Operator Scaling: Theory, Applications and Connections[1]

Avi Wigderson
Institute for Advanced Study, Princeton

July 6–8, 2017

[1]A tutorial given at CCC'17. I am indebted to Clément Canonne, Chin Ho Lee, Pritish Kamath, and Aditya Potukuchi who scribed the lectures so thoroughly, and to Ankit Garg and Rafael Oliveira for carefully editing them. Videos of the lectures are available at http://computationalcomplexity.org/Archive/2017/tutorial.html. There were three 2-hour lectures, and these notes, like the videos, are partitioned to chapters accordingly.

Contents

Chapter 1

Matrix and Operator Scaling

Scribed by: Clément Canonne (ccanonne@cs.columbia.edu)

Abstract

I will define the problems of matrix and operator scaling, their origins, motivations, connections and applications. I will describe algorithms for solving both of them, explaining the similarities and differences in their performance analyses. I will then give brief introductions to the parts of Invariant Theory, Non-commutative fields and Quantum information theory that are used in motivating and analyzing operator scaling, and the key notion of *capacity* which ties many of the parts together. I will hint at why this new algorithm has potential to solve new non-convex programs and exponentially large linear programs. Along the way we will encounter more familiar problems and topics in algorithms and complexity like perfect matchings and polynomial identity testing, and see different or more general ways to look at them.

I find it quite special that a single computational problem and algorithm for it connects so many areas from CS and math, only some of which I will have time to describe. So far, in CS they include algebraic complexity, optimization and formal language theory, and within Math aspects of non-commutative algebra, Invariant theory, analysis, quantum information theory and control theory. Some of these connections between these fields are new. Moreover, follow-up works in a variety of natural directions, computational and mathematical, bring even more areas to play, creates new collaborations and makes the developing area very exciting.

The structure of the notes follows the structure of the lectures, one chapter per lecture, to make them correspond to the available videos.[1] However we have added to them many details and references missing from the lectures. They also contain a long list of (homework) problems at all levels, some in the text itself and some in the final section, which the interested reader is encouraged to solve.

Much of my interest and understanding in these problems is based on several joint works with Ankit Garg, Pavel Hrubeš, Leonid Gurvits, and Rafael Oliveira.

[1] Available at http://computationalcomplexity.org/Archive/2017/tutorial.html

1.1 Symbolic Matrices

We define the central object which we will study in these lectures. Here they arise in the setting of arithmetic complexity, and for an excellent survey of the topic, especially some models and problems mentioned in these lectures, we refer the reader to [SY10].

Definition 1.1 (Symbolic Matrix)**.** Let $\mathbb{F}$ be a field, and $\mathbf{x} = \{x_1, \dots, x_m\}$ be a set of variables. A *symbolic matrix* is a matrix $L = (\ell_{ij})_{1 \le i,j \le n}$ where each entry $\ell_{ij}(x_1, \dots, x_m)$ is a linear form in the variables, with coefficients in $\mathbb{F}$. In particular, one can write

$$L = A_1 x_1 + \cdots + A_m x_m,$$

with $A_j \in \mathscr{M}_n(\mathbb{F})$ for all j, and consider the tuple $L = (A_1, \dots, A_m)$. (Without loss of generality, in this lectures, we can assume the A_j's are linearly independent, and so that $m \le n^2$. This will allow us to focus on n as the main size parameter of the input.)

This tuple L, which will be the input to many problems we discuss, can be considered from many viewpoints, each of them bringing its own insight. Viewed as a symbolic matrix, L is *non-singular* if there exists M such that $LM = ML = I$ where I is the identity matrix. As we will see, the allowed entries of M will depend on the context.
The fundamental problem we will consider is that of *singularity testing*: given a symbolic matrix L, is this matrix singular (over the appropriate field of fractions)? This problem appears in two settings, as we see below.

Commutative case The first setting is when our variables $\mathbf{x} = \{x_1, \dots, x_m\}$ commute. In this case, denoting as usual by $\mathbb{F}[\mathbf{x}]$ the ring over field $\mathbb{F}$ of polynomials in $\mathbf{x}$, and $\mathbb{F}(\mathbf{x})$ the field of rational functions, the problem is defined as follows.

