]\!]})\mathbf{X}^{(J)}_{\text{right}}\|_F \quad \text{by definitions of } \mathbf{B}, \mathbf{B}' \\
&= \|\mathbf{X}_{[\![q,t]\!]} - \mathbf{X}_{[\![q,s]\!]}\mathbf{X}_{[\![s+1,t]\!]}\|_F \quad \text{by Lemma C.1} \\
&= \inf_{\mathbf{Y}_i \in \Sigma^{\tilde{\boldsymbol{\pi}}_i}, i=1,2} \|\mathbf{X}_{[\![q,t]\!]} - \mathbf{Y}_1\mathbf{Y}_2\|_F \quad \text{by (5.1)} \\
&= E^{\boldsymbol{\beta}_s}(\mathbf{X}^{(J)}_{\text{left}}\mathbf{X}_{[\![q,t]\!]}\mathbf{X}^{(J)}_{\text{right}}) \quad \text{by Lemma E.1 below} \\
&= E^{\boldsymbol{\beta}_s}(\mathbf{B}) \quad \text{by definition of } \mathbf{B}.
\end{aligned}$$

The penultimate line used the following result, which we prove next.

LEMMA E.1. *Consider a non-redundant chainable architecture $\boldsymbol{\beta} := (\boldsymbol{\pi}_\ell)_{\ell=1}^L$, integers (q, s, t) such that $1 \leq q \leq s < t \leq L$, and denote*

$$\boldsymbol{\pi}'_1 := \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{q-1}, \quad \boldsymbol{\pi}'_2 := \boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t, \quad \boldsymbol{\pi}'_3 := \boldsymbol{\pi}_{t+1} * \ldots * \boldsymbol{\pi}_L,$$
$$\tilde{\boldsymbol{\pi}}_1 := \boldsymbol{\pi}_q * \ldots \boldsymbol{\pi}_s, \quad \tilde{\boldsymbol{\pi}}_2 := \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_t.$$

Assume that $\mathbf{X} \in \Sigma^{\boldsymbol{\pi}'_1}$ is left-$r(\boldsymbol{\pi}_{q-1}, \boldsymbol{\pi}_q)$-unitary (if $q > 1$) or the identity matrix of size $a_1 b_1 d_1$ (if $q = 1$), and that $\mathbf{Z} \in \Sigma^{\boldsymbol{\pi}'_3}$ is right-$r(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1})$-unitary (if $t < L$) or the identity matrix of size $a_L c_L d_L$ (if $t = L$). Then for any $\mathbf{Y} \in \Sigma^{\boldsymbol{\pi}'_2}$ we have :

$$E^{\boldsymbol{\beta}_s}(\mathbf{XYZ}) = \inf_{\mathbf{Y}_i \in \Sigma^{\tilde{\boldsymbol{\pi}}_i}, i=1,2} \|\mathbf{Y} - \mathbf{Y}_1\mathbf{Y}_2\|_F. \tag{E.1}$$

Before proving the lemma observe that for each pair of factors $\mathbf{Y}_i \in \Sigma^{\tilde{\boldsymbol{\pi}}_i}$, $i = 1, 2$, by Lemma 4.19 we have $(\mathbf{XY}_1, \mathbf{Y}_2\mathbf{Z}) \in \Sigma^{\boldsymbol{\beta}_s}$ (cf. Definition 7.1), hence the inequality

$$\begin{aligned}
E^{\boldsymbol{\beta}_s}(\mathbf{XYZ}) \leq \|\mathbf{XYZ} - \mathbf{XY}_1\mathbf{Y}_2\mathbf{Z}\|_F^2 &= \|\mathbf{X}\underbrace{(\mathbf{Y} - \mathbf{Y}_1\mathbf{Y}_2)}_{\in \Sigma^{\boldsymbol{\pi}'_2}}\mathbf{Z}\|_F^2 \\
&= \|\mathbf{Y} - \mathbf{Y}_1\mathbf{Y}_2\|_F^2 \quad \text{(by Lemma C.1)}.
\end{aligned}$$

Rather than proving the converse inequality, we proceed by characterizing both hand sides of (E.1) via spectral properties of appropriate blocks of $\mathbf{XYZ}$ (resp. of $\mathbf{Y}$).

Proof of Lemma E.1. We denote $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}_2})$ and $\mathcal{P}' = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3})$ the partitions of $[\![p]\!]$ (where p is the number of columns of matrices in $\Sigma^{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}$ and in $\Sigma^{\tilde{\boldsymbol{\pi}}_1}$, as well as the number of rows of matrices in $\Sigma^{\tilde{\boldsymbol{\pi}}_2}$ and in $\Sigma^{\tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3}$). By Lemma 4.8, the sets R_P, C_P for $P \in \mathcal{P}$ satisfy

$$R_P = \mathtt{supp}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}[:, i]), \quad C_P = \mathtt{supp}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_2}[i, :]), \quad \forall i \in P. \tag{E.2}$$

To avoid confusions, for $P \in \mathcal{P}'$ we use the distinct notation/property

$$R'_P = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}[:, i]), \quad C'_P = \mathtt{supp}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3}[i, :]), \quad \forall i \in P. \tag{E.3}$$

With $r := r(\boldsymbol{\pi}_s, \boldsymbol{\pi}_{s+1})$, by Lemma 4.14 we have $r(\tilde{\boldsymbol{\pi}}_1, \tilde{\boldsymbol{\pi}}_2) = r(\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1, \tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3) = r$.

With these notations, and denoting $\texttt{rankproj}_r(\cdot)$ a best rank-r approximation (in the Frobenius norm) to a matrix, we first characterize the left-hand side (LHS) and the right-hand side (RHS) of (E.1) separately. Exploiting the (left/right)-*-unitarity of $\mathbf{X}$ and $\mathbf{Z}$, by Lemma C.1 we have $\|\mathbf{XYZ}\|_F^2 = \|\mathbf{Y}\|_F^2$. Moreover, by Lemma 5.1, all constraint support patterns satisfy the assumptions of Theorem 3.4. Decomposing $\mathbf{XYZ}$ (resp. $\mathbf{Y}$) into the corresponding blocks $R_P \times C_P$, using Theorem 3.4, and summing the resulting equalities, we obtain

$$\texttt{LHS} = E^{\boldsymbol{\beta}_s}(\mathbf{XYZ})^2 = \underbrace{\|\mathbf{XYZ}\|_F^2}_{=\|\mathbf{Y}\|_F^2} - \sum_{P \in \mathcal{P}'} \|\texttt{rank}_r\big((\mathbf{XYZ})[R'_P, C'_P]\big)\|_F^2 \tag{E.4}$$

$$\texttt{RHS} = \|\mathbf{Y}\|_F^2 - \sum_{P \in \mathcal{P}} \|\texttt{rank}_r\big(\mathbf{Y}[R_P, C_P]\big)\|_F^2. \tag{E.5}$$

To conclude, we prove that $\mathcal{P} = \mathcal{P}' = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_s, r)$, and that

$$\|\texttt{rank}_r\big((\mathbf{XYZ})[R'_P, C'_P]\big)\|_F^2 = \|\texttt{rank}_r\big(\mathbf{Y}[R_P, C_P]\big)\|_F^2, \quad \forall P \in \mathcal{P}. \tag{E.6}$$

Proof that $\mathcal{P} = \mathcal{P}'$. Since $r(\tilde{\boldsymbol{\pi}}_1, \tilde{\boldsymbol{\pi}}_2) = r(\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1, \tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3) = r$ (this implies that the assumption $r \mid c$ holds where c is such that $\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1 = (a, b, c, d)$ – respectively such that $\tilde{\boldsymbol{\pi}}_1 = (a, b, c, d)$ – for some a, b, d), by Lemma C.4 and Lemma C.7 we have

$$\begin{aligned}
\mathcal{P}' &= \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}_2 * \boldsymbol{\pi}'_3}) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_s, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_s, r) \\
\mathcal{P} &= \mathcal{P}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}_2}) = \mathcal{P}_{\texttt{col}}(\tilde{\boldsymbol{\pi}}_1, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_s, r) = \mathcal{P}_{\texttt{col}}(\boldsymbol{\pi}_s, r).
\end{aligned}$$

Proof of (E.6)**.** An important step is to show that for each $P \in \mathcal{P}$, the matrix $(\mathbf{XYZ})[R'_P, C'_P]$ is, up to some permutation of rows and columns and addition of zero rows and columns, equal to $\mathbf{X}[:, R_P]\mathbf{Y}[R_P, C_P]\mathbf{Z}[C_P, :]$. For this, we prove that

$$\mathbf{XYZ} = \sum_{P \in \mathcal{P}} \mathbf{X}[:, R_P]\mathbf{Y}[R_P, C_P]\mathbf{Z}[C_P, :] \tag{E.7}$$

$$\texttt{supp}(\mathbf{X}[:, R_P]\mathbf{Y}[R_P, C_P]\mathbf{Z}[C_P, :]) \subseteq R'_P \times C'_P. \tag{E.8}$$

