Fast Computation of the Rank Profile Matrix and the Generalized Bruhat Decomposition

Jean-Guillaume Dumas, Clément Pernet, Ziad Sultan

Fast Computation of the Rank Profile Matrix and the Generalized Bruhat Decomposition

Jean-Guillaume Dumas

Université Grenoble Alpes, Laboratoire LJK, umr CNRS, BP53X, 51, av. des Mathématiques, F38041 Grenoble, France

Clément Pernet

Université Grenoble Alpes, Laboratoire de l'Informatique du Parallélisme, Université de Lyon, France.

Ziad Sultan

Université Grenoble Alpes, Laboratoire LJK and LIG, Inria, CNRS, Inovallée, 655, av. de l'Europe, F38334 St Ismier Cedex, France

Abstract

The row (resp. column) rank profile of a matrix describes the stair-case shape of its row (resp. column) echelon form. We here propose a new matrix invariant, the rank profile matrix, summarizing all information on the row and column rank profiles of all the leading sub-matrices. We show that this normal form exists and is unique over any ring, provided that the notion of McCoy's rank is used, in the presence of zero divisors. We then explore the conditions for a Gaussian elimination algorithm to compute all or part of this invariant, through the corresponding PLUQ decomposition. This enlarges the set of known Elimination variants that compute row or column rank profiles. As a consequence a new Crout base case variant significantly improves the practical efficiency of previously known implementations over a finite field. With matrices of very small rank, we also generalize the techniques of Storjohann and Yang to the computation of the rank profile matrix, achieving an $(r^\omega + mn)^{1+o(1)}$ time complexity for an $m \times n$ matrix of rank r, where ω is the exponent of matrix multiplication. Finally, by give connections to the Bruhat decomposition, and several of its variants and generalizations. Thus, our algorithmic improvements for the PLUQ factorization, and their implementations, directly apply to these decompositions. In particular, we show how a PLUQ decomposition revealing the rank profile matrix also reveals both a row and a column echelon form of the input matrix or of any of its leading sub-matrices, by a simple post-processing made of row and column permutations.

Key words: Gaussian elimination, Rank profile, Echelon form, PLUQ decomposition, Bruhat decomposition, McCoy's rank.

Contents

1. Introduction

Triangular matrix decompositions are widely used in computational linear algebra. Besides solving linear systems of equations, they are also used to compute other objects more specific to exact arithmetic: computing the rank, sampling a vector from the null-space, computing echelon forms and rank profiles.

The *row rank profile* (resp. *column rank profile*) of an $m \times n$ matrix A with rank r, denoted by RowRP(A) (resp. ColRP(A)), is the lexicographically smallest sequence of r indices of linearly independent rows (resp. columns) of A. An $m \times n$ matrix has generic row (resp. column) rank profile if its row (resp. column) rank profile is $(1, .., r)$. Lastly, an $m \times n$ matrix has generic rank profile if its r first leading principal minors are non-zero. Note that if a matrix has generic rank profile, then its row and column rank profiles are

⋆ This research was partly supported by the HPAC project of the French Agence Nationale de la Recherche (ANR 11 BS02 013).

Email addresses: jean-guillaume.dumas@imag.fr (Jean-Guillaume Dumas), clement.pernet@imag.fr (Clément Pernet), mailto:ziad.sultan@imag.fr (Ziad Sultan).

URLs: http://www-ljk.imag.fr/membres/Jean-Guillaume.Dumas/ (Jean-Guillaume Dumas), http://lig-membres.imag.fr/pernet/ (Clément Pernet), http://moais.imag.fr/membres/ziad.sultan (Ziad Sultan).

generic, but the converse is false: the matrix $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ does not have generic rank profile even if its row and column rank profiles are generic. The row support (resp. column support) of a matrix A, denoted by RowSupp(A) (resp. ColSupp(A)), is the subset of indices of its non-zero rows (resp. columns).

We recall that the row echelon form of an $m \times n$ matrix A is an upper triangular matrix $E = TA$, for a non-singular matrix T, with the zero rows of E at the bottom and the non-zero rows in stair-case shape: $\min\{j : a_{i,j} \neq 0\} < \min\{j : a_{i+1,j} \neq 0\}$. As T is non singular, the column rank profile of A is that of E, and therefore corresponds to the column indices of the leading elements in the staircase. Similarly the row rank profile of A is composed of the row indices of the leading elements in the staircase of the column echelon form of A.

Rank profile and triangular matrix decompositions. The rank profiles of a matrix and the triangular matrix decomposition obtained by Gaussian elimination are strongly related. The elimination of matrices with arbitrary rank profiles gives rise to several matrix factorizations and many algorithmic variants. In numerical linear algebra one often uses the PLUQ decomposition, with P and Q permutation matrices, L a lower unit triangular matrix and U an upper triangular matrix. The LSP and LQUP variants of [1] are used to reduce the complexity of rank deficient Gaussian elimination to that of matrix multiplication. Many other algorithmic decompositions exist allowing fraction free computations [2], in-place computations [3,4] or sub-cubic rank-sensitive time complexity [5,4]. The reader may refer to [4] for a detailed comparison between these matrix factorizations, and further details on the CUP (resp. PLE) variants, revealing the row (resp. column) rank profiles. All these algorithms, together with the schoolbook Gaussian elimination algorithm share the property that, for a row rank profile computation, the pivot search processes rows in order, and searches a pivot in all possible column position before declaring the row linearly dependent with the previous ones. As a consequence, blocking is limited to only one dimension (in this case the row dimension) leading to slab algorithms [6] operating on rectangular blocks of unbalanced dimensions. This reduces the data locality of the algorithm, and therefore penalizes the efficiency of implementations in practice. In parallel, this blocking also puts more constrains on the dependency between tasks [7].

Contribution with respect to the state of the art. In [8] we proposed a first Gaussian elimination algorithm, with a recursive splitting of both row and column dimensions, which simultaneously computes the row and column rank profile while preserving the sub-cubic rank-sensitive time complexity and keeping the computation in-place. It showed that slab blocking is not a necessary condition for a Gaussian elimination to reveal rank profiles. Consequently, we have further analyzed the conditions on the pivoting that reveal the rank profiles in [9], where we introduced a new matrix invariant, the rank profile matrix. This normal form contains the row and column rank profile information of the matrix and that of all its leading sub-matrices.

This normal form is closely related to a permutation matrix appearing in the Bruhat decomposition [10] and in related variants [11,12,13,14,15]. Still, no connection to the rank profiles were made. In another setting, the construction of matrix Schubert varieties in [16, Ch. 15] defines a similar invariant, but presents neither a matrix decomposition nor any computational aspects.

More precisely, the present paper gathers the key contributions of [8] and [9]:

- we define of a new matrix invariant over a field, the rank profile matrix, summarizing all information on the row and column rank profiles of all the leading sub-matrices;
- we study the conditions for a Gaussian elimination algorithm to compute all or part of this invariant, through the corresponding PLUQ decomposition;
- as a consequence, we show that the classical iterative CUP decomposition algorithm can actually be adapted to compute the rank profile matrix. Used, in a Crout variant, as a base-case to our implementation, it delivers a significant improvement in efficiency;
- we also show that both the row and the column echelon forms of a matrix can be recovered from some PLUQ decompositions thanks to an elementary post-processing algorithm.

Further, we here develop three novel aspects:

- we extend the notion of rank profile matrix over arbitrary rings;
- we make further connections with existing matrix decompositions, in particular the Bruhat and generalized Bruhat decompositions;
- lastly, we extend the recent algorithmic improvements of [17,18,19] for low rank matrices: indeed, we show here that the algorithm in [18] computes the rank profile matrix; we also propose an algorithmic variant reducing the leading constant by a factor of three; and we propose an algorithmic reduction computing the rank profile matrix in time bounded by $(r^\omega + mn)^{1+o(1)}$.

Organization of the article. We first introduce in Section 2 the rank profile matrix $\mathcal{R}_A$, a normal form summarizing all rank profile information of a matrix and of all its leading sub-matrices. This definition is extended over arbitrary rings in Section 2.2. There, two notions of rank are considered: the spanning rank and McCoy's rank and we show that only the latter is suited to the definition of the rank profile matrix. We then study the conditions that PLUQ decomposition algorithms must satisfy in order to reveal the rank profile structure of a matrix. For this, we decompose, in Section 3, the pivoting strategy of any PLUQ algorithm into two types of operations: the search of the pivot and the permutation used to move it to the main diagonal. We propose new search and new permutation strategies and show what rank profiles are computed using any possible combination of these operations. In particular we show three new pivoting strategy combinations that compute the rank profile matrix.

As an illustration, we show in Section 4 how these pivoting strategies instantiate in iterative or recursive algorithms, using slab or tile blocking. Connections are made to the most common elimination algorithms and we state in full details the recent tile recursive algorithm of [8], implementing the new pivoting strategy.

Section 5 shows how this better understanding of the pivoting conditions helped to design an iterative Crout CUP with rotations, to be used as a base case for the tile recursive algorithm. Using the analysis of Section 3, it is shown that it computes the rank profile matrix while improving the previous base case implementation by a significant speed-up.

We then show in Section 6 how a PLUQ decomposition revealing the rank profile matrix relates with other triangular matrix decompositions, such as the LEU and the classical, modified or generalized Bruhat decompositions, or the computation of both row and column echelon forms, from a single PLUQ decomposition.

Lastly, we focus in section 7 on the recent algorithmic improvements for the computation of the row or column rank profile for matrices with low rank. We show that

the algorithm in [18] actually computes the rank profile matrix, but with a classical complexity. Then we show how to adapt its fast improvement [19] to also compute the rank profile matrix, before discussing algorithmic variants that improve on the leading constant factor in the latter algorithm's complexity bound.

Notations. In the following, $0_{m\times n}$ denotes the $m \times n$ zero matrix. For two list of indices $\mathcal{P}$ and $\mathcal{Q}$, $A_{\mathcal{P},\mathcal{Q}}$ denotes the sub-matrix of A formed by the rows of index in $\mathcal{P}$ and the columns of index in $\mathcal{Q}$. In particular, $A_{i..j,k..l}$ denotes the contiguous block of coefficients in A of rows position between i and j and columns position between k and l. We may denote by $*$ the sequence of all possible row or column indices: e.g. $A_{i,*}$ denotes the i-th row of A.

To a permutation $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ we define the associated permutation matrix $P(\sigma)$, permuting rows by left multiplication: the rows of $P(\sigma)A$ are that of A permuted by σ. Reciprocally, for a permutation matrix P, we denote by $\sigma(P)$ the associated permutation.

2. The rank profile matrix

We propose in Theorem 3 the definition of the rank profile matrix, an invariant summarizing all information on the rank profiles of a matrix. As will be discussed in this section and in section 6, this invariant is closely related to the Bruhat decomposition [10] and its generalizations [12,20,16].

2.1. Definition over a field

We consider first matrices over a field K and a valid pivot in Gaussian elimination is any non-zero element of K.

Definition 1. An r-sub-permutation matrix is a matrix of rank r with only r non-zero entries equal to one.

Lemma 2. *An $m \times n$ r-sub-permutation matrix has at most one non-zero entry per row and per column, and can be written $P \begin{bmatrix} I_r & \\ & 0_{(m-r)\times(n-r)} \end{bmatrix} Q$ where P and Q are permutation matrices.*

Theorem 3. *Let $A \in \mathrm{K}^{m\times n}$ of rank r. There exists a unique $m \times n$ r-sub-permutation matrix $\mathcal{R}_A$ of which every leading sub-matrix has the same rank as the corresponding leading sub-matrix of A. This sub-permutation matrix is called the* rank profile matrix of A.

Proof. We prove existence by induction on the row dimension of the leading submatrices.

If $A_{1,1..n} = 0_{1\times n}$, setting $\mathcal{R}^{(1)} = 0_{1\times n}$ satisfies the defining condition. Otherwise, let j be the index of the invertible element in $A_{1,1..n}$ and set $\mathcal{R}^{(1)} = e_j^T$ the j-th n-dimensional row canonical vector, which satisfies the defining condition.

Now for a given $i \in \{1, \ldots, m\}$, suppose that there is a unique $i \times n$ rank profile matrix $\mathcal{R}^{(i)}$ such that $\mathrm{rank}(A_{1..i,1..j}) = \mathrm{rank}(\mathcal{R}_{1..i,1..j})$ for every $j \in \{1..n\}$. If $\mathrm{rank}(A_{1..i+1,1..n}) = \mathrm{rank}(A_{1..i,1..n})$, then $\mathcal{R}^{(i+1)} = \begin{bmatrix} \mathcal{R}^{(i)} \\ 0_{1\times n} \end{bmatrix}$. Otherwise, consider k, the smallest column index

such that $\text{rank}(A_{1..i+1,1..k}) = \text{rank}(A_{1..i,1..k}) + 1$ and set $\mathcal{R}^{(i+1)} = \begin{bmatrix} \mathcal{R}^{(i)} \\ e_k^T \end{bmatrix}$. Any leading sub-matrix of $\mathcal{R}^{(i+1)}$ has the same rank as the corresponding leading sub-matrix of A: first, for any leading subset of rows and columns with less than i rows, the case is covered by the induction; second define $\begin{bmatrix} B & u \\ v^T & x \end{bmatrix} = A_{1..i+1,1..k}$, where u, v are vectors and x is a scalar. From the definition of k, v is linearly dependent with B and thus any leading sub-matrix of $\begin{bmatrix} B \\ v^T \end{bmatrix}$ has the same rank as the corresponding sub-matrix of $\mathcal{R}^{(i+1)}$. Similarly, from the definition of k, the same reasoning works when considering more than k columns, with a rank increment by 1.
Lastly we show that $\mathcal{R}^{(i+1)}$ is an r_{i+1}-sub-permutation matrix. Indeed, u is linearly dependent with the columns of B: otherwise, $\text{rank}(\begin{bmatrix} B & u \end{bmatrix}) = \text{rank}(B) + 1$. From the definition of k we would then have $\text{rank}(\begin{bmatrix} B & u \\ v^T & x \end{bmatrix}) = \textit{rank}(\begin{bmatrix} B & u \end{bmatrix}) + 1 = \text{rank}(B) + 2 = \text{rank}(\begin{bmatrix} B \\ v^T \end{bmatrix}) + 2$, but this is a contradiction. Consequently, the k-th column of $\mathcal{R}^{(i)}$ is all zero, and $\mathcal{R}^{(i+1)}$ is an r_{i+1}-sub-permutation matrix.

