Linear Algebra and its Applications 509 (2016) 114–142

Contents lists available at ScienceDirect

Linear Algebra and its Applications

www.elsevier.com/locate/laa

Generalizing block LU factorization: A lower–upper–lower block triangular decomposition with minimal off-diagonal ranks

François Serre [*], Markus Püschel

Department of Computer Science, ETH Zurich, Switzerland

A R T I C L E   I N F O

Article history:
Received 6 March 2015
Accepted 23 July 2016
Available online 28 July 2016
Submitted by A. Frommer

MSC:
15A23
15A03

Keywords:
Block matrices
Block triangular decomposition
LU factorization
Rank minimization
Circuit complexity
Schur complement

A B S T R A C T

We propose a novel factorization of a non-singular matrix P, viewed as a 2×2-blocked matrix. The factorization decomposes P into a product of three matrices that are lower block-unitriangular, upper block-triangular, and lower block-unitriangular, respectively. Our goal is to make this factorization "as block-diagonal as possible" by minimizing the ranks of the off-diagonal blocks. We give lower bounds on these ranks and show that they are sharp by providing an algorithm that computes an optimal solution. The proposed decomposition can be viewed as a generalization of the well-known Block LU factorization using the Schur complement. Finally, we briefly explain one application of this factorization: the design of optimal circuits for a certain class of streaming permutations.

* Corresponding author.
E-mail addresses: serref@inf.ethz.ch (F. Serre), pueschel@inf.ethz.ch (M. Püschel).

http://dx.doi.org/10.1016/j.laa.2016.07.020
0024-3795/© 2016 Elsevier Inc. All rights reserved.

1. Introduction

Given is a non-singular matrix $P \in GL_{m+n}(\mathbb{K})$ over a field $\mathbb{K}$. We partition P as

$$P = \begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix}, \quad \text{such that } P_1 \text{ is } m \times m.$$

We denote the ranks of the submatrices with $p_i = \operatorname{rk} P_i$, $i = 1, 2, 3, 4$. Matrices are denoted with capital letters and vector spaces with calligraphic letters.

If P_1 is non-singular, then a block Gaussian elimination uniquely decomposes P into the form

$$P = \begin{pmatrix} I_m & \\ L & I_n \end{pmatrix} \begin{pmatrix} C_1 & C_2 \\ & C_4 \end{pmatrix}, \tag{1}$$

where I_m denotes the $m \times m$ identity matrix. The rank of $L = P_3 P_1^{-1}$ is equal to p_3, and C_4 is the Schur complement of P_1. Conversely, if such a decomposition exists for P, then P_1 is non-singular. This block LU decomposition has several applications including computing the inverse of P [1], solving linear systems [2], and in the theory of displacement structure [3]. The Schur complement is also used in statistics, probability and numerical analysis [4,5].

Analogously, the following decomposition exists if and only if P_4 is non-singular:

$$P = \begin{pmatrix} C_1 & C_2 \\ & C_4 \end{pmatrix} \begin{pmatrix} I_m & \\ R & I_n \end{pmatrix}. \tag{2}$$

This decomposition is again unique, and the rank of R is p_3.

In this article, we release the restrictions on P_4 and P_1 and propose the following decomposition for a general $P \in GL_{m+n}(\mathbb{K})$:

$$P = \begin{pmatrix} I_m & \\ L & I_n \end{pmatrix} \begin{pmatrix} C_1 & C_2 \\ & C_4 \end{pmatrix} \begin{pmatrix} I_m & \\ R & I_n \end{pmatrix}, \tag{3}$$

where in addition we want the three factors to be "as block-diagonal as possible," i.e., that $\operatorname{rk} L + \operatorname{rk} C_2 + \operatorname{rk} R$ is minimal.

1.1. Lower bounds

The following theorem provides bounds on the ranks of such a decomposition:

Theorem 1. *If a decomposition (3) exists for $P \in GL_{m+n}(\mathbb{K})$, it satisfies*

$$\operatorname{rk} C_2 = p_2, \tag{4}$$

$$\operatorname{rk} L \geq n - p_4, \tag{5}$$

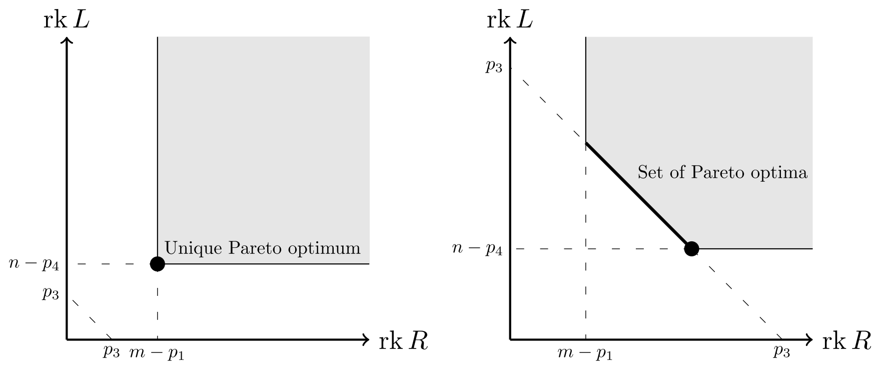


Fig. 1. Possible ranks for L and R. On the left graph, $p_3 < m + n - p_4 - p_1$. On the right graph, $p_3 > m + n - p_4 - p_1$. The dot shows the decomposition provided by Theorem 2.

$$\operatorname{rk} R \geq m - p_1, \tag{6}$$

$$\operatorname{rk} R + \operatorname{rk} L \geq p_3. \tag{7}$$

In particular, the rank of C_2 is fixed and we have

$$\operatorname{rk} R + \operatorname{rk} L \geq \max(p_3, n + m - p_4 - p_1). \tag{8}$$

We will prove this theorem in Section 4. Next, we assert that these bounds are sharp.

1.2. Optimal solution

The following theorem shows that the inequality (8) is sharp:

Theorem 2. *If $P \in GL_{m+n}(\mathbb{K})$, then there exists a decomposition (3) that satisfies*

$$\operatorname{rk} R + \operatorname{rk} L = \max(p_3, n + m - p_4 - p_1) \text{ and}$$
$$\operatorname{rk} L = n - p_4.$$

Additionally, such a decomposition can be computed with $O((m + n)^3)$ arithmetic operations.

We prove this theorem in Section 5 when $p_3 \leq m + n - p_4 - p_1$, and in Section 6 for the case $p_3 > m + n - p_4 - p_1$. In both cases, the proof is constructive and we provide a corresponding algorithm (Algorithms 3 and 4). Theorem 2 and the corresponding algorithms are the main contributions of this article.

Two cases. As illustrated in Fig. 1, two different cases appear from inequality (8). If $p_3 \leq m + n - p_4 - p_1$, bound (7) is not restrictive, and the optimal pair $(\operatorname{rk} L, \operatorname{rk} R)$ is unique and equals $(n - p_4, m - p_1)$. In the other case, where $p_3 > m + n - p_4 - p_1$, bound (7) becomes restrictive and several optimal pairs $(\operatorname{rk} L, \operatorname{rk} R)$ exist.

Example. As a simple example we consider the special case

$$P = \begin{pmatrix} & P_2 \\ P_3 & \end{pmatrix}, \quad \text{with } n = m.$$

In this case P_3, P_2 are non-singular and neither (1) nor (2) exists. Theorem 1 gives a lower bound of $\operatorname{rk} R + \operatorname{rk} L \geq 2n$, which implies that both R and L have full rank. Straightforward computation shows that for any non-singular L,

$$P = \begin{pmatrix} I_n & \\ L & I_n \end{pmatrix} \begin{pmatrix} L^{-1}P_3 & P_2 \\ & -LP_2 \end{pmatrix} \begin{pmatrix} I_n & \\ -(LP_2)^{-1}P_3 & I_n \end{pmatrix}$$

is an optimal solution. This also shows that the optimal decomposition (3) is in general not unique.

1.3. *Flexibility*

The following theorem adds flexibility to Theorem 2 and shows that a decomposition exists for any Pareto-optimal pair of non-diagonal ranks that satisfies the bounds of Theorem 1:

Theorem 3. *If $P \in GL_{m+n}(\mathbb{K})$ and $(l, r) \in \mathbb{N}^2$ satisfies $l \geq n - p_4$, $r \geq m - p_1$, and*

$$r + l = \max(p_3, n + m - p_4 - p_1),$$

then P has a decomposition (3) with $\operatorname{rk} L = l$ and $\operatorname{rk} R = r$.

In the case where $p_3 \leq m + n - p_4 - p_1$, the decomposition produced by Theorem 2 has already the unique optimal pair $(\operatorname{rk} L, \operatorname{rk} R) = (n - p_4, m - p_1)$. In the other case, we will provide a method in Section 7 to trade between the rank of R and the rank of L, until bound (6) is reached. By iterating this method over the decomposition obtained in Theorem 2, decompositions with various rank tradeoffs can be built.

Therefore, it is possible to build decomposition (3) for any pair $(\operatorname{rk} L, \operatorname{rk} R)$ that is a Pareto optimum of the given set of bounds. As a consequence, if $f : \mathbb{N}^2 \to \mathbb{R}$ is weakly increasing in both of its arguments, it is possible to find a decomposition that minimizes $f(\operatorname{rk} L, \operatorname{rk} R)$. Examples include $\min(\operatorname{rk} L, \operatorname{rk} R)$, $\max(\operatorname{rk} L, \operatorname{rk} R)$, $\operatorname{rk} L + \operatorname{rk} R$, $\operatorname{rk} L \cdot \operatorname{rk} R$ or $\sqrt{\operatorname{rk}^2 L + \operatorname{rk}^2 R}$.

Generalization of block LU factorization. In the case where P_4 is non-singular, Theorem 2 provides a decomposition that satisfies $\operatorname{rk} L = 0$. In other words, it reduces to the decomposition (2). Using Theorem 3, we can obtain a similar result in the case where P_1 is non-singular. Since in this case $m - p_1 = 0$, it is possible to choose $r = 0$, and thus obtain the decomposition (1).

