THEORY OF COMPUTING, Volume 14 (3), 2018, pp. 1–21
www.theoryofcomputing.org

SPECIAL ISSUE: CCC 2017

A Deterministic PTAS for the Commutative Rank of Matrix Spaces

Markus Bläser Gorav Jindal Anurag Pandey

Received July 15, 2017; Revised December 22, 2017; Published April 15, 2018

Abstract: We consider the problem of computing the commutative rank of a given matrix space $\mathcal{B} \subseteq \mathbb{F}^{n\times n}$, that is, given a basis of $\mathcal{B}$, find a matrix of maximum rank in $\mathcal{B}$. This problem is fundamental, as it generalizes several computational problems from algebra and combinatorics. For instance, checking if the commutative rank of the space is n, subsumes problems such as testing perfect matching in graphs and identity testing of algebraic branching programs. Finding an efficient deterministic algorithm for the commutative rank is a major open problem, although there is a simple and efficient randomized algorithm for it. Recently, there has been a series of results on computing the non-commutative rank of matrix spaces in deterministic polynomial time. Since the non-commutative rank of any matrix space is at most twice the commutative rank, one immediately gets a deterministic $1/2$-approximation algorithm for the commutative rank. It is a natural question whether this approximation ratio can be improved. In this paper, we answer this question affirmatively.

We present a deterministic polynomial-time approximation scheme (PTAS) for computing the commutative rank of a given matrix space. More specifically, given a matrix space $\mathcal{B} \subseteq \mathbb{F}^{n\times n}$ and a rational number $\varepsilon > 0$, we give an algorithm that runs in time $O(n^{4+3/\varepsilon})$ and

A conference version of this paper appeared in the Proceedings of the 32nd Computational Complexity Conference (CCC), 2017 [1].

ACM Classification: F.2.2, I.1.2

AMS Classification: 68Q25, 68W25, 68W30

Key words and phrases: approximation algorithm, algebraic complexity, commutative rank, matrix spaces, PTAS, Wong sequences

 DOI: 10.4086/toc.2018.v014a003

computes a matrix $A \in \mathcal{B}$ such that the rank of A is at least $(1-\varepsilon)$ times the commutative rank of $\mathcal{B}$. The algorithm is the natural greedy algorithm. It always takes the first set of k matrices that will increase the rank of the matrix constructed so far until it does not find any improvement, where the size k of the set depends on ε.

1 Introduction

Given a matrix space $\mathcal{B}$ by a basis $B_1, B_2, \dots, B_m$ over some underlying field $\mathbb{F}$, what is the maximum rank of any matrix contained in $\mathcal{B}$? This maximum rank is also called the *commutative rank* of the matrix space $\mathcal{B} = \langle B_1, B_2, \dots, B_m \rangle$. This problem was introduced by Edmonds [4] and is now known as Edmonds' Problem.

With a given basis $B_1, B_2, \dots, B_m$, we can associate a matrix B with homogeneous linear forms as entries by setting

$$B := \sum_{i=1}^{m} x_i B_i \,,$$

where $x_1, \dots, x_m$ are indeterminates. Every matrix in $\mathcal{B}$ is the homomorphic image of B under some substitution that assigns values from $\mathbb{F}$ to the variables. The commutative rank of the space $\mathcal{B}$ is same as the rank of the matrix B over the field of rational functions $\mathbb{F}(x_1, x_2, \dots, x_m)$, provided that $\mathbb{F}$ is large enough. For this reason, this problem is sometimes called the *symbolic matrix rank problem*, too.

The commutative rank problem is a fundamental problem, which generalizes several algorithmic problems in algebraic complexity theory and graph theory. The maximum matching problem in bipartite and general graphs is a special case of the commutative rank problem [15]. A graph has a perfect matching if and only if its Tutte matrix, which is a matrix with homogeneous linear forms as entries, has full rank. Even the so-called linear matroid parity problem is a special case of the commutative rank problem [19].

Valiant [23] showed that a formula of size s can be written as a determinant of an $(s+2)\times(s+2)$ matrix having affine linear forms as entries. (In fact, these entries are of a very special form, they are either constants or variables.) It follows that checking if a given matrix space has full rank is as hard as polynomial identity testing of formulas. (Note that we here have affine linear forms, but this is no problem, since the symbolic rank of $B_1 + x_2B_2 + \cdots + x_mB_m$ and $x_1B_1 + x_2B_2 + \cdots + x_mB_m$ is the same.) It is even known that polynomials computed by algebraic branching programs can be written as the determinant of a polynomial-size matrix with affine linear forms as entries, see [24, 22, 18]. So the problem of deciding whether a given matrix space is of full rank is as hard as polynomial identity testing of arithmetic branching programs.

The rank of B is the size of the largest non-vanishing minor of B. Any minor of B is a polynomial of degree at most n in the variables $x_1, x_2, \dots, x_m$. Therefore, for almost all substitutions, the homomorphic image of B will be a matrix of maximum rank. The Schwartz-Zippel lemma [25, 20] tells us that we can choose such a substitution uniformly from a large enough set. In this way, one immediately gets a very simple randomized algorithm for computing the commutative rank of $\mathcal{B}$, provided that the order of the field $\mathbb{F}$ is large enough. It is a major open problem in complexity theory to find an efficient deterministic algorithm for computing the commutative rank.

We remark that if the underlying field $\mathbb{F}$ is not large enough, then this problem is hard. Buss et al. [2] proved that the problem is NP-hard when the field $\mathbb{F}$ is of constant order.

1.1 Previous work

Since the general case of computing the commutative rank is as hard as identity testing for polynomials given as algebraic branching programs, several special cases of matrix spaces have been considered. Maybe the simplest case is when all the matrices B_i are of rank 1 [16, 12, 13]. This case subsumes the matching problem in bipartite graphs, for instance. For this case, deterministic polynomial-time algorithms are known [12, 13]. Recently, it was even demonstrated that computing the commutative rank in this case is in quasi-NC, that is, there are (almost) efficient deterministic parallel algorithms for this case [6, 10]. The next case is when the matrices B_i are skew-symmetric of rank 2. This case is also of special interest as the linear matroid parity problem reduces to it [15]. Several deterministic polynomial-time algorithms have been found for this case, see [21, 8, 17].

Analogous to the notion of commutative rank of a matrix space, there is also a notion of non-commutative rank (see the next section for a precise definition). The commutative rank of a matrix space is always a lower bound for the non-commutative rank. Matrix spaces for which the commutative rank and non-commutative rank are equal are called *compression spaces* [5]. A deterministic polynomial-time algorithm for checking if a compression space is of full rank (over fields of characteristic zero) was discovered by Gurvits [11]. Gurvits' algorithm was analyzed more carefully by Garg et al. [9] to demonstrate that the algorithm is actually a deterministic polynomial-time algorithm to check if a given matrix space has full non-commutative rank. Moreover, Garg et al. [9] devise a deterministic polynomial-time algorithm which can compute the non-commutative rank exactly, even when it is not maximum. This algorithm works for $\mathbb{C}$ and its subfields. Ivanyos et al. [14] extended this results to arbitrary fields, using a totally different algorithm.

Fortin and Reutenauer [7] show that the non-commutative rank of any matrix space is at most twice the commutative rank. So the algorithms in [9, 14] are deterministic polynomial-time algorithms that can compute a $1/2$-approximation of the commutative rank. Approximating the commutative rank of a matrix space can be seen as a relaxation of the polynomial identity testing problem. Improving on the $1/2$-approximation was formulated as an open problem by Garg et al. [9].

1.2 Our results

Our main result is an improvement on this approximation performance. We give a deterministic polynomial-time approximation scheme (PTAS) for approximating the commutative rank. In other words, given a basis $B_1, \dots, B_m$ of a matrix space $\mathcal{B}$ of $n \times n$-matrices and some rational number $\varepsilon > 0$, our algorithm outputs a matrix $A \in \mathcal{B}$ whose rank is at least $(1-\varepsilon) \cdot r$, where $r = \max\{\operatorname{rank}(B) \mid B \in \mathcal{B}\}$, provided that the order of the underlying field is larger than n. Our algorithm performs $O(n^{4+3/\varepsilon})$ many arithmetic operations, the size of each operand is linear in the sizes of the entries of the matrices $B_1, \dots, B_m$. So for fixed ε, the running time is polynomial in the input size.

Our algorithm is very simple, it is the natural greedy algorithm: Assume we have constructed a matrix A so far. Then the algorithm tries all subsets $B_{i_1}, \dots, B_{i_k}$ of $B_1, \dots, B_m$ of size k, where k depends on ε, and tests whether we can increase the rank of A by adding an appropriate linear combination of $B_{i_1}, \dots, B_{i_k}$. The main difficulty is to prove that when this algorithm stops, then A is an $(1-\varepsilon)$-approximation. Our analysis uses so-called *Wong sequences*.