SING: given input L over m (commutative) variables, answer "Is L singular in $\mathbb{F}(\mathbf{x})$?" (Equivalent to asking if $\det L \equiv 0$ over $\mathbb{F}[\mathbf{x}]$.)

Let us consider the complexity of this problem.

- This problem was first put forth by Edmonds [Edm67], who asked if this problem was in P (and we shall soon see why). Note that this is roughly the time the class P was defined, by Edmonds and others. This question remains open!
- Lovász [Lov79] showed it was in BPP (the algorithm works by assigning random values to the variables and evaluating the numeric determinant). This one of the earliest probabilistic algorithms!

Why is his problem a big deal? Two more complexity discoveries clarified it!

- Valiant [Val79] showed that SING captures most algebraic identities mathematicians care about, as it is (essentially) complete for testing whether a given arithmetic formula[2] computes the identically zero polynomial.

[2]More generally an ABP, namely an Arithmetic Branching Program.

- Kabanets and Impagliazzo [KI04] showed that derandomizing SING (i.e., showing SING $\in$ SUBEXP) would show that either VP $\neq$ VNP or NEXP $\not\subset$ $\mathrm{P}_{/\mathrm{poly}}$. In short, *striking* consequences of a striking nature: a single deterministic algorithm (upper bound) for a single (simple) problem will imply impressive lower bounds.

Responses: "hey, that looks hard, let's not work on that" (pessimist), or, "hey, in that case let's find an algorithm, if that's all it takes!" (optimist).
This is the main open problem motivating us. Unable to solve it, we move (in the best mathematical tradition) to a syntactically similar but different problem. This will open a treasure trove in itself, and will end up teaching us some new things about SING, as we shall see later, including an efficient deterministic approximation scheme for the rank of commutative symbolic matrices, an approach to PIT via invariant theory.

Non-commutative case In all the above, we assumed the variables x_i commute. What if they do not? $\rightsquigarrow$ leads to different and strange world. *Non-commutative version of singularity*:

NC-SING: given input L (over non-commutative variables), answer "Is L singular (non invertible) in $\mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$?"

Here $\mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$ denotes the *free skew field*, a very complicated object which we will define later. So, our computational problem is not defined yet, and certainly not motivated yet. We will see elementary formulations of this problem as well as plenty of motivations in the sequel. Note that there is no analog to the determinant formulation of SING. In non-commutative variables there is no "canonical" definition of determinant. The correct notion, which is very related to taking the inverse as we do here, is called "quasi-determinant" (a beautiful survey is here [GGRW02]).

The complexity of this problem NC-SING is what will occupy us in these lecture. The first thing to observe is that unlike its commutative sister SING, it was not even clear, and required hard work even to prove its decidability. Here is some of the evolution of understanding its complexity.

- [Coh73, Coh75]: NC-SING is decidable!
- [CR99]: NC-SING $\in$ PSPACE
- [GGOW16]: NC-SING $\in$ P! (over fields $\mathbb{F}$ of characteristic 0)
- [DM15, IQS15]: NC-SING $\in$ P! (over any field $\mathbb{F}$)

Question 1.2 (Igor Carboni–Oliveira). Is there any hope for a *parallel* algorithm? (i.e., NC-SING $\in$ NC?) (answer: it is known to be in RandomNC, due to a result of [DM15]).

The two polynomial time algorithms are of a very different nature: the one in [GGOW16] is *analytic* in nature, and the one in [IQS15] is *combinatorial / linear-algebraic*. We will focus now on developing the tools and ideas towards establishing the first polynomial time algorithm in characteristic 0, although many are relevant also to the second algorithm that works for all fields (indeed, the stronger results in Invariant Theory which enabled the second were discovered a few weeks after the first algorithm was published). The second algorithm has another advantage – it efficiently certifies its output! However, the first has extra features and computes certain quantities that are meaningless in positive characteristics, and these have extra motivations as we shall see at the end of the last lecture.

To explain "operator scaling", the main algorithm, we first survey its precursor, "matrix scaling."