1.4. Equivalent formulations

Lemma 1. *The following decomposition is equivalent to decomposition (3), with analogous constraints for the non-diagonal ranks:*

$$P = \begin{pmatrix} I_m & \\ L_3 & L_4 \end{pmatrix} \begin{pmatrix} C_1 & C_2 \\ & I_n \end{pmatrix} \begin{pmatrix} I_m & \\ R_3 & R_4 \end{pmatrix}. \tag{9}$$

In this case, the minimization of the non-diagonal ranks is exactly the same problem as in (3). However, an additional degree of freedom appears: any non-singular $n \times n$ matrix can be chosen for either L_4 or R_4.

It is also possible to decompose P into two matrices, one with a non-singular leading principal submatrix L_1 and the other one with a non-singular lower principal submatrix R_4:

$$P = \begin{pmatrix} L_1 & L_2 \\ L_3 & L_4 \end{pmatrix} \begin{pmatrix} R_1 & R_2 \\ R_3 & R_4 \end{pmatrix}. \tag{10}$$

Once again, the minimization of $\operatorname{rk} L_3 + \operatorname{rk} R_3$ is the same problem as in (3). The two other non-diagonal blocks satisfy $\operatorname{rk} L_2 + \operatorname{rk} R_2 \geq p_2$.

Proof. The lower non-diagonal ranks are invariant through the following steps:

(3) ⇒ (9). If P has a decomposition (3), a straightforward computation shows that

$$P = \begin{pmatrix} I_m & \\ L & C_4 \end{pmatrix} \begin{pmatrix} C_1 & C_2 \\ & I_n \end{pmatrix} \begin{pmatrix} I_m & \\ R & I_n \end{pmatrix},$$

which has the form of decomposition (9).

(9) ⇒ (10). If P has a decomposition (9), the multiplication of the two left factors leads to formulation (10). In fact, $L_1 = C_1$ and R_4 are both non-singular.

(10) ⇒ (3). If P has a decomposition (10), then using (1) on the left factor, and (2) on the right factor, and multiplying the two central matrices leads to formulation (3). □

1.5. Related work

Schur complement. Several efforts have been made to adapt the definition of Schur complement in the case of general P_1 and P_4. For instance, it is possible to define an indexed Schur complement of another non-singular principal submatrix [4], or use pseudo-inverses [6] for matrix inversion algorithms.

Alternative block decompositions. A common way to handle the case where P_1 is singular is to use a permutation matrix B that reorders the columns of P such that the new principal upper submatrix is non-singular [4]. Decomposition (1) then becomes

$$P = PBB^T = \begin{pmatrix} I_m & \\ L & I_n \end{pmatrix} \begin{pmatrix} C_1 & C_2 \\ & C_4 \end{pmatrix} .B^T$$

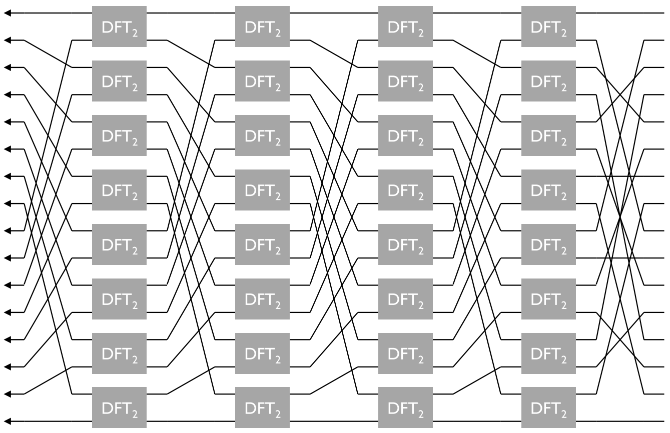


Fig. 2. Dataflow of a Pease FFT on 16 points from right (input) to left (output). Computing stages, consisting of DFTs on two points and pointwise multiplications (not shown), alternate with permutations.

However, B needs to swap columns with index $\geq m$; thus B^T does not have the required form considered in our work.

One can modify the above idea to choose B such that B^{-1} has the shape required by decomposition (3):

$$\begin{aligned} P &= \begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix} \begin{pmatrix} I_m & \\ -R & I_n \end{pmatrix} \begin{pmatrix} I_m & \\ R & I_n \end{pmatrix} \\ &= \begin{pmatrix} P_1 - RP_2 & P_2 \\ P_3 - RP_4 & P_4 \end{pmatrix} \begin{pmatrix} I_m & \\ R & I_n \end{pmatrix}. \end{aligned}$$

Then the problem is to design R such that $P_1 - RP_2$ is non-singular and $\operatorname{rk}(P_3 - RP_4) + \operatorname{rk} R$ is minimal. This basic idea is used in [7], where, however, only $\operatorname{rk} R$ is minimized, which, in general, does not produce optimal solutions for the problem considered here.

Finally, our decomposition also shares patterns with a block Cholesky decomposition, or the Block LDL decomposition, in the sense that they involve block uni-triangular matrices. However, the requirements on P and the expectations on the decomposition are different.

2. Application: optimal circuits for streaming permutations

The original motivation for considering our decomposition (3) was an important application in the domain of hardware design. Many algorithms in signal processing and communication consist of alternating computation and reordering (permutation) stages. Typical examples include fast Fourier transforms [8]; one example (a so-called Pease FFT) is shown in Fig. 2 for 16 data points. When mapped to hardware, permutations could become simple wires. However, usually, the data is not available in one cycle, but streamed in chunks over several cycles. Implementing such a "streaming" permutation in this scenario on an application-specific integrated circuit (ASIC) or on a field programmable gate array (FPGA) becomes complex, as it now requires both logic and memory [9,7,10]. It turns out that for an important subclass of permutations called "linear," the design of an optimal circuit (i.e., one with minimal logic) is equivalent to solving (3) in the field $\mathbb{F}_2$.

Next, we provide a few more details on this application starting with the necessary background information. However, we only provide a sketch; a more complete treatment can be found in [10] and in [7].

2.1. Linear permutations

For $0 \leq i < 2^{m+n}$, we denote with i_b the associated (bit) vector in $\mathbb{F}_2^{m+n}$ that contains the radix-2 digits of i, with the most significant digit on top. For instance, for $m+n=4$, we have

$$12_b = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \text{ and } 7_b = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Any invertible $(m+n) \times (m+n)$ matrix P over $\mathbb{F}_2$ induces a permutation $\pi(P)$ on $\{0, \ldots, 2^{m+n}-1\}$ that maps i to j, where $j_b = P \cdot i_b$.

For example, if we define

$$V_{m+n} = \begin{pmatrix} 1 & & \\ \vdots & \ddots & \\ 1 & & 1 \end{pmatrix},$$

then $\pi(V_3)$ is the mapping $0 \mapsto 0$, $1 \mapsto 1$, $2 \mapsto 2$, $3 \mapsto 3$, $4 \mapsto 7$, $7 \mapsto 4$, $5 \mapsto 6$, $6 \mapsto 5$. More generally, $\pi(V_{m+n})$ is the permutation that leaves the first half of the elements unchanged, and that reverts the second half.

The mapping $\pi : GL_{m+n}(\mathbb{F}_2) \to S_{2^{m+n}}$ is a group-homomorphism, and its range is called the group of linear permutations [11,12]. This group contains many of the permutations used in signal processing and communication algorithms, including stride permutations, bit-reversal, Hadamard reordering, and the Grey code reordering.

2.2. Streamed linear permutations (SLP)

We want to implement a circuit that performs a linear permutation on 2^{m+n} points. If we assume that this circuit has 2^n input (and output) ports, this means that the dataset has to be split into 2^m parts that are fed (streamed) as input over 2^m cycles. Similarly, the permuted output will be produced streamed over 2^m cycles. As an example consider Fig. 3 in which $2^n = 2$ and $2^m = 4$.

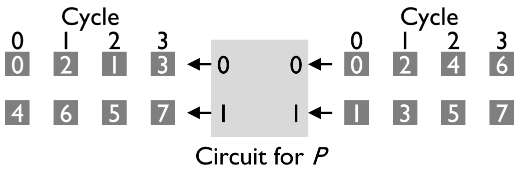


Fig. 3. Sketch of an implementation of the bit reversal permutation on 2^3 elements streamed on two ports. The dataset enters within 4 cycles (right), and is retrieved within 4 cycles (left).

With this convention, the element with the index $i = c \cdot 2^m + p$ arrives as input in the cth cycle at the pth port. Particularly, the upper m bits of i_b are the bit representation c_b of the input cycle c, while the lower n bits are p_b. For example, if $m = 3$ and $n = 2$, then the element indexed with

$$14_b = \begin{pmatrix} 0 \\ 1 \\ 1 \\ \hline 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 3_b \\ \hline 2_b \end{pmatrix}$$

will arrive during the third cycle on the second port.

Thus, it is natural to block the desired linear permutation

$$P = \begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix}$$

such that P_1 is $m \times m$. This implies that the element that arrives on port p during the cth cycle has to be routed to the output port $P_3 c_b + P_4 p_b$ at output cycle $P_1 c_b + P_2 p_b$.

P_3 has a particular role here, as it represents "how the routing between the different ports must vary during time," and directly influences the complexity of the implementation. For example, if $P_3 = 0$, then the output is always $P_4 p_b$ without variation during time.