For polynomial identity testing, one has to test whether a given matrix space has full rank or rank

$\leq n-1$. Therefore, our PTAS does not seem to help getting a polynomial-time algorithm for polynomial identity testing.

1.3 Organization of the paper

Section 2 describes the basic setup of the problem. It describes the basic notation, definitions, and related known lemmas and theorems. In Section 3, we first present the most basic variant of our greedy algorithm, which computes a $1/2$-approximation of the commutative rank in deterministic polynomial time. It describes the basic ideas but is much easier to analyze. This motivates our final algorithm which can compute arbitrary approximations to the commutative rank in deterministic polynomial time. To extend this $1/2$-approximation to arbitrary approximation, we introduce the notion of Wong sequences and Wong index in Section 4. Section 5 studies the relation between commutative rank and Wong index. In this section, we prove that the higher the Wong index is of a given matrix, the closer its rank is to the commutative rank of the given matrix space. This allows us to extend Algorithm 1 to arbitrary approximation by considering larger subsets. The algorithm for arbitrary approximation of the commutative rank and its proof of correctness and desired running time are given in Section 6. We conclude by giving some examples in Section 7 which demonstrate that the analysis of our algorithm is tight.

2 Preliminaries

Here, we introduce the basic definitions and notation needed to fully describe our algorithm. Henceforth we assume that the order of the underlying field is greater than n, the size of the input matrices, i. e., $|\mathbb{F}| > n$.

1. If V and W are vector spaces, then we use the notation $V \leq W$ to denote that V is a subspace of W.
2. We use $\mathbb{F}^{n\times n}$ to denote the set of all $n \times n$ matrices over a field $\mathbb{F}$.
3. $\mathrm{Im}(A)$ is used to denote the image of a matrix $A \in \mathbb{F}^{n\times n}$.
4. $\ker(A)$ is used to denote the kernel of a matrix $A \in \mathbb{F}^{n\times n}$.
5. $\dim(V)$ is used to denote the dimension of a vector space V.
6. For any subset S of a vector space U, $\langle S\rangle$ denotes the linear span of S.
7. For $A \in \mathbb{F}^{n\times n}$ and a vector space $U \leq \mathbb{F}^n$, the image of U under A is $A(U) = AU = \{Au \mid u \in U\}$.
8. The preimage of $W \leq \mathbb{F}^n$ under $A \in \mathbb{F}^{n\times n}$ is defined as $A^{-1}(W) = \{v \in V \mid Av \in W\}$.
9. The set $\{0,1,2,\dots,n\}$ is denoted by $[n]$.
10. We use the notation I_r to denote the $r \times r$ identity matrix.

Below are some of the basic definitions which we shall need.

Definition 2.1 (Matrix space)**.** A vector space $\mathcal{B} \leq \mathbb{F}^{n\times n}$ is called a *matrix space*.

We usually deal with matrix spaces whose generating set is given as the input. More precisely, we are given a matrix space

$$\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n\times n},$$

where we get the matrices $B_1, B_2, \ldots, B_m$ as the input. Note that without loss of generality, one can assume that $m \leq n^2$.

Definition 2.2 (Commutative rank)**.** The maximum rank of any matrix in a matrix space $\mathcal{B}$ is called the *commutative rank* of $\mathcal{B}$. We write $\operatorname{rank}(\mathcal{B})$ to denote this quantity.

We shall use the same notation $\operatorname{rank}(A)$ for denoting the usual rank of any matrix. Note that the rank of a matrix A is same as the commutative rank of the matrix space generated by A, that is, $\operatorname{rank}(A) = \operatorname{rank}(\langle A \rangle)$.

Definition 2.3 (Product of a matrix space and a vector space)**.** The image of a vector space U under a matrix space $\mathcal{A}$ is the span of the images of U under every $A \in \mathcal{A}$, that is $\mathcal{A}(U) := \mathcal{A}U := \langle \bigcup_{A\in\mathcal{A}} A(U) \rangle$. We also call this image $\mathcal{A}U$ to be the product of the matrix space $\mathcal{A}$ and the vector space U.

Definition 2.4 (c-shrunk subspace)**.** A vector space $V \leq \mathbb{F}^n$ is a *c-shrunk subspace* of a matrix space $\mathcal{B}$, if $\operatorname{rank}(\mathcal{B}V) \leq \dim(V) - c$.

Definition 2.5 (Non-commutative rank)**.** Given a matrix space $\mathcal{B} \leq \mathbb{F}^{n\times n}$, let r be the maximum non-negative integer such that there exists a r-shrunk subspace of the matrix space $\mathcal{B}$. Then $n - r$ is called the *non-commutative rank* of $\mathcal{B}$. We use the notation $\text{nc-rank}(\mathcal{B})$ to denote this quantity.

From the definition above, it is not clear why we call this quantity non-commutative rank. It can be shown that the quantity above equals the rank of the corresponding symbolic matrix when the variables $x_1, \ldots, x_m$ do not commute. For more natural and equivalent definitions as well as more background on non-commutative rank, we refer the reader to [9, 7].

Lemma 2.6. *For all fields $\mathbb{F}$ and for all matrix spaces $\mathcal{B} \leq \mathbb{F}^{n\times n}$, $\operatorname{rank}(\mathcal{B}) \leq \text{nc-rank}(\mathcal{B})$.*

Proof. Let $r = \text{nc-rank}(\mathcal{B})$. This means that there exists $V \leq \mathbb{F}^n$ such that $\operatorname{rank}(\mathcal{B}V) = \dim(V) - (n-r)$. Therefore, for all $B \in \mathcal{B}$, $\operatorname{rank}(BV) \leq \dim(V) - (n-r)$. Thus

$$\operatorname{rank}(\mathcal{B}) \leq n - (n-r) = r = \text{nc-rank}(\mathcal{B}). \qquad \square$$

The above lemma states that the non-commutative rank is at least as large as the commutative rank. But how large it can be compared to the commutative rank? The following theorem states that it is always less than twice the commutative rank.

Theorem 2.7 ([7, 3])**.** *For infinite fields $\mathbb{F}$ and all matrix spaces $\mathcal{B} \leq \mathbb{F}^{n\times n}$, we have :*

$$\text{nc-rank}(\mathcal{B}) \leq 2 \cdot \operatorname{rank}(\mathcal{B}).$$

Derksen and Makam also gave a family of examples where the ratio of non-commutative rank and commutative rank gets arbitrarily close to 2, hence showing that the bound above is sharp (see [3, Theorem 1.15]). To bound the commutative rank of a matrix space, we also need the following easy fact from linear algebra.

Fact 2.8. *Let M be a matrix of the following form:*

$$M = \begin{array}{r c l} & \overbrace{}^{r \text{ columns}} & \\ r \text{ rows} \{ & \begin{bmatrix} L & B \\ A & 0 \end{bmatrix} & \} \, n-r \text{ rows} \\ & \underbrace{}_{n-r \text{ columns}} & \end{array} \tag{2.1}$$

Also, let $\operatorname{rank}(A) = a$ *and* $\operatorname{rank}(B) = b$. *Then* $\operatorname{rank}(M) \leq r + \min\{a,b\}$.

3 1/2-approximation algorithm for the commutative rank

Here we present the simplest variant of the greedy algorithm which also achieves a $1/2$-approximation for the commutative rank. This algorithm looks for the first matrix that increases the rank of the current matrix and stops if it does not find such a matrix. Its analysis is much easier than the general case.

Algorithm 1 Greedy algorithm for $1/2$-approximating commutative rank

Input : A matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n \times n}$, input is a list of matrices $B_1, B_2, \ldots, B_m$.
Output: A matrix $A \in \mathcal{B}$ such that $\operatorname{rank}(A) \geq \frac{1}{2} \cdot \operatorname{rank}(\mathcal{B})$
Initialize $A = 0 \in \mathbb{F}^{n \times n}$ to the zero matrix.
while *Rank is increasing* **do**
    for *each* $1 \leq i \leq m$ **do**
        Check if there exists a $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda B_i) > \operatorname{rank}(A)$.
        if $\operatorname{rank}(A + \lambda B_i) > \operatorname{rank}(A)$ **then**
            Update $A = A + \lambda B_i$.
return A.

Lemma 3.1. *If $|\mathbb{F}| > n$, then Algorithm 1 runs in polynomial time and returns a matrix $A \in \mathcal{B}$ such that* $\operatorname{rank}(A) \geq (1/2) \cdot \operatorname{rank}(\mathcal{B})$.