Indeed, since $\boldsymbol{\pi}'_2 = \tilde{\boldsymbol{\pi}}_1 * \tilde{\boldsymbol{\pi}}_2$, by Lemma 4.8, we have $\texttt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_2}) = \bigcup_{P \in \mathcal{P}} R_P \times C_P$ where we recall that $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}_2})$, and the sets $R_P \times C_P$ are pairwise disjoint. Since $\mathbf{Y} \in \Sigma^{\boldsymbol{\pi}'_2}$ it follows that $\texttt{supp}(\mathbf{Y}) \subset \bigcup_{P \in \mathcal{P}} R_P \times C_P$, hence the decomposition (E.7). Moreover, by Lemma 4.8, the integers $k := |R_P|$ and $\ell := |C_P|$ are independent of $P \in \mathcal{P}$, and since $\mathbf{X} \in \Sigma^{\boldsymbol{\pi}'_1}$ and $\mathbf{Z} \in \Sigma^{\boldsymbol{\pi}'_3}$, for each $P \in \mathcal{P}$ we have

$$\begin{aligned}
\texttt{supp}(\mathbf{X}[:, R_P]\,\mathbf{Y}[R_P, C_P]\,\mathbf{Z}[C_P, :]) &\subseteq \texttt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_1}[:, R_P]\,\mathbf{1}_{k \times \ell}\,\mathbf{S}_{\boldsymbol{\pi}'_3}[C_P, :]) \\
&= \texttt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_1}[:, R_P]\,\mathbf{1}_{k \times 1}\,\mathbf{1}_{1 \times \ell}\,\mathbf{S}_{\boldsymbol{\pi}'_3}[C_P, :]).
\end{aligned}$$

By (E.2)-(E.3), for any $i \in P$, $\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}[:, i] = \mathbf{1}_{R_P}$ (the indicator vector of the set R_P), and $\texttt{supp}(\mathbf{S}_{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}) = \mathbf{1}_{R'_P}$, hence by elementary linear algebra and Proposition 4.7

$$\mathbf{S}_{\boldsymbol{\pi}'_1}[:, R_P]\mathbf{1}_{k \times 1} = \mathbf{S}_{\boldsymbol{\pi}'_1}\mathbf{1}_{R_P} = \mathbf{S}_{\boldsymbol{\pi}'_1}\mathbf{S}_{\tilde{\boldsymbol{\pi}}_1}[:, i] \propto \mathbf{S}_{\boldsymbol{\pi}'_1 * \tilde{\boldsymbol{\pi}}_1}[:, i] = \mathbf{1}_{R'_P}.$$

Similarly $\mathbf{1}_{1 \times \ell}\,\mathbf{S}_{\boldsymbol{\pi}'_3}[C_P, :] \propto \mathbf{1}_{C'_P}^\top$. This establishes (E.8): as a consequence the supports of the summands in the right-hand side of (E.7) are pairwise disjoint.

Thus, for any $P \in \mathcal{P}$, the product $\mathbf{X}[:, R_P]\mathbf{Y}[R_P, C_P]\mathbf{Z}[C_P, :]$ is, up to some permutation of rows and columns and deletion of zero rows and columns, equal to

$(\mathbf{XYZ})[R'_P, C'_P]$, as claimed. This implies that their best approximation of rank r has the same Frobenius norm. Therefore, to establish (E.6), we only need to prove

(E.9)
$$\|\mathtt{rankproj}_r\big(\mathbf{X}[:, R_P]\mathbf{Y}[R_P, C_P]\mathbf{Z}[C_P, :]\big)\|_F^2 = \|\mathtt{rankproj}_r\big(\mathbf{Y}[R_P, C_P]\big)\|_F^2, \ \forall P \in \mathcal{P}.$$

This is again a consequence of the left/right-r-unitarity of $\mathbf{X}/\mathbf{Z}$: by Corollary C.12, the columns (resp. rows) of $\mathbf{X}[:, R_P]$ (resp. $\mathbf{Z}[C_P, :]$) are orthonormal. ∎

E.2. Proof of Lemma 7.11 We consider a chainable $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ and $1 \leq s, q \leq L-1$. Our goal is to show that $E^{\boldsymbol{\beta}_s}(\mathbf{M}) \geq E^{\boldsymbol{\beta}_s}(\mathbf{N})$ where $\mathbf{N}$ is a projection of $\mathbf{M}$ onto $\mathcal{B}^{\boldsymbol{\beta}_q}$. When $s = q$ the result is trivial as $E^{\boldsymbol{\beta}_s}(\mathbf{M}) \geq 0 = E^{\boldsymbol{\beta}_q}(\mathbf{N}) = E^{\boldsymbol{\beta}_s}(\mathbf{N})$, so we focus on the case $s \neq q$. We give the detailed proof when $s < q$ and outline its adaptation when $s > q$.

Assume that $s < q$ and denote $(\boldsymbol{\pi}, \boldsymbol{\pi}')$ the chainable patterns such that $\boldsymbol{\beta}_s = (\boldsymbol{\pi}, \boldsymbol{\pi}')$ and $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$ (cf. Definition 3.3). By Lemma 4.8, all classes $P \in \mathcal{P}$ have the same cardinality, denoted r. By Theorem 3.4, recalling that $\varepsilon_r^2(\cdot)$ denotes the squared error of best rank-r approximation in the Frobenius norm, we have $E^{\boldsymbol{\beta}_s}(\mathbf{M}) = c_{\mathbf{M}} + \sum_{P\in\mathcal{P}} \varepsilon_r^2(\mathbf{M}[R_P, C_P])$ where $c_{\mathbf{M}} \geq 0$, and similarly for $E^{\boldsymbol{\beta}_s}(\mathbf{N})$ with $c_{\mathbf{N}} = 0$ since $\mathtt{supp}(\mathbf{N}) \subseteq \mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L}$ (by Proposition 4.7, since $\mathbf{N} \in \mathcal{B}^{\boldsymbol{\beta}_q}$). Considering an arbitrary $P \in \mathcal{P}$, we will prove that

$$\varepsilon_r^2(\mathbf{M}[R_P, C_P]) \geq \varepsilon_r^2(\mathbf{N}[R_P, C_P]). \tag{E.10}$$

This will yield the conclusion. To establish (E.10) we first observe that, as a simple consequence of Eckart–Young–Mirsky theorem on low rank matrix approximation [10], for $\mathbf{U} \in \mathbb{R}^{m\times n}$ and $r \leq \min(m, n)$ we have

$$\varepsilon_r^2(\mathbf{U}) = \mathrm{Tr}(\mathbf{U}\mathbf{U}^\top) - \sum_{i=1}^r \lambda_i(\mathbf{U}\mathbf{U}^\top) = \mathrm{Tr}(\mathbf{U}^\top\mathbf{U}) - \sum_{i=1}^r \lambda_i(\mathbf{U}^\top\mathbf{U}), \tag{E.11}$$

with $\lambda_i(\cdot)$ the i-th largest eigenvalue of a symmetric matrix. We also will use the following lemma (the proof of all intermediate lemmas is slightly postponed).

LEMMA E.2. *If $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$ are symmetric positive semi-definite (PSD) then*

$$\mathrm{Tr}(\mathbf{A}) - \sum_{i=1}^q \lambda_i(\mathbf{A}) \leq \mathrm{Tr}(\mathbf{A}+\mathbf{B}) - \sum_{i=1}^q \lambda_i(\mathbf{A}+\mathbf{B}), \quad \forall 1 \leq q \leq n. \tag{E.12}$$

Denoting $\mathbf{K} := \mathbf{M} - \mathbf{N}$ and

$$\mathbf{U} := \mathbf{M}[R_P, C_P], \quad \mathbf{V} := \mathbf{N}[R_P, C_P], \quad \mathbf{W} := \mathbf{U} - \mathbf{V} = \mathbf{K}[R_P, C_P],$$

we will soon show that

$$\mathbf{V}\mathbf{W}^\top = \mathbf{0}_{|R_P|\times|R_P|}. \tag{E.13}$$

Since $\mathbf{U} = \mathbf{V} + \mathbf{W}$, this implies $\mathbf{U}\mathbf{U}^\top = \mathbf{V}\mathbf{V}^\top + \mathbf{V}\mathbf{W}^\top + \mathbf{W}\mathbf{V}^\top + \mathbf{W}\mathbf{W}^\top = \mathbf{V}\mathbf{V}^\top + \mathbf{W}\mathbf{W}^\top$, hence $\mathbf{B} := \mathbf{U}\mathbf{U}^\top - \mathbf{V}\mathbf{V}^\top = \mathbf{W}\mathbf{W}^\top \succeq 0$. With $\mathbf{A} := \mathbf{V}\mathbf{V}^\top$ we have $\mathbf{A} + \mathbf{B} = \mathbf{U}\mathbf{U}^\top$ and the following derivation then yields (E.10) as claimed:

$$\begin{aligned}\varepsilon_r^2(\mathbf{M}[R_P, C_P]) &\overset{\text{(E.11)}}{=} \mathrm{Tr}(\mathbf{U}\mathbf{U}^\top) - \sum_{i=1}^r \lambda_i(\mathbf{U}\mathbf{U}^\top) = \mathrm{Tr}(\mathbf{A}+\mathbf{B}) - \sum_{i=1}^r \lambda_i(\mathbf{A}+\mathbf{B}) \\ &\overset{\text{(E.12)}}{\geq} \mathrm{Tr}(\mathbf{B}) - \sum_{i=1}^r \lambda_i(\mathbf{B}) \overset{\text{(E.11)}}{=} \varepsilon_r^2(\mathbf{N}[R_P, C_P]).\end{aligned}$$

To prove (E.13) we use the following lemma.