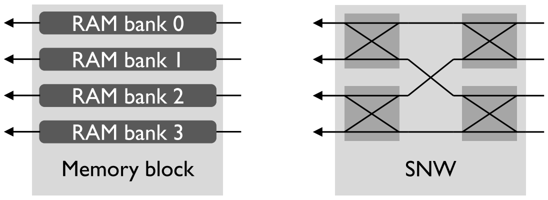
In fact, a theorem in [10] formalizes this intuition:

Theorem 4. *A full-throughput implementation of an SLP for P with 2^n ports that only uses 2×2-switches for routing requires at least $p_3 \cdot 2^{n-1}$ many switches.*

2.3. Implementing SLPs on hardware

Two special cases of SLPs can be directly translated to a hardware implementation. The first kind are the permutations that only permute across time, i.e., that do not change the port number of the elements. Thus, they satisfy $P_3 c_b + P_4 p_b = p_b$ for all c and p, and therefore have the form

$$P = \begin{pmatrix} P_1 & P_2 \\ & I_n \end{pmatrix}.$$



(a) RAM bank array (b) Switching network

Fig. 4. Two hardware blocks to implement particular types of SLPs. On the left, an array of 4 RAM banks. Write and read addresses are used to delay differently elements. This block can be used to implement SLPs that do not permute across ports. On the right is an example of a switching network consisting of 4 2×2-switches. Switching networks can be used to implement SLPs that do not permute across time.

These SLPs can be implemented using an array of 2^n blocks of RAMs as shown in Fig. 4(a).

Conversely, SLPs that only permute across the ports within each cycle have the form

$$P = \begin{pmatrix} I_m & \\ P_3 & P_4 \end{pmatrix}.$$

They do not require memory and can be implemented using a network of $p_3 \cdot 2^{n-1}$ 2×2-switches as was shown in [10,7]. This result, combined with decomposition (9) and Theorem 2 yields an immediate corollary:

Theorem 5. *For a given P, the associated SLP can be implemented using a RAM bank array (Fig. 4(a)) framed by two switching networks (Fig. 4(b)) with a total of* $\max(p_3, n+m-p_4-p_1) \cdot 2^{n-1}$ 2×2*-switches.*

Our Algorithms 3 and 4, introduced later, provide an efficient method to compute this architecture.

The total number of 2×2-switches, $\max(p_3, n+m-p_4-p_1)\cdot 2^{n-1}$, matches the bound of Theorem 4 in the case where $n+m \leq p_4+p_3+p_1$.

In the other case, this decomposition doesn't provide an optimal solution with respect to Theorem 4. However, a "transposed" version of decomposition (9) (obtained by transposing back (3) on P^T, and then using the same method as the one we used to get (9)) provides an implementation consisting of a central switching network, and two extremal memory blocks:

$$P = \begin{pmatrix} L_1 & L_2 \\ & I_n \end{pmatrix} \begin{pmatrix} I_m & \\ C_3 & C_4 \end{pmatrix} \begin{pmatrix} R_1 & R_2 \\ & I_n \end{pmatrix},$$

where $\operatorname{rk} C_3 = p_3$.

This implementation uses the minimal number of 2×2-switches, but uses the double amount of RAM banks compared to the previous structure.

3. Preliminaries

In this section, we will prove some basic lemmas that we will use throughout this article.

3.1. Properties of the blocks of an invertible matrix

In this subsection, we derive some direct consequences of the invertibility of P on the range and the nullspace of its submatrices.

Lemma 2. *The following properties are immediate from the structure of P:*

$$\ker P_4 \cap \ker P_2 = \{0\} \tag{11}$$

$$\ker P_3 \cap \ker P_1 = \{0\} \tag{12}$$

$$\operatorname{im} P_4 + \operatorname{im} P_3 = \mathbb{K}^n \tag{13}$$

$$\operatorname{im} P_2 + \operatorname{im} P_1 = \mathbb{K}^m \tag{14}$$

$$P_3(\ker P_1) \cap P_4(\ker P_2) = \{0\} \tag{15}$$

$$P_1(\ker P_3) \cap P_2(\ker P_4) = \{0\} \tag{16}$$

Proof. We prove here equation (15). If $x \in \ker P_1$ and $y \in \ker P_2$ satisfy $P_3x = P_4y$, we have

$$\begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix} \cdot \begin{pmatrix} x \\ -y \end{pmatrix} = \begin{pmatrix} P_1x - P_2y \\ P_3x - P_4y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Since P is non-singular, $x = y = 0$, as desired. □

These equalities yield the dimensions of the following subspaces:

Corollary 1.

$$\dim P_2(\ker P_4) = \dim \ker P_4 = n - p_4 \tag{17}$$

$$\dim P_1(\ker P_3) = \dim \ker P_3 = m - p_3 \tag{18}$$

$$\dim P_4(\ker P_2) = \dim \ker P_2 = n - p_2 \tag{19}$$

$$\dim P_3(\ker P_1) = \dim \ker P_1 = m - p_1 \tag{20}$$

$$\dim \operatorname{im} P_4 \cap \operatorname{im} P_3 = p_4 + p_3 - n \tag{21}$$

$$\dim \operatorname{im} P_2 \cap \operatorname{im} P_1 = p_2 + p_1 - m \tag{22}$$

Proof. We prove equation (17). Since, by (11), $\ker P_4 \cap \ker P_2 = \{0\}$, the dimension of the image of $\ker P_4$ under P_2 has the same dimension as $\ker P_4$, which is $n - p_4$.

Equation (21) is a consequence of equation (13) and of the rank-nullity theorem. □

Table 1
We summarize matrix operators to perform basic operations on subspaces. M is a general matrix, while A and B are matrices with m rows that represent the subspaces $\mathcal{A}$ and $\mathcal{B}$; i.e. $\mathcal{A} = \langle A\rangle$ and $\mathcal{B} = \langle B\rangle$. Inspection of these routines shows that they all can be implemented with $O(m^3)$ runtime.

Operation related to subspaces	Associated matrix operation	Correspondence
Kernel of a matrix M	$\overline{\ker}\, M$	$\langle \overline{\ker}\, M\rangle = \ker M$
Direct sum of subspaces $\mathcal{A}, \mathcal{B}$	$(A \quad B)$	$(A \quad B) = \mathcal{A} \oplus \mathcal{B}$
Intersection of subspaces $\mathcal{A}, \mathcal{B}$	$A \overline{\cap} B$	$\langle A \overline{\cap} B\rangle = \mathcal{A} \cap \mathcal{B}$
Complement of a subspace $\mathcal{B}$ in $\mathcal{A}$	$A \overline{\ominus} B$	$\langle A \overline{\ominus} B\rangle \oplus \mathcal{B} = \mathcal{A}$

3.2. Algorithms on linear subspaces in matrix form

The algorithms we present in this article heavily rely on operations on subspaces of $\mathbb{K}^m$. To make the representation of these algorithms more practical for implementation, we introduce a matrix representation for subspaces and formulate the subspace operations needed in this paper on this representation.

We represent a linear subspace as an associated matrix[1] whose columns form a basis of this subspace. In other words, if $\mathcal{A}$ is a subspace of $\mathbb{K}^m$ of dimension n, then we represent it using a $m \times n$ matrix A such that $\operatorname{im} A = \mathcal{A}$. In this case, and only in this case, we will use the notation $\langle A\rangle = \operatorname{im} A$ to emphasize that the columns of A form a linear independent set.

With this notation we can formulate subspace computations as computations on their associated matrices as explained in the following. To formally emphasize this correspondence, these operations on matrices will carry the same symbol as the subspace computation (e.g., $\cap$) they represent augmented with an overline (e.g., $\overline{\cap}$). The operations are collected in Table 1. All algorithms in this paper are written as sequences of these matrix operations. Because of this, we implemented the algorithms by first designing an object oriented infrastructure that provides these operations. Then we could directly map the algorithms, as they are formulated, to code.

Direct sum of two subspaces. If $\langle A\rangle, \langle B\rangle \leq \mathbb{K}^m$ are two subspaces, then the direct sum can be computed by concatenating the two matrices: $\langle A\rangle \oplus \langle B\rangle = \langle (A \quad B)\rangle$.

Null space of a matrix. Gaussian elimination can be used to compute the null space of a given $m \times n$ matrix M. Indeed, if the reduced column echelon form of the matrix $\begin{pmatrix} M \\ I_n \end{pmatrix}$ is blocked into the form $\begin{pmatrix} M_1 & \\ M_3 & M_4 \end{pmatrix}$, where M_1 has m rows and no zero column, then $\ker M = \langle M_4\rangle$. We denote this computation with $\overline{\ker}\, M = M_4$.

Intersection of two subspaces. For two subspaces $\operatorname{im} A, \operatorname{im} B \leq \mathbb{K}^m$, the intersection can be computed using the Zassenhaus algorithm. Namely, if the reduced column echelon form of $\begin{pmatrix} A & B \\ A & \end{pmatrix}$ is blocked into the form $\begin{pmatrix} C_1 & \\ C_3 & C_4 \end{pmatrix}$, where C_1 and C_4 have m rows and no zero column, then $\operatorname{im} A \cap \operatorname{im} B = \langle C_4\rangle$. We denote this computation with $A \overline{\cap} B = C_4$.

[1] We allow the existence of matrices with 0 column to represent the trivial subspace of $\mathbb{K}^m$.

Complement of a subspace within another one. Let $\operatorname{im} A \leq \operatorname{im} B \leq \mathbb{K}^m$. Then, a complement $\langle C\rangle$ of $\operatorname{im} A$ in $\operatorname{im} B$, i.e., a space that satisfies $\langle C\rangle \oplus \operatorname{im} A = \operatorname{im} B$, can be obtained as described in Algorithm 1.[2] We denote this operation with $C = B\overline{\ominus}A$.