1.2 The Commutative Case, SING, Perfect Matchings, and Matrix Scaling

The problem SING was first introduced in [Edm67], because of its connection to the *perfect matching* problem in bipartite graphs. To explain it, recall that to an $n \times n$ bipartite graph G corresponds a Boolean matrix A_G with entries (i, j) set to 1 if (i, j) is an edge and zero otherwise; then, one can observe that G has a perfect matching iff $\operatorname{perm}(A_G) > 0$.
Now, considering the *symbolic* matrix L_G corresponding to setting the (i, j) entry to x_{ij} when there is an edge (i, j) in G (and 0 otherwise), we further get that

$$G \text{ has a perfect matching } \Leftrightarrow \operatorname{perm}(A_G) > 0 \Leftrightarrow \det(L_G) \not\equiv 0.$$

This establishes the perfect matching problem in bipartite graphs as a very special case SING, in which the linear forms in every entry of the symbolic matrix are either 0 or a distinct variable. This viewpoint will soon lead us to a very different algorithm for matching than the familiar one taught in algorithms courses.

1.2.1 Matrix Scaling ([Sin64, LSW00])

To introduce the main algorithm in these lectures, for the *operator scaling* problem, we will first develop an algorithm for a simpler (and older) one, *matrix scaling*. This will turn out to also solve (in a very different way than usual) the perfect matching problem in bipartite graphs. We begin by defining the *(exact) matrix scaling* problem.

(EXACT) MATRIX SCALING: Given $A = (a_{ij})_{1\leq i,j\leq n} \in \mathcal{M}_n(\mathbb{R})$ with non-negative entries, determine whether there exists R, C diagonal such that $B := RAC$ is doubly stochastic, i.e. such that $\mathbb{1} = B\mathbb{1} = B^T\mathbb{1}$.

Remark 1.3. A generalization of the above question is that of (r, c)*-scaling*, where one asks that the sum of row and column vectors be equal to some prescribed vectors r, c, respectively (instead of $\mathbb{1}$ and $\mathbb{1}$).

Why should we care about Matrix Scaling? Well,

- Many practical linear system solvers start by scaling (if possible) their input matrix as a pre-processing step, to improve stability of the numerical algorithm[3]

[3] It turns out that scaling to double stochastic may not always improve numerical stability of a linear system solver [ALOW]. Its prevalent use seems to suggest people believe it helps in many inputs.

- Applications to signal processing
- Implications to deterministic approximation of the permanent
- Perfect matching algorithms
- Connections to hyperbolic polynomials
- etc.

This is the *exact* matrix scaling problem. More relevant to us, and indeed to many of the applications above, is its *approximate* version, which only require that B be arbitrarily close to a doubly stochastic matrix.

MATRIX SCALING: Given $A = (a_{ij})_{1\leq i,j\leq n} \in \mathcal{M}_n(\mathbb{R})$ with non-negative entries, determine whether for every $\varepsilon \in (0,1)$ there exists $R_\varepsilon, C_\varepsilon$ diagonal such that $B_\varepsilon := R_\varepsilon A C_\varepsilon$ is approximately doubly stochastic: $(1-\varepsilon)\mathbb{1} \leq B\mathbb{1}, B^T\mathbb{1} \leq (1+\varepsilon)\mathbb{1}$.

When a matrix A can be scaled as above to an approximately doubly stochastic matrix, we say that A is *scalable*.

Exercise 1.4. With the same notations as before, G has a perfect matching $\Leftrightarrow$ $\operatorname{perm}(A_G) > 0 \Leftrightarrow \det(L_G) \not\equiv 0 \Leftrightarrow A_G$ is scalable.

The intuition for the algorithm, informally presented below, is quite simple and the only surprising part is that it actually works. It is basically a greedy approach. Since we have two sets of conditions to satisfy (namely, normalized rows, and normalized columns), a natural idea is to alternately satisfy the one at a time, and repeat this process: given $A = (a_{ij})_{1\leq i,j\leq n} \in \mathcal{M}_n(\mathbb{R})$ with non-negative entries,

1. Normalize the rows of A so that they all sum to 1, getting a matrix A_1
2. Normalize the columns of A_1 so that they all sum to 1, getting a matrix A_2
3. Repeat these two steps some number of time.

This type of approach, of iteratively satisfying or optimizing only some of the constraints in a complex problems, and then others, is often called "alternate minimization." There are many examples in which this general heuristic is used in optimization. However, sometimes it does not converge – or even if it does, may not converge quickly. A famous example is the Lemke–Houson algorithm for finding a Nash equilibrium in 2-player games, which alternatingly picks an optimal response for one player given strategy of the other. This algorithm does converges, but in worst-case exponential time. This one will be efficient.