To prove uniqueness, suppose there exist two distinct rank profile matrices $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$ for a given matrix A and let (i, j) be the lexicographically minimal coordinates where $\mathcal{R}^{(1)}_{i,j} \neq \mathcal{R}^{(2)}_{i,j}$. As a consequence the rank of the (i, j)-leading submatrices of $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$ differ but should both be equal to $\text{rank}(A_{1..i,1..j})$, a contradiction. □

Example 4. $A = \begin{bmatrix} 2&0&3&0 \\ 1&0&0&0 \\ 0&0&4&0 \\ 0&2&0&1 \end{bmatrix}$ has $\mathcal{R}_A = \begin{bmatrix} 1&0&0&0 \\ 0&0&1&0 \\ 0&0&0&0 \\ 0&1&0&0 \end{bmatrix}$ for rank profile matrix over $\mathbb{Q}$.

Remark 5. The permutation matrix introduced in the *modified Bruhat decomposition* of [20], and defined there only for invertible matrices, is also the matrix E introduced in Malaschonok's LEU decomposition [14, Theorem 1]. In the latter paper, an algorithm for this decomposition was only shown over a field for $m = n = 2^k$, and no connection was made to the relation with ranks and rank profiles. We have shown in [8, Corollary 1] that E is in fact the rank profile matrix and we made the explicit connection in in [8, Corollary 1], as recalled in Section 6. We here generalize the existence to arbitrary rank r and dimensions m and n and after proving its uniqueness, we propose this definition as a new matrix normal form. We next also generalize the construction over a ring in Section 2.2.

The rank profile matrix has the following properties:

Lemma 6. *Let A be a matrix.*
(1) $\mathcal{R}_A$ is diagonal *if A has* generic rank profile*.*
(2) $\mathcal{R}_A$ is a permutation *matrix if A is* invertible
(3) RowRP(A) = RowSupp($\mathcal{R}_A$); ColRP(A) = ColSupp($\mathcal{R}_A$).
Moreover, for all $1 \leq i \leq m$ and $1 \leq j \leq n$, we have:
(4) RowRP($A_{1..i,1..j}$) = RowSupp($(\mathcal{R}_A)_{1..i,1..j}$)
(5) ColRP($A_{1..i,1..j}$) = ColSupp($(\mathcal{R}_A)_{1..i,1..j}$),

These properties show how to recover the row and column rank profiles of A and of any of its leading sub-matrix.

2.2. Generalization over a ring

Over a ring, several notions of rank exist: the spanning rank, McCoy's rank, Smith's rank, etc.

Definition 7. Let A be an $m \times n$ matrix over a ring R:

- *McCoy's rank*, denoted by $\mathrm{r}_{\mathcal{M}}$, is the largest r such that no size s minor of A with $s > r$ is a unit. Equivalently, it is also the largest r such that there exists a selection of r columns whose right nullspace is the zero vector [21, Theorem 51]. Over a principal ideal ring (PIR), this is the number of invertible determinantal divisors.
- The *spanning rank*, denoted by $\mathrm{s_r}$, the smallest r such that $A = BC$, where B is $m \times r$ and C is $r \times n$. Over a principal ideal ring (PIR), this is equivalent to Smith's rank [22, Definiton 2.2], the number of non-zero determinantal divisors [23, Proposition 1.1.(d)].

First, the rank profile matrix over a field gives rise to a rank profile matrix over any principal ideal ring.

Corollary 8. *Let $\mathbb{D}$ be a principal ideal domain (PID) and let $A \in \mathbb{D}^{m\times n}$ with McCoy's rank r. There exists a unique $m \times n$ r-sub-permutation matrix $\mathcal{R}_A$ of which every leading sub-matrix has the same rank as the corresponding leading sub-matrix of A. This sub-permutation matrix is called the* rank profile matrix of A.

Proof. From [23, Proposition 1.6], over a PID with field of fractions K, $\mathrm{s_r} = \mathrm{r}_{\mathcal{M}} = \mathrm{rank_K}$. Thus $\mathcal{R}_A$ over K satisfies the requirements over $\mathbb{D}$ and is the unique such matrix. □

However, when the ring R contains zero divisors, the notion of spanning rank differs from that of McCoy's rank. Lemma 9 shows that the notion of rank profile matrix can not be defined from the spanning rank.

Lemma 9. *Over $\mathbb{Z}/4\mathbb{Z}$, no matrix with exactly 1 or 2 non-zero entries is such that its leading sub-matrices have the same spanning ranks as that of $A = \left[\begin{smallmatrix} 0 & 2 \\ 2 & 1 \end{smallmatrix}\right]$.*

Proof. The spanning rank of A is 1, as $A = \left[\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right]\left[\begin{smallmatrix} 2 & 1 \end{smallmatrix}\right]$. However, note that the spanning rank of the leading 1×1 sub-matrix of A is 0, while that of the leading 1×2 and the leading 2×1 sub-matrices of A are equal to one. Hence, if a matrix $R = \left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ is such that any of its leading sub-matrices has the same spanning rank as the corresponding sub-matrix of A, then $a = 0$ and $b, c \neq 0$. Now as R must have spanning rank one, there must exist x, y, z, t, such that $\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]\left[\begin{smallmatrix} z & t \end{smallmatrix}\right] = \left[\begin{smallmatrix} 0 & b \\ c & d \end{smallmatrix}\right]$. As $b, c \neq 0$, then x, y, z, t are all non zero and since $a = 0$, we have $x = z = 2$, but then, $z, t \in \{-1, 1\}$ otherwise $b = 0$ or $c = 0$. Consequently $d \neq 0$. □

With McCoy's rank, on the contrary, the rank profile matrix of the matrix in Lemma 9 could be defined as the matrix $\left[\begin{smallmatrix} 0 & 0 \\ 0 & 1 \end{smallmatrix}\right]$. Indeed, remark that the McCoy's rank of the sub-matrix $\left[\begin{smallmatrix} 0 & 2 \end{smallmatrix}\right]$ is 0. Hence, we focus in the remaining of this section on McCoy's rank and only assume that R is a commutative ring R with a unit element 1. We will need the following lemmata.

Lemma 10. *Consider a matrix $B \in \mathrm{R}^{m\times n}$, and two vectors u and v. Then $\delta_1 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \end{bmatrix}\right) - \mathrm{r}_{\mathcal{M}}(B)$ and $\delta_2 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B \\ v^T \end{bmatrix}\right) - \mathrm{r}_{\mathcal{M}}(B)$ satisfy $\delta_i \in \{0, 1\}$.*

Proof. Appending a column to B can not reduce the maximal number of columns having a right null-space reduced to the zero vector, hence $\delta_1 \geq 0$. Let $r = \mathrm{r}_{\mathcal{M}}(B)$. If $r = n$, then $\mathrm{r}_{\mathcal{M}}([\, B\ u\,]) \leq n+1$ and $\delta \leq 1$. Otherwise, this means that any subset of $r+1$ columns of B vanishes in a linear combination with some non-zero coefficients. Consequently, any subset of $r+2$ columns of $[\, B\ u\,]$, which necessarily includes at least $r+1$ columns of B, vanishes in a linear combination with some non-zero coefficients, thus showing that $\delta_1 \leq 1$. Applying the same reasoning on $\left[\begin{smallmatrix} B \\ v^T \end{smallmatrix}\right]^T$, proves that $\delta_2 \in \{0, 1\}$. □

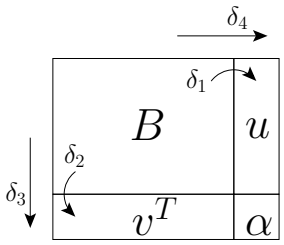
Lemma 11. *Consider an $m \times n$ matrix B, two vectors, u and v, and a scalar α. Denote by $\delta_1, \delta_2, \delta_3, \delta_4$ the discrepancies in rank of the following matrices:*

(1) $\mathrm{r}_{\mathcal{M}}(B) + \delta_1 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \end{bmatrix}\right)$

(2) $\mathrm{r}_{\mathcal{M}}(B) + \delta_2 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B \\ v^T \end{bmatrix}\right)$

(3) $\mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \end{bmatrix}\right) + \delta_3 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix}\right)$

(4) $\mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B \\ v^T \end{bmatrix}\right) + \delta_4 = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix}\right)$



Then:

(i) ($\delta_1 = 0$ *and* $\delta_4 = 1$) *is equivalent to* ($\delta_2 = 0$ *and* $\delta_3 = 1$).

(ii) ($\delta_1 = 1$ *and* $\delta_3 = 1$) *is equivalent to* ($\delta_2 = 1$ *and* $\delta_4 = 1$).

(iii) ($\delta_1 = 1$ *and* $\delta_3 = 0$) *implies* ($\delta_2 = 0$ *and* $\delta_4 = 1$).

(iv) ($\delta_1 = 1$) *implies* ($\delta_4 = 1$).

(v) ($\delta_2 = 1$) *implies* ($\delta_3 = 1$).

Proof. Let $r = \mathrm{r}_{\mathcal{M}}(B)$. From Lemma 10, all the discrepancies are either zero or one, since the proposed ranks involve matrices with only one extra row or column. From the definitions of δ_1 and δ_3, on the one hand, those of δ_2 and δ_4 on the other hand, we have that

$$r + \delta_2 + \delta_4 = r + \delta_1 + \delta_3. \tag{1}$$

Thus:

(i) if ($\delta_1 = 0$ and $\delta_4 = 1$), then $\delta_2 + 1 = \delta_3$ and thus the only possibility is ($\delta_2 = 0$ and $\delta_3 = 1$). Reciprocally, $\delta_4 = \delta_1 + 1$ yields ($\delta_1 = 0$ and $\delta_4 = 1$) as only possibility.

(ii) Similarly there is only way for the rank to be augmented by 2.

(iii) Suppose that $\delta_1 = 1$ and $\delta_3 = 0$. Then $\mathrm{r}_{\mathcal{M}}([\, B\ u\,]) = \mathrm{r}_{\mathcal{M}}(\left[\begin{smallmatrix} B & u \\ v^T & \alpha \end{smallmatrix}\right]) = r + 1$. There exist subsets of $r+1$ indices $\mathcal{I}$ and $\mathcal{J}$ such that the sub-matrix $[\, B\ u\,]_{\mathcal{I},\mathcal{J}}$ has a unit determinant. Since $\delta_1 = 1$ then necessarily $n+1 \in \mathcal{J}$, otherwise $\mathrm{r}_{\mathcal{M}}(B) = r+1$. As

$\mathrm{r}_{\mathcal{M}}(B_{\mathcal{I},\mathcal{J}\setminus\{n+1\}}) = r$, there is a $k \in \mathcal{I}$, such that $X = B_{\mathcal{I}\setminus\{k\},\mathcal{J}\setminus\{n+1\}}$ has a unit determinant and is therefore invertible over R. Consider the permutation matrices P and Q such that $PBQ = \left[\begin{smallmatrix} X & Y \\ Z & T \end{smallmatrix}\right]$. Let $\left[\begin{smallmatrix} b_1 \\ b_2 \end{smallmatrix}\right] = Pu$ where b_1 has dimension r. We have

$$PBQ \begin{bmatrix} X^{-1} & -X^{-1}Y \\ & I_{n-r} \end{bmatrix} = \begin{bmatrix} I_r & 0 \\ M & N \end{bmatrix}.$$

From [24, 4.11.(c)], we know that multiplication by matrices with unit determinant preserves $\mathrm{r}_{\mathcal{M}}$. Note that N does not contain a unit, otherwise $\mathrm{r}_{\mathcal{M}}(B) = r+1$. Now we also have

$$P \begin{bmatrix} B & u \end{bmatrix} \begin{bmatrix} Q & \\ & 1 \end{bmatrix} \begin{bmatrix} X^{-1} & -X^{-1}Y & -X^{-1}b_1 \\ & I_{n-r} & 0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} I_r & 0 & 0 \\ M & N & \beta \end{bmatrix},$$

where $\beta = -Mb_1 + b_2$ is a vector containing at least one unit u at index i_β and

$$\begin{bmatrix} P & \\ & 1 \end{bmatrix} \begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix} \begin{bmatrix} Q & \\ & 1 \end{bmatrix} \begin{bmatrix} X^{-1} & -X^{-1}Y & -X^{-1}b_1 \\ & I_{n-r} & 0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} I_r & 0 & 0 \\ M & N & \beta \\ w^T & x^T & \gamma \end{bmatrix} = C.$$

If x^T would contain a unit at index j_x, we could form the subsets of indices $\mathcal{I} = \{1, \ldots, r, i_\beta, m+1\}$ and $\mathcal{J} = \{1, \ldots, r, j_x, n+1\}$ such that $C_{\mathcal{I},\mathcal{J}}$ has a unit determinant, showing that $\mathrm{r}_{\mathcal{M}}(\left[\begin{smallmatrix} B & u \\ v^T & \alpha \end{smallmatrix}\right]) \geq r+2$ which is a contradiction. Therefore, since

$$\begin{bmatrix} P & \\ & 1 \end{bmatrix} \begin{bmatrix} B \\ v^T \end{bmatrix} Q \begin{bmatrix} X^{-1} & -X^{-1}Y \\ & I_{n-r} \end{bmatrix} = \begin{bmatrix} I_r & 0 \\ M & N \\ w^T & x^T \end{bmatrix},$$

we deduce that $\mathrm{r}_{\mathcal{M}}(\left[\begin{smallmatrix} B \\ v^T \end{smallmatrix}\right]) = r$. Hence, $\delta_2 = 0$, and from (1), we have $\delta_4 = 1$.

(iv) Combining (ii) and (iii) leads directly to (iv).