Algorithm 1 Complement of a subspace within another.

```
Input: Two matrices $A$ and $B$ with $m$ rows
Output: A matrix $A\overline{\ominus}B$ such that $\langle A\overline{\ominus}B\rangle \oplus \operatorname{im} B = \operatorname{im} A$
  $C \leftarrow$ A $m \times 0$ matrix
  for each column vector $b$ of $B$ do
    if $\operatorname{rk}(A \quad C \quad b) > \operatorname{rk}(A \quad C)$ then
      $C \leftarrow (C \quad b)$
    end if
  end for
  return $C$
```

3.3. Double complement

Lemma 3. *Let $\mathcal{C}$ be a finite-dimensional vector space and let $\mathcal{A}, \mathcal{B} \leq \mathcal{C}$ with $\dim \mathcal{A} \geq \dim \mathcal{B}$. Then, there exists a space $\mathcal{S} \leq \mathcal{C}$ such that*

$$\begin{cases} \mathcal{S} \oplus \mathcal{A} = \mathcal{C}, \\ \mathcal{S} \cap \mathcal{B} = \{0\}. \end{cases}$$

Proof. We first consider the case where $\mathcal{C} = \mathcal{A} + \mathcal{B}$. We denote with $\mathcal{P}$ (resp. $\mathcal{Q}$) a complement of $\mathcal{A} \cap \mathcal{B}$ in $\mathcal{A}$ (resp. in $\mathcal{B}$).

$$\begin{tikzcd}
 & \mathcal{C} = \mathcal{A} + \mathcal{B} \arrow[dl, dash] \arrow[dr, dash] & \\
\mathcal{A} \arrow[dr, dash, "\mathcal{P}"'] & & \mathcal{B} \arrow[dl, dash, "\mathcal{Q}"] \\
 & \mathcal{A} \cap \mathcal{B} &
\end{tikzcd}$$

We show first that $\mathcal{P} \cap \mathcal{Q} = \{0\}$. Let $v \in \mathcal{P} \cap \mathcal{Q}$. As $\mathcal{P} \leq \mathcal{A}$ and $\mathcal{Q} \leq \mathcal{B}$, we have $v \in \mathcal{A} \cap \mathcal{B}$. Therefore, $v \in \mathcal{P} \cap \mathcal{A} \cap \mathcal{B} = \{0\}$, as desired.

We now denote with $b = \{b_1, \ldots, b_p\}$ (resp. $b' = \{b'_1, \ldots, b'_q\}$) a basis of $\mathcal{P}$ (resp. $\mathcal{Q}$), implying $q \leq p$. Considering $w = \{b_1 + b'_1, \ldots, b_q + b'_q\}$, the following holds:

i) w is linear independent: if $\{\alpha_1, \ldots, \alpha_q\} \in \mathbb{K}^q$ is such that $\sum \alpha_i w_i = 0$, then $\sum \alpha_i b_i = -\sum \alpha_i b'_i$. As $\sum \alpha_i b_i \in \mathcal{P}$ and $\sum \alpha_i b'_i \in \mathcal{Q}$, it comes that $\sum \alpha_i b'_i = \sum \alpha_i b_i = 0$. It follows that for all i, $\alpha_i = 0$, yielding the result.

[2] This algorithm can be implemented to run in a cubic arithmetical time by keeping a reduced column echelon form of $(A \quad C)$, which makes it possible to check the condition within the loop in quadratic time.

ii) $\langle w\rangle \cap \mathcal{A} = \{0\}$: If $v \in \langle w\rangle \cap \mathcal{A}$, then there exists $\{\alpha_1,\dots,\alpha_q\} \in \mathbb{K}^q$ such that $v = \sum \alpha_i(b_i + b'_i) \in \mathcal{A}$. It implies $\sum \alpha_i b'_i \in \mathcal{A}$. As the left hand side is in $\mathcal{Q}$, it comes that $\sum \alpha_i b'_i = 0$. It follows that for all i, $\alpha_i = 0$, yielding the result.

iii) $\langle w\rangle \cap \mathcal{B} = \{0\}$: Same proof as above.

Then, since $(\mathcal{A}\cap\mathcal{B}) \oplus \mathcal{Q} = \mathcal{B}$, we have $\dim \mathcal{C} = \dim A + \dim B - \dim(\mathcal{A}\cap\mathcal{B}) = \dim A + q$. Therefore, $\dim\langle w\rangle = q = \dim \mathcal{C} - \dim \mathcal{A}$, and $\mathcal{S} = \langle w\rangle$ satisfies the desired conditions.

In the general case, where $\mathcal{C} > \mathcal{A} + \mathcal{B}$, we use the same method and simply add a complement $\mathcal{S}'$ of $\mathcal{A} + \mathcal{B}$ in $\mathcal{C}$ to the solution. □

Algorithm 2 uses the method in this proof to compute a basis of $\mathcal{S}$, given $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$. Note that if $\dim \mathcal{A} = \dim \mathcal{B}$, then $\mathcal{S}$ is a complement of both $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{C}$.

Algorithm 2 "Double complement" algorithm (Lemma 3).

```
Input: $A$, $B$ and $C$, such that $\operatorname{im} A, \operatorname{im} B \le \operatorname{im} C$ and $\operatorname{rk} A \ge \operatorname{rk} B$
Output: A matrix $S$ such that $\langle S\rangle \oplus \operatorname{im} A = \operatorname{im} C$ and $\langle S\rangle \cap \operatorname{im} B = \{0\}$.
  $P \leftarrow A \overline{\ominus} (A \overline{\cap} B)$
  $Q \leftarrow B \overline{\ominus} (A \overline{\cap} B)$
  $P' \leftarrow P$ truncated such that $P'$ and $Q$ have the same size
  $S' \leftarrow C \overline{\ominus} \begin{pmatrix} A & B \end{pmatrix}$
  return $\begin{pmatrix} S' & P' + Q \end{pmatrix}$
```

4. Proof of Theorem 1

We start with an auxiliary result that asserts that a decomposition of the form (3) is characterized by L.

Lemma 4. *Decomposition (3) exists if and only if L is chosen such that $P_4 - LP_2$ is non-singular. In this case,*

$$\operatorname{rk} R = \operatorname{rk}(P_3 - LP_1). \tag{23}$$

Proof. We have

$$\begin{pmatrix} I_m & \\ L & I_n \end{pmatrix}^{-1} P = \begin{pmatrix} P_1 & P_2 \\ P_3 - LP_1 & P_4 - LP_2 \end{pmatrix}. \tag{24}$$

This matrix can be uniquely decomposed as in (2) if and only if $P_4 - LP_2$ is non-singular, and we have the desired value for $\operatorname{rk} R$. □

Now we start the actual proof of Theorem 1. If we assume that decomposition (3) exists for P, then Lemma 4 yields

$$P = \begin{pmatrix} I_m & \\ L & I_n \end{pmatrix} \cdot \begin{pmatrix} P_1 - P_2(P_4 - LP_2)^{-1}(P_3 - LP_1) & P_2 \\ & P_4 - LP_2 \end{pmatrix} \cdot \begin{pmatrix} I_m & \\ (P_4 - LP_2)^{-1}(P_3 - LP_1) & I_n \end{pmatrix}. \tag{25}$$

It follows:

- (4) is obvious from (25).
- $\mathbb{K}^n = \operatorname{im}(P_4 - LP_2) \le \operatorname{im} P_4 + \operatorname{im} L$. Thus, $n \le p_4 + \operatorname{rk} L$, which yields (5).
- $\operatorname{im}(P_3 - LP_1) = (P_3 - LP_1)(\mathbb{K}^n) \ge (P_3 - LP_1)(\ker P_1) = P_3(\ker P_1)$. (6) now follows from (20) and (23).
- (7) is a direct computation:

$$\begin{aligned} p_3 &= \operatorname{rk}(P_3 - LP_1 + LP_1) \\ &\le \operatorname{rk}(P_3 - LP_1) + \operatorname{rk}(LP_1) \\ &\le \operatorname{rk} R + \operatorname{rk} L. \end{aligned}$$

5. Proof of Theorem 2, case $p_3 \le m + n - p_4 - p_1$

In this section, we provide an algorithm to construct an appropriate decomposition, in the case where $p_3 \le m + n - p_4 - p_1$ (Fig. 1 left). This means that, using Lemma 4, we have to build a matrix L that satisfies

$$\begin{cases} P_4 - LP_2 \text{ is non-singular,} \\ \operatorname{rk} L = n - p_4, \\ \operatorname{rk}(P_3 - LP_1) = m - p_1. \end{cases}$$

5.1. *Sufficient conditions*

We first derive a set of sufficient conditions that ensure that L satisfies the two following properties: $P_4 - LP_2$ is non-singular (Lemma 5) and $\operatorname{rk}(P_3 - LP_1) = m - p_1$ (Lemma 6).

Lemma 5. *If* $\operatorname{rk} L = n - p_4$ *and* $\operatorname{im} P_4 \oplus LP_2(\ker P_4) = \mathbb{K}^n$, *then* $P_4 - LP_2$ *is non-singular.*

Proof. We denote with $\mathcal{U}$ a complement of $\ker P_4$ in $\mathbb{K}^n$, i.e., $\mathbb{K}^n = \ker P_4 \oplus \mathcal{U}$. This implies $\operatorname{im} P_4 = P_4(\mathbb{K}^n) = P_4(\mathcal{U})$. Now, let $\operatorname{im} P_4 \oplus LP_2(\ker P_4) = \mathbb{K}^n$. Hence, $\dim LP_2(\ker P_4) = n - p_4 = \operatorname{rk} L$ from which we get $\operatorname{im} L = LP_2(\ker P_4)$. In particular, $LP_2(\mathcal{U}) \le \operatorname{im} L = LP_2(\ker P_4)$. Further,

$$\begin{aligned} \operatorname{im}(P_4 - LP_2) &= (P_4 - LP_2)(\mathcal{U} \oplus \ker P_4) \\ &= (P_4 - LP_2)(\mathcal{U}) + LP_2(\ker P_4). \end{aligned}$$

As $LP_2(\mathcal{U}) \leq LP_2(\ker P_4)$ and $\operatorname{im} P_4 \cap LP_2(\ker P_4) = \{0\}$, we have

$$\begin{aligned}\operatorname{im}(P_4 - LP_2) &= P_4(\mathcal{U}) + LP_2(\ker P_4)\\ &= \operatorname{im} P_4 + LP_2(\ker P_4)\\ &= \mathbb{K}^n,\end{aligned}$$

as desired. □

Lemma 6. *If, for every vector v of* $\operatorname{im} P_1$*, L satisfies $Lv \in P_3P_1^{-1}(\{v\})$, then* $\operatorname{rk}(P_3 - LP_1) = m - p_1$.