Proof. Let A be the matrix returned by Algorithm 1. Assume that A has rank r. We know that there exist non-singular matrices P and Q such that

$$PAQ = \begin{pmatrix} I_r & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{pmatrix}, \tag{3.1}$$

where I_r is the $r \times r$ identity matrix. Now consider the matrix space

$$P\mathcal{B}Q := \langle PB_1Q, PB_2Q, \ldots, PB_mQ \rangle .$$

This does not change anything with respect to the rank. Also the way the algorithms works is not changed by such a transformation. So for the analysis, we can replace $\mathcal{B}$ by $P\mathcal{B}Q$. Consider any general matrix $A+x_1B_1+x_2B_2+\cdots+x_mB_m$ in $\mathcal{B}$. We decompose it as

$$A+x_1B_1+x_2B_2+\cdots+x_mB_m = \left(\begin{array}{c|c} M_1 & M_2 \\ \hline M_3 & M_4 \end{array}\right). \tag{3.2}$$

Here M_1 is an $r\times r$ matrix, M_2 is an $r\times(n-r)$ matrix, M_3 is a $(n-r)\times r$ matrix and M_4 is a $(n-r)\times(n-r)$ matrix. M_1, M_2, M_3, and M_4 have (affine) linear forms in variables $\mathbf{x}=(x_1,x_2,\ldots,x_m)$ as their entries.

Now we claim that the bottom right block M_4 is the zero matrix. Assume otherwise. Assume that the (s,t)-entry of the above matrix is non-zero with $s,t>r$. Consider the $(r+1)\times(r+1)$ submatrix of $A+x_1B_1+x_2B_2+\cdots+x_mB_m$, obtained by adding the s^{th} row (from M_3) and the t^{th} column (from M_2) to M_1. We shall denote this submatrix by C, which looks like

$$C=\begin{pmatrix} 1+\ell_{11}(\mathbf{x}) & \ell_{12}(\mathbf{x}) & \ldots & \ell_{1r}(\mathbf{x}) & a_1(\mathbf{x}) \\ \ell_{21}(\mathbf{x}) & 1+\ell_{22}(\mathbf{x}) & \ldots & \ell_{2r}(\mathbf{x}) & a_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \ell_{r1}(\mathbf{x}) & \ell_{r2}(\mathbf{x}) & \ldots & 1+\ell_{rr}(\mathbf{x}) & a_r(\mathbf{x}) \\ b_1(\mathbf{x}) & b_2(\mathbf{x}) & \ldots & b_r(\mathbf{x}) & c(\mathbf{x}) \end{pmatrix}. \tag{3.3}$$

The $\ell_{i,j}$, a_i, b_j, and c are homogeneous linear forms in $\mathbf{x}$. By our choice, $c(\mathbf{x})\neq 0$. It is not hard to see that

$$\det(C)=c(\mathbf{x})+\text{terms of degree at least }2\,. \tag{3.4}$$

Thus there are $\lambda\in\mathbb{F}$ and $i\in[m]$ such that $\det(C(\alpha))\neq 0$, where α is the assignment to the variables $\mathbf{x}=(x_1,x_2,\ldots,x_m)$ obtained by setting $x_k=0$ when $k\neq i$ and $x_i=\lambda$. Since the degree of $\det(C)$ is at most n, by Schwartz-Zippel lemma it follows that we have to check at most $n+1$ values for λ.

These choices of $i\in[m]$ and $\lambda\in\mathbb{F}$ would allow Algorithm 1 to find a matrix A of larger rank. Thus Algorithm 1 would keep finding a matrix A of larger rank when the matrix M_4 is non-zero. Hence it can only stop when M_4 is the zero matrix. If M_4 is the zero matrix, then by using Fact 2.8 we know that $\operatorname{rank}(\mathcal{B})\leq 2r$. Thus when Algorithm 1 stops, it outputs a matrix A such that $\operatorname{rank}(A)\geq(1/2)\cdot\operatorname{rank}(\mathcal{B})$.

The running time is obviously polynomial since the while loop is executed at most n times and we have to check at most $n+1$ values for λ. The size of the numbers that occur in the rank check is polynomial in the size of the entries of $B_1,\ldots,B_m$. □

4 Wong sequences and Wong index

In this section, we introduce the notion of Wong sequences which is crucially used in our proofs. For a more comprehensive exposition, we refer the reader to [12].

Definition 4.1 (Second Wong sequence)**.** Let $\mathcal{B}\leq\mathbb{F}^{n\times n}$ be a matrix space and $A\in\mathcal{B}$. The sequence of subspaces $(W_i)_{i\in[n]}$ of $\mathbb{F}^n$ is called the *second Wong sequence* of $(A,\mathcal{B})$, where $W_0=\{0\}$, and $W_{i+1}=\mathcal{B}A^{-1}(W_i)$.

In [12], first Wong sequences are also introduced. But for our purpose, just the notion of *second Wong sequence* is enough. It is easy to see that $W_0 \leq W_1 \leq W_2 \leq \cdots \leq W_n$, see [12].

Next, we introduce the notion of pseudo-inverses. They are helpful in computing the Wong sequences. We remark that we need the notion of Wong sequence only for the analysis, our algorithm is completely oblivious to Wong sequences.

Definition 4.2 (Pseudo-Inverse)**.** A non-singular matrix $A' \in \mathbb{F}^{n\times n}$ is called a *pseudo-inverse* of a linear map $A \in \mathbb{F}^{n\times n}$ if the restriction of A' to $\mathrm{Im}(A)$ is the inverse of the restriction of A to a direct complement of $\ker(A)$.

Unlike the usual inverse of a non-singular matrix, a pseudo-inverse of a matrix is not necessarily unique. But it always exists and if A is non-singular, then it is unique and coincides with the usual inverse.

The following lemma demonstrates the role of pseudo-inverses in computing Wong sequences. This lemma and its proof are implicit in the proof of Lemma 10 in [12]. We prove it here for the sake of completeness. The lemma essentially states that we can replace the preimage computation in the Wong sequence by multiplication with a pseudo-inverse.

Lemma 4.3. *Let $\mathbb{F}$ be any field and $\mathcal{B} \leq \mathbb{F}^{n\times n}$ be a matrix space, $A \in \mathcal{B}$, A' be a pseudo-inverse of A and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A,\mathcal{B})$. Then for all $1 \leq i \leq n$, we have*

$$W_i = (\mathcal{B}A')^i(\ker(AA'))$$

as long as $W_{i-1} \subseteq \mathrm{Im}(A)$.

Proof. We prove the statement by induction on i. Since $\ker(AA') = (A')^{-1}(\ker(A))$, we get that

$$(\mathcal{B}A')(\ker(AA')) = \mathcal{B}A'(A')^{-1}(\ker(A)) = \mathcal{B}\ker(A) = W_1\,.$$

This proves the base case of $i = 1$. To prove that $W_i = (\mathcal{B}A')^i(\ker(AA'))$, we need to prove that $(\mathcal{B}A')^i(\ker(AA')) \subseteq W_i$ and $W_i \subseteq (\mathcal{B}A')^i(\ker(AA'))$. By the induction hypothesis, we just need to prove that $(\mathcal{B}A')(W_{i-1}) \subseteq W_i$ and $W_i \subseteq (\mathcal{B}A')(W_{i-1})$.

First we prove the easy direction, that is $(\mathcal{B}A')(W_{i-1}) \subseteq W_i$. Since $W_{i-1} \subseteq \mathrm{Im}(A)$, we have that $A'(W_{i-1}) \subseteq A^{-1}(W_{i-1})$. Thus $(\mathcal{B}A')(W_{i-1}) \subseteq \mathcal{B}A^{-1}(W_{i-1}) = W_i$.

Now we prove that $W_i \subseteq (\mathcal{B}A')(W_{i-1})$. Since $W_{i-1} \subseteq \mathrm{Im}(A)$, we get that $A^{-1}(W_{i-1}) = A'W_{i-1} + \ker(A)$. Thus $W_i = \mathcal{B}A^{-1}(W_{i-1}) \subseteq \mathcal{B}A'W_{i-1} + \mathcal{B}\ker(A)$. We have $\mathcal{B}\ker(A) = W_1 \subseteq W_{i-1}$, this implies that $W_i \subseteq \mathcal{B}A'W_{i-1} + W_{i-1}$. Since $A \in \mathcal{B}$ and $W_{i-1} = AA'W_{i-1}$, we get that $W_{i-1} \subseteq \mathcal{B}A'W_{i-1}$. This in turn implies that $W_i \subseteq \mathcal{B}A'W_{i-1} + \mathcal{B}A'W_{i-1} = (\mathcal{B}A')(W_{i-1})$. $\square$

Given a matrix space $\mathcal{B}$ and a matrix $A \in \mathcal{B}$, how can one check that A is of maximum rank in $\mathcal{B}$, i. e., $\mathrm{rank}(A) = \mathrm{rank}(\mathcal{B})$? The following lemma in [12] gives a sufficient condition for A to be of maximum rank in $\mathcal{B}$.