Let us define the algorithm more formally, and then explain its rather simple analysis. Define the scaling factors for a matrix A as $R(A) := \operatorname{diag}(R_1, \ldots, R_n)$ and $C(A) := \operatorname{diag}(C_1, \ldots, C_n)$, where R_i (resp. C_i) is the i-th row (resp. column) sum of A. After each iteration we check how close the current matrix is to a doubly-stochastic one, and if it is, we halt. We refer to this algorithm as "Algorithm S," for *Sinkhorn*.[4]

[4]We will see later the analogue in the non-commutative case, "Algorithm G", for *Gurvits*).

Algorithm 1 Algorithm S

```
Require: $A = (a_{ij})_{1\le i,j\le n} \in \mathcal{M}_n(\mathbb{R})$ with non-negative entries
 1: for poly($n$) steps do
 2:     $A \leftarrow R(A)^{-1} \cdot A$
 3:     if $\|C(A) - I\|_2 \le \frac{1}{\sqrt{n}}$ then
 4:         return yes                                  ▷ $\mathrm{perm}(A) > 0$
 5:     end if
 6:     $A \leftarrow A \cdot C(A)^{-1}$
 7:     if $\|R(A) - I\|_2 \le \frac{1}{\sqrt{n}}$ then
 8:         return yes                                  ▷ $\mathrm{perm}(A) > 0$
 9:     end if
10: end for
11: return no                                          ▷ $\mathrm{perm}(A) = 0$
```

We will only aim at a polynomial time upper bound on the complexity of Algorithm S. A lot of effort has been invested in making matrix scaling very efficient; here is some partial list of results (see the survey [Ide16]).

- [Sin64]: $\mathrm{poly}(b, 1/\varepsilon)$ where b is the bit size of the instance (Sinkhorn only proved that his iterative algorithm converges while the first general complexity analysis appeared in [GY98] and [LSW00])
- [KK96]: $\mathrm{poly}(b, \log(1/\varepsilon))$
- [LSW00]: $\mathrm{poly}(n, \log(1/\varepsilon))$ (a strongly polynomial time algorithm)
- [CMTV17, ALOW17]: near linear time

Analysis of the algorithm of [LSW00] Denote the intermediate matrices produced by the algorithm execution by $A^{(0)} = A, \dots, A^{(t)}, \dots$. For simplicity, assume first that A is $\{0,1\}$-valued.[5] Our progress measure will be $\mathrm{perm}(A^{(t)})$; here is the outline of the three stages of the analysis.

1. $\mathrm{perm}(A^{(t)}) \le 1$ for all $t \ge 1$ (as the columns or rows sum to 1);
2. $\mathrm{perm}(A^{(1)}) \ge n^{-n}$;
3. For $t \ge 1$, $\frac{\mathrm{perm}(A^{(t+1)})}{\mathrm{perm}(A^{(t)})} \ge \exp(1/6n)$. This is because, for even t (and analogously for odd t),

$$\frac{\mathrm{perm}(A^{(t+1)})}{\mathrm{perm}(A^{(t)})} = \prod_j \frac{1}{R_j(A^{(t)})} \ge \exp(1/6n)\,.$$

Exercise 1.5. Show that if $\mathrm{perm}(A) = 0$ then Algorithm 1 does not converge.

Exercise 1.6. Show 2.

Exercise 1.7. Show 3 using the fact that column sums are 1 and $\|R(A) - I\|_2 \ge \frac{1}{\sqrt{n}}$.

[5]Otherwise, in the time analysis, replace n by the total bit size of the matrix entries (where we assume rational input).

Remark 1.8. Following [GY98], define the *capacity* of a non-negative matrix A as

$$\operatorname{cap}(A) := \inf_{x>0} \frac{\prod_{j=1}^{n} (Ax)_j}{\prod_{j=1}^{n} x_j}.$$

Then, instead of the permanent, one can use the capacity as progress measure in Algorithm 1. (And will have the advantage, as we shall see, to generalize to the operator scaling question.)