Proof. In the proof of Theorem 1 (Section 4) we already showed that $\operatorname{im}(P_3 - LP_1) \geq P_3(\ker P_1)$.

Let now $Lv \in P_3P_1^{-1}(\{v\})$ for all $v \in \operatorname{im} P_1$. If $u \in \mathbb{K}^m$, we have

$$\begin{aligned}(P_3 - LP_1)u &= P_3u - LP_1u\\ &\in P_3u - P_3P_1^{-1}(\{P_1u\})\\ &\leq P_3u - P_3(u + \ker P_1)\\ &\leq P_3(\ker P_1).\end{aligned}$$

Therefore, $\operatorname{im}(P_3 - LP_1) = P_3(\ker P_1)$. Thus, $\operatorname{rk} P_3 - LP_1 = m - p_1$. □

The following lemma summarizes the two previous results:

Lemma 7. *Let $\mathcal{Y}$ be a complement of* $\operatorname{im} P_4$ *and $\mathcal{T}$ a complement of $P_1(\ker P_3)$ in* $\operatorname{im} P_1$. *If*

$$\begin{cases}\operatorname{im} L = \mathcal{Y},\\ L \cdot P_2(\ker P_4) = \mathcal{Y},\\ L \cdot v \in P_3P_1^{-1}(\{v\}), \forall v \in \mathcal{T},\\ L \cdot P_1(\ker P_3) = \{0\},\end{cases}$$

then L is an optimal solution.[3]

5.2. Building L

We now build a matrix L that satisfies the previous set of sufficient conditions. For all v in a complement $\mathcal{T}$ of $P_1(\ker P_3)$, L has to satisfy $Lv \in P_3P_1^{-1}(\{v\})$. We first show

[3] The proposed set of sufficient conditions is stronger than necessary; if we replace the last condition with $L \cdot P_1(\ker P_3) \leq P_3(\ker P_1)$, if and only if holds.

that, given a suitable domain and image, it is possible to build a bijective linear mapping that satisfies this property.

Lemma 8. *Let $P_1(\ker P_3) \oplus \mathcal{T} = \operatorname{im} P_1$ and $P_3(\ker P_1) \oplus \mathcal{V} = \operatorname{im} P_3$. If we define the subspace*

$$\mathcal{F} = P_1^{-1}(\mathcal{T}) \cap P_3^{-1}(\mathcal{V}),$$

then the mapping $f : \mathcal{T} \to \mathcal{V}$, such that for all $v \in \mathcal{F}, f(P_1 v) = P_3 v$, is well defined and is an isomorphism.

Proof. We prove the lemma by first considering two functions f_1 and f_2 that are P_1 and P_3 restricted to $\mathcal{F}$ as shown in the diagram. We show that both are isomorphisms. Then $f = f_2 \circ f_1^{-1}$ is the desired function.

$$\begin{tikzcd}
\mathcal{T} \arrow[rr, "f"] & & \mathcal{V} \\
& \mathcal{F} \arrow[ul, "f_1 :\ x \mapsto P_1 x"] \arrow[ur, "f_2 :\ x \mapsto P_3 x"'] &
\end{tikzcd}$$

We begin with the surjectivity of f_1. Let $x \in \mathcal{T}$. As $\mathcal{T} \leq \operatorname{im} P_1$, there exists a vector v such that $P_1 v = x$. The coset $v + \ker P_1$ is obviously a subset of $P_1^{-1}(\mathcal{T})$. Additionally, its image under P_3, the coset $P_3(v + \ker P_1) = P_3 v + P_3 \ker P_1$ contains a unique representative $P_3 v_f$ in $\mathcal{V}$, since $\operatorname{im} P_3 = P_3(\ker P_1) \oplus \mathcal{V}$. Therefore, $v_f \in P_3^{-1}(\mathcal{V})$, and thus $v_f \in \mathcal{F}$ and $f_1(v_f) = x$, as desired.

We now prove that f_1 is injective. Let $v \in \ker f_1 \leq \mathcal{F}$. We have $P_3 v \in \mathcal{V}$. Since $v \in \ker P_1$, $P_3 v \in P_3 \ker P_1$. Since $P_3(\ker P_1) \cap \mathcal{V} = \{0\}$, $P_3 v = 0$ and thus $v \in \ker P_3$. Equation (12) shows that $v = 0$, as desired. Thus, f_1 is bijective.

The proof that $f_2 : \mathcal{F} \to \mathcal{V}, v \mapsto P_3 v$ is bijective is analogous. It follows that $f = f_2 \circ f_1^{-1}$ is the desired isomorphism. □

As explained below, we now build a matrix L that matches the conditions listed in Lemma 7. As they involve two spaces that may not be in a direct sum, $P_2(\ker P_4)$ and a complement of $P_1(\ker P_3)$ in $\operatorname{im} P_1$, some precautions must be taken.

We first construct the image $\mathcal{Y}$ of L. It must be a complement of $\operatorname{im} P_4$ and must contain a complement $\mathcal{Y}_1$ of $P_3(\ker P_1)$ in $\operatorname{im} P_3$. From $p_3 \leq m + n - p_4 - p_1$ we get $m - p_1 \geq p_4 + p_3 - n$ and thus $\dim(P_3(\ker P_1)) \geq \dim(\operatorname{im} P_4 \cap \operatorname{im} P_3)$ using (20) and (21). Therefore, we can use the Lemma 3 to build a space $\mathcal{Y}_1$ such that

$$\begin{cases} \mathcal{Y}_1 \oplus P_3(\ker P_1) = \operatorname{im} P_3, \\ \mathcal{Y}_1 \cap \operatorname{im} P_4 \cap \operatorname{im} P_3 = \{0\}. \end{cases}$$

We then complete $\mathcal{Y}_1$ to form a complement $\mathcal{Y}$ of $\operatorname{im} P_4$.

We now decompose $\mathbb{K}^m$ the following way:

$$\underbrace{\mathcal{X}_1 \oplus \mathcal{X}_2}_{P_2(\ker P_4)} \oplus \overbrace{\mathcal{X}_2 \oplus \mathcal{X}_3 \oplus P_1(\ker P_3)}^{\operatorname{im} P_1} \oplus \mathcal{X}_4 = \mathbb{K}^m.$$

We define $\mathcal{X}_2 = P_2(\ker P_4) \cap \operatorname{im} P_1$. $\mathcal{X}_2 \cap P_1(\ker P_3) = \{0\}$ according to equation (16). Then, we define $\mathcal{X}_3$ as a complement of $P_1(\ker P_3) \oplus \mathcal{X}_2$ in $\operatorname{im} P_1$ and $\mathcal{X}_1$ as a complement of $\mathcal{X}_2$ in $P_2(\ker P_4)$. $\mathcal{X}_4$ is defined as a complement of $\mathcal{X}_1 \oplus \mathcal{X}_2 \oplus \mathcal{X}_3 \oplus P_1(\ker P_3)$.

Finally, we build L through the associated mapping, itself defined using a direct sum of linear mappings defined on the following subspaces of $\mathbb{K}^m$:

- We use Lemma 8 to construct a bijective linear mapping f from $\mathcal{T} = \mathcal{X}_2 \oplus \mathcal{X}_3$ onto $\mathcal{V} = \mathcal{Y}_1$. By definition, for all $v \in \mathcal{T}$, f verifies $f(v) \in P_3 P_1^{-1}(\{v\})$. Furthermore, as f is bijective, its restriction on $\mathcal{X}_2$ is itself bijective onto $f(\mathcal{X}_2)$.
- We complete this bijective linear mapping with g, a bijective linear mapping between $\mathcal{X}_1$ and a complement $\mathcal{Y}_2$ of $f(\mathcal{X}_2)$ in $\mathcal{Y}$. Such a mapping exists as we have $\dim \mathcal{X}_1 - \dim \mathcal{Y}_2 = \dim \mathcal{X}_1 + \dim \mathcal{X}_2 - \dim \mathcal{Y} = \dim(P_2(\ker P_4)) - (n - p_4) = 0$. This way, the restriction of $f \oplus g$ on $\mathcal{X}_1 \oplus \mathcal{X}_2 = P_2(\ker P_4)$ is bijective onto $\mathcal{Y}$.
- We consider the mapping h that maps $P_1(\ker P_3) \oplus \mathcal{X}_4$ to $\{0\}$.

$$\begin{tikzcd}
\mathcal{X}_1 \arrow[d, "g"] & \mathcal{X}_2 \oplus \mathcal{X}_3 \arrow[d, "f"] & P_1(\ker P_3) \oplus \mathcal{X}_4 \arrow[d, "h"] \\
\mathcal{Y}_2 & \mathcal{Y}_1 & \{0\}
\end{tikzcd}$$

The matrix associated with the linear mapping $f \oplus g \oplus h$ satisfies all the conditions of Lemma 7, and is therefore an optimal solution.

This method is summarized in Algorithm 3, which allows us to construct a solution for Theorem 2. Its key part is the construction of a basis of $\mathcal{F}$, which uses a generalized pseudo-inverse $P_1^\dagger$ (resp. $P_3^\dagger$) of P_1 (resp. P_3), i.e., a matrix verifying $P_1 P_1^\dagger P_1 = P_1$ (resp. $P_3 P_3^\dagger P_3 = P_3$). This algorithm is a main contribution of this article.

Inspection of this algorithm shows that its arithmetic cost is $O((m+n)^3)$.