LEMMA E.3. *Let $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ be a chainable architecture, and $1 \leq i < j \leq L-1$. For $\ell \in \{i, j\}$ denote $\mathcal{P}_\ell := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_\ell}, \mathbf{S}_{\boldsymbol{\pi}_{\ell+1} * \ldots * \boldsymbol{\pi}_L})$. For each $P \in \mathcal{P}_i$, $Q \in \mathcal{P}_j$*

1. *If $R_P \cap R_Q \neq \emptyset$ then $R_P \subseteq R_Q$*
2. *If $C_P \cap C_Q \neq \emptyset$ then $C_Q \subseteq C_P$ (reverse inclusion compared to R_P and R_Q).*

For each $P \in \mathcal{P}_i$ denote $\mathcal{P}_j(P) := \{Q \in \mathcal{P}_j : R_P \cap R_Q \neq \emptyset \text{ and } C_P \cap C_Q \neq \emptyset\}$,. Similarly, for $Q \in \mathcal{P}_j$, $\mathcal{P}_i(Q) := \{P \in \mathcal{P}_i : R_P \cap R_Q \neq \emptyset \text{ and } C_P \cap C_Q \neq \emptyset\}$. We have

3. *C_P is the disjoint union of C_Q, $Q \in \mathcal{P}_j(P)$, and $R_P \subseteq R_Q$ for each $Q \in \mathcal{P}_j(P)$.*
4. *R_Q is the disjoint union of R_P, $P \in \mathcal{P}_i(Q)$, and $C_Q \subseteq C_P$ for each $P \in \mathcal{P}_i(Q)$.*

To prove (E.13) for an arbitrary $P \in \mathcal{P}$, first recall that $\boldsymbol{\beta}_s = (\boldsymbol{\pi}, \boldsymbol{\pi}')$ with $\boldsymbol{\pi} = \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_s$, $\boldsymbol{\pi}' = \boldsymbol{\pi}_{s+1} * \ldots * \boldsymbol{\pi}_L$ and that $\mathcal{P} = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}}, \mathbf{S}_{\boldsymbol{\pi}'})$. Similarly $\boldsymbol{\beta}_q = (\tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\pi}}')$ with $\tilde{\boldsymbol{\pi}} = \boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_q$, $\tilde{\boldsymbol{\pi}}' = \boldsymbol{\pi}_{q+1} * \ldots * \boldsymbol{\pi}_L$, and we introduce $\mathcal{P}' := \mathcal{P}(\mathbf{S}_{\tilde{\boldsymbol{\pi}}}, \mathbf{S}_{\tilde{\boldsymbol{\pi}}'})$. With the notation of Lemma E.3, we have $\mathcal{P} = \mathcal{P}_i$ and $\mathcal{P}' = \mathcal{P}_j$ with $i = s < q = j$, and we denote $\mathcal{P}'(P) := \mathcal{P}_j(P)$. By Lemma E.3:

- C_P is the disjoint union of C_Q, $Q \in \mathcal{P}'(P)$;
- $R_P \subseteq R_Q$ for each $Q \in \mathcal{P}'(P)$.

As a result of the first fact, $\mathbf{V} := \mathbf{N}[R_P, C_P]$ is (up to column permutation) the horizontal concatenation of blocks $\mathbf{N}[R_P, C_Q]$, $Q \in \mathcal{P}'(P)$, and similarly for $\mathbf{W} := \mathbf{K}[R_P, C_P]$. Establishing (E.13) is thus equivalent to proving that

$$\mathbf{N}[R_P, C_Q]\mathbf{K}[R_P, C_Q]^\top = \mathbf{K}[R_P, C_Q]\mathbf{N}[R_P, C_Q]^\top = \mathbf{0}, \quad \forall Q \in \mathcal{P}'(P) \tag{E.14}$$

since

$$\mathbf{V}\mathbf{W}^\top = \sum_{Q \in \mathcal{P}'(P)} \mathbf{N}[R_P, C_Q]\mathbf{K}[R_P, C_Q]^\top.$$

To prove (E.14), consider $Q \in \mathcal{P}'(P)$: since $R_P \subseteq R_Q$, we have

$$\begin{aligned} \operatorname{rowspan}(\mathbf{K}[R_P, C_Q]) &\subseteq \operatorname{rowspan}(\mathbf{K}[R_Q, C_Q]), \\ \operatorname{rowspan}(\mathbf{N}[R_P, C_Q]) &\subseteq \operatorname{rowspan}(\mathbf{N}[R_Q, C_Q]) \end{aligned}$$

with $\operatorname{rowspan}(\cdot)$ the span of the rows of a matrix. We use the following classical lemma.

LEMMA E.4. *Consider a non-redundant and chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$. Denote $\mathcal{P}_j = \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_j}, \mathbf{S}_{\boldsymbol{\pi}_{j+1} * \ldots * \boldsymbol{\pi}_L})$, where $j \in [\![L-1]\!]$. If $\mathbf{B}$ is a projection of a matrix $\mathbf{A}$ onto $\boldsymbol{\beta}_j$, then we have:*

$$\begin{aligned} \operatorname{colspan}(\mathbf{A}[R_Q, C_Q] - \mathbf{B}[R_Q, C_Q]) \perp \operatorname{colspan}(\mathbf{B}[R_Q, C_Q]), \qquad \forall Q \in \mathcal{P}_j, \\ \operatorname{rowspan}(\mathbf{A}[R_Q, C_Q] - \mathbf{B}[R_Q, C_Q]) \perp \operatorname{rowspan}(\mathbf{B}[R_Q, C_Q]), \qquad \forall Q \in \mathcal{P}_j. \end{aligned}$$

By Lemma E.4 with $j = q$ we further have:

$$\operatorname{rowspan}(\mathbf{K}[R_Q, C_Q]) = \operatorname{rowspan}(\mathbf{M}[R_Q, C_Q] - \mathbf{N}[R_Q, C_Q]) \perp \operatorname{rowspan}(\mathbf{N}[R_Q, C_Q])$$

hence $\operatorname{rowspan}(\mathbf{K}[R_P, C_Q]) \perp \operatorname{rowspan}(\mathbf{N}[R_P, C_Q])$ and (E.14) holds as claimed.

We proceed similarly when $s > q$: we prove an analog of (E.10) for each $Q \in \mathcal{P}'$ instead of each $P \in \mathcal{P}$. For this we establish a variant of (E.13), where $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are the $R_Q \times C_Q$ blocks instead of $R_P \times C_P$:

$$\mathbf{V}^\top \mathbf{W} = \mathbf{0}_{|C_q| \times |C_q|}, \tag{E.15}$$

so that $\mathbf{U}^\top \mathbf{U} = \mathbf{A} + \mathbf{B}$ where $\mathbf{A} := \mathbf{V}^\top \mathbf{V}$ and $\mathbf{B} = \mathbf{W}^\top \mathbf{W} \succeq 0$. Since $s > q$, by Lemma E.3 again, $\mathbf{V} = \mathbf{N}[R_P, C_P]$ and $\mathbf{W} = \mathbf{K}[R_P, C_P]$ are *vertical* concatenations

of $\mathbf{N}[R_Q, C_P]$ and $\mathbf{K}[R_Q, C_P]$, $P \in \mathcal{P}'(Q)$, respectively, (E.15) can be deduced from an analog to (E.14):

$$\text{(E.16)}\quad \mathbf{N}[R_Q, C_P]^\top \mathbf{K}[R_Q, C_P] = \mathbf{K}[R_Q, C_P]^\top \mathbf{N}[R_Q, C_P] = \mathbf{0}, \forall P \in \mathcal{P}(Q) := \mathcal{P}_i(Q),$$

(with the notations of Lemma E.3), which is a direct consequence of

$$\begin{aligned}
&\operatorname{colspan}(\mathbf{K}[R_Q, C_P]) \subseteq \operatorname{colspan}(\mathbf{K}[R_Q, C_Q]),\\
&\operatorname{colspan}(\mathbf{N}[R_Q, C_P]) \subseteq \operatorname{colspan}(\mathbf{N}[R_Q, C_Q])\\
&\operatorname{colspan}(\mathbf{K}[R_Q, C_P]) \perp \operatorname{colspan}(\mathbf{N}[R_Q, C_P]).
\end{aligned}$$

where $\operatorname{colspan}(\cdot)$ is the linear span of the columns of a matrix.