(v) Applying (iv) on the transpose of A yields (v). □

Lemma 11, point (iii), is typically what is not true with the spanning rank: on the counter-example matrix $A_1 = \left[\begin{smallmatrix} 0 & 2 \\ 2 & 1 \end{smallmatrix}\right]$ of Lemma 9, we have $\delta_1 = 1$, $\delta_3 = 0$, but $\delta_2 = 1$ and $\delta_4 = 0$. We are now ready to extend the proof of existence of the rank profile matrix to matrices over a ring, using the notion of McCoy's rank.

Definition 12. An *r-sub-permutation matrix* is a matrix of McCoy rank r with only $\mathrm{r}_{\mathcal{M}}$ non-zero entries equal to one.

Lemma 13. *An $m \times n$ r-sub-permutation matrix has at most one non-zero entry per row and per column, and can be written $P \left[\begin{smallmatrix} I_r & \\ & 0_{(m-r)\times(n-r)} \end{smallmatrix}\right] Q$ where P and Q are permutation matrices.*

Proof. Let A be an $m \times n$ $\mathrm{r}_{\mathcal{M}}$-sub-permutation matrix. Let μ and ν be respectively the number of non-zero columns and rows in A. As A has only $r = \mathrm{r}_{\mathcal{M}}(A)$ non-zero elements, we have that $s = \min(\mu, \nu) \leq r$. If $s = \mu$, then A writes $A = B \begin{bmatrix} I_s & 0 \end{bmatrix} P$ where B is the $m \times s$ matrix formed by the s non-zero columns of A and P a permutation matrix. From [24, Lemma 4.14], we have that $\mathrm{r}_{\mathcal{M}}(A) \leq \min\{\mathrm{r}_{\mathcal{M}}(B), s = \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} I_s & 0 \end{bmatrix}\right), \mathrm{r}_{\mathcal{M}}(P)\}$. This decomposition thus shows that the rank of A is at most s and the only way for A to have a rank r is to have one non zero element per column. The same reasoning applies on the non-zero rows when $s = \nu$. □

Theorem 14. *Let $A \in \mathrm{R}^{m \times n}$ of McCoy rank r. There exist a unique $m \times n$ r-sub-permutation matrix $\mathcal{R}_A$ of which every leading sub-matrix has the same McCoy rank as the corresponding leading sub-matrix of A. This sub-permutation matrix is called the* (McCoy) rank profile matrix of A.

Proof. We prove existence by induction on the row dimension of the leading submatrices.

Consider the first row of A. If it does not contain any unit, then setting $\mathcal{R}^{(1)} = 0_{1 \times n}$ satisfies the defining condition. Otherwise, let j be the index of the leftmost unit element in $A_{1,1..n}$ and set $\mathcal{R}^{(1)} = e_j^T$ the j-th n-dimensional canonical row vector, which satisfies the defining condition.

Now suppose that there exists a rank profile matrix $\mathcal{R}^{(s)}$ of which every leading sub-matrix has the same McCoy rank as the corresponding leading sub-matrix of $A_{1..s,*}$. Let $r_s = \mathrm{r}_{\mathcal{M}}(A_{1..s,*})$. Then, from Lemma 10, either $r_{s+1} = r_s$ or $r_{s+1} = r_s + 1$:

(i) On the one hand, if $r_{s+1} = r_s$, let j_0 be the largest index such that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j_0}) < r_s$. We then use Lemma 11, point (iii), with $B = A_{1..s,1..j_0}$, $\begin{bmatrix} B & u \end{bmatrix} = A_{1..s,1..(j_0+1)}$, $\begin{bmatrix} B \\ v^T \end{bmatrix} = A_{1..s+1,1..j_0}$, $\begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix} = A_{1..s+1,1..(j_0+1)}$: we thus have that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j_0}) = \mathrm{r}_{\mathcal{M}}(A_{1..(s+1),1..j_0}) = r_s - 1$ and $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j_0+1+k}) = \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..j_0+1+k}) = r_S$ for all $k = 0..(n - j_0 - 1)$.

Let now j_1 be the largest index such that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j_1}) < (r_s - 1)$, the same reasoning shows that the McCoy ranks of the leading matrices with $j_1 + 1$ to j_0 columns and s or $s+1$ rows are identical. Continue with j_i until there are no more columns to consider. This shows that $\left[\begin{array}{c} \mathcal{R}^{(s)} \\ \hline 0 \end{array}\right]$ is a suitable rank profile matrix for $A_{1..(s+1),*}$.

(ii) On the other hand, if $r_{s+1} = r_s + 1$, it means in particular, that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..n}) \neq \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..n})$. We thus again consider the leading matrices with s or $s+1$ rows. Let now j be the smallest index such that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j}) \neq \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..j})$. As there is only one extra row in the latter, we must have exactly

$$\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j}) + 1 = \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..j}) \tag{2}$$

We show now that if e_j is the j-th canonical vector then $\mathcal{T} = \left[\begin{array}{c} \mathcal{R}^{(s)} \\ \hline e_j^T \end{array}\right]$ is a suitable rank profile matrix for $A_{1..s+1,*}$: we use Lemma 11, point (i), with $B = A_{1..s,1..j-1}$, $\begin{bmatrix} B & u \end{bmatrix} = A_{1..s,1..j}$, $\begin{bmatrix} B \\ v^T \end{bmatrix} = A_{1..s+1,1..j-1}$, $\begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix} = A_{1..s+1,1..j}$. From the definition of j in Equation (2), we have that $\mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \end{bmatrix}\right) \neq \mathrm{r}_{\mathcal{M}}\left(\begin{bmatrix} B & u \\ v^T & \alpha \end{bmatrix}\right)$ and therefore that

$\delta_3 = 1$. As j is the smallest with this property, it means that $\mathrm{r}_{\mathcal{M}}\left(\left[\begin{smallmatrix} B \end{smallmatrix}\right]\right) = \mathrm{r}_{\mathcal{M}}\left(\left[\begin{smallmatrix} B \\ v^T \end{smallmatrix}\right]\right)$ and therefore that $\delta_2 = 0$. Lemma 11 then asserts that $\delta_1 = 0$, i.e. that $\mathrm{r}_{\mathcal{M}}\left(\left[\begin{smallmatrix} B & u \end{smallmatrix}\right]\right) = \mathrm{r}_{\mathcal{M}}(B)$, or that $\mathrm{r}_{\mathcal{M}}(A_{1..s,1..j}) = \mathrm{r}_{\mathcal{M}}(A_{1..s,1..j-1})$. Thus, by induction, the latter equality also holds for their rank profile matrices: $\mathrm{r}_{\mathcal{M}}(\mathcal{R}^{(s)}_{1..s,1..j}) = \mathrm{r}_{\mathcal{M}}(\mathcal{R}^{(s)}_{1..s,1..j-1})$. From the definition of the rank profile matrix, this shows that the j-th column of $\mathcal{R}^{(s)}$ is zero and thus that the candidate matrix $\mathcal{T}$ has the following shape:

$$\mathcal{T} = \left[\begin{array}{c} \mathcal{R}^{(s)} \\ \hline e_j^T \end{array}\right] = \begin{bmatrix} * & 0 & * \\ 0 & 1 & 0 \end{bmatrix}. \tag{3}$$

Now any leading submatrix of $\mathcal{T}$ with less than s rows is also a leading submatrix of $\mathcal{R}^{(s)}$ so that the induction takes place. There remains to show that $\mathrm{r}_{\mathcal{M}}(\mathcal{T}_{*,1..t}) = \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..t})$ for all $t \in 1..n$. For $t = 1..j-1$ this is guaranteed by the fact that j is the smallest index modifying the rank. For $t = j$ the shape of $\mathcal{T}$ shows that $\mathrm{r}_{\mathcal{M}}\left(\mathcal{R}^{(s)}_{1..s,1..j}\right)$ is exactly one less than the rank of $\mathcal{T}_{*,1..j}$. By induction we know that $\mathrm{r}_{\mathcal{M}}\left(\mathcal{R}^{(s)}_{1..s,1..j}\right) = \mathrm{r}_{\mathcal{M}}(A_{1..s,1..j})$ so that Equation (2) now yields that $\mathrm{r}_{\mathcal{M}}(\mathcal{T}_{*,1..j}) = \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..j})$. Finally, for $t = j+1..n$, Lemma 11, point (v), ensures that once there is a discrepancy in (McCoy) rank with the last row, this discrepancy is maintained for every added column. This shows that for $t \geq j$, we have:

$$\mathrm{r}_{\mathcal{M}}(A_{1..s,1..t}) + 1 = \mathrm{r}_{\mathcal{M}}(A_{1..s+1,1..t}). \tag{4}$$

The shape of $\mathcal{T}$ also shows that for $t \geq j$, we have:

$$\mathrm{r}_{\mathcal{M}}\left(\mathcal{R}^{(s)}_{*,1..t}\right) + 1 = \mathrm{r}_{\mathcal{M}}(\mathcal{T}_{*,1..t}). \tag{5}$$

Now the induction equates both left hand sides of the above Equations (4) and (5), so that both right hand sides also agree. Thus $\mathcal{T}$ satisfies the rank profile requirements of the theorem for $A_{1..s+1,*}$.

The proof of uniqueness is the same as over a field: consider (i,j) the lexicographically minimal coordinates where two possible rank profile matrices $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$ differ. The rank of the (i,j)-leading sub-matrices of $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$ differ but should both be equal to $\mathrm{rank}(A_{1..i,1..j})$, a contradiction. □

3. When does a PLUQ algorithm reveal the rank profile matrix?

From now on, for the sake of simplicity, we consider algorithms over a field.

3.1. Ingredients of a PLUQ decomposition algorithm

Over a field, the LU decomposition generalizes to matrices with arbitrary rank profiles, using row and column permutations (in some cases such as the CUP, or LSP decompositions, the row permutation is embedded in the structure of the C or S matrices). However such PLUQ decompositions are not unique and not all of them will necessarily reveal rank profiles and echelon forms. We will characterize the conditions for a PLUQ decomposition algorithm to reveal the row or column rank profile or the rank profile matrix.

We consider the four types of operations of a Gaussian elimination algorithm in the processing of the k-th pivot:

Pivot search: finding an element to be used as a pivot,
Pivot permutation: moving the pivot in diagonal position (k, k) by column and/or row permutations,
Update: applying the elimination at position (i, j): $a_{i,j} \leftarrow a_{i,j} - a_{i,k}a_{k,k}^{-1}a_{k,j}$,
Normalization: dividing the k-th row (resp. column) by the pivot.
Choosing how each of these operation is done, and when they are scheduled results in an elimination algorithm. Conversely, any Gaussian elimination algorithm computing a PLUQ decomposition can be viewed as a set of specializations of each of these operations together with a scheduling.

The choice of doing the normalization on rows or columns only determines which of U or L will be unit triangular. The scheduling of the updates vary depending on the type of algorithm used: iterative, recursive, slab or tiled block splitting, with right-looking, left-looking or Crout variants [25]. Neither the normalization nor the update impact the capacity to reveal rank profiles and we will thus now focus on the pivot search and the permutations.

Choosing a search and a permutation strategy fixes the matrices P and Q of the PLUQ decomposition obtained and, as we will see, determines the ability to recover information on the rank profiles. Once these matrices are fixed, the L and the U factors are unique. We introduce the pivoting matrix.

Definition 15. The pivoting matrix of a PLUQ decomposition $A = PLUQ$ of rank r is the r-sub-permutation matrix

$$\Pi_{P,Q} = P \begin{bmatrix} I_r & \\ & 0_{(m-r)\times(n-r)} \end{bmatrix} Q.$$

The r non-zero elements of $\Pi_{P,Q}$ are located at the initial positions of the pivots in the matrix A. Thus $\Pi_{P,Q}$ summarizes the choices made in the search and permutation operations.

3.1.1. Pivot search

The search operation vastly differs depending on the field of application. In numerical dense linear algebra, numerical stability is the main criterion for the selection of the pivot. In sparse linear algebra, the pivot is chosen so as to reduce the fill-in produced by the update operation. In order to reveal some information on the rank profiles, a notion of precedence has to be used: a usual way to compute the row rank profile is to search in a given row for a pivot and only move to the next row if the current row was found to be all zeros. This guarantees that each pivot will be on the first linearly independent row, and therefore the row support of $\Pi_{P,Q}$ will be the row rank profile. The precedence here is that the pivot's coordinates must minimize the order for the first coordinate (the row index). As a generalization, we consider the most common preorders of the cartesian product $\{1, \dots m\} \times \{1, \dots n\}$ inherited from the natural orders of each of its components and describe the corresponding search strategies, minimizing this preorder:
Row order: $(i_1, j_1) \preceq_{\text{row}} (i_2, j_2)$ iff $i_1 \leq i_2$: *search for any invertible element in the first non-zero row.*
Column order: $(i_1, j_1) \preceq_{\text{col}} (i_2, j_2)$ iff $j_1 \leq j_2$. *search for any invertible element in the first non-zero column.*
Lexicographic order: $(i_1, j_1) \preceq_{\text{lex}} (i_2, j_2)$ iff $i_1 < i_2$ or $i_1 = i_2$ and $j_1 \leq j_2$: *search for the leftmost non-zero element of the first non-zero row.*

Reverse lexicographic order: $(i_1, j_1) \preceq_{\text{revlex}} (i_2, j_2)$ iff $j_1 < j_2$ or $j_1 = j_2$ and $i_1 \leq i_2$: *search for the topmost non-zero element of the first non-zero column.*

Product order: $(i_1, j_1) \preceq_{\text{prod}} (i_2, j_2)$ iff $i_1 \leq i_2$ and $j_1 \leq j_2$: *search for any non-zero element at position (i, j) being the only non-zero of the leading (i, j) sub-matrix.*

Example 16. Consider the matrix $\begin{bmatrix} 0 & 0 & 0 & a & b \\ 0 & c & d & e & f \\ g & h & i & j & k \\ l & m & n & o & p \end{bmatrix}$, where each literal is a non-zero element. The minimum non-zero elements for each preorder are the following:
Row order: a, b,
Column order: g, l,
Lexicographic order: a,
Reverse lexic. order: g,
Product order: a, c, g.