5.3. Example

We illustrate our algorithm with a concrete example. Motivated by our main application (Section 2) we choose as base field $\mathbb{K} = \mathbb{F}_2$. For $m = 4$ and $n = 3$, we consider the matrix

Algorithm 3 Constructing L (Theorem 2), case $p_3 \leq m+n-p_4-p_1$.

```
Input: $m$,$n$ and $P \in GL_{m+n}(\mathbb{K})$ such that $p_3 \leq m+n-p_4-p_1$
Output: $L$
  $Y_1 \leftarrow$ Algorithm 2 with $A = P_3 \cdot \overline{\ker}\, P_1$, $B = P_4 \overline{\sqcap} P_3$, $C = P_3$
  $Y \leftarrow (Y_1 \quad I_m \overline{\ominus} (Y_1 \quad P_4))$
  $X_2 \leftarrow (P_2 \cdot \overline{\ker}\, P_4) \overline{\sqcap} P_1$
  $X_3 \leftarrow P_1 \overline{\ominus} ((P_1 \cdot \ker P_3) \quad X_2))$
  $X_1 \leftarrow (P_2 \cdot \overline{\ker}\, P_4) \overline{\ominus} X_2$
  $X_4 \leftarrow I_m \overline{\ominus} (P_1 \quad P_2 \cdot \overline{\ker}\, P_4)$
  $F \leftarrow (\overline{\ker}\, P_1 \quad P_1^{\dagger} \cdot (X_2 \quad X_3)) \overline{\sqcap} (\overline{\ker}\, P_3 \quad P_3^{\dagger} \cdot Y_1)$
  $Y_2 \leftarrow Y \overline{\ominus} (P_3 \cdot ((\overline{\ker}\, P_1 \quad P_1^{\dagger} \cdot X_2) \overline{\sqcap} F))$
  $L_R \leftarrow (P_1 \cdot F \quad X_1 \quad P_1 \cdot \overline{\ker}\, P_3 \quad X_4)$
  $L_L \leftarrow (P_3 \cdot F \quad Y_2 \quad Z)$, where $Z$ is a zero filled matrix such that $L_L$ has the same number of columns as $L_R$
  return $L_L \cdot L_R^{-1}$
```

$$P = \begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix} = \left(\begin{array}{cccc|ccc} 1 & 1 & & & 1 & & 1 \\ & & 1 & & & 1 & \\ 1 & 1 & & 1 & 1 & 1 & 1 \\ & & & & & 1 & 1 \\ \hline 1 & & 1 & 1 & & & \\ 1 & 1 & & & 1 & 1 & 1 \\ & 1 & & & & & \end{array}\right).$$

We observe $p_3 = 3 \leq m+n-p_4-p_1 = 4+3-1-3$. Therefore, we can use Algorithm 3 to compute a suitable L.

The first step is to compute Y_1. We have

$$P_3 \cdot \overline{\ker}\, P_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \text{ and } P_4 \overline{\sqcap} P_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

Using Algorithm 2, we get

$$Y_1 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix}.$$

Then, we complete it to form a complement of $\overline{\operatorname{im}}\, P_4$:

$$Y = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}.$$

The next step computes the different domains:

$$X_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix},\ X_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix},\ X_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ and } X_4 = ().$$

To compute F, we need pseudo-inverses of P_1 and P_3:

$$P_1^\dagger = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix} \text{ and } P_3^\dagger = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

We then obtain

$$F = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Then, we compute Y_2:

$$Y_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.$$

Now we can compute L. With

$$L_R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, L_L = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

we get

$$L = L_L \cdot L_R^{-1} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

The final decomposition is now obtained using (25):

$$P = \left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline 1 & 1 & 1 & 1 & 1 & & \\ 1 & & & & & 1 & \\ 1 & & & & & & 1 \end{array}\right) \cdot \left(\begin{array}{cccc|ccc} & 1 & & & 1 & & 1 \\ & & 1 & & & 1 & \\ & 1 & & 1 & 1 & 1 & 1 \\ 1 & & & & & 1 & 1 \\ \hline & & & & & 1 & 1 \\ & & & & & 1 & \\ & & & & 1 & & 1 \end{array}\right) \cdot$$

$$\left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline & & & & 1 & & \\ & & & & & 1 & \\ 1 & & & & & & 1 \end{array}\right).$$

This decomposition satisfies $\operatorname{rk} L = 2$ and $\operatorname{rk} R = 1$, thus matching the bounds of Theorem 1.

If we consider the application presented in Section 2, this decomposition provides a way to implement in hardware the permutation associated with P on 128 elements, arriving in chunks of 8 during 16 cycles. This yields an implementation consisting of a permutation network of 4 2×2-switches, followed by a block of 8 RAM banks, followed by another permutation network with 8 2×2-switches.

6. Proof of Theorem 2, case $p_3 \geq m + n - p_4 - p_1$

In this case, the third inequality in Theorem 1 is restrictive (Fig. 1 right). Using again Lemma 4, we have to build a matrix L satisfying

$$\begin{cases} P_4 - LP_2 \text{ is non-singular,} \\ \operatorname{rk} L = n - p_4, \\ \operatorname{rk}(P_3 - LP_1) = p_4 + p_3 - n. \end{cases}$$

As in the previous section, we will first provide a set of sufficient conditions for L and then build it.

6.1. *Sufficient conditions*

The set of conditions that we will derive in this subsection will be slightly more complex than in the previous section, as we cannot reach the intrinsic bound of $\operatorname{rk}(P_3 - LP_1)$. Particularly, we cannot use Lemma 6 directly.

Lemma 9. *If $\mathcal{W}$ is such that $\mathcal{W} \oplus \operatorname{im} P_4 = \mathbb{K}^n$ and $\mathcal{T}$ is such that*

$$\begin{cases} \mathcal{T} \cap P_1(\ker P_3) = \{0\}, \\ \mathcal{T} \leq \operatorname{im} P_1, \\ \dim \mathcal{T} = n - p_4, \end{cases}$$

then if L satisfies

$$\begin{cases} \operatorname{im} L = \mathcal{W}, \\ L \cdot P_2(\ker P_4) = \mathcal{W}, \\ L \cdot v \in P_3 P_1^{-1}(\{v\}),\ \forall v \in \mathcal{T}, \\ L \cdot P_1(\ker P_3) = \{0\}, \end{cases}$$

then L is a solution[4] *that verifies* $\operatorname{rk} L = n - p_4$ *and* $\operatorname{rk}(P_3 - LP_1) = p_4 + p_3 - n$.

[4] If we replace the last condition with $L \cdot P_1(\ker P_3) \leq P_3(\ker P_1)$, this set of conditions is actually equivalent to having an optimal solution L that satisfies $\operatorname{rk} L = n - p_4$.

Proof. Let L be a matrix that satisfies the conditions above. Using Lemma 5 as before, we get that $\operatorname{rk} L = n - p_4$ and $P_4 - LP_2$ invertible.

Now, with the definition of $\mathcal{T}$, we can define a dimension $p_4 + p_3 + p_1 - m - n$ space $\mathcal{T}'$ such that $\operatorname{im} P_1 = P_1(\ker P_3) \oplus \mathcal{T} \oplus \mathcal{T}'$. Then, we define a matrix L' such that

$$\begin{cases} L' \cdot v \in Lv - P_3 P_1^{-1}(\{v\}), \text{ for all } v \in \mathcal{T}', \\ L' \cdot v = 0, \text{ for all } v \text{ in } P_1(\ker P_3) \oplus \mathcal{T}, \\ L' \cdot v = 0, \text{ for all } v \text{ in a complement of } \operatorname{im} P_1, \end{cases}$$

L' is therefore a rank $p_4 + p_3 + p_1 - m - n$ matrix such that for all $v \in \operatorname{im} P_1, (L - L')v \in P_3 P_1^{-1}(\{v\})$. We apply Lemma 6 on $L - L'$ and get

$$\begin{aligned} \operatorname{rk}(P_3 - LP_1) &= \operatorname{rk}(P_3 - (L - L')P_1 - L'P_1) \\ &\leq \operatorname{rk}(P_3 - (L - L')P_1) + \operatorname{rk}(L'P_1) \\ &\leq \dim P_3(\ker P_1) + \operatorname{rk} L' \\ &\leq p_4 + p_3 - n, \end{aligned}$$

as desired. □

6.2. Building L

We will build a matrix L that matches the conditions listed in Lemma 9. As before, we consider the image $\mathcal{Y}$ of L first. We will design it such that it is a complement of $\operatorname{im} P_4$, and that is contained in a complement $\mathcal{Y}'$ of $P_3(\ker P_1)$ in $\operatorname{im} P_3$. Using $p_3 \geq m+n-p_4-p_1$ with (20) and (21) we get $\dim(P_3 \ker P_1) \leq \dim(\operatorname{im} P_4 \cap \operatorname{im} P_3)$. With Lemma 3, we can construct a space $\mathcal{Y}$ that satisfies

$$\begin{cases} \mathcal{Y} \oplus (\operatorname{im} P_4 \cap \operatorname{im} P_3) = \operatorname{im} P_3, \\ \mathcal{Y} \cap P_3(\ker P_1) = \{0\}. \end{cases}$$

This space satisfies $\mathcal{Y} \oplus \operatorname{im} P_4 = \operatorname{im} P_3 + \operatorname{im} P_4 = \mathbb{K}^n$ according to (13), and can be completed to a complement $\mathcal{Y}'$ of $P_3(\ker P_1)$ in $\operatorname{im} P_3$. Note that we will use $\mathcal{Y}'$ only to define f; the image of L will be $\mathcal{Y}$.

Now, as before, we build L through the associated mapping, itself defined using a direct sum of linear mappings defined on the same subspaces of $\mathbb{K}^m$ as in Section 5.2.

- We use Lemma 8 to construct a first bijective linear mapping f' between $\mathcal{T}' = \mathcal{X}_2 \oplus \mathcal{X}_3$ and $\mathcal{Y}'$. As f' is bijective, we can define $\mathcal{T} = f'^{-1}(\mathcal{Y})$ and $f = f'|_{\mathcal{T}}$. Thus, $\mathcal{T}$ satisfies the properties in Lemma 9 and L the condition for all $v \in \mathcal{T}, Lv \in P_3 P_1^{-1}(\{v\})$.
- Then, we consider a complement $\mathcal{X}_1'$ of $\mathcal{T} \cap \mathcal{X}_2$ in $P_2(\ker P_4)$, a complement $\mathcal{Y}_2'$ of $f(\mathcal{T} \cap \mathcal{X}_2)$ in $\mathcal{Y}$ and a bijective linear mapping g between $\mathcal{X}_1'$ and $\mathcal{Y}_2'$. This way, the restriction of $f \oplus g$ on $P_2(\ker P_4)$ is bijective onto $\mathcal{Y}$.