Lemma 4.4 (Lemma 10 in [12])**.** *Assume that $|\mathbb{F}| > n$. Let $A \in \mathcal{B} \leq \mathbb{F}^{n\times n}$, and let A' be a pseudo-inverse of A. If we have that for all $i \in [n]$,*

$$W_i = (\mathcal{B}A')^i(\ker(AA')) \subseteq \mathrm{Im}(A)\,, \tag{4.1}$$

then A is of maximum rank in $\mathcal{B}$.

Thus, the above lemma shows that if A is not of maximum rank in $\mathcal{B}$, then we have $W_i \nsubseteq \mathrm{Im}(A)$ for some $i \in [n]$. For our purposes, we need to quantify when exactly this happens. Therefore we define:

Definition 4.5 (Wong Index)**.** Let $\mathcal{B} \leq \mathbb{F}^{n\times n}$ be a matrix space, $A \in \mathcal{B}$ and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A,\mathcal{B})$. Let $k \in [n]$ be the maximum integer such that $W_k \subseteq \mathrm{Im}(A)$. Then k is called the *Wong index* of $(A,\mathcal{B})$. We shall denote it by $w(A,\mathcal{B})$.

Using the above definition, another way to state Lemma 4.4 is that if the Wong index $w(A,\mathcal{B})$ of $(A,\mathcal{B})$ is n, then A is of maximum rank in $\mathcal{B}$. But can one say more? In next section, we explore this connection. We shall prove that the closer $w(A,\langle A,B\rangle)$ is to n, the closer the rank of A is to the commutative rank of $\langle A,B\rangle$.

The converse of Lemma 4.4 is not true in general. But the converse is true in the special case when $\mathcal{B}$ is spanned by just two matrices. Fortunately, for our algorithm we only require the converse to be true in this special case. The following fact from [12] formally states this idea.

Fact 4.6 (Restatement of Fact 11 in [12])**.** *Assume that $|\mathbb{F}| > n$ and let $A,B \in \mathbb{F}^{n\times n}$. If A is of maximum rank in $\langle A,B\rangle$, then the Wong index $w(A,\langle A,B\rangle)$ of $(A,\langle A,B\rangle)$ is n.*

In order to extend the simple greedy algorithm for rank increment described in Section 3 for arbitrary approximation of the commutative rank, we use the Wong index defined above. To achieve that, we need the relation between the commutative rank and Wong index, which we establish in the next section.

5 Relation between rank and Wong index

We prove that the natural greedy strategy works, essentially by showing that one of the following happens:

1. The Wong index of the matrix obtained by the greedy algorithm at a given step is high enough, in which case, we show that the matrix already has the desired rank. Lemma 5.4 formalizes this.
2. We can increase the rank by a greedy step. Lemma 5.5 formalizes this.

In the above spirit, we quantify the connection between the commutative rank and Wong index in this section, using a series of lemmas. First we need the following lemma which demonstrates that the second Wong sequence remains "almost" the same under invertible linear maps.

Lemma 5.1. *Let $\mathbb{F}$ be any field, $A \in \mathcal{B} \leq \mathbb{F}^{n\times n}$ and $(W_i)_{i\in[n]}$ be the second Wong sequence of $(A,\mathcal{B})$. If $P \in \mathbb{F}^{n\times n}$ and $Q \in \mathbb{F}^{n\times n}$ are invertible matrices, then the second Wong sequence of $(PAQ,P\mathcal{B}Q)$ is $(PW_i)_{i\in[n]}$. In particular, $w(A,\mathcal{B}) = w(PAQ,P\mathcal{B}Q)$.*

Proof. Consider the i^{th} entry W_i' in the second Wong sequence of $(PAQ,P\mathcal{B}Q)$. We prove that $W_i' = PW_i$ for all $i \in [n]$. We use induction on i. The statement is trivially true for $i = 0$. By the induction hypothesis, we have,

$$W_i' = P\mathcal{B}Q(PAQ)^{-1}PW_{i-1} = P\mathcal{B}QQ^{-1}A^{-1}P^{-1}PW_{i-1} = P\mathcal{B}A^{-1}(W_{i-1}) = PW_i\,. \qquad \square$$

The following technical lemma relates the Wong index with a sequence of vanishing matrix products.

Lemma 5.2. *Let $\mathbb{F}$ be any field and $A,B \in \mathbb{F}^{n\times n}$. Assume*

$$A = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}$$

and express the matrix B as

$$B = \begin{array}{r} r \text{ rows} \left\{ \vphantom{B_{11}} \right. \\ \end{array} \begin{bmatrix} \overbrace{B_{11}}^{r \text{ columns}} & B_{12} \\ B_{21} & \underbrace{B_{22}}_{n-r \text{ columns}} \end{bmatrix} \begin{array}{l} \\ \left. \vphantom{B_{22}} \right\} n-r \text{ rows} \end{array} \tag{5.1}$$

Let $\ell \le n$ be the maximum integer such that first ℓ elements of the sequence of matrices

$$B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^{i}B_{12}, \ldots \tag{5.2}$$

are equal to the zero matrix. Then $\ell = w(A, \langle A,B\rangle)$.

Proof. Notice that I_n is a pseudo-inverse of A, so we can assume $A' = I_n$ in the statement of Lemma 4.3. Consider the second Wong sequence of $(A, \langle A,B\rangle)$. By Lemma 4.3, it equals $((\langle A,B\rangle A')^i(\ker(AA'))$. Since we can assume that $A' = I_n$, this sequence is $((\langle A,B\rangle)^i(\ker(A))$. $\ker(A) \le \mathbb{F}^n$ contains exactly the vectors having the first r entries equal to zero and $\operatorname{Im}(A)$ contains exactly the vectors which have the last $n-r$ entries equal to zero. Let $k = w(A, \langle A,B\rangle)$, we want to show that $k = \ell$.

First we show that $\ell \ge k$. For this, we need to show that

$$B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \cdots = B_{21}B_{11}^{k-2}B_{12} = 0\,.$$

If $k = 0$ then we do not need to show anything. Otherwise $k > 0$. Consider the first entry W_1 of second Wong sequence of $(A, \langle A,B\rangle)$. By Lemma 4.3, we know that $W_1 = \langle A,B\rangle \ker(A)$. As $\ker(A) \le \mathbb{F}^n$ contains exactly the vectors which have first r entries to be zero, if B_{22} was not zero then $B\ker(A)$ would contain a vector with a non-zero entry in the last $n-r$ coordinates. This would violate the assumption $W_1 \subseteq \operatorname{Im}(A)$. Thus $B_{22} = 0$. Now we use induction on length of the sequence

$$B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^{i}B_{12}\,.$$

Our induction hypothesis assumes that for $i \ge 1$

$$B^i = \begin{array}{r} r \text{ rows} \left\{ \vphantom{B_{11}^{i}} \right. \\ \end{array} \begin{bmatrix} \overbrace{B_{11}^{i} + \sum_{j=0}^{i-2} B_{11}^{j}B_{12}B_{21}B_{11}^{i-2-j}}^{r \text{ columns}} & B_{11}^{i-1}B_{12} \\ B_{21}B_{11}^{i-1} & \underbrace{0}_{n-r \text{ columns}} \end{bmatrix} \begin{array}{l} \\ \left. \vphantom{B_{11}^{i}} \right\} n-r \text{ rows} \end{array} \tag{5.3}$$

and

$$B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \cdots = B_{21}B_{11}^{i-2}B_{12} = 0\,.$$

We just proved the base case of $i = 1$. Consider the following evaluation of $B^{i+1} = B \cdot B^i$

$$B^{i+1} = \begin{array}{r} r \text{ rows}\left\{\right. \\ \end{array} \left[\begin{array}{cc} \overbrace{B_{11}^{i+1} + \sum_{j=0}^{i-2} B_{11}^{j+1} B_{12} B_{21} B_{11}^{i-2-j} + B_{12} B_{21} B_{11}^{i-1}}^{r \text{ columns}} & B_{11}^{i} B_{12} \\ B_{21} B_{11}^{i} + \sum_{j=0}^{i-2} B_{21} B_{11}^{j} B_{12} B_{21} B_{11}^{i-2-j} & \underbrace{B_{21} B_{11}^{i-1} B_{12}}_{n-r \text{ columns}} \end{array}\right] \begin{array}{l} \\ \left.\right\} n-r \text{ rows} \end{array} \tag{5.4}$$