Remark 1.9. The reason for the previous remark is that permanent and capacity have the same *multiplicative property*: if one obtains B by multiplying a row or a column of a matrix A by any positive scalar λ, then $\frac{\operatorname{perm}(B)}{\operatorname{perm}(A)} = \lambda = \frac{\operatorname{cap}(B)}{\operatorname{cap}(A)}$.

Exercise 1.10. Prove the fact above about the multiplicative property property of permanent and capacity.

1.3 The non-commutative case, NC-SING, and Operator Scaling ([Gur04, GGOW16])

In this section we will introduce the operator scaling problem and an algorithm for it, both as (quantum) generalizations of the matrix scaling problem and the algorithm from the previous section. Note that until now we viewed L (or a tuple of matrices) as a symbolic matrix. We will now view it as an *operator*. Throughout these lectures we will see other ways in which to interpret it in various areas, and these viewpoints of the same object are essential to the theory presented here. (In the rest of this chapter, we take $\mathbb{F} = \mathbb{C}$.) The following table provides a parallel between the concepts used matrix scaling and the operator scaling settings.

Matrix scaling	**Operator scaling**
L_1	L_2
vectors	matrices
positive	PSD
A non-negative matrix	$L = (A_1, \ldots, A_m)$, m-tuple of matrices
R, C diagonal	$R, C \in \mathrm{GL}_n(\mathbb{F})$
$RAC = B$ doubly stochastic	$B_i = RA_iC$ doubly stochastic (see below)

Definition 1.11 (CPM and Double Stochasticity)**.** $L = (A_1, \ldots, A_m)$ defines a map by setting

$$L(P) := \sum_i A_i P A_i^\dagger$$

for $P \in \mathcal{M}_n(\mathbb{C})$. This map is called a *completely positive map (CPM)*, and one of its properties is that $P \succcurlyeq 0$ implies $L(P) \succcurlyeq 0$. (Note that we also have the dual operator $L^\dagger(P) := \sum_i A_i^\dagger P A_i$, analogous to matrix transpose.) Then, L is said to be *(exactly) doubly stochastic* if $L(I) = I$ and $L^\dagger(I) = I$.

Remark 1.12. One can generalize the above to ask that $L(P) = Q$ and $L^\dagger(I) = I$, for some prescribed PSD matrices P, Q. This is known as the *Schrödinger bridge* problem, analogous to the (r, c)-scaling problem in classical matrix scaling, which generalizes $(\mathbb{1}, \mathbb{1})$, or doubly stochastic scaling.

Definition 1.13 (Approximate Double Stochasticity)**.** Let $L = (A_1, \ldots, A_m)$ be a CPM, and define

$$R(L) := \sum_i A_i A_i^\dagger = L(I)$$
$$C(L) := \sum_i A_i^\dagger A_i = L^\dagger(I)$$

L is *approximately doubly stochastic scalable* if, for every $\varepsilon \in (0,1)$ there exists $R_\varepsilon, C_\varepsilon$ such that, for $L_\varepsilon := R_\varepsilon L C_\varepsilon$, we have $\|R(L_\varepsilon) - I\|_2 \le \varepsilon$ and $\|C(L_\varepsilon) - I\|_2 \le \varepsilon$.

OPERATOR SCALING: Given a tuple of matrices $L = (A_1, \ldots, A_m)$, determine whether L is *approximately doubly stochastic scalable*; that is, if for every $\varepsilon \in (0,1)$ there exists $R_\varepsilon, C_\varepsilon$ such that, for $L_\varepsilon := R_\varepsilon L C_\varepsilon$, we have $\|R(L_\varepsilon) - I\|_2 \le \varepsilon$ and $\|C(L_\varepsilon) - I\|_2 \le \varepsilon$.

We get the following immediate generalization of the matrix scaling algorithm, alternatingly normalizing the operator or its dual. Note that taking square roots is possible and efficient as these are PSD matrices.