3.1.2. Pivot permutation

The pivot permutation moves a pivot from its initial position to the leading diagonal. Besides this constraint all possible choices are left for the remaining values of the permutation. Most often, it is done by row or column transpositions, as it clearly involves a small amount of data movement. However, these transpositions can break the precedence relations in the set of rows or columns, and can therefore prevent the recovery of the rank profile information. A pivot permutation that leaves the precedence relations unchanged will be called k-monotonically increasing.

Definition 17. A permutation of $\sigma \in \mathcal{S}_n$ is called k-monotonically increasing if its last $n - k$ values form a monotonically increasing sequence.

In particular, the last $n - k$ rows of the associated row-permutation matrix P_σ are in row echelon form. For example, the cyclic shift between indices k and i, with $k < i$ defined as $R_{k,i} = (1, \ldots, k-1, i, k, k+1, \ldots, i-1, i+1, \ldots, n)$, that we will call a (k, i)-rotation, is an elementary k-monotonically increasing permutation.

Example 18. The $(1, 4)$-rotation $R_{1,4} = (4, 1, 2, 3)$ is a 1-monotonically increasing permutation. Its row permutation matrix is $\begin{bmatrix} 0 & & & 1 \\ 1 & & & \\ & 1 & & \\ & & 1 & 0 \end{bmatrix}$. In fact, any (k, i)-rotation is a k-monotonically increasing permutation.

Monotonically increasing permutations can be composed as stated in Lemma 19.

Lemma 19. *If $\sigma_1 \in \mathcal{S}_n$ is a k_1-monotonically increasing permutation and $\sigma_2 \in \mathcal{S}_{k_1} \times \mathcal{S}_{n-k_1}$ a k_2-monotonically increasing permutation with $k_1 < k_2$ then the permutation $\sigma_2 \circ \sigma_1$ is a k_2-monotonically increasing permutation.*

Proof. The last $n - k_2$ values of $\sigma_2 \circ \sigma_1$ are the image of a sub-sequence of $n - k_2$ values from the last $n - k_1$ values of σ_1 through the monotonically increasing function σ_2. □

Therefore an iterative algorithm, using rotations as elementary pivot permutations, maintains the property that the permutation matrices P and Q at any step k are k-monotonically increasing. A similar property also applies with recursive algorithms.

3.2. How to reveal rank profiles

A PLUQ decomposition reveals the row (resp. column) rank profile if it can be read from the first r values of the permutation matrix P (resp. Q). Equivalently, by Lemma 6, this means that the row (resp. column) support of the pivoting matrix $\Pi_{P,Q}$ equals that of the rank profile matrix.

Definition 20. The decomposition $A = PLUQ$ reveals:
(1) the row rank profile if $\text{RowSupp}(\Pi_{P,Q}) = \text{RowSupp}(\mathcal{R}_A)$,
(2) the col. rank profile if $\text{ColSupp}(\Pi_{P,Q}) = \text{ColSupp}(\mathcal{R}_A)$,
(3) the rank profile matrix if $\Pi_{P,Q} = \mathcal{R}_A$.

Example 21. $A = \begin{bmatrix} 2&0&3&0 \\ 1&0&0&0 \\ 0&0&4&0 \\ 0&2&0&1 \end{bmatrix}$ has $\mathcal{R}_A = \begin{bmatrix} 1&0&0&0 \\ 0&0&1&0 \\ 0&0&0&0 \\ 0&1&0&0 \end{bmatrix}$ for rank profile matrix over $\mathbb{Q}$. Now the pivoting matrix obtained from a PLUQ decomposition with a pivot search operation following the row order (any column, first non-zero row) could be the matrix $\Pi_{P,Q} = \begin{bmatrix} 0&0&1&0 \\ 1&0&0&0 \\ 0&0&0&0 \\ 0&1&0&0 \end{bmatrix}$. As these matrices share the same row support, the matrix $\Pi_{P,Q}$ reveals the row rank profile of A.

Remark 22. Example 21, suggests that a pivot search strategy minimizing row and column indices could be a sufficient condition to recover both row and column rank profiles at the same time, regardless the pivot permutation. However, this is unfortunately not the case. Consider for example a search based on the lexicographic order (first non-zero column of the first non-zero row) with transposition permutations, run on the matrix: $A = \begin{bmatrix} 0&0&1 \\ 2&3&0 \end{bmatrix}$. Its rank profile matrix is $\mathcal{R}_A = \begin{bmatrix} 0&0&1 \\ 1&0&0 \end{bmatrix}$ whereas the pivoting matrix could be $\Pi_{P,Q} = \begin{bmatrix} 0&0&1 \\ 0&1&0 \end{bmatrix}$, which does not reveal the column rank profile. This is due to the fact that the column transposition performed for the first pivot changes the order in which the columns will be inspected in the search for the second pivot.

We will show that if the pivot permutations preserve the order in which the still unprocessed columns or rows appear, then the pivoting matrix will equal the rank profile matrix. This is achieved by the monotonically increasing permutations.

Theorem 23 shows how the ability of a PLUQ decomposition algorithm to recover the rank profile information relates to the use of monotonically increasing permutations. More precisely, it considers an arbitrary step in a PLUQ decomposition where k pivots have been found in the elimination of an $\ell \times p$ leading sub-matrix A_1 of the input matrix A.

Theorem 23. *Consider a partial PLUQ decomposition of an $m \times n$ matrix A:*

$$A = P_1 \begin{bmatrix} L_1 & \\ M_1 & I_{m-k} \end{bmatrix} \begin{bmatrix} U_1 & V_1 \\ & H \end{bmatrix} Q_1$$

where $\begin{bmatrix} L_1 \\ M_1 \end{bmatrix}$ is $m \times k$ lower triangular and $\begin{bmatrix} U_1 & V_1 \end{bmatrix}$ is $k \times n$ upper triangular, and let A_1 be some $\ell \times p$ leading sub-matrix of A, for $\ell, p \geq k$. Let $H = P_2 L_2 U_2 Q_2$ be a PLUQ

decomposition of H. Consider the PLUQ decomposition

$$A = \underbrace{P_1 \begin{bmatrix} I_k & \\ & P_2 \end{bmatrix}}_{P} \underbrace{\begin{bmatrix} L_1 & \\ P_2^T M_1 & L_2 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} U_1 & V_1 Q_2^T \\ & U_2 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} I_k & \\ & Q_2 \end{bmatrix} Q_1}_{Q}.$$

Consider the following clauses:
(i) $RowRP(A_1) = RowSupp(\Pi_{P_1,Q_1})$
(ii) $ColRP(A_1) = ColSupp(\Pi_{P_1,Q_1})$
(iii) $\mathcal{R}_{A_1} = \Pi_{P_1,Q_1}$
(iv) $RowRP(H) = RowSupp(\Pi_{P_2,Q_2})$
(v) $ColRP(H) = ColSupp(\Pi_{P_2,Q_2})$
(vi) $\mathcal{R}_H = \Pi_{P_2,Q_2}$
(vii) P_1^T *is* k*-monotonically increasing or (*P_1^T *is* ℓ*-monotonically increasing and* $p = n$*)*

(viii) Q_1^T *is* k*-monotonically increasing or (*Q_1^T *is* p*-monotonically increasing and* $\ell = m$*)*
Then,
(a) if (i) or (ii) or (iii) then $H = \begin{bmatrix} 0_{(\ell-k)\times(p-k)} & * \\ * & * \end{bmatrix}$
(b) if (vii) then ((i) and (iv)) $\Rightarrow RowRP(A) = RowSupp(\Pi_{P,Q})$*;*
(c) if (viii) then ((ii) and (v)) $\Rightarrow ColRP(A) = ColSupp(\Pi_{P,Q})$*;*
(d) if (vii) and (viii) then (iii) and (vi) $\Rightarrow \mathcal{R}_A = \Pi_{P,Q}$*.*

Proof. Let $P_1 = \begin{bmatrix} P_{11} & E_1 \end{bmatrix}$ and $Q_1 = \begin{bmatrix} Q_{11} \\ F_1 \end{bmatrix}$ where E_1 is $m \times (m-k)$ and F_1 is $(n-k) \times n$.

On one hand we have

$$A = \underbrace{\begin{bmatrix} P_{11} & E_1 \end{bmatrix} \begin{bmatrix} L_1 \\ M_1 \end{bmatrix} \begin{bmatrix} U_1 & V_1 \end{bmatrix} \begin{bmatrix} Q_{11} \\ F_1 \end{bmatrix}}_{B} + E_1 H F_1. \tag{6}$$

On the other hand,

$$\begin{aligned} \Pi_{P,Q} &= P_1 \begin{bmatrix} I_k & \\ & P_2 \end{bmatrix} \begin{bmatrix} I_r & \\ & 0_{(m-r)\times(n-r)} \end{bmatrix} \begin{bmatrix} I_k & \\ & Q_2 \end{bmatrix} Q_1 \\ &= P_1 \begin{bmatrix} I_k & \\ & \Pi_{P_2,Q_2} \end{bmatrix} Q_1 = \Pi_{P_1,Q_1} + E_1 \Pi_{P_2,Q_2} F_1. \end{aligned}$$

Let $\overline{A}_1 = \begin{bmatrix} A_1 & 0 \\ 0 & 0_{(m-\ell)\times(n-p)} \end{bmatrix}$ and denote by B_1 the $\ell \times p$ leading sub-matrix of B.

(a) The clause (i) or (ii) or (iii) implies that all k pivots of the partial elimination were found within the $\ell \times p$ sub-matrix A_1. Hence $\text{rank}(A_1) = k$ and we can write $P_1 = \begin{bmatrix} P_{11} & \\ 0_{(m-\ell)\times k} & E_1 \end{bmatrix}$ and $Q_1 = \begin{bmatrix} Q_{11} & 0_{k\times(n-p)} \\ & F_1 \end{bmatrix}$, and the matrix A_1 writes

$$A_1 = \begin{bmatrix} I_\ell & 0 \end{bmatrix} A \begin{bmatrix} I_p \\ 0 \end{bmatrix} = B_1 + \begin{bmatrix} I_\ell & 0 \end{bmatrix} E_1 H F_1 \begin{bmatrix} I_p \\ 0 \end{bmatrix}. \tag{7}$$

Now $\text{rank}(B_1) = k$ as a sub-matrix of B of rank k and since

$$\begin{aligned} B_1 &= \begin{bmatrix} P_{11} & \begin{bmatrix} I_\ell & 0 \end{bmatrix} \cdot E_1 \end{bmatrix} \begin{bmatrix} L_1 \\ M_1 \end{bmatrix} \begin{bmatrix} U_1 & V_1 \end{bmatrix} \begin{bmatrix} Q_{11} \\ F_1 \cdot \begin{bmatrix} I_p \\ 0 \end{bmatrix} \end{bmatrix} \\ &= P_{11}L_1U_1Q_{11} + \begin{bmatrix} I_\ell & 0 \end{bmatrix} E_1 M_1 \begin{bmatrix} U_1 & V_1 \end{bmatrix} Q_1 \begin{bmatrix} I_p \\ 0 \end{bmatrix} \end{aligned}$$

where the first term, $P_{11}L_1U_1Q_{11}$, has rank k and the second term has a disjoint row support.

Finally, consider the term $\begin{bmatrix} I_\ell & 0 \end{bmatrix} E_1 H F_1 \begin{bmatrix} I_p \\ 0 \end{bmatrix}$ of equation (7). As its row support is disjoint with that of the pivot rows of B_1, it has to be composed of rows linearly dependent with the pivot rows of B_1 to ensure that $\text{rank}(A_1) = k$. As its column support is disjoint with that of the pivot columns of B_1, we conclude that it must be the zero matrix. Therefore the leading $(\ell - k) \times (p - k)$ sub-matrix of $E_1 H F_1$ is zero.

(b) From (a) we know that $A_1 = B_1$. Thus $\text{RowRP}(B) = \text{RowRP}(A_1)$. Recall that $A = B + E_1 H F_1$. No pivot row of B can be made linearly dependent by adding rows of $E_1 H F_1$, as the column position of the pivot is always zero in the latter matrix. For the same reason, no pivot row of $E_1 H F_1$ can be made linearly dependent by adding rows of B. From (i), the set of pivot rows of B is $\text{RowRP}(A_1)$, which shows that

$$\text{RowRP}(A) = \text{RowRP}(A_1) \cup \text{RowRP}(E_1 H F_1). \tag{8}$$

Let $\sigma_{E_1} : \{1..m-k\} \to \{1..m\}$ be the map representing the sub-permutation E_1 (i.e. such that $E_1[\sigma_{E_1}(i), i] = 1 \ \forall i$). If P_1^T is k-monotonically increasing, the matrix E_1 has full column rank and is in column echelon form, which implies that

$$\begin{aligned} \text{RowRP}(E_1 H F_1) &= \sigma_{E_1}(\text{RowRP}(H F_1)) \\ &= \sigma_{E_1}(\text{RowRP}(H)), \end{aligned} \tag{9}$$

since F_1 has full row rank. If P_1^T is ℓ monotonically increasing, we can write $E_1 = \begin{bmatrix} E_{11} & E_{12} \end{bmatrix}$, where the $m \times (m - \ell)$ matrix E_{12} is in column echelon form. If $p = n$, the matrix H writes $H = \begin{bmatrix} 0_{(\ell-k)\times(n-k)} \\ H_2 \end{bmatrix}$. Hence we have $E_1 H F_1 = E_{12} H_2 F_1$ which also implies

$$\text{RowRP}(E_1 H F_1) = \sigma_{E_1}(\text{RowRP}(H)).$$

From equation (7), the row support of $\Pi_{P,Q}$ is that of $\Pi_{P_1,Q_1} + E_1\Pi_{P_2,Q_2}F_1$, which is the union of the row support of these two terms as they are disjoint. Under the conditions of point (b), this row support is the union of $\text{RowRP}(A_1)$ and $\sigma_{E_1}(\text{RowRP}(H))$, which is, from (9) and (8), $\text{RowRP}(A)$.

(c) Similarly as for point (b).