Since $i+1 \leq k$, we must have $B_{21} B_{11}^{i-1} B_{12} = 0$, otherwise we would have $W_{i+1} \not\subseteq \operatorname{Im}(A)$. Also we know by the induction hypothesis that

$$B_{22} = B_{21} B_{12} = B_{21} B_{11} B_{12} = \cdots = B_{21} B_{11}^{i-2} B_{12} = 0\,,$$

which implies that

$$B^{i+1} = B \cdot B^i = \begin{array}{r} r \text{ rows}\left\{\right. \\ \end{array} \left[\begin{array}{cc} \overbrace{B_{11}^{i+1} + \sum_{j=0}^{i-1} B_{11}^{j} B_{12} B_{21} B_{11}^{i-1-j}}^{r \text{ columns}} & B_{11}^{i} B_{12} \\ B_{21} B_{11}^{i} & \underbrace{0}_{n-r \text{ columns}} \end{array}\right] \begin{array}{l} \\ \left.\right\} n-r \text{ rows} \end{array} \tag{5.5}$$

Now we show that $k \geq \ell$. Since $k = w(A, \langle A, B\rangle)$, for all $1 \leq i \leq k$, B^i can be written as

$$B^{i} = \begin{array}{r} r \text{ rows}\left\{\right. \\ \end{array} \left[\begin{array}{cc} \overbrace{B_{11}^{i} + \sum_{j=0}^{i-2} B_{11}^{j} B_{12} B_{21} B_{11}^{i-2-j}}^{r \text{ columns}} & B_{11}^{i-1} B_{12} \\ B_{21} B_{11}^{i-1} & \underbrace{0}_{n-r \text{ columns}} \end{array}\right] \begin{array}{l} \\ \left.\right\} n-r \text{ rows} \end{array} \tag{5.6}$$

Note that $\langle A, B\rangle^i$ is spanned by all matrices of the form $M_1 M_2 \cdots M_i$ with $M_j = A$ or $M_j = B$, $1 \leq j \leq i$. Since we have that $W_k \subseteq \operatorname{Im}(A)$, we know that $M_1 M_2 \cdots M_k \ker(A) \subseteq \operatorname{Im}(A)$ for any product $M_1 M_2 \cdots M_k$ as above. Now let us see what condition one needs such that $W_{k+1} \not\subseteq \operatorname{Im}(A)$ is true. Since A is the identity on $\operatorname{Im}(A)$, only B^{k+1} can take $\ker(A)$ out of $\operatorname{Im}(A)$ for $W_{k+1} \not\subseteq \operatorname{Im}(A)$ to be true. By a similar argument as above, this happens only when $B_{21} B_{11}^{k-1} B_{12} \neq 0$, thus $\ell \leq k$. $\square$

Now, having established the connection between Wong index and the sequence of vanishing matrix products, we prove another technical lemma establishing the relation between the length of this sequence and the commutative rank.

Lemma 5.3. *Let $\mathbb{F}$ be any field, $B \in \mathbb{F}^{n \times n}$ and*

$$B = \begin{array}{r} r \textit{ rows}\left\{\right. \\ \end{array} \left[\begin{array}{cc} \overbrace{B_{11}}^{r \textit{ columns}} & B_{12} \\ B_{21} & \underbrace{B_{22}}_{n-r \textit{ columns}} \end{array}\right] \begin{array}{l} \\ \left.\right\} n-r \textit{ rows} \end{array} \tag{5.7}$$

Consider the sequence of matrices

$$B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^{j}B_{12} \ldots .$$

If the first $k \geq 1$ matrices in this sequence are equal to the zero matrix and B_{11} is non-singular, then $\operatorname{rank}(B) \leq r(1+1/k)$.

Proof. If $\operatorname{rank}(B_{12}) \leq r/k$, then we are done by using the Fact 2.8. So we can assume without loss of generality that $\operatorname{rank}(B_{12}) > r/k$. Now suppose that

$$\dim\langle \operatorname{Im}(B_{12}) \cup \operatorname{Im}(B_{11}B_{12}) \cup \cdots \cup \operatorname{Im}(B_{11}^{k-2}B_{12})\rangle \geq (k-1)\operatorname{rank}(B_{12}) .$$

We note that $\operatorname{Im}(B_{12}), \operatorname{Im}(B_{11}B_{12}), \ldots, \operatorname{Im}(B_{11}^{k-2}B_{12})$, are subspaces of $\ker(B_{21})$. Further using the rank-nullity theorem, we get

$$\operatorname{rank}(B_{21}) < r - \frac{r\cdot(k-1)}{k} = \frac{r}{k} .$$

By using Fact 2.8, we again get that $\operatorname{rank}(B) \leq r(1+1/k)$.

In the above discussion, we assumed that

$$\dim\langle \operatorname{Im}(B_{12}) \cup \operatorname{Im}(B_{11}B_{12}) \cup \cdots \cup \operatorname{Im}(B_{11}^{k-2}B_{12})\rangle \geq (k-1)\operatorname{rank}(B_{12}).$$

What if this is not the case? We still want to use the same idea as above but we want to ensure this assumption. For this purpose, we use a series of elementary column operations on B to transform it to a new matrix B^*, which satisfies the above assumption. Since the rank of a matrix is invariant under elementary column operations, we obtain the desired rank bound. Now we show how to obtain this matrix B^* using a series of elementary column operations on B. Whenever we apply these elementary column operations on B, we shall also maintain the invariant that

$$B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \cdots = B_{21}B_{11}^{k-2}B_{12} = 0 .$$

Suppose

$$\dim\langle \operatorname{Im}(B_{12}) \cup \operatorname{Im}(B_{11}B_{12}) \cup \cdots \cup \operatorname{Im}(B_{11}^{k-2}B_{12})\rangle < (k-1)\operatorname{rank}(B_{12}) . \tag{5.8}$$

Let $\rho := \operatorname{rank}(B_{12})$. First, we can assume that B_{12} has exactly ρ non-zero columns. This can be achieved by performing elementary column operations on the last $n-r$ columns. This does not change the matrix $B_{22} = 0$. Furthermore, these column operations correspond to replacing B_{12} by $B_{12}\cdot S$ for some invertible $(n-r)\times(n-r)$-matrix S. Since

$$B_{22} = B_{21}B_{12} = B_{21}B_{11}B_{12} = \cdots = B_{21}B_{11}^{k-2}B_{12} = 0$$

implies

$$B_{21}B_{12}S = B_{21}B_{11}B_{12}S = \cdots = B_{21}B_{11}^{k-2}B_{12}S = 0 ,$$

we keep our invariant. We will call the new matrix again B_{12}.

Note that the image of a matrix is its column span. Since every matrix $B_{11}^{i}B_{12}$ has exactly ρ non-zero columns (since B_{12} has ρ non-zero columns and B_{11} is non-singular), assumption in equation (5.8) means that there is a linear dependence between these columns. That means there are vectors $y_0, y_1, \dots, y_{k-2} \in \mathbb{F}^{n-r}$, not all equal to zero, such that

$$\sum_{i=0}^{k-2} B_{11}^i B_{12} \cdot y_i = 0\,.$$

Moreover, we can assume that these vectors only have non-zero entries in the places that corresponds to non-zero columns of B_{12}. First we show that we can assume $y_0 \neq 0$. Suppose $0 \leq j \leq k-2$ is the least integer such that $y_j \neq 0$. So we left multiply the equation

$$\sum_{i=0}^{k-2} B_{11}^i B_{12} \cdot y_i = 0$$

by $(B_{11}^j)^{-1}$, giving us

$$(B_{11}^j)^{-1} \sum_{i=0}^{k-2} B_{11}^i B_{12} \cdot y_i = \sum_{i=j}^{k-2} B_{11}^{i-j} B_{12} \cdot y_i = 0\,.$$

By renumbering the indices, this can be re-written as

$$\sum_{i=0}^{k-2-j} B_{11}^i B_{12} \cdot y_i = 0\,.$$

Thus we can assume that $y_0 \neq 0$. (The new sum runs only up to $k-2-j$, for the missing summands, we choose the corresponding y_i to be zero.)