To conclude, we now prove Lemma E.2, Lemma E.3, and Lemma E.4.

Proof of Lemma E.2. The set $\mathbb{S}^n$ of symmetric $n \times n$ matrices is convex, and the function $f : \mathbb{S}^n \mapsto \mathbb{R} : \mathbf{A} \mapsto \sum_{i=1}^q \lambda_i(\mathbf{A})$ is convex [1, Problem 3.26]. In addition, f is positively homogeneous, i.e., $f(t\mathbf{A}) = tf(\mathbf{A}), \forall t \geq 0$. Therefore, for any $\mathbf{A}, \mathbf{B} \in \mathbb{S}^n$, we have:

$$f(\mathbf{A}) + f(\mathbf{B}) = 2 \cdot \frac{1}{2}\left(f(\mathbf{A}) + f(\mathbf{B})\right) \geq 2f\left(\frac{\mathbf{A}+\mathbf{B}}{2}\right) = f(\mathbf{A}+\mathbf{B})$$

so that if in addition $\mathbf{B} \in \mathbb{S}^n$ is positive semi-definite (PSD) we get

$$\begin{aligned}
&\Big[\operatorname{Tr}(\mathbf{A}+\mathbf{B}) - \sum_{i=1}^q \lambda_i(\mathbf{A}+\mathbf{B})\Big] - \Big[\operatorname{Tr}(\mathbf{A}) - \sum_{i=1}^q \lambda_i(\mathbf{A})\Big]\\
&\qquad = [\operatorname{Tr}(\mathbf{A}+\mathbf{B}) - f(\mathbf{A}+\mathbf{B})] - [\operatorname{Tr}(\mathbf{A}) - f(\mathbf{A})]\\
&\qquad = \operatorname{Tr}(\mathbf{B}) - [f(\mathbf{A}+\mathbf{B}) - f(\mathbf{A})]\\
&\qquad \geq \operatorname{Tr}(\mathbf{B}) - f(\mathbf{B}) \geq 0.
\end{aligned}$$

The fact that $\mathbf{B}$ is PSD was only used in the last inequality. ∎

Proof of Lemma E.3. **Preliminaries.** Denote $\boldsymbol{\pi}_\ell = (a_\ell, b_\ell, c_\ell, d_\ell)$ for $\ell \in [\![L]\!]$. By Definition 7.1 and the definition (4.3) of the operator $*$, we have

$$\boldsymbol{\beta}_i = \Big(\underbrace{\big(a_1, \tfrac{b_1 d_1}{d_i}, \tfrac{a_i c_i}{a_1}, d_i\big)}_{=:\boldsymbol{\pi}=(a,b,c,d)}, \underbrace{\big(a_{i+1}, \tfrac{b_{i+1} d_{i+1}}{d_L}, \tfrac{a_L c_L}{a_{i+1}}, d_L\big)}_{=:\boldsymbol{\pi}'=(a',b',c',d')}\Big).$$

Consider an arbitrary column $\mathbf{s}$ of $\mathbf{S}_{\boldsymbol{\pi}}$. By the structure of $\mathbf{S}_{\boldsymbol{\pi}}$ (cf. Figure 1) there exists a block index $1 \leq k \leq a = a_1$ and an index $1 \leq \ell \leq d = d_i$ such that $\mathbf{s}$ is equal to the ℓ-th column of the k-th "group" of cd columns of $\mathbf{S}_{\boldsymbol{\pi}}$, so that

$$\begin{aligned}
\texttt{supp}(\mathbf{s}) &= [\![(k-1)bd+1, kbd]\!] \cap \{t \in \mathbb{Z} : t \equiv \ell \mod d\}\\
&= \underbrace{[\![(k-1)b_1 d_1 + 1, kb_1 d_1]\!]}_{=:T_k} \cap \{t \in \mathbb{Z} : t \equiv \ell \mod d_i\} =: R^i_{k,\ell}.
\end{aligned}$$

Similarly, for each row $\mathbf{s}'$ of $\mathbf{S}_{\boldsymbol{\pi}'}$, we have

$$\begin{aligned}
\texttt{supp}(\mathbf{s}') &= [\![(k-1)c'd' + 1, kc'd']\!] \cap \{t \in \mathbb{Z} : t \equiv \ell \mod d'\}\\
&= [\![(k-1)\tfrac{a_L c_L d_L}{a_{i+1}} + 1, k\tfrac{a_L c_L d_L}{a_{i+1}}]\!] \cap \underbrace{\{t \in \mathbb{Z} : t \equiv \ell \mod d_L\}}_{=:T'_\ell} =: C^i_{k,\ell}
\end{aligned}$$

for some $1 \leq k \leq a' = a_{i+1}$, $1 \leq \ell \leq d' = d_L$. The same holds with j instead of i.

Claims 1 and 2. Consider now $P \in \mathcal{P}_i$ and $Q \in \mathcal{P}_j$. Assume that $R_P \cap R_Q \neq \emptyset$. By the definition of $\mathcal{P}_i$ and $\mathcal{P}_j$ (Definition 3.3) and the above considerations, there exists indices k, k', ℓ, ℓ' such that $R_P = R^i_{k,\ell}$ and $R_Q = R^j_{k',\ell'}$, hence $R_P \cap R_Q \subset T_k \cap T_{k'}$, and therefore $k = k'$. As a result, R_P (resp. R_Q) is exactly the subset of all integers in T_k satisfying a certain congruence modulo d_i (resp. modulo d_j). Since $i < j$, by chainability, we have $d_j \mid d_i$, hence there are only two possibilities for $\{t \in \mathbb{Z}, t \equiv \ell \mod d_i\}$ and $\{t \in \mathbb{Z}, t \equiv \ell' \mod d_j\}$: they are either disjoint or the former is a subset of the latter. Since $R_P \cap R_Q \neq \emptyset$ the case of an empty intersection is excluded, and we obtain $R_P \subseteq R_Q$ as claimed.

A similar reasoning shows that if $C_P \cap C_Q \neq \emptyset$ then $C_P = C^i_{k,\ell}$ and $C_Q = C^j_{k',\ell'}$ with $\ell = \ell'$. Denoting $p = a_L c_L d_L / a_{i+1}$ and $q = a_L c_L d_L / a_{j+1}$, we have $p/q = a_{j+1}/a_{i+1}$. By chainability, since $i < j$, we have $a_{i+1} \mid a_{j+1}$ hence $q \mid p$. It follows that C_P (resp. C_Q) is exactly the subset of all integers in T'_ℓ belonging to $[\![(k-1)p+1, kp]\!]$ (resp. $[\![(k'-1)q+1, k'q]\!]$). Since C_P intersects C_Q, these two intervals must intersect, and it is easy to check that since $q \mid p$ the second must then be a subset of the first. We obtain $C_Q \subseteq C_P$ as claimed.

Claims 3 and 4. We prove Claim 3, the proof of Claim 4 is analogous. Consider $P \in \mathcal{P}_i$. The fact that $R_P \subseteq R_Q$ for each $Q \in \mathcal{P}_j(P)$ is a direct consequence of Claim 1 and the definition of $\mathcal{P}_j(P)$. Let us now show that the sets C_Q, $Q \in \mathcal{P}_j(P)$ are pairwise disjoint. For this consider $Q, Q' \in \mathcal{P}_j(P)$ such that $C_Q \cap C_{Q'} \neq \emptyset$. By the first claim of Lemma E.3 and the definition of $\mathcal{P}_j(P)$, we have $R_P \subseteq R_Q$ and $R_P \subseteq R_{Q'}$, hence

$$(R_Q \times C_Q) \cap (R_{Q'} \times C_{Q'}) \supseteq R_P \times (C_Q \cap C_{Q'}) \neq \emptyset.$$

By Lemma 4.14 the pair $(\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1} * \ldots, \boldsymbol{\pi}_L)$ is chainable, hence by Lemma 5.1 it satisfies the assumptions of Theorem 3.4, i.e., the sets $R_Q \times C_Q$, $Q \in \mathcal{P}_j$ are pairwise disjoint. This shows that $Q = Q'$.