Algorithm 2 Algorithm G

```
Require: $L = (A_1, \ldots, A_m)$ a CPM
 1: for $t$ steps do
 2:     $L \leftarrow R(L)^{-1/2} \cdot L$
 3:     if $\|C(L) - I\|_2 \le \frac{1}{\sqrt{n}}$ then
 4:         return yes                                  ▷ $\mathrm{cap}(L) > 0$
 5:     end if
 6:     $L \leftarrow L \cdot C(L)^{-1/2}$
 7:     if $\|R(L) - I\|_2 \le \frac{1}{\sqrt{n}}$ then
 8:         return yes                                  ▷ $\mathrm{cap}(L) > 0$
 9:     end if
10: end for
11: return no                                          ▷ $\mathrm{cap}(L) = 0$
```

In order to analyze the convergence properties of this algorithm, we will need a notion of progress measure. (Recall that in the classical case, this was the permanent, or equivalently the capacity of the intermediate matrices.) This leads to defining the quantum analogue of the capacity:

Definition 1.14 (Capacity of a CPM)**.** Let $L = (A_1, \ldots, A_m)$ be a CPM. Then, the *capacity* of L is defined as

$$\mathrm{cap}(L) := \inf_{P \succ 0} \frac{\det L(P)}{\det P}.$$

Note that, in this case, we have $\frac{\operatorname{cap}(RLC)}{\operatorname{cap}(L)} = \det(R)^2 \det(C)^2$.

Remark 1.15. Computing capacity is *not* a convex optimization problem; however, [GGOW16] gave an FPTAS for computing it.

Theorem 1.16 ([Gur04])**.** *Algorithm 2 converges on input L iff* $\operatorname{cap}(L) > 0$.

Outline. As in the classical case, we have three main steps to the proof. Denote the intermediate operators produced by the algorithm execution by $L^{(0)} = L, \ldots, L^{(t)}, \ldots$.

1. $\operatorname{cap}(L^{(t)}) \le 1$ for all $t \ge 1$;
2. $\operatorname{cap}(L^{(1)}) > 0$;
3. For $t \ge 1$, $\frac{\operatorname{cap}(L^{(t+1)})}{\operatorname{cap}(L^{(t)})} \ge \exp(1/6n)$.

□

Exercise 1.17. Show that if $\operatorname{cap}(L) = 0$ then Algorithm 2 does not converge.

Exercise 1.18. Show item 2 using a proof similar to that of the matrix scaling case (Exercise 1.7).

Theorem 1.19 ([GGOW16])**.** *Algorithm 2 converges on input L* in polynomial time *iff* $\operatorname{cap}(L) > 0$.

Outline. As in the classical case, we have three main steps to the proof. The proof is similar as that of the previous theorem, but the key is to improve item 2 to show that $\operatorname{cap}(L^{(1)}) \ge n^{-O(n)}$ (assuming L has low bit-complexity). □

We note that such a bound on $\operatorname{cap}(L^{(1)})$ was proved in [Gur04] for the special case that L (as a space of matrices) contains a non-singular matrix (in other words, $L \not\subseteq$ SING). The main hurdle will be generalizing it to all L for which $\operatorname{cap}(L) > 0$, which will turn out equivalent to $L \not\subseteq$ NC-SING.

This key to the analysis will require important ideas and results from both non-commutative algebra (the theory of free skew fields) and from commutative invariant theory (mostly of quiver representations), which we will discuss in the subsequent lectures.

1.4 Exercises

Problem 1.20 (Hall's theorem)**.** Prove Hall's bipartite matching theorem. It says that an $n \times n$ bipartite graph has a perfect matching iff for every subset S of vertices on the left side, $|N(S)| \ge |S|$, where $N(S)$ denotes the neighbours of S.

Problem 1.21 (Doubly stochastic matrices)**.** Prove that the polytope of doubly stochastic matrices is the convex hull of permutation matrices.

Problem 1.22. Prove that $\operatorname{cap}(A) > 0$ iff the bipartite graph defined by support of A has a perfect matching.

Problem 1.23 (Capacity of doubly stochastic matrices)**.** Prove that the capacity of a doubly stochastic matrix is 1.

Chapter 2

Algebraic Complexity and Invariant Theory

Scribed by: Pritish Kamath (pritish@mit.edu)

Abstract

I will start with background in algebraic complexity motivating commutative and non-commutative PIT. I will introduce the commutative and non-commutative rank of symbolic matrices, and their relation. Some characterization of non-commutative rank will in particular clarify why the operator scaling algorithm efficiently solves a large family of systems of quadratic equations. I will then survey basic definitions and problems in Invariant Theory, including the nullcone, invariant rings and their degrees, and then prove degree bounds on the group action we care about. Then we shall see how to use this information to analyze the operator scaling algorithm.