By writing

$$\sum_{i=0}^{k-2} B_{11}^i B_{12} \cdot y_i = 0 \qquad \text{as} \qquad B_{12} \cdot y_0 + B_{11} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1} B_{12} y_i = 0\,,$$

we see that there is a linear dependence between the columns of B_{12} and B_{11}. Let $\ell \in [n-r]$ be such that ℓ^{th} entry of y_0 is non-zero. Therefore, we can make the ℓ^{th} column of B_{12} zero by adding a multiple of

$$\sum_{i=1}^{k-2} B_{11}^i B_{12} \cdot y_i$$

and maybe adding some multiple of some other columns of B_{12} to it. This will decrease the rank of B_{12} by 1.

We claim that our invariant is still fulfilled. First, we add

$$B_{11} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1} B_{12} \cdot y_i$$

to the ℓ^{th} column of B_{12} and this will also add

$$B_{21} \cdot \sum_{i=1}^{k-2} B_{11}^{i-1} B_{12} \cdot y_i$$

to the ℓ^{th} column of B_{22}. Since the invariant was fulfilled before the operation, B_{22} will stay zero. As seen before, column operations within the last $n-r$ columns do not change B_{22}. Thus, one of the $n-r$ columns on the right-hand side (i. e., the side composed of B_{12} and B_{22}) of B became zero. We can remove this column from our consideration. Let B' and B'_{12} the matrices obtained from B and B_{12} by removing this zero column. Since the columns of B'_{12} are a subset of the columns of B_{12},

$$B_{21}B_{12} = B_{21}B_{11}B_{12} = \cdots = B_{21}B_{11}^{k-2}B_{12} = 0$$

implies that

$$B_{21}B'_{12} = B_{21}B_{11}B'_{12} = \cdots = B_{21}B_{11}^{k-2}B'_{12} = 0\,.$$

Therefore, our invariant is still valid.

We repeat this process until the equation (5.8) is not true anymore. Note that this happens for sure when $\operatorname{rank}(B_{12}) = 0$. At the end of this process we get a matrix B^* such that

$$\dim\langle \operatorname{Im}(B^*_{12}) \cup \operatorname{Im}(B_{11}B^*_{12}) \cup \cdots \cup \operatorname{Im}(B_{11}^{k-2}B^*_{12})\rangle \geq (k-1)\operatorname{rank}(B^*_{12})\,.$$

Now the rank bound follows from the argument given above. □

Finally, combining the above three lemmas, the following lemma gives the desired quantitative relation between the commutative rank and Wong index, essential to the analysis of our algorithm. It shows that higher the Wong index of the given matrix, the better it approximates the rank of the space.

Lemma 5.4. *Let $\mathbb{F}$ be any field, $A \in \mathcal{B} = \langle B_1, B_2, \ldots, B_m\rangle \leq \mathbb{F}^{n\times n}$ and $B = \sum_{i=1}^m x_i B_i$, then*

$$\operatorname{rank}(\mathcal{B}) = \operatorname{rank}(\langle A, B\rangle) \leq \operatorname{rank}(A)\left(1 + \frac{1}{w(A, \langle A, B\rangle)}\right). \tag{5.9}$$

Proof. Let $\operatorname{rank}(A) = r$. We use $\mathcal{C}$ to denote the matrix space $\langle A, B\rangle$, note that this space is being considered over the rational function field $\mathbb{F}(x_1, x_2, \ldots, x_m)$.

We know that there exist matrices $P, Q \in \mathbb{F}^{n\times n}$ such that

$$PAQ = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}. \tag{5.10}$$

Notice that $\operatorname{Im}(PAQ) = P\operatorname{Im}(A)$. Thus by Lemma 5.1, $w(A, \mathcal{C}) = w(PAQ, P\mathcal{C}Q)$. Also, it is easy to see that $\operatorname{rank}(A) = \operatorname{rank}(PAQ)$ and $\operatorname{rank}(\mathcal{C}) = \operatorname{rank}(P\mathcal{C}Q)$. Hence it is enough to show that

$$\operatorname{rank}(P\mathcal{C}Q) \leq \operatorname{rank}(PAQ)\left(1 + \frac{1}{w(PAQ, P\mathcal{C}Q)}\right). \tag{5.11}$$

For the sake of simplicity, we just write $P\mathcal{C}Q$ as $\mathcal{C}$ and PAQ as A. Thus we have

$$A = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}. \tag{5.12}$$

We write B as

$$B = \begin{array}{cc} & \begin{array}{cc} r \text{ columns} & n-r \text{ columns} \end{array} \\ \begin{array}{c} r \text{ rows} \\ n-r \text{ rows} \end{array} & \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \end{array} \tag{5.13}$$

We get that B_{11} is non-singular over the field $\mathbb{F}(x_1, x_2, \ldots, x_m)$ since $A \in \mathcal{B}$. Also, we get by Lemma 5.2 that the first $w(A, \mathcal{C})$ entries of the sequence of matrices

$$B_{22}, B_{21}B_{12}, B_{21}B_{11}B_{12}, \ldots, B_{21}B_{11}^{i}B_{12} \ldots$$

are zero matrices. Now we apply Lemma 5.3 to obtain that

$$\operatorname{rank}(B) = \operatorname{rank}(\mathcal{B}) = \operatorname{rank}(\mathcal{C}) \leq \operatorname{rank}(A)\left(1 + \frac{1}{w(A,\mathcal{C})}\right). \tag{5.14}$$

□

Lemma 5.5. *If $|\mathbb{F}| > n$, $A \in \mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n \times n}$, $B = \sum_{i=1}^{m} x_i B_i$ and $w(A, \langle A, B \rangle) < k$ for some $k \in [n]$, then there exist $1 \leq i_1, i_2, \ldots, i_k \leq m$ and $\lambda_1, \lambda_2, \ldots, \lambda_k \in \mathbb{F}$ such that $w(A, \langle A, C \rangle) < k$, where $C = \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \cdots + \lambda_k B_{i_k}$.*

Proof. Let $\operatorname{rank}(A) = r$. We know that there exist matrices $P, Q \in \mathbb{F}^{n \times n}$ such that

$$PAQ = \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}. \tag{5.15}$$

Let $A' = PAQ$, $\mathcal{B}' = P\mathcal{B}Q$ and $B' = \sum_{i=1}^{m} x_i P B_i Q$. We write B' as

$$B' = \begin{array}{cc} & \begin{array}{cc} r \text{ columns} & n-r \text{ columns} \end{array} \\ \begin{array}{c} r \text{ rows} \\ n-r \text{ rows} \end{array} & \begin{bmatrix} B'_{11} & B'_{12} \\ B'_{21} & B'_{22} \end{bmatrix} \end{array} \tag{5.16}$$

By using Lemma 5.1, we know that $w(A, \langle A, B \rangle) = w(A', \langle A', B' \rangle) < k$. By using Lemma 5.2, we get that there exists $t \leq k$ such that $B'_{21}(B'_{11})^{t-2}B'_{12} \neq 0$ and

$$(B')^t = \begin{array}{cc} & \begin{array}{cc} r \text{ columns} & n-r \text{ columns} \end{array} \\ \begin{array}{c} r \text{ rows} \\ n-r \text{ rows} \end{array} & \begin{bmatrix} B''_{11} & B''_{12} \\ B''_{21} & B'_{21}(B'_{11})^{t-2}B'_{12} \end{bmatrix} \end{array} \tag{5.17}$$

for some matrices $B''_{11}, B''_{12}, B''_{21}$. Since the entries of the matrix $B'_{21}(B'_{11})^{t-2}B'_{12}$ are polynomials in the variables $x_1, x_2, \ldots, x_m$ of degree at most k, there exists an assignment to these variables by field constants, assigning at most k variables non-zero values such that $B'_{21}(B'_{11})^{t-2}B'_{12}$ evaluates to a non-zero matrix. By using Lemma 5.2 again, this assignment gives us a matrix $C' \in \mathcal{B}'$ such that $w(A', \langle A', C' \rangle) < k$. By using Lemma 5.1, the same assignment of the variables gives us a matrix $C \in \mathcal{B}$ such that $w(A, \langle A, C \rangle) < k$. □

6 Final algorithm

Suppose we have a matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n\times n}$, $B = \sum_{i=1}^m x_i B_i$ and a matrix $A \in \mathcal{B}$. Our goal is find a matrix D in $\mathcal{B}$ such that its rank is "close" to the commutative rank of $\mathcal{B}$. If the Wong index $w(A, \langle A, B \rangle)$ of A in $\langle A, B \rangle$ is "large," then we know by Lemma 5.4 that rank of of A is "close" to the commutative rank of $\mathcal{B}$, which is equal to the commutative rank of $\langle A, B \rangle$. What if this Wong index $w(A, \langle A, B \rangle)$ is "small"? Then we know by Lemma 5.5 that by trying out a small number (that means, $m^{w(A,\mathcal{B})+1}$) of possibilities of combinations of B_i, we can find a matrix $C \in \mathcal{B}$ such that Wong index $w(A, \langle A, B \rangle)$ of A in $\langle A, C \rangle$ is also "small." Using Fact 4.6, we obtain that rank of A is not maximum in $\langle A, C \rangle$. Thus there exists $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda C) > \operatorname{rank}(A)$. And we can find this λ quite efficiently. Also, $A + \lambda C \in \mathcal{B}$. Thus we can efficiently find a matrix of bigger rank if we are given a matrix of "small" Wong index. This idea is formalized in the following Algorithm.