Finally, since $\cup_{P \in \mathcal{P}_i} R_P \times C_P = \cup_{Q \in \mathcal{P}_j} R_Q \times C_Q = \mathtt{supp}(\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L})$ (see, e.g., Lemma 4.8), we have $\cup_{P \in \mathcal{P}_i} R_P = \cup_{Q \in \mathcal{P}_j} R_Q = [\![a_1 b_1 d_1]\!]$ and $\cup_{P \in \mathcal{P}_i} C_P = \cup_{Q \in \mathcal{P}_j} C_Q = [\![a_L b_L d_L]\!]$. Combined with the proved inclusions $R_P \subseteq R_Q$ and $C_Q \subseteq C_P$ (under non-empty intersection conditions), this implies that R_Q (resp. C_P) is the (disjoint) union of all R_P (resp. C_Q) that intersect it. ∎

Proof of Lemma E.4. By (A.3), $\mathbf{B}$ is a projection of $\mathbf{A}$ onto $\mathcal{B}^{\boldsymbol{\beta}_j}$ if and only if $\mathtt{supp}(\mathbf{B}) \subseteq \mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L}$ and its subblocks satisfy

$$\mathbf{B}[R_P, C_P] \in \mathtt{rankproj}_{|P|}(\mathbf{A}[R_P, C_P])$$

where we recall that $\mathtt{rankproj}_r(\mathbf{X}) := \arg\min_{\mathbf{M}:\mathtt{rank}(\mathbf{M}) \leq r} \|\mathbf{X} - \mathbf{M}\|_F$. It is thus sufficient to prove the following claim for the low-rank matrix approximation problem: Given a matrix $\mathbf{D} \in \mathbb{R}^{m \times n}$, if a matrix $\mathbf{C}$ is a projection of $\mathbf{D}$ onto the set $\mathcal{M}_r$ of matrices of rank at most r, then:

$$\mathrm{colspan}(\mathbf{D} - \mathbf{C}) \perp \mathrm{colspan}(\mathbf{C}) \quad \text{and} \quad \mathrm{rowspan}(\mathbf{D} - \mathbf{C}) \perp \mathrm{rowspan}(\mathbf{C}).$$

This result is classical. We re-prove it here for self-completeness. One can re-write the projection onto $\mathcal{M}_r$ as the following optimization problem:

$$\underset{\mathbf{X} \in \mathbb{R}^{m \times r}, \mathbf{Y} \in \mathbb{R}^{r \times n}}{\text{Minimize}} \; f(\mathbf{X}, \mathbf{Y}) \quad \text{where} \quad f(\mathbf{X}, \mathbf{Y}) := \tfrac{1}{2} \|\mathbf{X}\mathbf{Y} - \mathbf{D}\|_F^2.$$

At an optimum $\mathbf{C} = \mathbf{X}\mathbf{Y}$ (which always exists, take the truncated SVD for example):

$$\frac{\partial f}{\partial \mathbf{X}} = (\mathbf{X}\mathbf{Y} - \mathbf{D})\mathbf{Y}^\top = \mathbf{0}, \qquad \frac{\partial f}{\partial \mathbf{Y}} = \mathbf{X}^\top(\mathbf{X}\mathbf{Y} - \mathbf{D}) = \mathbf{0}.$$

or equivalently: $\mathrm{rowspan}(\mathbf{C}-\mathbf{D}) \perp \mathrm{rowspan}(\mathbf{Y})$ and $\mathrm{colspan}(\mathbf{C}-\mathbf{D}) \perp \mathrm{colspan}(\mathbf{X})$. Since $\mathrm{colspan}(\mathbf{C}) \subseteq \mathrm{colspan}(\mathbf{X}), \mathrm{rowspan}(\mathbf{C}) \subseteq \mathrm{rowspan}(\mathbf{Y})$, this yields the claim. ∎

E.3. Proof for (7.14)

Proof. The proof is given when σ is the identity permutation $\sigma = (1, \dots, L-1)$. The proof when σ is the "converse" permutation $\sigma = (L-1, \dots, 1)$ is similar, replacing rows by columns, left-*-unitarity by right-*-unitarity, etc. We begin by preliminaries on key matrices involved in the expression of the matrices $\mathbf{B}_J$, $\mathbf{B}_{J-1}$ and $\mathbf{B}_p$ appearing in (7.14). We then highlight simple orthogonality conditions which imply (7.14). Finally we prove these orthogonality conditions.

Preliminaries. Since $\sigma_\ell = \ell$, $\ell \in [\![L-1]\!]$, it is not difficult to check by induction on $J \in [\![L-1]\!]$ that at the Jth iteration of Algorithm 6.1, at line 18, we have

1. `partition` $= (I_1, \dots, I_J)$ where $I_\ell = \{\ell\}$, $\ell \in [\![J-1]\!]$ and $I_J = [\![J, L]\!]$;
2. $s = J$ (from line 9).
3. $j = J$ and $I_j = I_J = [\![J, L]\!]$, i.e., $q = J$, $t = L$ (line 10).

Denoting $(\hat{\mathbf{X}}_\ell)_{\ell=1}^L$ the final output of Algorithm 6.1, one can also check by induction that the list `factors` obtained at the end of the J-th iteration (cf. lines 18 and 20 of Algorithm 6.1) is a tuple of the form:

$$(\hat{\mathbf{X}}_1, \dots, \hat{\mathbf{X}}_{J-1}, \mathbf{X}_{[\![J,J]\!]}, \mathbf{X}_{[\![J+1,L]\!]}).$$

Indeed, the value of the first $J-1$ factors in the list `factors` are left-r_ℓ-unitary factors for appropriate r_ℓ, $\ell \in [\![J-1]\!]$ (cf. Lemma 6.3). Therefore, their values during pseudo-orthonormalization operations in the next iterations $J+1, J+2, \dots, L-1$ do not change anymore, which means that they are equal to $\hat{\mathbf{X}}_1, \dots, \hat{\mathbf{X}}_{J-1}$. The factor $\mathbf{X}_{[\![J,J]\!]} = \mathbf{X}_{[\![q,s]\!]}$ will be pseudo-orthonormalized at the next iteration if $J < L-1$.

Expression of $\mathbf{B}_J$. By the convention of (7.9) and the above observation, we have $\mathbf{X}_{\mathrm{left}}^{(J)} = \mathbf{X}_{[\![1,1]\!]} \dots \mathbf{X}_{[\![J-1,J-1]\!]} = \prod_{\ell=1}^{J-1} \hat{\mathbf{X}}_\ell$ (by convention this is the identity if $J = 1$) and $\mathbf{X}_{\mathrm{right}}^{(J)} = \mathbf{I}_{a_L b_L d_L}$. The matrices $\mathbf{X}_{[\![q,s]\!]}$ $\mathbf{X}_{[\![s+1,t]\!]}$, $\mathbf{X}_{I_j}$ of line 18 of Algorithm 6.1 correspond to $\mathbf{X}_{[\![J,J]\!]}$, $\mathbf{X}_{[\![J+1,L]\!]}$ and $\mathbf{X}_{[\![J,L]\!]}$, hence the matrix $\mathbf{B}_J$ from (7.10) is

$$\mathbf{B}_J := \mathbf{X}_{\mathrm{left}}^{(J)} \mathbf{X}_{[\![q,s]\!]} \mathbf{X}_{[\![s+1,t]\!]} \mathbf{X}_{\mathrm{right}}^{(J)} = \mathbf{X}_{\mathrm{left}}^{(J)} \mathbf{X}_{[\![J,J]\!]} \mathbf{X}_{[\![J+1,L]\!]}, \quad \forall J \in [\![L-1]\!]. \tag{E.17}$$

Given the nature of (7.14) we also express $\mathbf{B}_{J-1}$ and $\mathbf{B}_p$ for $p \in [\![J, L-1]\!]$.

Equation (E.17) also reads $\mathbf{B}_J = \mathbf{X}_{[\![1,1]\!]} \dots \mathbf{X}_{[\![J,J]\!]} \mathbf{X}_{[\![J+1,L]\!]}$, hence for $J \in [\![2, L-1]\!]$

$$\begin{aligned} \mathbf{B}_{J-1} &= \mathbf{X}_{[\![1,1]\!]} \dots \mathbf{X}_{[\![J-1,J-1]\!]} \mathbf{X}_{[\![J,L]\!]} \\ &= \mathbf{X}_{\mathrm{left}}^{(J)} \mathbf{X}_{[\![J,L]\!]}. \end{aligned} \tag{E.18}$$

This indeed holds for all $J \in [\![L-1]\!]$: for $J = 1$, the convention below (7.10) yields $\mathbf{B}_{J-1} = \mathbf{B}_0 := \mathbf{A}$, and $\mathbf{X}_{[\![1,L]\!]} = \mathbf{A}$ (line 5 of Algorithm 6.1). As $\mathbf{X}_{\mathrm{left}}^{(J)}$ is the identity, this shows (E.18) for $J = 1$.

For $p > J$ we have $\mathbf{X}_{\mathrm{left}}^{(p)} = \mathbf{X}_{\mathrm{left}}^{(J)} \hat{\mathbf{X}}_J \dots \hat{\mathbf{X}}_{p-1}$, and one easily deduces from (E.17) that

$$\mathbf{B}_p = \mathbf{X}_{\mathrm{left}}^{(J)} \left(\prod_{\ell=J}^{p-1} \hat{\mathbf{X}}_\ell \right) \mathbf{X}_{[\![p,p]\!]} \mathbf{X}_{[\![p+1,L]\!]}, \quad \forall J, p \in [\![L-1]\!] \text{ such that } p \geq J, \tag{E.19}$$

where again by convention an empty matrix product is the identity.