The rest of the algorithm in similar to the previous case:

- We consider the mapping h that maps $P_1(\ker P_3)$ to $\{0\}$.
- To complete the definition of L, we take a mapping h' between a complement $\mathcal{X}_4'$ of $\mathcal{X}_1' \oplus \mathcal{T} \oplus P_1(\ker P_3)$ and $\{0\}$.

$$\begin{tikzcd}
\mathcal{X}_1' \arrow[d, "g"] & \mathcal{T} = f'^{-1}(\mathcal{Y}) \arrow[d, "f"] & P_1(\ker P_3) \arrow[d, "h"] & \mathcal{X}_4' \arrow[dl, "h'"] \\
\mathcal{Y}_2' & \mathcal{Y} & \{0\} &
\end{tikzcd}$$

The matrix associated with the mapping $f \oplus g \oplus h \oplus h'$ satisfies all the conditions of Lemma 9, and is therefore an optimal solution.

Algorithm 4 summarizes this method, and allows to construct a solution for Theorem 2, in the case where $p_3 > m + n - p_4 - p_1$. This algorithm is a main contribution of this article.

Algorithm 4 Constructing L (Theorem 2), case $p_3 > m + n - p_4 - p_1$.

```
Input: $m$,$n$, $P \in GL_{m+n}(\mathbb{K})$ such that $p_3 > m + n - p_4 - p_1$
Output: $L$
  $Y \leftarrow$ Algorithm 2 with $A = P_4 \overline{\cap} P_3$, $B = P_3 \cdot \overline{\ker} P_1$ and $C = P_3$
  $X_2 \leftarrow (P_2 \cdot \overline{\ker} P_4) \overline{\cap} P_1$
  $X_3 \leftarrow P_1 \overline{\ominus} \left( P_1 \cdot \overline{\ker} P_3 \quad X_2 \right)$
  $F \leftarrow \left( \overline{\ker} P_1 \quad P_1^\dagger \cdot ( X_2 \quad X_3 ) \right) \overline{\cap} \left( \overline{\ker} P_3 \quad P_3^\dagger \cdot Y \right)$
  $X_1' \leftarrow (P_2 \cdot \ker P_4) \overline{\ominus} ((P_1 \cdot F) \overline{\cap} X_2)$
  $X_4 \leftarrow I_m \overline{\ominus} \left( X_1' \quad P_1 \cdot F \quad P_1 \cdot \overline{\ker} P_3 \right)$
  $Y_2' \leftarrow Y \overline{\ominus} (P_3 \cdot (F \overline{\cap} \left( P_1^\dagger \cdot X_2 \quad \overline{\ker} P_1 \right)))$
  $L_R \leftarrow \left( P_1 \cdot F \quad X_1' \quad P_1 \cdot \overline{\ker} P_3 \quad X_4 \right)$
  $L_L \leftarrow \left( P_3 \cdot F \quad Y_2' \quad Z \right)$, where $Z$ is a zero filled matrix such that $L_L$ has the same number of columns
  as $L_R$
  return $L_L \cdot L_R^{-1}$
```

As in the previous case, this algorithm has an arithmetic cost cubic in $m + n$.

6.3. Example

We now consider, for $\mathbb{K} = \mathbb{F}_2$, $m = 4$ and $n = 3$, the matrix

$$P = \begin{pmatrix} P_1 & P_2 \\ P_3 & P_4 \end{pmatrix} = \left(\begin{array}{cccc|ccc} & 1 & 1 & 1 & 1 & & \\ 1 & & & 1 & & 1 & 1 \\ & 1 & 1 & 1 & & 1 & 1 \\ 1 & 1 & & 1 & & 1 & 1 \\ \hline 1 & & & 1 & & 1 & \\ & & & 1 & & 1 & \\ 1 & & 1 & 1 & 1 & 1 & \end{array} \right).$$

We observe $p_3 = 3 > m + n - p_4 - p_1 = 4 + 3 - 2 - 3$. Therefore, we use Algorithm 4 to compute a suitable L.

The first step is to compute Y. We have

$$P_3 \cdot \overline{\ker}\, P_1 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \text{ and } P_4 \overline{\cap} P_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

Using Algorithm 2, we get

$$Y_1 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix}.$$

Then, we complete it to form a complement of $\operatorname{im} P_4$:

$$Y = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The next step computes the different domains:

$$X_2 = () \text{ and } X_3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

To compute F, we need pseudo-inverses of P_1 and P_3:

$$P_1^\dagger = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \text{ and } P_3^\dagger = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

and get

$$F = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}.$$

Next we compute the remaining subspaces that depend on F:

$$X_1' = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}, X_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \text{ and } Y_2' = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

Now we can compute L. With

$$L_R = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \ L_L = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix},$$

we get

$$L = L_L \cdot L_R^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The final decomposition is obtained using (25):

$$P = \left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline & 1 & & & & & \\ & & & & & & \\ & 1 & & & & & \end{array}\right) \cdot \left(\begin{array}{cccc|ccc} & 1 & & 1 & 1 & & \\ 1 & & & & & 1 & 1 \\ & 1 & 1 & & & 1 & 1 \\ 1 & 1 & & & & 1 & 1 \\ \hline & & & & & & 1 \\ & & & & & 1 & \\ & & & & 1 & & 1 \end{array}\right) \cdot$$
$$\left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline & & 1 & & 1 & & \\ & & & 1 & & 1 & \\ & & & & & & 1 \end{array}\right).$$

This decomposition satisfies $\operatorname{rk} L = 1$ and $\operatorname{rk} R = 2$, thus matching the bounds of Theorem 1.

As before, if we consider the application of Section 2. The decomposition shows that we can implement in hardware the permutation associated with P on 128 elements, arriving in chunks of 8 during 16 cycles through a permutation network of 8 2×2-switches, followed by a block of 8 RAM banks, followed by another permutation network with 4 2×2-switches.

7. Rank exchange

The solution built in Section 6.2 satisfies $\operatorname{rk} L = n - p_4$ and $\operatorname{rk}(P_3 - LP_1) = p_4 + p_3 - n$. In this section, we will show that it is possible to construct a solution for all possible pairs $(\operatorname{rk} L, \operatorname{rk}(P_3 - LP_1))$ matching the bounds in Theorem 1. First, we will construct a rank 1 matrix L' that will trade a rank of L for a rank of $P_3 - LP_1$ (i.e., $\operatorname{rk}(L + L') = 1 + \operatorname{rk} L$,

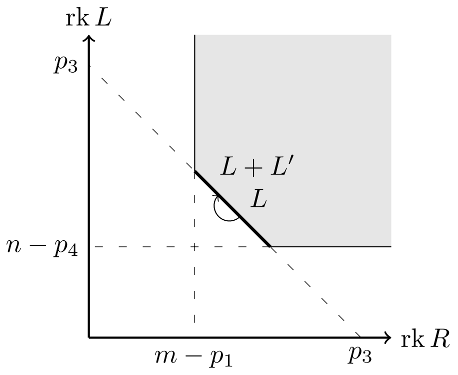


Fig. 5. L' trades a rank of L for a rank of R on the associated decomposition.

$\operatorname{rk}(P_3 - (L+L')P_1) = \operatorname{rk}(P_3 - LP_1) - 1$ and $P_4 - (L+L')P_2$ is non-singular) (see Fig. 5). This method can then be applied several times, until $\operatorname{rk}(P_3 - LP_1)$ reaches its own bound, $m - p_1$.

We assume that L satisfies the following conditions:

$$\begin{cases} P_4 - LP_2 \text{ is non-singular,} \\ \operatorname{rk} L + \operatorname{rk}(P_3 - LP_1) = p_3, \\ \operatorname{rk}(P_3 - LP_1) > m - p_1. \end{cases}$$

As in the previous sections, we first formulate sufficient conditions on L', before building it.

7.1. Sufficient conditions

We now define $C = P_1 - P_2(P_4 - LP_2)^{-1}(P_3 - LP_1)$.

Lemma 10. *If $z \in \mathbb{K}^m$ satisfies $z \notin \ker(P_3 - LP_1) + \ker P_1$ and L' satisfies*

$$\begin{cases} \operatorname{rk} L' = 1, \\ L'P_1 \ker(P_3 - LP_1) = \{0\}, \\ L'P_1 z = (P_3 - LP_1)z, \\ L'Cz \neq 0. \end{cases}$$

Then $L + L'$ is an optimal solution to our problem that satisfies $\operatorname{rk}(P_3 - (L+L')P_1) = \operatorname{rk}(P_3 - LP_1) - 1$.

Proof. We first prove that $P_4 - (L+L')P_2 = (I - L'P_2(P_4 - LP_2)^{-1})(P_4 - LP_2)$ is non-singular. Let $x \in \ker(I - L'P_2(P_4 - LP_2)^{-1})$. x satisfies

$$x - L'P_2(P_4 - LP_2)^{-1}x = 0.$$

Therefore, $x \in \operatorname{im} L'$. As $\operatorname{rk} L' = 1$, $\exists \lambda \in \mathbb{K}$, $x = \lambda(P_3 - LP_1)z = \lambda L' P_1 z$. It comes that

$$\lambda L' P_1 z - \lambda L' P_2 (P_4 - LP_2)^{-1}(P_3 - LP_1)z = 0.$$

Finally, $\lambda L' C z = 0$, which implies, as $L' C z \neq 0$, $\lambda = 0$, as desired.