(d) From (a) we have still $A_1 = B_1$. Now since $\text{rank}(B) = \text{rank}(B_1) = \text{rank}(A_1) = k$, there is no other non-zero element in $\mathcal{R}_B$ than those in $\mathcal{R}_{\overline{A_1}}$ and $\mathcal{R}_B = \mathcal{R}_{\overline{A_1}}$. The row and column support of $\mathcal{R}_B$ and that of E_1HF_1 are disjoint. Hence

$$\mathcal{R}_A = \mathcal{R}_{\overline{A_1}} + \mathcal{R}_{E_1HF_1}. \tag{10}$$

If both P_1^T and Q_1^T are k-monotonically increasing, the matrix E_1 is in column echelon form and the matrix F_1 in row echelon form. Consequently, the matrix E_1HF_1 is a copy of the matrix H with k zero-rows and k zero-columns interleaved, which does not impact the linear dependency relations between the non-zero rows and columns. As a consequence

$$\mathcal{R}_{E_1HF_1} = E_1\mathcal{R}_HF_1. \tag{11}$$

Now if Q_1^T is k-monotonically increasing, P_1^T is ℓ-monotonically increasing and $p = n$, then, using notations of point (b), $E_1HF_1 = E_{12}H_2F_1$ where E_{12} is in column echelon form. Thus $\mathcal{R}_{E_1HF_1} = E_1\mathcal{R}_HF_1$ for the same reason. The symmetric case where Q_1^T is p-monotonically increasing and $\ell = m$ works similarly. Combining equations (7), (10) and (11) gives $\mathcal{R}_A = \Pi_{P,Q}$.

□

4. Algorithms for the rank profile matrix

Using Theorem 23, we deduce what rank profile information is revealed by a PLUQ algorithm by the way the Search and the Permutation operations are done. Table 1 summarizes these results, and points to instances known in the literature, implementing the corresponding type of elimination. More precisely, we first distinguish in this table the ability to compute the row or column rank profile or the rank profile matrix, but we also indicate whether the resulting PLUQ decomposition preserves the monotonicity of the rows or columns. Indeed some algorithm may compute the rank profile matrix, but break the precedence relation between the linearly dependent rows or columns, making it unusable as a base case for a block algorithm of higher level.

4.1. Iterative algorithms

We start with iterative algorithms, where each iteration handles one pivot at a time. Here Theorem 23 is applied with $k = 1$, and the partial elimination represents how one pivot is being treated. The elimination of H is done by induction.

4.1.1. Row and Column order Search

The row order pivot search operation is of the form: *any non-zero element in the first non-zero row*. Each row is inspected in order, and a new row is considered only when the previous row is all zeros. With the notations of Theorem 23, this means that A_1 is the leading $\ell \times n$ sub-matrix of A, where ℓ is the index of the first non-zero row of A. When permutations P_1 and Q_1, moving the pivot from position (ℓ, j) to (k, k) are

Search	Row Perm.	Col. Perm.	Reveals	Monotonicity	Instance
Row order	Transposition	Transposition	RowRP		[1,4]
Col. order	Transposition	Transposition	ColRP		[26,4]
Lexico.	Transposition	Transposition	RowRP		[5]
	Transposition	Rotation	RowRP, ColRP, $\mathcal{R}$	Col.	here
	Rotation	Rotation	RowRP, ColRP, $\mathcal{R}$	Row, Col.	here
Rev. lexico.	Transposition	Transposition	ColRP		[5]
	Rotation	Transposition	RowRP, ColRP, $\mathcal{R}$	Row	here
	Rotation	Rotation	RowRP, ColRP, $\mathcal{R}$	Row, Col.	here
Product	Rotation	Transposition	RowRP	Row	here
	Transposition	Rotation	ColRP	Col	here
	Rotation	Rotation	RowRP, ColRP, $\mathcal{R}$	Row, Col.	[8]

Table 1. Pivoting Strategies revealing rank profiles

transpositions, the matrix Π_{P_1,Q_1} is the element $E_{\ell,j}$ of the canonical basis. Its row rank profile is (ℓ) which is that of the $\ell \times n$ leading sub-matrix A_1. Finally, the permutation P_1 is ℓ-monotonically increasing, and Theorem 23 case (b) can be applied to prove by induction that any such algorithm will reveal the row rank profile: $\text{RowRP}(A) = \text{RowSupp}(\Pi_{P,Q})$. The case of the column order search is similar.

4.1.2. Lexicographic order based pivot search

In this case the Pivot Search operation is of the form: *first non-zero element in the first non-zero row*. The lexicographic order being compatible with the row order, the above results hold when transpositions are used and the row rank profile is revealed. If in addition column rotations are used, $Q_1 = R_{1,j}$ which is 1-monotonically increasing. Now $\Pi_{P_1,Q_1} = E_{\ell,j}$ which is the rank profile matrix of the $\ell \times n$ leading sub-matrix A_1 of A. Theorem 23 case (d) can be applied to prove by induction that any such algorithm will reveal the rank profile matrix: $\mathcal{R}_A = \Pi_{P,Q}$. Lastly, the use of row rotations, ensures that the order of the linearly dependent rows will be preserved as well. Algorithm 2 is an instance of Gaussian elimination with a lexicographic order search and rotations for row and column permutations.

The case of the reverse lexicographic order search is similar. As an example, the algorithm in [5, Algorithm 2.14] is based on a reverse lexicographic order search but with transpositions for the row permutations. Hence it only reveals the column rank profile.

4.1.3. Product order based pivot search

The search here consists in finding any non-zero element $A_{\ell,p}$ such that the $\ell \times p$ leading sub-matrix A_1 of A is all zeros except this coefficient. If the row and column permutations are the rotations $R_{1,\ell}$ and $R_{1,p}$, we have $\Pi_{P_1,Q_1} = E_{\ell,p} = \mathcal{R}_{A_1}$. Theorem 23 case (d) can be applied to prove by induction that any such algorithm will reveal the rank profile matrix: $\mathcal{R}_A = \Pi_{P,Q}$. An instance of such an algorithm is given in [8, Algorithm 2]. If P_1 (resp. Q_1) is a transposition, then Theorem 23 case (c) (resp. case (b)) applies to show by induction that the columns (resp. row) rank profile is revealed.

4.2. Recursive algorithms

A recursive Gaussian elimination algorithm can either split one of the row or column dimension, cutting the matrix in wide or tall rectangular slabs, or split both dimensions, leading to a decomposition into tiles.

4.2.1. Slab recursive algorihtms

Most algorithms computing rank profiles are slab recursive [1,26,5,4]. When the row dimension is split, this means that the search space for pivots is the whole set of columns, and Theorem 23 applies with $p = n$. This corresponds to a either a row or a lexicographic order. From case(b), one shows that, with transpositions, the algorithm recovers the row rank profile, provided that the base case does. If in addition, the elementary column permutations are rotations, then case (d) applies and the rank profile matrix is recovered. Finally, if rows are also permuted by monotonically increasing permutations, then the PLUQ decomposition also respects the monotonicity of the linearly dependent rows and columns. The same reasoning holds when splitting the column dimension.

4.2.2. Tile recursive algorithms

Tile recursive Gaussian elimination algorithms [8,14,27] are more involved, especially when dealing with rank deficiencies. Algorithm 1 recalls the tile recursive algorithm that the authors proposed in [8].

Here, the search area A_1 has arbitrary dimensions $\ell \times p$, often specialized as $m/2 \times n/2$. As a consequence, the pivot search can not satisfy neither row, column, lexicographic or reverse lexicographic orders. Now, if the pivots selected in the elimination of A_1 minimize the product order, then they necessarily also respect this order as pivots of the whole matrix A. Now, from (a), the remaining matrix H writes $H = \begin{bmatrix} 0_{(\ell-k)\times(p-k)} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$ and its elimination is done by two independent eliminations on the blocks H_{12} and H_{21}, followed by some update of H_{22} and a last elimination on it. Here again, pivots minimizing the row order on H_{21} and H_{12} are also pivots minimizing this order for H, and so are those of the fourth elimination. Now the block row and column permutations used in [8, Algorithm 1] to form the PLUQ decomposition are r-monotonically increasing. Hence, from case (d), the algorithm computes the rank profile matrix and preserves the monotonicity. If only one of the row or column permutations are rotations, then case (b) or (c) applies to show that either the row or the column rank profile is computed.

5. Improvements in practice

In [8], we identified the ability to recover the rank profile matrix via the use of the product order search and of rotations. Hence we proposed an implementation combining a tile recursive algorithm and an iterative base case, using these search and permutation strategies.

The analysis of sections 3.2 and 4 shows that other pivoting strategies can be used to compute the rank profile matrix, and preserve the monotonicity. We present here a new base case algorithm and its implementation over a finite field that we wrote in the `FFLAS-FFPACK` library [1] . It is based on a lexicographic order search and row and column

[1] FFLAS-FFPACK Git rev. d420b4b, http://linbox-team.github.io/fflas-ffpack/, linked against OpenBLAS-v0.2.9.

Algorithm 1 PLUQ

Input: $A = (a_{ij})$ a $m \times n$ matrix over a field
Output: P, Q: $m \times m$ and $n \times n$ permutation matrices, r, the rank of A
Output: $A \leftarrow \left[\begin{smallmatrix} L\backslash U & V \\ M & 0 \end{smallmatrix}\right]$ where L and U are $r \times r$ resp. unit lower and upper triangular, and $A = P \left[\begin{smallmatrix} L \\ M \end{smallmatrix}\right] \left[\begin{smallmatrix} U & V \end{smallmatrix}\right] Q$.

if $\min(m, n) \leq$ Threshold **then**
  Call a base case implementation
end if

Split $A = \left[\begin{smallmatrix} A_1 & A_2 \\ A_3 & A_4 \end{smallmatrix}\right]$ where A_1 is $\lfloor \frac{m}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$.

Decompose $A_1 = P_1 \left[\begin{smallmatrix} L_1 \\ M_1 \end{smallmatrix}\right] \left[\begin{smallmatrix} U_1 & V_1 \end{smallmatrix}\right] Q_1$ ▷ PLUQ(A_1)

$\left[\begin{smallmatrix} B_1 \\ B_2 \end{smallmatrix}\right] \leftarrow P_1^T A_2$ ▷ PermR(A_2, P_1^T)

$\left[\begin{smallmatrix} C_1 & C_2 \end{smallmatrix}\right] \leftarrow A_3 Q_1^T$ ▷ PermC(A_3, Q_1^T)

Here $A = \left[\begin{smallmatrix} L_1\backslash U_1 & V_1 & B_1 \\ M_1 & 0 & B_2 \\ C_1 & C_2 & A_4 \end{smallmatrix}\right]$.

$D \leftarrow L_1^{-1} B_1$ ▷ TRSM(L_1, B_1)

$E \leftarrow C_1 U_1^{-1}$ ▷ TRSM(C_1, U_1)

$F \leftarrow B_2 - M_1 D$ ▷ MM(B_2, M_1, D)

$G \leftarrow C_2 - E V_1$ ▷ MM(C_2, E, V_1)

$H \leftarrow A_4 - E D$ ▷ MM(A_4, E, D)

Here $A = \left[\begin{smallmatrix} L_1\backslash U_1 & V_1 & D \\ M_1 & 0 & F \\ \hline E & G & H \end{smallmatrix}\right]$.

Decompose $F = P_2 \left[\begin{smallmatrix} L_2 \\ M_2 \end{smallmatrix}\right] \left[\begin{smallmatrix} U_2 & V_2 \end{smallmatrix}\right] Q_2$ ▷ PLUQ(F)

Decompose $G = P_3 \left[\begin{smallmatrix} L_3 \\ M_3 \end{smallmatrix}\right] \left[\begin{smallmatrix} U_3 & V_3 \end{smallmatrix}\right] Q_3$ ▷ PLUQ(G)

$\left[\begin{smallmatrix} H_1 & H_2 \\ H_3 & H_4 \end{smallmatrix}\right] \leftarrow P_3^T H Q_2^T$ ▷ PermR(H, P_3^T); PermC(H, Q_2^T)

$\left[\begin{smallmatrix} E_1 \\ E_2 \end{smallmatrix}\right] \leftarrow P_3^T E$ ▷ PermR(E, P_3^T)

$\left[\begin{smallmatrix} M_{11} \\ M_{12} \end{smallmatrix}\right] \leftarrow P_2^T M_1$ ▷ PermR(M_1, P_2^T)

$\left[\begin{smallmatrix} D_1 & D_2 \end{smallmatrix}\right] \leftarrow D Q_2^T$ ▷ PermR(D, Q_2^T)

$\left[\begin{smallmatrix} V_{11} & V_{12} \end{smallmatrix}\right] \leftarrow V_1 Q_3^T$ ▷ PermR(V_1, Q_3^T)

Here $A = \left[\begin{smallmatrix} L_1\backslash U_1 & V_{11} & V_{12} & D_1 & D_2 \\ M_{11} & 0 & 0 & L_2\backslash U_2 & V_2 \\ M_{12} & 0 & 0 & M_2 & 0 \\ E_1 & L_3\backslash U_3 & V_3 & H_1 & H_2 \\ E_2 & M_3 & 0 & H_3 & H_4 \end{smallmatrix}\right]$.

$I \leftarrow H_1 U_2^{-1}$ ▷ TRSM(H_1, U_2)

$J \leftarrow L_3^{-1} I$ ▷ TRSM(L_3, I)

$K \leftarrow H_3 U_2^{-1}$ ▷ TRSM(H_3, U_2)

$N \leftarrow L_3^{-1} H_2$ ▷ TRSM(L_3, H_2)

$O \leftarrow N - J V_2$ ▷ MM(N, J, V_2)

$R \leftarrow H_4 - K V_2 - M_3 O$ ▷ MM(H_4, K, V_2); MM(H_4, M_3, O)

Decompose $R = P_4 \left[\begin{smallmatrix} L_4 \\ M_4 \end{smallmatrix}\right] \left[\begin{smallmatrix} U_4 & V_4 \end{smallmatrix}\right] Q_4$ ▷ PLUQ(R)

$\left[\begin{smallmatrix} E_{21} & M_{31} & 0 & K_1 \\ E_{22} & M_{32} & 0 & K_2 \end{smallmatrix}\right] \leftarrow P_4^T \left[\begin{smallmatrix} E_2 & M_3 & 0 & K \end{smallmatrix}\right]$ ▷ PermR

$\left[\begin{smallmatrix} D_{21} & D_{22} \\ V_{21} & V_{22} \\ 0 & 0 \\ O_1 & O_2 \end{smallmatrix}\right] \leftarrow \left[\begin{smallmatrix} D_2 \\ V_2 \\ 0 \\ O \end{smallmatrix}\right] Q_4^T$ ▷ PermC

Here $A = \begin{bmatrix} L_1\backslash U_1 & V_{11} & V_{12} & D_1 & D_{21} & D_{22} \\ M_{11} & 0 & 0 & L_2\backslash U_2 & V_{21} & V_{22} \\ M_{12} & 0 & 0 & M_2 & 0 & 0 \\ E_1 & L_3\backslash U_3 & V_3 & I & O_1 & O_2 \\ E_{21} & M_{31} & 0 & K_1 & L_4\backslash U_4 & V_4 \\ E_{22} & M_{32} & 0 & K_2 & M_4 & 0 \end{bmatrix}$.