Algorithm 2 Greedy algorithm for $(1-\varepsilon)$-approximating commutative rank

Input : A matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n\times n}$, given as a list of basis matrices $B_1, B_2, \ldots, B_m$. An approximation parameter $0 < \varepsilon < 1$.
Output: A matrix $A \in \mathcal{B}$ such that $\operatorname{rank}(A) \geq (1-\varepsilon) \cdot \operatorname{rank}(\mathcal{B})$
Initialize $A = 0 \in \mathbb{F}^{n\times n}$ to the zero matrix.
Assign $\ell = \lceil \frac{1}{\varepsilon} - 1 \rceil$.
while *Rank is increasing* **do**
  for *each* $\{i_1, i_2, \ldots, i_\ell\} \in \binom{[m]\setminus\{0\}}{\ell}$ **do**
    `/* This means we try all combinations of matrices` $B_{i_1}, B_{i_2}, \ldots, B_{i_\ell}$ `*/`
    Check if there exist $\lambda_1, \lambda_2, \ldots, \lambda_\ell \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \cdots + \lambda_\ell B_{i_\ell}) > \operatorname{rank}(A)$.
    if $\operatorname{rank}(A + \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \cdots + \lambda_\ell B_{i_\ell}) > \operatorname{rank}(A)$ **then**
      Update $A = A + \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \cdots + \lambda_\ell B_{i_\ell}$.
return A.

The following theorem proves the correctness of Algorithm 2. Let s be an upper bound on the bit size of the entries of $B_1, \ldots, B_m$.

Theorem 6.1. *Assume that* $|\mathbb{F}| > n$. *Algorithm 2 runs in time* $O((mn)^{1/\varepsilon} \cdot M(n, s + \log n) \cdot n)$ *and returns a matrix* $A \in \mathcal{B}$ *such that* $\operatorname{rank}(A) \geq (1-\varepsilon) \cdot \operatorname{rank}(\mathcal{B})$*, where* $M(n,t)$ *is the time required to compute the rank of an* $n \times n$ *matrix with entries of bit size at most* t*.*

Proof. Suppose $B = \sum_{i=1}^m x_i B_i$ and A be the rank r matrix returned by Algorithm 2. Let k be the Wong index $w(A, \langle A, B \rangle)$ of $(A, \langle A, B \rangle)$. By Lemma 5.4, we know that $\operatorname{rank}(\mathcal{B}) \leq r(1 + 1/k)$. Thus

$$r \geq \left(1 - \frac{1}{k+1}\right) \operatorname{rank}(\mathcal{B}).$$

If $\varepsilon \geq 1/(k+1)$, then we are done. Otherwise we have that $\varepsilon < 1/(k+1)$, i. e., $k < 1/\varepsilon - 1$. Since $\ell = \lceil 1/\varepsilon - 1 \rceil$, we also have $w(A, \langle A, B \rangle) < \ell$. By using Lemma 5.5, we get that there exist $1 \leq i_1, i_2, \ldots, i_\ell \leq$

m and $\lambda_1, \lambda_2, \dots, \lambda_\ell \in \mathbb{F}$ such that that $w(A, \langle A, C\rangle) < \ell$, where $C = \lambda_1 B_{i_1} + \lambda_2 B_{i_2} + \dots + \lambda_\ell B_{i_\ell}$. By using Fact 4.6, we get that A is not of maximum rank in $\langle A, C\rangle$. Thus there exists $\lambda \in \mathbb{F}$ such that $\operatorname{rank}(A + \lambda C) > A$, and we shall detect this in Algorithm 2 since we try all possible choices of $i_1, i_2, \dots, i_\ell$.

The desired running time can be proved easily. The outer while loop runs at most n times, thus the total running time is at most n times the running time of one iteration. One iteration of the outer loop has

$$\binom{[m] \setminus \{0\}}{\ell} = O(m^{1/\varepsilon})$$

iterations of the inner for loop. By using the Schwartz–Zippel Lemma [25, 20], one iteration of inner for loop needs to try at most $(n+1)^\ell = O(n^{1/\varepsilon})$ possible values of $\lambda_1, \lambda_2, \dots, \lambda_\ell \in \mathbb{F}$. And then we perform two instances of rank computation. The stated running time follows. □

Remark 6.2. Algorithm 2 runs in time $O((mn)^{1/\varepsilon} \cdot n \cdot M(n))$ in the algebraic RAM model. Here $M(n)$ is the time required to compute the rank of an $n \times n$ matrix in the algebraic RAM model. It is known that $M(n) = O(n^\omega)$ with ω being the exponent of matrix multiplication. Since one can assume that $m \le n^2$, Algorithm 2 runs in time $O(n^{3/\varepsilon + \omega + 1})$ in the algebraic RAM model.

The statement of the above remark and the trivial fact that $\omega \le 3$, gives us the running time stated in the abstract.

Remark 6.3. With a more refined analysis, it can be seen that Algorithm 2 uses

$$O\big((mn)^{1/\varepsilon} \cdot n \cdot M(n, s + \log n)\big)$$

bit operations if the entries of the input matrices $B_1, B_2, \dots, B_m$ have bit-size at most s. Here $M(n,t)$ is the bit complexity of computing the rank of a matrix whose entries have bit-size at most t. The additional $\log n$ in the bit-size comes from the fact that the entries of the final matrix A are by a polynomial factor (in n) larger than the entries of the B_i due to the update steps.

7 Tight examples

We conclude by giving some tight examples, which show that the analysis of the approximation performance of the greedy approximation scheme cannot be improved. Consider the following matrix space of $n \times n$-matrices:

$$\left(\begin{array}{cccc|cccc} * & 0 & \dots & 0 & * & 0 & \dots & 0 \\ 0 & * & \dots & 0 & 0 & * & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & * & 0 & 0 & \dots & * \\ \hline 0 & 0 & \dots & 0 & * & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & * & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & * \end{array}\right) \tag{7.1}$$

Each block has size $n/2 \times n/2$. This space consists of all matrices where we can substitute arbitrary values for the $*$ and the basis consists of all matrices where exactly one $*$ is replaced by 1 and all others are set to 0. Assume that $\varepsilon = 1/2$, that means, that the greedy algorithm only looks at sets of size $\ell = 1$. Furthermore, assume that the matrix A constructed so far is

$$A = \begin{pmatrix} 0 & I_{\frac{n}{2}} \\ 0 & 0 \end{pmatrix}. \tag{7.2}$$

Any single basis matrix cannot improve the rank of A, since either its non-zero column is contained in the column span of A or its non-zero row is contained in the row span of A. On the other hand, the matrix space contains a matrix of full rank n, namely, the identity matrix.

The next space for the case $\ell = 2$ looks like this:

$$\left(\begin{array}{cccc|cccc|cccc}
* & 0 & \dots & 0 & * & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\
0 & * & \dots & 0 & 0 & * & \dots & 0 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \dots & * & 0 & 0 & \dots & * & 0 & 0 & \dots & 0 \\
\hline
0 & 0 & \dots & 0 & * & 0 & \dots & 0 & * & 0 & \dots & 0 \\
0 & 0 & \dots & 0 & 0 & * & \dots & 0 & 0 & * & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \dots & 0 & 0 & 0 & \dots & * & 0 & 0 & \dots & * \\
\hline
0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & * & 0 & \dots & 0 \\
0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & * & \dots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & *
\end{array}\right) \tag{7.3}$$

and the corresponding matrix A is

$$A = \begin{pmatrix} 0 & I_{\frac{2n}{3}} \\ 0 & 0 \end{pmatrix}. \tag{7.4}$$

By an argument similar to above, it is easy to see that we need at least three matrices to improve the rank of A, so the algorithm gets stuck with a $2/3$-approximation.

The above scheme generalizes to arbitrary values of ℓ in the obvious way.

Acknowledgments. We would like to thank Ankit Garg and Rafael Oliveira for pointing out some inaccuracies in an earlier draft of the paper. We would also like to thank the anonymous reviewers for their helpful and constructive comments that have helped us to improve the final version of the paper.