We now prove that $\operatorname{rk}(P_3 - (L+L')P_1) = \operatorname{rk}(P_3 - LP_1) - 1$. We have already $\ker(P_3 - (L+L')P_1) \leq \ker(P_3 - LP_1)$ as $L'P_1 \ker(P_3 - LP_1) = \{0\}$. We also have $(P_3 - (L'+L)P_1)z = 0$. As $z \notin \ker(P_3 - LP_1) + \ker P_1$, $\ker(P_3 - (L+L')P_1) \geq \ker(P_3 - LP_1) \oplus \langle z \rangle$. Therefore,

$$\dim \ker(P_3 - (L+L')P_1) \geq 1 + \dim \ker(P_3 - LP_1),$$

as desired. □

7.2. Building L'

Lemma 11. $\ker(P_3 - LP_1) \cap \ker P_1 = \{0\}$.

Proof. This is a consequence of (24): the block column $\begin{pmatrix} P_1 \\ P_3 - LP_1 \end{pmatrix}$ has full rank. □

Thus, we have $\dim(\ker(P_3 - LP_1) \oplus \ker P_1) = 2m - p_1 - \operatorname{rk}(P_3 - LP_1) < m$. Decomposition (25) shows that C is non-singular, and using Lemma 11, we have $\dim C^{-1}P_1 \ker(P_3 - LP_1) = \dim P_1 \ker(P_3 - LP_1) = \dim \ker(P_3 - LP_1) = m - \operatorname{rk}(P_3 - LP_1)$. Using Lemma 3, we can build a space $\mathcal{Z}$ such that

$$\begin{cases} \mathcal{Z} \oplus \ker(P_3 - LP_1) \oplus \ker P_1 = \mathbb{K}^m, \\ \mathcal{Z} \cap C^{-1}P_1 \ker(P_3 - LP_1) = \{0\}. \end{cases}$$

We can now pick a nonzero element $z \in \mathcal{Z}$ and build a corresponding L':

- If $Cz \in P_1 \ker(P_3 - LP_1) \oplus \langle P_1 z \rangle$: We take a complement $\mathcal{A}$ of $P_1 \ker(P_3 - LP_1) \oplus \langle P_1 z \rangle$ and build L' such that

$$\begin{cases} L' P_1 z = (P_3 - LP_1)z, \\ L'(P_1 \ker(P_3 - LP_1) \oplus \mathcal{A}) = \{0\}. \end{cases}$$

  We have $L'Cz \neq 0$. In fact, Cz can be uniquely decomposed in the form $k + \lambda P_1 z$, where $k \in P_1 \ker(P_3 - LP_1)$ and $\lambda \in \mathbb{K}$. As $z \notin C^{-1}P_1 \ker(P_3 - LP_1)$, $\lambda \neq 0$. Then, $L'Cz = L'k + L'\lambda P_1 z = 0 + \lambda(P_3 - LP_1)z \neq 0$.

- If $Cz \notin P_1 \ker(P_3 - LP_1) \oplus \langle P_1 z\rangle$: The vector $a = Cz - P_1 z$ is outside of $P_1 \ker(P_3 - LP_1) \oplus \langle P_1 z\rangle$. Therefore, it is possible to build a complement $\mathcal{A}$ of $P_1 \ker(P_3 - LP_1) \oplus \langle P_1 z\rangle$ that contains a. Then, we build L' as before:

$$\begin{cases} L'P_1 z = (P_3 - LP_1)z, \\ L'(P_1 \ker(P_3 - LP_1) \oplus \mathcal{A}) = \{0\}. \end{cases}$$

  As in the previous case, we have $L'Cz = L'a + L'P_1 z = 0 + (P_3 - LP_1)z \neq 0$.

In both cases, the matrix L' we built satisfies the conditions of Lemma 10. Therefore, $L + L'$ is the desired solution.

Algorithm 5 summarizes this method, and allows to build a new optimal solution from a pre-existing one, with a different trade-off. This algorithm is a main contribution of this article.

Algorithm 5 Exchanging ranks between L and R (Theorem 3).

```
Input: $P$ and a solution $L$ such that $\mathrm{rk}(P_3 - LP_1) > m - p_1$
Output: A new optimal solution $L$ with a rank incremented by 1
  $K \leftarrow \overline{\ker}(P_3 - LP_1)$
  $C \leftarrow P_1 - P_2(P_4 - LP_2)^{-1}(P_3 - LP_1)$
  $Z \leftarrow$ Algorithm 2 with $A = \begin{pmatrix} K & \overline{\ker}\, P_1 \end{pmatrix}$, $B = C^{-1}P_1K$ and $C = I_m$
  $z \leftarrow$ first column of $Z$
  if $Cz \in \begin{pmatrix} P_1 \cdot K & P_1 z \end{pmatrix}$ then
    $A \leftarrow I_m \overline{\ominus} P_1 \cdot \begin{pmatrix} K & z \end{pmatrix}$
  else
    $a \leftarrow (C - P_1)z$
    $A \leftarrow \begin{pmatrix} I_m \overline{\ominus} \begin{pmatrix} P_1K & P_1 z & a \end{pmatrix} & a \end{pmatrix}$
  end if
  $L'_R \leftarrow \begin{pmatrix} P_1 z & P_1 K & A \end{pmatrix}$
  $L'_L \leftarrow \begin{pmatrix} (P_3 - LP_1)z & F \end{pmatrix}$, where $F$ is a zero filled matrix such that $L'_L$ has the same number of columns
  as $L'_R$
  return $L + L'_L \cdot L'^{-1}_R$
```

7.3. Example

To illustrate Algorithm 5, we continue the example of Section 6.3, and the matrix L that we found. We have

$$K = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \text{ and } C = \begin{pmatrix} & 1 & & 1 \\ 1 & & & \\ & 1 & 1 & \\ 1 & 1 & & \end{pmatrix}.$$

Using Algorithm 2, we get

$$Z = z = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

As $Cz = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ is not included in $(P_1K \quad P_1z) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, we compute A as a complement of $(P_1K \quad P_1z)$ that contains $a = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$:

$$\mathcal{A} = \left\langle \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right\rangle.$$

Now, we compute L', using

$$L'_R = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix},\ L'_L = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},\ \text{and } L' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}.$$

Finally, we get the new decomposition, using, as usual, Equation (25):

$$P = \left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline & 1 & & & 1 & & \\ & & & & & 1 & \\ & & 1 & 1 & & & 1 \end{array}\right) \cdot \left(\begin{array}{cccc|ccc} & 1 & 1 & 1 & 1 & & \\ 1 & & & & & 1 & 1 \\ & 1 & 1 & & & 1 & 1 \\ 1 & 1 & & & & 1 & 1 \\ \hline & & & & & & 1 \\ & & & & & 1 & \\ & & & & 1 & 1 & \end{array}\right) \cdot$$
$$\left(\begin{array}{cccc|ccc} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ \hline & & & & 1 & & \\ & & & 1 & & 1 & \\ & & & & & & 1 \end{array}\right).$$

As expected, the left off-diagonal rank has increased by one, while the right one has decreased by one. The two different decompositions that we now have for P cover all the possible tradeoffs that minimize the off-diagonal ranks.

8. Conclusion

In this paper, we introduced a novel block matrix decomposition that generalizes the classical block-LU factorization. A 2×2-blocked invertible matrix is decomposed into a product of three matrices: lower block unitriangular, upper block triangular, and lower block unitriangular matrix, such that the sum of the off-diagonal ranks are minimal. We provided an algorithm that computes an optimal solution with an asymptotic number of operations cubic in the matrix size. We note that we implemented the algorithm for finite fields, for rational numbers, for Gaussian rational numbers and for exact real arithmetic for validation. For a floating point implementation, numerical issues may arise.

The origin of the considered decomposition, as we explained, is in the design of optimal circuits for a certain class of streaming permutations that are very relevant in practice. However, we believe that because of its simple and natural structure, the matrix decomposition is also of pure mathematical interest. Specifically, it would be interesting to investigate if the proposed decomposition is a special case of a more general problem that involves, for example, finer block structures.

Acknowledgements

We would like to thank Jiří Matoušek and Paolo Bientinesi for helpful discussions. We are very sad about Jiří's untimely death and wish there would have been more time to collaborate.

References

[1] E. Chow, Y. Saad, Approximate inverse techniques for block-partitioned matrices, SIAM J. Sci. Comput. 18 (6) (1997) 1657–1675, http://dx.doi.org/10.1137/S1064827595281575.
[2] J.W. Demmel, N.J. Higham, R.S. Schreiber, Block LU factorization, Numer. Linear Algebra Appl. 2 (2) (1992) 173–190.
[3] V. Pan, Structured Matrices and Polynomials: Unified Superfast Algorithms, Springer Science & Business Media, 2001.
[4] F. Zhang, The Schur Complement and its Applications, vol. 4, Springer, 2006.
[5] R.W. Cottle, Manifestations of the Schur complement, Linear Algebra Appl. 8 (3) (1974) 189–211.
[6] D. Carlson, E. Haynsworth, T. Markham, A generalization of the Schur complement by means of the Moore–Penrose inverse, SIAM J. Appl. Math. 26 (1) (1974) 169–175, http://dx.doi.org/10.1137/0126013.
[7] M. Püschel, P.A. Milder, J.C. Hoe, Permuting streaming data using RAMs, J. ACM 56 (2) (2009) 10:1–10:34.
[8] R. Tolimieri, M. An, C. Lu, Algorithms for Discrete Fourier Transforms and Convolution, 2nd edition, Springer, 1997.
[9] K.K. Parhi, Systematic synthesis of DSP data format converters using life-time analysis and forward-backward register allocation, IEEE Trans. Circuits Syst. II. Analog Digit. Signal Process. 39 (7) (1992) 423–440, http://dx.doi.org/10.1109/82.160168.
[10] F. Serre, T. Holenstein, M. Püschel, Optimal circuits for streamed linear permutations using RAM, in: International Symposium on Field-Programmable Gate Arrays (FPGA), ACM, 2016, pp. 215–223.
[11] M.C. Pease, The indirect binary n-cube microprocessor array, IEEE Trans. Comput. 26 (5) (1977) 458–473.
[12] J. Lenfant, S. Tahé, Permuting data with the Omega network, Acta Inform. 21 (6) (1985) 629–641.