$S \leftarrow \begin{bmatrix} I_{r_1+r_2} & & & & & \\ & & I_{k-r_1-r_2} & & & \\ & I_{r_3+r_4} & & & & \\ & & & & & I_{m-k-r_3-r_4} \end{bmatrix}$

$T \leftarrow \begin{bmatrix} I_{r_1} & & & & & \\ & & & I_{r_2} & & \\ & I_{r_3} & & & & \\ & & & & I_{r_4} & \\ & & I_{k-r_1-r_3} & & & \\ & & & & & I_{n-k-r_2-r_4} \end{bmatrix}$

$P \leftarrow \text{Diag}(P_1 \begin{bmatrix} I_{r_1} & \\ & P_2 \end{bmatrix}, P_3 \begin{bmatrix} I_{r_3} & \\ & P_4 \end{bmatrix})S$

$Q \leftarrow T\text{Diag}(\begin{bmatrix} I_{r_1} & \\ & Q_3 \end{bmatrix} Q_1, \begin{bmatrix} I_{r_2} & \\ & Q_4 \end{bmatrix} Q_2)$

$A \leftarrow S^T A T^T$ ▷ $\texttt{PermR}(A, S^T); \texttt{PermC}(A, T^T)$

Here $A = \begin{bmatrix} L_1\backslash U_1 & D_1 & V_{11} & D_{21} & V_{12} & D_{22} \\ M_{11} & L_2\backslash U_2 & 0 & V_{21} & 0 & V_{22} \\ E_1 & I & L_3\backslash U_3 & O_1 & V_3 & O_2 \\ E_{21} & K_1 & M_{31} & L_4\backslash U_4 & 0 & V_4 \\ M_{12} & M_2 & 0 & 0 & 0 & 0 \\ E_{22} & K_2 & M_{32} & M_4 & 0 & 0 \end{bmatrix}$

Return $(P, Q, r_1 + r_2 + r_3 + r_4, A)$

rotations. Moreover, the schedule of the update operations is that of a Crout elimination, for it reduces the number of modular reductions, as shown in [28, § 3.1]. Algorithm 2 summarizes this variant.

Algorithm 2 Crout variant of PLUQ with lexicographic search and column rotations

1: $k \leftarrow 1$
2: **for** $i = 1 \dots m$ **do**
3: $\quad A_{i,k..n} \leftarrow A_{i,k..n} - A_{i,1..k-1} \times A_{1..k-1,k..n}$
4: $\quad$ **if** $A_{i,k..n} = 0$ **then**
5: $\quad\quad$ Loop to next iteration
6: $\quad$ **end if**
7: $\quad$ Let $A_{i,s}$ be the left-most non-zero element of row i.
8: $\quad A_{i+1..m,s} \leftarrow A_{i+1..m,s} - A_{i+1..m,1..k-1} \times A_{1..k-1,s}$
9: $\quad A_{i+1..m,s} \leftarrow A_{i+1..m,s}/A_{i,s}$
10: $\quad$ Bring $A_{*,s}$ to $A_{*,k}$ by column rotation
11: $\quad$ Bring $A_{i,*}$ to $A_{k,*}$ by row rotation
12: $\quad k \leftarrow k + 1$
13: **end for**

In the following experiments, we report the median of the real time for up to 60 computations (depending on the size of the instance), with $n \times n$ matrices of rank $r = n/2$ (left) or $r = n/8$ (right). In order to ensure that the row and column rank profiles of these matrices are random, we construct them as the product $A = L\mathcal{R}U$, where L and U are random non-singular lower and upper triangular matrices and $\mathcal{R}$ is an $m \times n$ r-sub-permutation matrix whose non-zero elements positions are chosen uniformly at

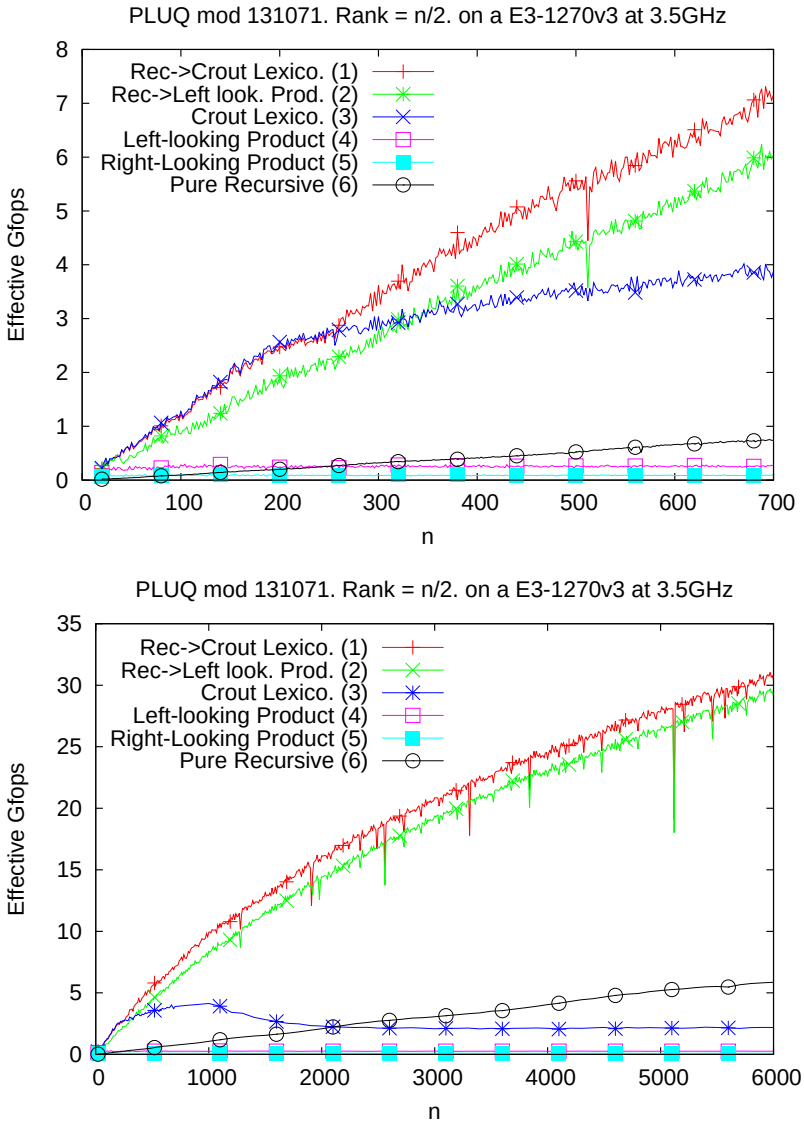


Fig. 1. Computation speed of PLUQ decomposition base cases.

random. The effective speed is obtained by dividing an estimate of the arithmetic cost $(2mnr + 2/3r^3 - r^2(m+n))$ by the computation time.

Figure 1 shows the computation speed of Algorithm 2 (3), compared to that of the pure recursive algorithm (6), and to our previous base case algorithm [8], using a product order search, and either a left-looking (4) or a right-looking (5) schedule. At $n = 200$, the left-looking variant (4) improves over the right looking variant (5) by a factor of about 3.18 as it performs fewer modular reductions. Then, the Crout variant (3) again improves variant (4) by a factor of about 9.29. Lastly we also show the speed of the final implementation, formed by the tile recursive algorithm cascading to either the Crout

base case (1) or the left-looking one (2). The threshold where the cascading to the base case occurs is experimentally set to its optimum value, i.e. 256 for variant (1) and 40 for variant (2). This illustrates that the gain on the base case efficiency leads to a higher threshold, and improves the efficiency of the cascade implementation (by an additive gain of about 1 effective Gfops in the range of dimensions considered).

Note that the experiments reported here differ from that of [9]. We found a mistake in the code generating the random matrices, making their row rank profile generic, which led to a reduced amount of row permutations and therefore a much higher computation speed. After fixing this issue, we noticed a slow-down of most base case implementations, but all variants still compare in the same way. The spikes in curves (1) and (2) showing drops in computation speed in a few points are reproducible, and seem to correspond to pathological dimensions, for which the blocking generates worst case cache misses scenari.

6. Relations with other triangularizations

6.1. The LEU decomposition

The LEU decomposition of [14] involves a lower triangular matrix L, an upper triangular matrix U and a r-sub-permutation matrix E. It is also called the *modified Bruhat decomposition* in [20]. Theorem 24 shows how to recover an LEU decomposition from a PLUQ decomposition revealing the rank profile matrix.

Theorem 24. *Let $A = PLUQ$ be a PLUQ decomposition revealing the rank profile matrix ($\Pi_{P,Q} = \mathcal{R}_A$). Then an LEU decomposition of A with $E = \mathcal{R}_A$ is obtained as follows (only using row and column permutations):*

$$A = \underbrace{P \begin{bmatrix} L & 0_{m\times(m-r)} \end{bmatrix} P^T}_{\overline{L}} \underbrace{P \begin{bmatrix} I_r & \\ & 0 \end{bmatrix} Q}_{E} \underbrace{Q^T \begin{bmatrix} U \\ 0_{(n-r)\times n} \end{bmatrix} Q}_{\overline{U}} \tag{12}$$

Proof. First $E = P \left[\begin{smallmatrix} I_r & \\ & 0 \end{smallmatrix}\right] Q = \Pi_{P,Q} = \mathcal{R}_A$. Then there only needs to show that $\overline{L}$ is lower triangular and $\overline{U}$ is upper triangular. Suppose that $\overline{L}$ is not lower triangular, let i be the first row index such that $\overline{L}_{i,j} \neq 0$ for some $i < j$. First $j \in \text{RowRP}(A)$ since the non-zero columns in $\overline{L}$ are placed according to the first r values of P. Remarking that $A = P \left[\begin{smallmatrix} L & 0_{m\times(n-r)} \end{smallmatrix}\right] \left[\begin{smallmatrix} U & \\ 0 & I_{n-r} \end{smallmatrix}\right] Q$, and since right multiplication by a non-singular matrix does not change row rank profiles, we deduce that $\text{RowRP}(\Pi_{P,Q}) = \text{RowRP}(A) = \text{RowRP}(\overline{L})$. If $i \notin \text{RowRP}(A)$, then the i-th row of $\overline{L}$ is linearly dependent with the previous rows, but none of them has a non-zero element in column $j > i$. Hence $i \in \text{RowRP}(A)$.

Let (a, b) be the position of the coefficient $\overline{L}_{i,j}$ in L, that is $a = \sigma_P^{-1}(i), b = \sigma_P^{-1}(j)$. Let also $s = \sigma_Q(a)$ and $t = \sigma_Q(b)$ so that the pivots at diagonal position a and b in L respectively correspond to ones in $\mathcal{R}_A$ at positions (i, s) and (j, t). Consider the $\ell \times p$ leading sub-matrices A_1 of A where $\ell = \max_{x=1..a-1}(\sigma_P(x))$ and $p = \max_{x=1..a-1}(\sigma_Q(x))$. On one hand (j, t) is an index position in A_1 but not (i, s), since otherwise $\text{rank}(A_1) = b$. Therefore, $(i, s) \not\prec_{prod} (j, t)$, and $s > t$ as $i < j$. As coefficients (j, t) and (i, s) are pivots in $\mathcal{R}_A$ and $i < j$ and $t < s$, there can not be a non-zero element above (j, t) at row i when it is chosen as a pivot. Hence $\overline{L}_{i,j} = 0$ and $\overline{L}$ is lower triangular. The same reasoning applies to show that $\overline{U}$ is upper triangular. □

Remark 25. Note that the LEU decomposition with $E = \mathcal{R}_A$ is not unique, even for invertible matrices. As a counter-example, the following decomposition holds for any $a \in \mathrm{K}$:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ a & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -a \\ 0 & 1 \end{bmatrix} \tag{13}$$

However, uniqueness of the full rank LEU decomposition can be achieved by imposing an additional constraint on the L matrix: there is a unique LEU decomposition where L is such that $E^T L E$ is also lower triangular [15, Th. 2.1]. We remark that for the LEU decomposition proposed in (12), this condition becomes:

$$\text{the matrix } E^T \bar{L} E = Q^T_{*,1..r} L_{1..r,1..r} Q_{*,1..r} \text{ must be lower triangular.} \tag{14}$$

Whenever $Q_{*,1..r}$ is in column echelon form, which means that the sequence of the pivot's column positions is monotonically increasing, then the above matrix is lower triangular. A sufficient condition to ensure it, is thus that the pivoting strategy of the corresponding PLUQ decomposition minimizes the reverse lexicographic order.

Corollary 26. *Let $A = PLUQ$ be a full-rank PLUQ decomposition computed with a pivoting strategy minimizing the reverse lexicographic order and performing row rotations. Then the LEU decomposition of equation* (12) *is the unique modified Bruhat decomposition as defined in [15, Theorem 2.1].*

6.2. The Bruhat decomposition

The Bruhat decomposition, that has inspired Malaschonok's LEU decomposition [14], is another decomposition with a central permutation matrix [10,11,13,12]. Bruhat's theorem is stated in terms of Weyl groups but reduces to the following:

Theorem 27 ([13])**.** *Any invertible matrix A can be written as $A = VPU$ for V and U uppper triangular invertible matrices and P a permutation matrix. The latter decomposition is called the* Bruhat decomposition *of A.*

It was then naturally extended to singular square matrices in [12] and an algorithm over principal ideals domains given in [29]. Corollary 28 generalizes it to matrices with arbitrary dimensions, and relates it to the PLUQ decomposition.