References

[1] MARKUS BLÄSER, GORAV JINDAL, AND ANURAG PANDEY: Greedy strikes again: A deterministic PTAS for commutative rank of matrix spaces. In *Proc. 32nd Computational Complexity Conf.*

(CCC'17), volume 79 of *LIPIcs*, pp. 33:1–33:16. DROPS, 2017. [doi:10.4230/LIPIcs.CCC.2017.33] 1

[2] JONATHAN F. BUSS, GUDMUND SKOVBJERG FRANDSEN, AND JEFFREY O. SHALLIT: The computational complexity of some problems of linear algebra. *J. Comput. System Sci.*, 58(3):572–596, 1999. Preliminary version in STACS'97. [doi:10.1006/jcss.1998.1608] 2

[3] HARM DERKSEN AND VISU MAKAM: On non-commutative rank and tensor rank. *Linear and Multilinear Algebra*, pp. 1–16, 2017. [doi:10.1080/03081087.2017.1337058, arXiv:1606.06701] 5, 6

[4] JACK R. EDMONDS: Systems of distinct representatives and linear algebra. *J. Res. Nat. Bur. Standards Sect. B*, 71(4):241–245, 1967. [doi:10.6028/jres.071B.033] 2

[5] DAVID EISENBUD AND JOE HARRIS: Vector spaces of matrices of low rank. *Advances in Mathematics*, 70(2):135–155, 1988. [doi:10.1016/0001-8708(88)90054-0] 3

[6] STEPHEN A. FENNER, ROHIT GURJAR, AND THOMAS THIERAUF: Bipartite perfect matching is in quasi-NC. In *Proc. 48th STOC*, pp. 754–763. ACM Press, 2016. [doi:10.1145/2897518.2897564, arXiv:1601.06319] 3

[7] MARC FORTIN AND CHRISTOPHE REUTENAUER: Commutative/noncommutative rank of linear matrices and subspaces of matrices of low rank. *Séminaire Lotharingien de Combinatoire*, 52:B52f, 2004. Available from EuDML. 3, 5

[8] HAROLD N. GABOW AND MATTHIAS F. M. STALLMANN: An augmenting path algorithm for linear matroid parity. *Combinatorica*, 6(2):123–150, 1986. Preliminary version in FOCS'84. [doi:10.1007/BF02579169] 3

[9] ANKIT GARG, LEONID GURVITS, RAFAEL MENDES DE OLIVEIRA, AND AVI WIGDERSON: A deterministic polynomial time algorithm for non-commutative rational identity testing. In *Proc. 57th FOCS*, pp. 109–117. IEEE Comp. Soc. Press, 2016. [doi:10.1109/FOCS.2016.95, arXiv:1511.03730] 3, 5

[10] ROHIT GURJAR AND THOMAS THIERAUF: Linear matroid intersection is in quasi-NC. In *Proc. 49th STOC*, pp. 821–830. ACM Press, 2017. [doi:10.1145/3055399.3055440] 3

[11] LEONID GURVITS: Classical complexity and quantum entanglement. *J. Comput. System Sci.*, 69(3):448–484, 2004. Preliminary version in STOC'03. [doi:10.1016/j.jcss.2004.06.003] 3

[12] GÁBOR IVANYOS, MAREK KARPINSKI, YOUMING QIAO, AND MIKLOS SANTHA: Generalized Wong sequences and their applications to Edmonds' problems. *J. Comput. System Sci.*, 81(7):1373–1386, 2015. Preliminary version in STACS'14. [doi:10.1016/j.jcss.2015.04.006, arXiv:1307.6429] 3, 7, 8, 9

[13] GÁBOR IVANYOS, MAREK KARPINSKI, AND NITIN SAXENA: Deterministic polynomial time algorithms for matrix completion problems. *SIAM J. Comput.*, 39(8):3736–3751, 2010. [doi:10.1137/090781231, arXiv:0907.0774] 3

[14] GÁBOR IVANYOS, YOUMING QIAO, AND K. VENKATA SUBRAHMANYAM: Constructive non-commutative rank computation in deterministic polynomial time, 2015-2018. [arXiv:1512.03531] 3

[15] LÁSZLÓ LOVÁSZ: On determinants, matchings, and random algorithms. In *Proc. 2nd Internat. Conf. Fundamentals of Computation Theory (FCT'79)*, pp. 565–574, 1979. 2, 3

[16] LÁSZLÓ LOVÁSZ: Singular spaces of matrices and their application in combinatorics. *Boletim da Sociedade Brasileira de Matemática-Bulletin/Brazilian Mathematical Society*, 20(1):87–99, 1989. [doi:10.1007/BF02585470] 3

[17] LÁSZLÓ LOVÁSZ AND MICHAEL D. PLUMMER: *Matching Theory*. Volume 367. AMS Chelsea Publ., 2009. 3

[18] MEENA MAHAJAN AND V. VINAY: Determinant: Combinatorics, algorithms, and complexity. *Chicago J. Theor. Comput. Sci.*, (5), 1997. [doi:10.4086/cjtcs.1997.005] 2

[19] JAMES B. ORLIN: A fast, simpler algorithm for the matroid parity problem. In *Proc. 13th Internat. Conf. on Integer Programming and Combinatorial Optimization (IPCO'08)*, volume 5035 of *Lecture Notes in Comp. Sci.*, pp. 240–258. Springer, 2008. [doi:10.1007/978-3-540-68891-4_17] 2

[20] JACOB T. SCHWARTZ: Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980. Preliminary version in EUROSAM'79. [doi:10.1145/322217.322225] 2, 17

[21] F. BRUCE SHEPHERD AND ADRIAN VETTA: The demand-matching problem. *Math. Oper. Res.*, 32(3):563–578, 2007. Preliminary version in IPCO'02. [doi:10.1287/moor.1070.0254] 3

[22] SEINOSUKE TODA: Counting problems computationally equivalent to computing the determinant. *Technical Report CSIM 91-07*, 1991. 2

[23] LESLIE G. VALIANT: Completeness classes in algebra. In *Proc. 11th STOC*, pp. 249–261. ACM Press, 1979. [doi:10.1145/800135.804419] 2

[24] V. VINAY: Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proc. 6th IEEE Conf. on Structure in Complexity Theory (SCT'91)*, pp. 270–284. IEEE Comp. Soc. Press, 1991. [doi:10.1109/SCT.1991.160269] 2

[25] RICHARD ZIPPEL: Probabilistic algorithms for sparse polynomials. In *Proc. Symbolic and Algebraic Comput. (EUROSAM'79)*, volume 72 of *Lecture Notes in Comp. Sci.*, pp. 216–226. Springer, 1979. [doi:10.1007/3-540-09519-5_73] 2, 17

AUTHORS

Markus Bläser
Department of Computer Science, Saarland University
Saarland Informatics Campus, Saarbrücken, Germany
mblaeser@cs.uni-saarland.de
http://www-cc.cs.uni-saarland.de/mblaeser/

Gorav Jindal
Max-Planck-Institut für Informatik & Saarbrücken Graduate School of Computer Science
Saarland Informatics Campus, Saarbrücken, Germany
gjindal@mpi-inf.mpg.de
http://people.mpi-inf.mpg.de/~gjindal/

Anurag Pandey
Max-Planck-Institut für Informatik & Saarbrücken Graduate School of Computer Science
Saarland Informatics Campus, Saarbrücken, Germany
apandey@mpi-inf.mpg.de
http://people.mpi-inf.mpg.de/~apandey/

ABOUT THE AUTHORS

MARKUS BLÄSER is a full professor of computer science at Saarland University, Saarbrücken, Germany. Before joining Saarbrücken, he was an assistant professor at ETH Zurich. He obtained his Ph. D. from the University of Bonn under the supervision of Arnold Schönhage and his Habilitation from the University of Lübeck under the supervision of Rüdiger Reischuk.

GORAV JINDAL did his undergraduate studies at the Computer Science department of IIT Delhi. He did his master studies at the Computer Science department of Saarland University. Since 2014, he has been a Ph. D. student at the Max-Planck-Institut für Informatik under the supervision of Markus Bläser. Gorav's research interests lie in algebraic complexity theory, computer algebra and lower bounds in general.

ANURAG PANDEY did his bachelor's and master's studies at the Electrical Engineering department of IIT Kanpur. His master's thesis was supervised by Nitin Saxena. Since 2015, he has been a Ph. D. student at Max-Planck-Institut für Informatik under the supervision of Markus Bläser. Anurag's research interests mostly revolve around algebraic complexity theory.