Expression of $\langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_p \rangle$. From now on we fix $J \in [\![L]\!]$ and $p \in [\![J, L-1]\!]$, and rewrite $\langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_p \rangle$ where we recall that the inner-product is associated to the Frobenius norm on matrices. Denote

$$\begin{aligned} \widetilde{\mathbf{B}}_{J-1} &:= \mathbf{X}_{[\![J,L]\!]}, \\ \widetilde{\mathbf{B}}_J &:= \mathbf{X}_{[\![J,J]\!]} \mathbf{X}_{[\![J+1,L]\!]}, \\ \widetilde{\mathbf{B}}_p &:= \left(\prod_{\ell=J}^{p-1} \hat{\mathbf{X}}_\ell \right) \mathbf{X}_{[\![p,p]\!]} \mathbf{X}_{[\![p+1,L]\!]}. \end{aligned} \tag{E.20}$$

Note that the expression of $\widetilde{\mathbf{B}}_p$ falls back on $\mathbf{B}_J$ when $p = J$. We observe that for $k \in \{J-1, J, p\}$, $\widetilde{\mathbf{B}}_k$ is a $\boldsymbol{\pi}'$-factor with $\boldsymbol{\pi}' = \boldsymbol{\pi}_J * \ldots * \boldsymbol{\pi}_L$ by Proposition 4.7 since $\mathbf{X}_{[\![p,p]\!]} \in \Sigma^{\boldsymbol{\pi}_p}$, $\mathbf{X}_{[\![p+1,L]\!]} \in \Sigma^{\boldsymbol{\pi}_{p+1} * \cdots * \boldsymbol{\pi}_L}$ and (when $p > J$) $\hat{\mathbf{X}}_\ell \in \Sigma^{\boldsymbol{\pi}_\ell}$ for $\ell \in [\![J, p-1]\!]$.

Since the factors $\hat{\mathbf{X}}_\ell$ for $\ell \in [\![J-1]\!]$ are $\boldsymbol{\pi}_\ell$-factors, with chainable patterns $\boldsymbol{\pi}_\ell$, and left-r_ℓ-unitary for appropriate r_ℓ's, by Lemma 4.12 and a recursive use of Lemma C.2 their product $\mathbf{X} := \mathbf{X}_{\text{left}}^{(J)}$ is a left-r-unitary $\boldsymbol{\pi}$-factor with $\boldsymbol{\pi} := (\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_{J-1})$ and $r = r(\boldsymbol{\pi}_{J-1}, \boldsymbol{\pi}_J)$. As $\mathbf{Y}_1 := \widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J$ and $\mathbf{Y}_2 := \widetilde{\mathbf{B}}_p$ are $\boldsymbol{\pi}'$-factors, and since by Lemma 4.14, $r(\boldsymbol{\pi}, \boldsymbol{\pi}') = r(\boldsymbol{\pi}_{J-1}, \boldsymbol{\pi}_J) = r$, left-$r$-unitarity of $\mathbf{X}$ and the parallelogram law yield

$$\langle \mathbf{X}\mathbf{Y}_1, \mathbf{X}\mathbf{Y}_2 \rangle = \frac{1}{2}(\|\mathbf{X}\mathbf{Y}_1\|_F^2 + \|\mathbf{X}\mathbf{Y}_2\|_F^2 - \|\mathbf{X}(\mathbf{Y}_1 + \mathbf{Y}_2)\|_F^2) = \langle \mathbf{Y}_1, \mathbf{Y}_2 \rangle.$$

Combining this with the expressions of $\mathbf{B}_{J-1}, \mathbf{B}_J, \mathbf{B}_p$, we obtain:

$$\langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_p \rangle = \langle \mathbf{X}_{\text{left}}^{(J)}(\widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J), \mathbf{X}_{\text{left}}^{(J)} \widetilde{\mathbf{B}}_p \rangle = \langle \widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J, \widetilde{\mathbf{B}}_p \rangle.$$

Orthogonality conditions and their proof. Consider the partition $\mathcal{P} := \mathcal{P}(\mathbf{S}_{\boldsymbol{\pi}_{\text{left}}}, \mathbf{S}_{\boldsymbol{\pi}_{\text{right}}})$ with $\boldsymbol{\pi}_{\text{left}} := \boldsymbol{\pi}_J$ and $\boldsymbol{\pi}_{\text{right}} = \boldsymbol{\pi}_{J+1} * \ldots * \boldsymbol{\pi}_L$. For each $k \in \{J-1, J, p\}$, since $\widetilde{\mathbf{B}}_k$ is a $\boldsymbol{\pi}'$-factor and $\boldsymbol{\pi}' = \boldsymbol{\pi}_{\text{left}} * \boldsymbol{\pi}_{\text{right}}$, by Lemma 4.8 we have $\texttt{supp}(\widetilde{\mathbf{B}}_k) \subseteq \bigcup_{P \in \mathcal{P}} R_P \times C_P$ hence

$$\langle \widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J, \widetilde{\mathbf{B}}_p \rangle = \sum_{P \in \mathcal{P}} \langle (\widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J)[R_P, C_P], \widetilde{\mathbf{B}}_p[R_P, C_P] \rangle.$$

It follows that $\langle \mathbf{B}_{J-1} - \mathbf{B}_J, \mathbf{B}_p \rangle = 0$ is implied by the orthogonality conditions

$$\langle (\widetilde{\mathbf{B}}_{J-1} - \widetilde{\mathbf{B}}_J)[R_P, C_P], \widetilde{\mathbf{B}}_p[R_P, C_P] \rangle = 0, \quad \forall P \in \mathcal{P}. \tag{E.21}$$

To conclude it remains to prove that (E.21) indeed holds for a fixed $P \in \mathcal{P}$. By the definition of $\mathbf{X}_{[\![J,J]\!]} = \mathbf{X}_{[\![q,s]\!]}$, $\mathbf{X}_{[\![J+1,L]\!]} = \mathbf{X}_{[\![s+1,t]\!]}$ and $\mathbf{X}_{[\![J,L]\!]} = \mathbf{X}_{[\![q,t]\!]}$ at line 18 of Algorithm 6.1 and line 3 of Algorithm 3.1, we obtain that

$$\begin{aligned} \widetilde{\mathbf{B}}_J[R_P, C_P] &\overset{\text{(E.20)}}{=} (\mathbf{X}_{[\![J,J]\!]} \mathbf{X}_{[\![J+1,L]\!]})[R_P, C_P] \\ &\overset{\text{Lemma A.1}}{=} \mathbf{X}_{[\![J,J]\!]}[R_P, P] \mathbf{X}_{[\![J+1,L]\!]}[P, C_P] \end{aligned}$$

is the best rank-$|P|$ approximation of

$$\widetilde{\mathbf{B}}_{J-1}[R_P, C_P] \overset{\text{(E.20)}}{=} \mathbf{X}_{[\![J,L]\!]}[R_P, C_P].$$

If the two matrices are equal then (E.21) trivially holds, otherwise we have

$$\begin{aligned}\texttt{rank}(\widetilde{\mathbf{B}}_J[R_P, C_P]) &= \texttt{rank}(\mathbf{X}_{[\![J,J]\!]}[R_P, P]) = |P| \\ \operatorname{colspan}(\widetilde{\mathbf{B}}_J[R_P, C_P]) &= \operatorname{colspan}(\mathbf{X}_{[\![J,J]\!]}[R_P, P]).\end{aligned}$$

Since $\mathbf{X}_{[\![J,J]\!]}$ is pseudo-orthonormalized into $\hat{\mathbf{X}}_J$ at the next iteration $J+1$ (by design of Algorithm C.1), we also have

$$\operatorname{colspan}(\mathbf{X}_{[\![J,J]\!]}[R_P, P]) = \operatorname{colspan}(\hat{\mathbf{X}}_J[R_P, P]).$$

Combining these equalities with Lemma E.4 we obtain

$$\operatorname{colspan}(\hat{\mathbf{X}}_J[R_P, P]) = \operatorname{colspan}(\widetilde{\mathbf{B}}_J[R_P, C_P]) \perp \operatorname{colspan}((\widetilde{\mathbf{B}}_J - \widetilde{\mathbf{B}}_{J-1})[R_P, C_P]).$$

The orthogonality condition (E.21) is then implied by the fact that

$$\operatorname{colspan}(\widetilde{\mathbf{B}}_p[R_P, C_P]) \subseteq \operatorname{colspan}(\hat{\mathbf{X}}_J[R_P, P])$$

which is trivial if $p = J$, and if $p > J$ follows from the fact by (E.20) we have $\widetilde{\mathbf{B}}_p = \hat{\mathbf{X}}_J \mathbf{Z}$ for some $\mathbf{Z} :=\in \Sigma^{\boldsymbol{\pi}_{\text{right}}}$. ∎

Appendix F. On the generalization of the complementary low-rank property We show that the generalized complementary low-rank property associated with a chainable $\boldsymbol{\beta}$ given in Definition 7.6 coincides, under some assumption on $\boldsymbol{\beta}$, with the classical definition of the complementary low-rank property given in [23].