Corollary 28. *Any $m \times n$ matrix of rank r has a VPU decomposition, where V and U are upper triangular matrices, and P is a r-sub-permutation matrix.*

Proof. Let J_n be the unit anti-diagonal matrix. From the LEU decomposition of $J_n A$, we have $A = \underbrace{J_n L J_n}_{V} \underbrace{J_n E}_{P} U$ where V is upper triangular. □

Remark 29. [15] gives also a unique *generalized Bruhat decomposition*, $A = XPY$, but with non square and non triangular matrices X, Y. There, P is called the *Bruhat permutation* but contains only the r non zero rows and columns of the rank profile matrix. We show in Section 6.5, that X and Y are actual row and column echelon forms of the matrix.

6.3. Relation to LUP and PLU decompositions

The LUP decomposition $A = LUP$ only exists for matrices with generic row rank profile (including matrices with full row rank). Corollary 30 shows upon which condition the permutation matrix P equals the rank profile matrix $\mathcal{R}_A$. Note that although the rank profile of A is trivial in such cases, the matrix $\mathcal{R}_A$ still carries important information on the row and column rank profiles of all leading sub-matrices of A.

Corollary 30. *Let A be an $m \times n$ matrix.*

If A has generic column rank profile, then any PLU decomposition $A = PLU$ computed using reverse lexicographic order search and row rotations is such that $\mathcal{R}_A = P \begin{bmatrix} I_r \\ 0 \end{bmatrix}$. In particular, $P = \mathcal{R}_A$ if $r = m$.

If A has generic row rank profile, then any LUP decomposition $A = LUP$ computed using lexicographic order search and column rotations is such that $\mathcal{R}_A = \begin{bmatrix} I_r & 0 \end{bmatrix} P$. In particular, $P = \mathcal{R}_A$ if $r = n$.

Proof. Consider A has generic column rank profile. From table 1, any PLUQ decomposition algorithm with a reverse lexicographic order based search and rotation based row permutation is such that $\Pi_{P,Q} = P \begin{bmatrix} I_r \\ \end{bmatrix} Q = \mathcal{R}_A$. Since the search follows the reverse lexicographic order and the matrix has generic column rank profile, no column will be permuted in this elimination, and therefore $Q = I_n$. The same reasoning hold for when A has generic row rank profile. □

Note that the L and U factors in a PLU decomposition are uniquely determined by the permutation P. Hence, when the matrix has full row rank, $P = \mathcal{R}_A$ and the decomposition $A = \mathcal{R}_A LU$ is unique. Similarly the decomposition $A = LU\mathcal{R}_A$ is unique when the matrix has full column rank. Now when the matrix is rank deficient with generic row rank profile, there is no longer a unique PLU decomposition revealing the rank profile matrix: any permutation applied to the last $m-r$ columns of P and the last $m-r$ rows of L yields a PLU decomposition where $\mathcal{R}_A = P \begin{bmatrix} I_r \\ \end{bmatrix}$.

Lastly, we remark that the only situation where the rank profile matrix $\mathcal{R}_A$ can be read directly as a sub-matrix of P or Q is as in corollary 30, when the matrix A has generic row or column rank profile. Consider a PLUQ decomposition $A = PLUQ$ revealing the rank profile matrix ($\mathcal{R}_A = P \begin{bmatrix} I_r \\ \end{bmatrix} Q$) such that $\mathcal{R}_A$ is a sub-matrix of P. This means that $P = \mathcal{R}_A + S$ where S has disjoint row and column support with $\mathcal{R}_A$. We have $\mathcal{R}_A = (\mathcal{R}_A + S) \begin{bmatrix} I_r \\ \end{bmatrix} Q = (\mathcal{R}_A + S) \begin{bmatrix} Q_1 \\ 0_{(n-r)\times n} \end{bmatrix}$. Hence $\mathcal{R}_A (I_n - \begin{bmatrix} Q_1 \\ 0_{(n-r)\times n} \end{bmatrix}) = S \begin{bmatrix} Q_1 \\ 0_{(n-r)\times n} \end{bmatrix}$ but the row support of these matrices are disjoint, hence $\mathcal{R}_A \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix} = 0$ which implies that A has generic column rank profile. Similarly, one shows that $\mathcal{R}_A$ can be a sub-matrix of Q only if A has a generic row rank profile.

6.4. Computing Echelon forms

Usual algorithms computing an echelon form [1,26,5,4] use a slab block decomposition (with row or lexicographic order search), which implies that pivots appear in the order of the echelon form. The column echelon form is simply obtained as $C = PL$ from the PLUQ decomposition.

We extend here this approach to any PLUQ decomposition that reveals the rank profile matrix ($\mathcal{R}_A = \Pi_{P,Q}$) and show how both the row and column echelon forms can be recovered by only permutations on the L and U matrices.

Besides the ability to compute both echelon forms from one Gaussian elimination, this enables one to choose from a broader range of algorithmic variants for this Gaussian elimination.

Consider a PLUQ decomposition $A = PLUQ$ revealing the rank profile matrix and let $P_C = P_{*,1..r}$ be the first r columns of P. The ones in P_C indicate the row position of the pivots in the column echelon form of A, but they may not appear in the same column order. There exist an $r \times r$ permutation matrix S sorting the columns of P_C, i.e. such that $P_S = P_C S$ is in column echelon.

Lemma 31. *The matrix* $\begin{bmatrix} PLS & 0_{m\times n-r} \end{bmatrix}$ *is a column echelon form of* A.

Proof. $PLS = P \begin{bmatrix} L & 0_{m\times(m-r)} \end{bmatrix} P^T P \begin{bmatrix} S \\ 0_{(m-r)\times r} \end{bmatrix} = \bar{L} P_S$. From Theorem 24, the matrix $\bar{L} = P \begin{bmatrix} L & 0_{m\times(m-r)} \end{bmatrix} P^T$ is lower triangular. Multiplying $\bar{L}$ on the right by P_S simply removes the zero columns in $\bar{L}$ putting it in column echelon form. Now the relation $A = \begin{bmatrix} PLS & 0_{m\times(n-r)} \end{bmatrix} \begin{bmatrix} S^T & \\ & I_{n-r} \end{bmatrix} \begin{bmatrix} U \\ 0_{(n-r)\times r} \; I_{n-r} \end{bmatrix} Q$ shows that $\begin{bmatrix} PLS & 0_{m\times(n-r)} \end{bmatrix}$ is right-equivalent to A and is therefore a column echelon form for A. □

Equivalently, the same reasoning applies for the recovery of a row echelon form of A. Algorithm 3 summarizes how a row and a column echelon form of A can be computed by sorting the first r values of the permutations σ_P and σ_Q.

Algorithm 3 Echelon forms from a PLUQ decomposition

Input: P, L, U, Q, a PLUQ decomposition of A with $\mathcal{R}_A = \Pi_{P,Q}$
Output: C: a column echelon form of A
Output: R: a row echelon form of A
1: $(p_1, .., p_r) = \text{Sort}(\sigma_P(1), .., \sigma_P(r))$
2: $(q_1, .., q_r) = \text{Sort}(\sigma_Q(1), .., \sigma_Q(r))$
3: $\tau = (\sigma_P^{-1}(p_1), .., \sigma_P^{-1}(p_r)) \in S_r$
4: $\chi = (\sigma_Q^{-1}(q_1), .., \sigma_Q^{-1}(q_r)) \in S_r$
5: $S_\tau = P(\tau)$
6: $S_\chi = P(\chi)^T$
7: **Return** $C \leftarrow P \begin{bmatrix} LS_\tau & 0_{m\times(n-r)} \end{bmatrix}$ and $R \leftarrow \begin{bmatrix} S_\chi U \\ 0_{(m-r)\times n} \end{bmatrix} Q$

Remark 32. Remark that a row and column echelon form of the $i \times j$ leading submatrix can be computed by removing rows of PL below index i and filtering out the pivots of column index greater than j. The latter is achieved by replacing lines 1 and 2 in Algorithm 3 by:

1': $(p_1, .., p_s) = \text{Sort}(\{\sigma_P(k) : \sigma_Q(k) \leq j\})$
2': $(q_1, .., q_s) = \text{Sort}(\{\sigma_Q(k) : \sigma_P(k) \leq i\})$.

6.5. The generalized Bruhat decomposition

The generalization of the Bruhat decomposition for rank deficient matrices of [15, Theorem 2.4] is of the form $A = XFY$ where X is in column echelon form, Y in row echelon form, and F is a $r \times r$ permutation matrix.

Such a decomposition immediately follows from the echelon forms computed by algorithm 3. Indeed, by Lemma 31, PLS_τ is in column echelon form and $S_\chi UQ$ is in row echelon form. The identity

$$A = \underbrace{(PLS_\tau)}_{X} \underbrace{S_\tau^T S_\chi^T}_{F} \underbrace{(S_\chi UQ)}_{Y} \tag{15}$$

then directly gives an $A = XFY$ decomposition.

In a similar way as in the full-rank case (see Section 6.1) [15, Theorem 2.4] requires an additional condition on the X matrix to ensure the uniqueness of the decomposition: $F^T X_{\mathrm{RowRP}(X),*} F$ must be lower triangular, where $X_{I,*}$ denotes the lower triangular submatrix of X formed by the rows of its row rank profile. This condition is not satisfied in general by the X and Y matrices computed from Algorithm 3 and a PLUQ decomposition revealing the rank profile matrix. As a counter example, the matrix $\left[\begin{smallmatrix} 0 & 1 \\ 1 & a \end{smallmatrix}\right]$ has two PLUQ decompositions revealing the rank profile matrix $F = J_2 = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$: $A = J_2 I_2 \left[\begin{smallmatrix} 1 & a \\ 0 & 1 \end{smallmatrix}\right] I_2$ and $A = I_2 \left[\begin{smallmatrix} 1 & 0 \\ a & 1 \end{smallmatrix}\right] I_2 J_2$. For the latter one, $F^T X F = \left[\begin{smallmatrix} 1 & a \\ 0 & 1 \end{smallmatrix}\right]$ which is not lower triangular.

In a same way as in equation (14), we derive an equivalent condition for the uniqueness of the generalized Bruhat decomposition. We first need the following Lemma.

Lemma 33. *Let C be an $m \times r$ matrix in column echelon form, and P the $m \times r$ r-sub-permutation matrix with ones at the location of the pivots of C. Then*

$$C_{RowRP(X),*} = P^T C.$$

As $P_{*,1..r} S_\tau$ is the r-sub-permutation matrix with ones on the locations of the pivots of the column echelon form X, we deduce that

$$F^T X_{\mathrm{RowRP}(X),*} F = S_\chi S_\tau S_\tau^T (P_{*,1..r})^T P L S_\tau S_\tau^T S_\chi^T = S_\chi L_{1..r,1..r} S_\chi^T.$$

Again, a sufficient condition for this matrix to be lower triangular is that $S_\chi = I_r$ which happens if the PLUQ decomposition has been computed using a pivoting respecting the reverse lexicographic order.

Corollary 34. *Let $A = PLUQ$ be a PLUQ decomposition computed with a pivoting strategy minimizing the reverse lexicographic order and performing row rotations. Then the XFY decomposition of equation* (15) *is the unique generalized Bruhat decomposition as defined in [15, Theorem 2.4].*

7. Improvement for low rank matrices

7.1. Storjohann and Yang's algorithm

An alternative way to compute row and column rank profiles has recently been proposed by Storjohann and Yang in [18,19], reducing the time complexity from a deterministic $O(mnr^{\omega-2})$ to a Monte Carlo probabilistic $2r^3 + O(r^2(\log m + \log n) + mn)$ in [18]

first, and to $(r^\omega + mn)^{1+o(1)}$ in [19]. We show how these results can be extended to the computation of the rank profile matrix.

The technique originates from the oracle system solving technique of [30, §2]. There, a linear system is solved or proved to be inconsistent, by the incremental construction of a non-singular sub-matrix of the system's matrix, and the computation of its inverse by rank one updates.

Applied to a right-hand-side vector b sampled uniformly from the column space of a matrix, this technique is used to incrementally build a list of row indices $\mathcal{P} = [i_1, \ldots, i_s]$ and of column indices $\mathcal{Q} = [j_1, \ldots j_s]$, for $s = 1, \ldots, r$, forming an incrementally growing invertible sub-matrix $A_{\mathcal{P},\mathcal{Q}}$. Each coordinate (i_s, j_s) is that of a pivot as in a standard Gaussian elimination algorithm, but the complexity improvement comes from the fact that the elimination is only performed on this $s \times s$ submatrix, and not on the linearly dependent rows and columns of the matrix A. To achieve this, the search for pivots is done by limiting the Schur complement computation to the last column of the augmented system $\left[\begin{array}{c|c} A & b \end{array}\right]$:

$$b - A_{*,\mathcal{Q}} \cdot (A_{\mathcal{P},\mathcal{Q}})^{-1} \cdot b_{\mathcal{P}}.$$

Under a randomized assumption, the first non-zero component of this vector indicates the value i_s of the next linearly independent row. A similar approach is used for determining the value of j_s.

As shown in [18, Theorem 6], the algorithm ends up computing the row rank profile in $\mathcal{P}$ and the column rank profile in $\mathcal{Q}$. In fact, it computes all information of the rank profile matrix.

Theorem 35. *The r-permutation matrix whose ones are located at positions (i_s, j_s) for $1 \leq s \leq r$ is precisely the rank profile matrix of A.*

Proof. In this algorihtm, the pivot search looks for the first non-zero element in the Schur complement of b which, with probability $1 - 1/\#\mathrm{K}$, will correspond to the first linearly independent row. The exact Schur complement can then be computed on this row only:

$$A_{i_s,*} - A_{i_s,\mathcal{Q}} \cdot (A_{\mathcal{P},\mathcal{Q}})^{-1} \cdot A_{\mathcal{P},*},$$

so as to find the column position j_s of the first non zero element in it. This search strategy minimizes the Lexicographic Order of the pivot coordinates. No permutation strategy is being used, instead the columns where pivots have already been found are zeroed out in the Schur complement computation, which has the same effect as a Column rotation. From the 4th line of Table 1, this pivoting strategy reveals the rank profile matrix. □

The complexity of the direct Monte Carlo algorithm, in $O((m+n)r^2)$, hence directly applies for the computation of the rank profile matrix. This complexity is reduced to $2r^3 + O(r^2(\log n + \log m) + mn)$ by the introduction of linear independence oracles in a divide and conquer scheme. Upon success of the probabilistic asumption, the values for $\mathcal{P}$ and $\mathcal{Q}$ are unchanged. Consequently, [18, Theorem 19, Corollary 21] also hold for the computation of the rank profile matrix.

Corollary 36. *There exist a Monte Carlo algorithm computing the rank profile matrix that has running time bounded by $2r^3 + (r^2 + m + n + |A|)^{1+o(1)}$, where $|A|$ denotes the number of non-zero elements in A.*

The r^3 term in this complexity is from the iterative construction of the inverses of the non-singular sub-matrices of order s for $1 \leq s \leq r$, by rank one updates. To reduce this complexity to $O(r^\omega)$, Storjohann and Yang propose in [19] a relaxed computation of this online matrix inversion. In order to group arithmetic operations into matrix multiplications, their approach is to anticipate part of the updates on the columns that will be appended in the future. This however requires that one has an a priori knowledge on the position of r linearly independent columns of the initial matrix A and, even further, that the sub-matrix formed by these linearly independent columns has generic rank profile. The first condition is ensured by the Monte Carlo selection of linearly independent columns of [17, Theorem 2.11] and the second by the use of a lower triangular Toeplitz preconditioner as in [31, Theorem 2]. In the process, the row rank profile can be recovered, but all information on the column rank profile is lost, and the rank profile matrix can thus not be recovered.

Yet, this algorithm can be run twice (once on A and once on A^T), to recover the row and the column rank profiles, and extract the corresponding $r \times r$ invertible sub-matrix $A_{\mathcal{P},\mathcal{Q}}$ of A. Following Corollary 30, it then suffices to compute a PLU decomposition of this sub-matrix, using an appropriate pivoting strategy, to recover its rank profile matrix and therefore that of A as well.

Algorithm 4 Low Rank Profile Matrix

Input: A, an $m \times n$ matrix of rank r over a field K.
Output: $\mathcal{R}_A$ the rank profile matrix of A or FAILURE

1: Compute the Row rank profile $\mathcal{P} = [i_1, \ldots, i_r]$ of A using [19]
2: Compute the Column rank profile $\mathcal{Q} = [j_1, \ldots, j_r]$ of A using [19] applied on A^T.
3: Let $B = A_{\mathcal{P},\mathcal{Q}}$
4: Compute $B = LUP$ a LUP decomposition of B using a lexicographic pivot search and rotations for the column permutations.
5: Return $\mathcal{R}_A = P_{\mathcal{P}} \begin{bmatrix} P & \\ & 0_{(m-r)\times(n-r))} \end{bmatrix} P_{\mathcal{Q}}^T$

Theorem 37. *Algorithm 4 is Monte Carlo probabilistic and computes the rank profile matrix of A in time $(r^\omega + m + n + |A|)^{1+o(1)}$ field operations in* K.

Note that the last operation consists in the introduction of zero rows and columns to expand the $r \times r$ permutation matrix P into the $m \times n$ matrix $\mathcal{R}_A$. The position of these zero rows and columns are deduced from the row rank profile $\mathcal{P}$ and the column rank profile $\mathcal{Q}$.

7.2. Online LU decomposition

The rank profile algorithms of Storjohann and Yang rely on an online computation of a matrix inverse, either by rank one updates in [18] or by a relaxed variant [19], that corresponds to an online version of the classic divide and conquer algorithm [32]. We remark that these inversion algorithms can be advantageously replaced by LU decomposition algorithms. Following the argument in [4, Fig 2], the use of LU decomposition for linear system solving offers a speed-up factor of about 3, compared to the use of a matrix inverse. Indeed, solving a system from an LU decomposition is done by two back substitutions, which has the same costs of $2n^2 + O(n)$ field operations, as applying

the inverse of the matrix to the right-hand-side vector. But the cost computing an LU decomposition is about three times as fast as computing a matrix inverse ($2/3n^3$ versus $2n^3$ when classic matrix arithmetic is used). We present now how to replace the online matrix inverse by an online LU decomposition in [18,19].

First consider the iterative construction of the LU decomposition of $A_{\mathcal{P},\mathcal{Q}}$ by rank one updates. Suppose that $A_{s-1} = LU$ with L lower triangular with a unit diagonal, and U upper triangular. Then we have

$$\left[\begin{array}{c|c} A_{s-1} & u \\ \hline v & d \end{array}\right] = \left[\begin{array}{c|c} L & \\ \hline vU^{-1} & 1 \end{array}\right] \cdot \left[\begin{array}{c|c} U & L^{-1}u \\ \hline & w \end{array}\right], \tag{16}$$

where $w = d - vU^{-1}L^{-1}u$. This rank one update of the LU decomposition of the work matrix costs $2s^2 + O(s)$ to compute for an $(s-1) \times (s-1)$ matrix $A_{\mathcal{P},\mathcal{Q}}$ (compared with $6s^2+O(s)$ in [18, Lemma 2]). The remaining of the paper can then be used, replacing every multiplication by the pre-computed matrix inverse $x \leftarrow (A_{\mathcal{P},\mathcal{Q}})^{-1} \cdot b$ by two consecutive triangular system solving: $y \leftarrow L^{-1}b; x \leftarrow U^{-1}y$. This leads to a Monte Carlo algorithm computing the rank profile matrix in time $2/3r^3 + (r^2 + m + n + |A|)^{(1+o(1))}$.

Note that the decomposition in (16) uses two kinds of matrix vector products: vU^{-1} and $L^{-1}u$ contrarily to the matrix inverse formula (17) used in [18] and [19]

$$\left[\begin{array}{c|c} A_{s-1} & u \\ \hline v & d \end{array}\right]^{-1} = \left[\begin{array}{c|c} I_{s-1} & -A_{s-1}^{-1}u_s(d - v_sA_{s-1}^{-1}u_s)^{-1} \\ \hline & (d - v_sA_{s-1}^{-1}u_s)^{-1} \end{array}\right] \cdot \left[\begin{array}{c|c} I_{s-1} & \\ \hline -v_s & 1 \end{array}\right] \cdot \left[\begin{array}{c|c} A_{s-1}^{-1} & \\ \hline & 1 \end{array}\right], \tag{17}$$

where only one matrix-vector product, $A_{s-1}^{-1}u_s$, appears. The relaxation is achieved there by anticipating part of these products into larger matrix-matrix multiplications. This requires that all of the column vectors that will be treated are known in advance: this is ensured by the selection of r linearly independent columns of [17]. Since no matrix-vector product involves row vectors v_s, it is still possible to select and add them incrementally one after the other, thus allowing to compute the row rank profile.

Now, relaxing the LU decomposition of equation (16) in a similar way would require to also anticipate computations on the rows, namely vU^{-1}. This would be either too expensive, if no pre-selection of r linearly independent rows is performed, or such a pre-selection would lose the row rank profile information that has to be computed. Hence we can not propose a similar improvement of the leading constant in the relaxed case.

References

[1] O. H. Ibarra, S. Moran, R. Hui, A generalization of the fast LUP matrix decomposition algorithm and applications, J. of Algorithms 3 (1) (1982) 45–56. doi:10.1016/0196-6774(82)90007-4.

[2] D. J. Jeffrey, LU factoring of non-invertible matrices, ACM Comm. Comp. Algebra 44 (1/2) (2010) 1–8. doi:10.1145/1838599.1838602.

[3] J.-G. Dumas, P. Giorgi, C. Pernet, Dense linear algebra over prime fields, ACM TOMS 35 (3) (2008) 1–42. doi:10.1145/1391989.1391992.

[4] C.-P. Jeannerod, C. Pernet, A. Storjohann, Rank-profile revealing Gaussian elimination and the CUP matrix decomposition, J. Symbolic Comput. 56 (2013) 46–68. doi:10.1016/j.jsc.2013.04.004.

[5] A. Storjohann, Algorithms for matrix canonical forms, Ph.D. thesis, ETH-Zentrum, Zürich, Switzerland (Nov. 2000). doi:10.3929/ethz-a-004141007.

[6] K. Klimkowski, R. A. van de Geijn, Anatomy of a parallel out-of-core dense linear solver, in: International Conference on Parallel Processing, Vol. 3, CRC Press, 1995, pp. 29–33.

[7] J.-G. Dumas, C. Pernet, Z. Sultan, Recursion based parallelization of exact dense linear algebra routines for Gaussian elimination, Parallel Computingdoi:10.1016/j.parco.2015.10.003.

[8] J.-G. Dumas, C. Pernet, Z. Sultan, Simultaneous computation of the row and column rank profiles, in: M. Kauers (Ed.), Proc. ISSAC'13, ACM Press, 2013, pp. 181–188. doi:10.1145/2465506.2465517.

[9] J.-G. Dumas, C. Pernet, Z. Sultan, Computing the rank profile matrix, in: Proceedings of the 2015 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC '15, ACM, New York, NY, USA, 2015, pp. 149–156. doi:10.1145/2755996.2756682.

[10] F. Bruhat, Sur les représentations induites des groupes de Lie, Bulletin de la Société Mathématique de France 84 (1956) 97–205.
URL http://eudml.org/doc/86911

[11] J. Della Dora, Sur quelques algorithmes de recherche de valeurs propres, Theses, Université Joseph-Fourier - Grenoble I, université : Université scientifique et Médicale de Grenoble (Jul. 1973).
URL https://tel.archives-ouvertes.fr/tel-00010274

[12] D. Y. Grigor'ev, Analogy of Bruhat decomposition for the closure of a cone of Chevalley group of a classical serie, Soviet Mathematics Doklady 23 (2) (1981) 393–397.

[13] N. Bourbaki, Groupes et Algègres de Lie, no. Chapters 4–6 in Elements of mathematics, Springer, 2008.

[14] G. I. Malaschonok, Fast generalized Bruhat decomposition, in: CASC'10, Vol. 6244 of LNCS, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 194–202. doi:10.1007/978-3-642-15274-0_16.

[15] W. Manthey, U. Helmke, Bruhat canonical form for linear systems, Linear Algebra and its Applications 425 (2–3) (2007) 261 – 282, special Issue in honor of Paul Fuhrmann. doi:10.1016/j.laa.2007.01.022.

[16] E. Miller, B. Sturmfels, Combinatorial commutative algebra, Vol. 227, Springer, 2005. doi:10.1007/b138602.

[17] H. Y. Cheung, T. C. Kwok, L. C. Lau, Fast Matrix Rank Algorithms and Applications, J. ACM 60 (5) (2013) 31:1–31:25. doi:10.1145/2528404.

[18] A. Storjohann, S. Yang, Linear Independence Oracles and Applications to Rectangular and Low Rank Linear Systems, in: Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation, ISSAC '14, ACM, New York, NY, USA, 2014, pp. 381–388. doi:10.1145/2608628.2608673.

[19] A. Storjohann, S. Yang, A Relaxed Algorithm for Online Matrix Inversion, in: Proceedings of the 2015 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC '15, ACM, New York, NY, USA, 2015, pp. 339–346. doi:10.1145/2755996.2756672.

[20] E. Tyrtyshnikov, Matrix Bruhat decompositions with a remark on the QR (GR) algorithm, Linear Algebra and its Applications 250 (1997) 61 – 68. doi:10.1016/0024-3795(95)00453-X.

[21] N. H. McCoy, Rings and ideals, Carus Monograph Series, no. 8, The Open Court Publishing Company, LaSalle, Ill., 1948.

[22] J.-G. Dumas, B. D. Saunders, G. Villard, On efficient sparse integer matrix Smith normal form computations, Journal of Symbolic Computation 32 (1/2) (2001) 71–99. doi:10.1006/jsco.2001.0451.

[23] W. C. Brown, Null ideals and spanning ranks of matrices, Communications in Algebra 26 (8) (1998) 2401–2417. doi:10.1080/00927879808826285.

[24] W. C. Brown, Matrices over Commutative Rings, Chapman & Hall Pure and Applied Mathematics, CRC Press, 1992.

[25] J. J. Dongarra, L. S. Duff, D. C. Sorensen, H. A. V. Vorst, Numerical Linear Algebra for High Performance Computers, SIAM, 1998.

[26] W. Keller-Gehrig, Fast algorithms for the characteristic polynomial, Th. Comp. Science 36 (1985) 309–317. doi:10.1016/0304-3975(85)90049-0.

[27] J.-G. Dumas, J.-L. Roch, On parallel block algorithms for exact triangularizations, Parallel Computing 28 (11) (2002) 1531–1548. doi:10.1016/S0167-8191(02)00161-8.

[28] J.-G. Dumas, T. Gautier, C. Pernet, Z. Sultan, Parallel computation of echelon forms, in: Euro-Par 2014 Parallel Proc., LNCS (8632), Springer, 2014, pp. 499–510. doi:10.1007/978-3-319-09873-9_42.

[29] G. Malaschonok, Generalized Bruhat decomposition in commutative domains, in: V. Gerdt, W. Koepf, E. Mayr, E. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing, Vol. 8136 of Lecture Notes in Computer Science, Springer International Publishing, 2013, pp. 231–242. doi:10.1007/978-3-319-02297-0_20.

[30] T. Mulders, A. Storjohann, Rational Solutions of Singular Linear Systems, in: Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation, ISSAC '00, ACM, New York, NY, USA, 2000, pp. 242–249. doi:10.1145/345542.345644.

[31] E. Kaltofen, B. D. Saunders, On Wiedemann's method of solving sparse linear systems, in: H. F. Mattson, T. Mora, T. R. N. Rao (Eds.), Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, no. 539 in Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1991, pp. 29–38.

[32] P. V. Strassen, Gaussian elimination is not optimal, Numerische Mathematik 13 (4) (1969) 354–356. doi:10.1007/BF02165411.