F.1. Definition of the classical complementary low-rank property [23] We start by reformulating the definition of [23], based on the following terminology. A cluster tree T of a set of indices $[\![n]\!]$ with depth L is a tree where:

- the nodes are subsets of $[\![n]\!]$;
- the root is $[\![n]\!]$;
- each non-leaf node has non-empty children that partition their parent;
- the only leaves are at level L.

By convention the root nodes are at level 0. The set of all nodes at level $\ell \in [\![L]\!]$ is denoted $T(\ell)$. Notice that, by definition of a cluster tree T, the set of nodes $T(\ell)$ form a partition of the root node for each level ℓ, and $T(\ell+1)$ is finer than $T(\ell)$, in the following sense.

Definition F.1 (Finer partitions [15, Definition 1.11]). *Given two partitions P and $\tilde{P}$ of $[\![n]\!]$, P is* finer *than $\tilde{P}$ if for all $I \in P$ there is a $\tilde{I} \in \tilde{P}$ with $I \subseteq \tilde{I}$.*

We can now give a definition that covers the classical notion of complementary low-rank property [23].

Definition F.2 ("Classical" complementary low-rank property). *Consider two cluster trees T^{row} and T^{col} of $[\![m]\!]$ and $[\![n]\!]$ with the same depth L, and a set of integer* rank constraints $\mathcal{R} := \{r_{R,C} \,|\, (R,C) \in T^{row}(\ell) \times T^{col}(L-\ell+1),\, \ell \in [\![L]\!]\} \subseteq \mathbb{N}$. *A matrix $\mathbf{A}$ of size $m \times n$ satisfies the* complementary low-rank property for T^{row} and T^{col} with rank constraints $\mathcal{R}$ if the submatrix $\mathbf{A}[R, C]$ has rank at most $r_{R,C}$ for each $(R, C) \in T^{\text{row}}(\ell) \times T^{\text{col}}(L-\ell+1)$ and $\ell \in [\![L]\!]$.

With this definition, a matrix $\mathbf{A}$ satisfies the complementary low-rank property of [23, 25] if it is ϵ-close in the Frobenius norm to a matrix $\tilde{\mathbf{A}}$ satisfying Definition F.2 in the particular case where T^{row} and T^{col} are *dyadic trees* or *quadtrees*, where it is assumed that $\max_{R,C} r_{R,C}$ is bounded poly-logarithmically in $1/\epsilon$ and in matrix size.

F.2. Relation with the generalized complementary low-rank property (Definition 7.6) A cluster tree yields a hierarchical partitioning of a given set of indices. Similarly, under some appropriate conditions, a chainable architecture $\boldsymbol{\beta}$ also yields a hierarchical partitioning of the row and column indices, which leads to two cluster trees.

PROPOSITION F.3. *Consider a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ where $\boldsymbol{\pi}_\ell := (a_\ell, b_\ell, c_\ell, d_\ell)$ for $\ell \in [\![L]\!]$, and denote $m \times n$ the size of the matrices in $\mathcal{B}^{\boldsymbol{\beta}}$, as well as*

$$\mathbf{S}_{q,t} := \mathbf{S}_{\boldsymbol{\pi}_q * \ldots * \boldsymbol{\pi}_t}, \quad 1 \leq q \leq t \leq L. \tag{F.1}$$

Recalling the notation from Definition 3.3*, define for all $\ell \in [\![L-1]\!]$:*

$$\begin{aligned} \mathcal{P}_{L-\ell}^{\mathrm{row}} &:= \{R_P \,|\, P \in \mathcal{P}(\mathbf{S}_{1,\ell}, \mathbf{S}_{\ell+1,L})\}, \\ \mathcal{P}_{\ell}^{\mathrm{col}} &:= \{C_P \,|\, P \in \mathcal{P}(\mathbf{S}_{1,\ell}, \mathbf{S}_{\ell+1,L})\}. \end{aligned} \tag{F.2}$$

Assume that $a_1 = d_L = 1$. Then:

1. *For each $\ell \in [\![L-1]\!]$, $\mathcal{P}_\ell^{\mathrm{row}}$ is a partition of $[\![m]\!]$, and $\mathcal{P}_\ell^{\mathrm{col}}$ is a partition of $[\![n]\!]$. Moreover $\mathcal{P}_{\ell+1}^{\mathrm{row}}$ (resp. $\mathcal{P}_{\ell+1}^{\mathrm{col}}$) is finer than $\mathcal{P}_\ell^{\mathrm{col}}$ (resp. than $\mathcal{P}_\ell^{\mathrm{row}}$) when $\ell \leq L-2$.*
2. *Consequently, when completed by $\mathcal{P}_0^{\mathrm{row}} := [\![m]\!]$ and $\mathcal{P}_0^{\mathrm{col}} := [\![n]\!]$, the collections $\{\mathcal{P}_\ell^{\mathrm{row}}\}_{\ell=0}^{L-1}$ and $\{\mathcal{P}_\ell^{\mathrm{col}}\}_{\ell=0}^{L-1}$ yield two cluster trees, denoted $T_{\boldsymbol{\beta}}^{row}$ and $T_{\boldsymbol{\beta}}^{col}$, of depth $L-1$ with root node $[\![m]\!]$ and $[\![n]\!]$, respectively.*
3. *For each $\ell \in [\![L-1]\!]$ we have*

$$\{(R_P, C_P) : P \in \mathcal{P}(\mathbf{S}_{1,\ell}, \mathbf{S}_{\ell+1,L})\} = T_{\boldsymbol{\beta}}^{row}(L-\ell) \times T_{\boldsymbol{\beta}}^{col}(\ell). \tag{F.3}$$

We postpone the proof of Proposition F.3 to immediately highlight that under its assumptions the general complementary low-rank property of Definition 7.6 coincides with the classical one of Definition F.2.

COROLLARY F.4. *Consider a chainable architecture $\boldsymbol{\beta} = (\boldsymbol{\pi}_\ell)_{\ell=1}^L$ where $\boldsymbol{\pi}_\ell := (a_\ell, b_\ell, c_\ell, d_\ell)$ for $\ell \in [\![L]\!]$, and assume that $a_1 = d_L = 1$. For any matrix $\mathbf{A}$ of appropriate size, the following are equivalent:*

- $\mathbf{A}$ *satisfies the* generalized *complementary low-rank property (Definition* 7.6*) associated with $\boldsymbol{\beta}$;*
- $\mathbf{A}$ *satisfies the* classical *complementary low-rank property (Definition* F.2*) for the cluster trees $(T_{\boldsymbol{\beta}}^{row}, T_{\boldsymbol{\beta}}^{col})$ defined in Proposition* F.3*, with rank constraint $\mathcal{R}$ such that for each $\ell \in [\![L-1]\!]$ and every $(R, C) \in T_{\boldsymbol{\beta}}^{row}(L-\ell) \times T_{\boldsymbol{\beta}}^{col}(\ell)$ we have $r_{R,C} = r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1})$.*

Proof. Since $\boldsymbol{\beta}$ is chainable and $a_1 = d_L = 1$, by Lemma 4.12 we have $\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L = (1, m, n, 1)$ for some integers m, n (which turn out to be the dimensions of matrix $\mathbf{A}$) hence $\mathbf{S}_{\boldsymbol{\pi}_1 * \ldots * \boldsymbol{\pi}_L} = \mathbf{1}_{m \times n}$. Therefore, by Definition 7.6, a matrix $\mathbf{A}$ satisfies the general complementary low-rank property associated with $\boldsymbol{\beta}$ if, and only if, $\mathtt{rank}(\mathbf{A}[R_P, C_P]) \leq r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1})$ for each $P \in \mathcal{P}(\mathbf{S}_{1,\ell}, \mathbf{S}_{\ell+1,L})$ and $\ell \in [\![L-1]\!]$ (we use the shorthand (F.1)). By Proposition F.3, this is precisely a reformulation of the classical complementary low-rank property (Definition F.2) for the trees $(T_{\boldsymbol{\beta}}^{\mathrm{row}}, T_{\boldsymbol{\beta}}^{\mathrm{col}})$.

Proof of Proposition F.3. The second claim is an immediate consequence of the first one. We only prove the first claim for the column partitions $\{\mathcal{P}_\ell^{\mathrm{col}}\}_{\ell=1}^{L-1}$, since the proof is similar for the row partitions $\{\mathcal{P}_\ell^{\mathrm{row}}\}_{\ell=1}^{L-1}$. Given $\ell \in [\![L-1]\!]$, let us first show that $\mathcal{P}_\ell^{\mathrm{col}}$ is a partition of $[\![n]\!]$ for any $\ell \in [\![L-1]\!]$, where $n := a_L c_L d_L$. By (4.5): $\boldsymbol{\pi}_{\ell+1} * \ldots * \boldsymbol{\pi}_L = \left(a_{\ell+1}, \frac{b_{\ell+1} d_{\ell+1}}{d_L}, \frac{a_L c_L}{a_{\ell+1}}, d_L\right) = \left(a_{\ell+1}, b_{\ell+1} d_{\ell+1}, \frac{a_L c_L}{a_{\ell+1}}, 1\right)$ since

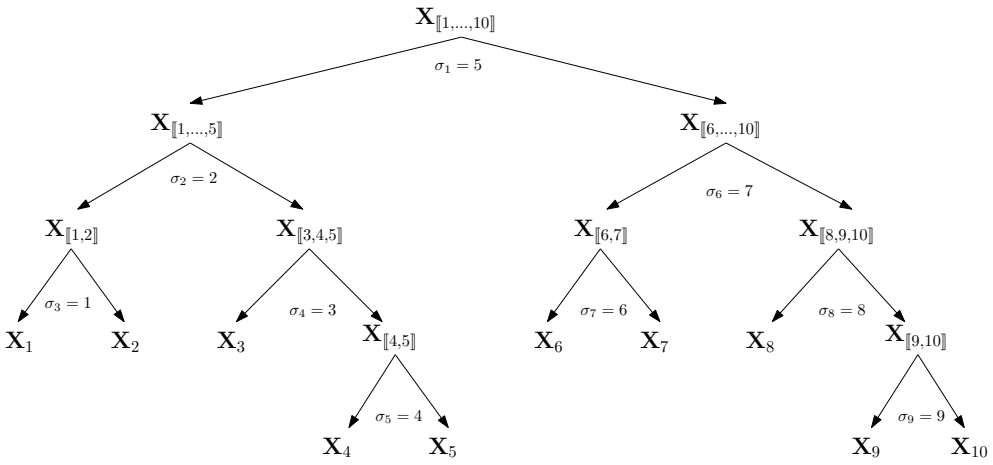


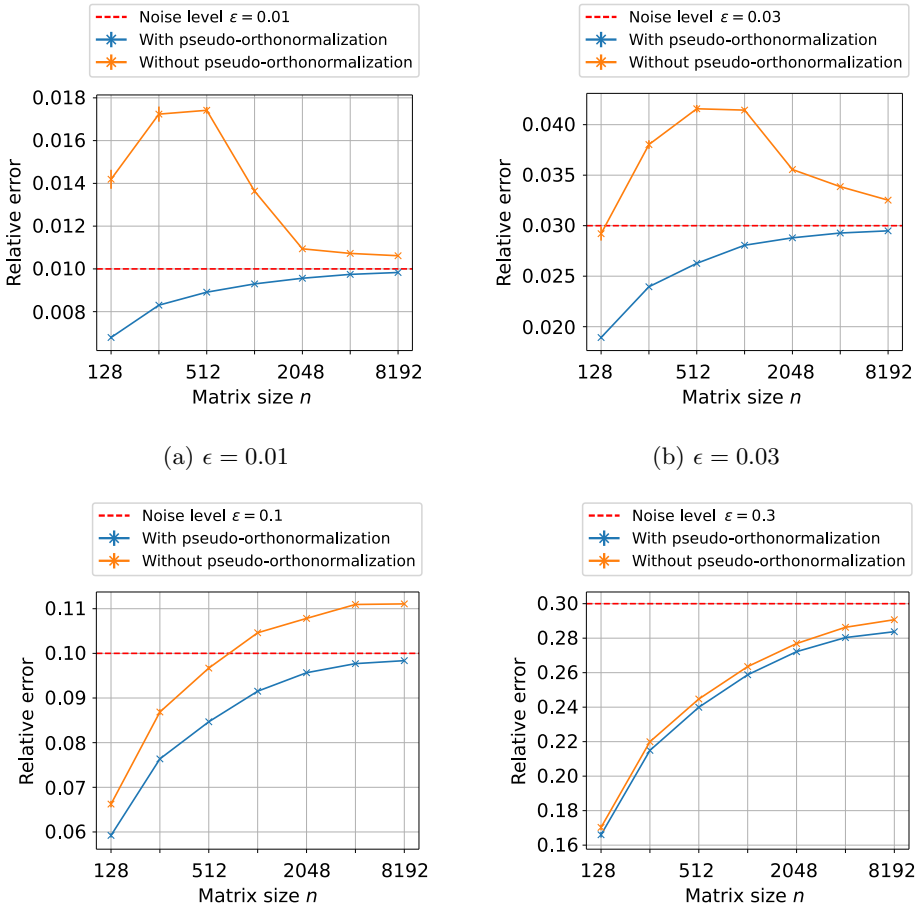
Fig. 7: Illustration of the permutation $\sigma = (5, 2, 1, 3, 4, 7, 6, 8, 9)$ corresponding to the balanced factor-bracketing tree of $[\![10]\!]$.

$d_L = 1$. Therefore, by Definition 4.1, the elements of $\mathcal{P}_\ell^{\text{col}}$ partition the integer set $[\![n]\!]$ (with $n := a_L c_L = a_L c_L d_L$) into consecutive intervals of length c_ℓ, where $c_\ell := \frac{a_L c_L}{a_{\ell+1}}$. It remains to show that each element of $\mathcal{P}_\ell^{\text{col}}$ can itself be partitioned into elements of $\mathcal{P}_{\ell+1}^{\text{col}}$. Since the latter are consecutive intervals of length $c_{\ell+1}$, this is a direct consequence of the fact that c_ℓ is an integer multiple of $c_{\ell+1}$: indeed, since $(\boldsymbol{\pi}_{\ell+1}, \boldsymbol{\pi}_{\ell+2})$ is chainable, we have: $a_{\ell+1} \mid a_{\ell+2}$, $\gamma = a_{\ell+2}/a_{\ell+1} \in \mathbb{N}$ and $c_\ell = \gamma c_{\ell+1}$.

To prove (F.3) observe that the left hand side is trivially a subset of $T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell) \times T_{\boldsymbol{\beta}}^{\text{col}}(\ell)$, by the definition of $T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell) = \mathcal{P}_{L-\ell}^{\text{row}}$ and $T_{\boldsymbol{\beta}}^{\text{col}}(\ell) = \mathcal{P}_\ell^{\text{col}}$ (cf (F.2)). The equality will follow from the fact that both sides share the same number of elements: this is a direct consequence of the fact that $|T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell)| = d_\ell$, $|T_{\boldsymbol{\beta}}^{\text{col}}(\ell)| = a_{\ell+1}$, a property that we prove immediately below. Indeed, this property implies that $|T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell) \times T_{\boldsymbol{\beta}}^{\text{col}}(\ell)| = a_{\ell+1} d_\ell$, and by Lemma 4.14, the number of equivalence classes P of $\mathcal{P}(\mathbf{S}_{1,\ell}, \mathbf{S}_{\ell+1,L})$ is also $\frac{a_\ell c_\ell d_\ell}{r(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_{\ell+1})} = a_{\ell+1} d_\ell$. To conclude, we prove that $|T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell)| = d_\ell$, $|T_{\boldsymbol{\beta}}^{\text{col}}(\ell)| = a_{\ell+1}$. By Lemma 4.12, we have $\boldsymbol{\pi}_{\ell+1} * \ldots * \boldsymbol{\pi}_L = \left(a_{\ell+1}, b_{\ell+1} d_{\ell+1}, \frac{a_L c_L}{a_{\ell+1}}, 1\right)$ since $d_L = 1$ hence $\mathbf{S}_{\ell+1,L}$ is a block diagonal matrix with $a_{\ell+1}$ dense blocks of the same size. Tracing back all definitions this implies that $T_{\boldsymbol{\beta}}^{\text{col}}(\ell) = \mathcal{P}_\ell^{\text{col}}$ is made of exactly $a_{\ell+1}$ consecutive intervals of the same size. The proof is similar for $T_{\boldsymbol{\beta}}^{\text{row}}(L-\ell)$. ∎

Appendix G. Numerical experiments: additionnal details and results

In complement to Section 8, we include further experimental details and results. In Figure 4b, we observed that the approximation error obtained by the hierarchical algorithm *with* orthonormalization operations (Algorithm 6.1) is always smaller than the noise level $\epsilon = 0.1$, as opposed to the one obtained without orthonormalization. In fact, we have the same observation for any values of $\epsilon \in \{0.01, 0.03, 0.1, 0.3\}$, as shown in Figure 8.



(a) $\epsilon = 0.01$ (b) $\epsilon = 0.03$

(c) $\epsilon = 0.1$ (d) $\epsilon = 0.3$

Fig. 8: Relative approximation errors *vs.* the matrix size n, for Algorithm 5.1 (without orthonormalization) and Algorithm 6.1 (with orthonormalization), for the instance of Problem (1.1) described in Subsection 8.2 with $r = 4$. We show mean and standard deviation on the error bars over 10 repetitions of the experiment.