2.1 Brief Recap & Outline of this lecture

Gurvits [Gur04] showed that Algorithm 2 converges iff $\mathrm{cap}(L) \neq 0$ (see Theorem 1.16), where $\mathrm{cap}(L)$ was defined as,

$$\mathrm{cap}(L) = \inf_{P \succ 0} \frac{\det(L(P))}{\det(P)}$$

To prove polynomial time convergence of Algorithm 2, we need to prove the following theorem.

Theorem 2.1. *For any $L = (A_1, \ldots, A_m) \in (\mathscr{M}_n(\mathbb{C}))^m$, if $\mathrm{cap}(L) > 0$, then $\mathrm{cap}(L) > \exp(-n^{O(1)})$.*

Remark 2.2. In the above theorem, we implicitly assume that L has "low" bit complexity ($\mathrm{poly}(n)$). Also we will only apply the above theorem for normalized L i.e. $C(L) = I$ (or $R(L) = I$). So we will assume that $C(L) = I$ (or $R(L) = I$) holds for any L that we consider throughout this lecture.

Indeed, proving Theorem 2.1 is the main result of [GGOW16], and will be the main focus of this lecture. To prove this, we will take several detours into non-commutative algebra and commutative invariant theory.[1]

We mention a basic fact due to Gurvits, regarding $\mathrm{cap}(L)$[2].

[1] As it turns out, invariant theory is not needed to prove Theorem 2.1, as it is shown in the most recent version of [GGOW16].

[2] It will be extremely useful in the application to Brascamp-Lieb ineuqalities at the end of the lectures, yielding an alternative proof of their feasibility criteria.

Fact 2.3. $\text{cap}(L) = 0$ *iff L is rank decreasing, where, we say that L is rank decreasing if there exists some $P \succcurlyeq 0$, such that* $\text{rank}(L(P)) < \text{rank}(P)$.

Exercise 2.4. Prove Fact 2.3. [*Hint:* See Problem 2.37.]

We will see later that L is rank decreasing iff $L \in \text{NC-SING}$. As mentioned before, we want to show that if $\text{cap}(L) > 0$, then Algorithm 2 converges in polynomially many steps. Gurvits established this result for the specific (yet important) case where L is *commutatively singular* (recall definition below).

Definition 2.5. Let $L(\mathbf{x}) = \sum_{i=1}^{m} A_i x_i$. We say that $L \in \text{SING}$ if $\det(L(\mathbf{x})) \equiv 0$ (over $\mathbb{F}[\mathbf{x}]$), i.e. if L is not invertible over $\mathbb{F}(\mathbf{x})$.

Theorem 2.6 ([Gur04])**.** *If L is* commutatively non-singular *(i.e. $L \notin \text{SING}$), then* $\text{cap}(L) > n^{-O(n)}$, *which implies convergence of Algorithm 2 in $t < n^{O(1)}$ steps.*

Exercise 2.7. Prove Theorem 2.6. [*Hint*: See [Gur04, GGOW16].]

The above theorem will actually be very crucial to proving the more general Theorem 2.1.

A useful observation is that if L is doubly stochastic then $\text{cap}(L) = 1$. Moreover, this is numerically robust, in the sense that $\text{cap}(L)$ is close to 1 iff L is close to being doubly stochastic.

Also, recall that $\frac{\text{cap}(RLC)}{\text{cap}(L)} = \det(R)^2 \cdot \det(C)^2$ (as seen in Lecture 1). Thus if we keep track of the matrices we are pre and post-multiplying with in Algorithm 2, then we can recover $\text{cap}(L)$ to arbitrary accuracy.

The main results of [GGOW16] can be summarized as:

- NC-SING is decidable in polynomial time, i.e. $\text{NC-SING} \in \text{P}$
- There is an FPTAS for computing $\text{cap}(L)$.
- Algorithm 2 is "continuous" in its input, and can be used to provide quantitative estimates for continuity of capacity.

The last two items will be useful in the very end of the lectures for the application Brascamp-Lieb inequalities.

As mentioned earlier, to prove Theorem 2.1, we are going to take detours into non-commutative algebra and commutative invariant theory.

2.2 Non-commutative Algebraic Circuits

An algebraic circuit C is defined as a directed acyclic graph (DAG), with

- input nodes labeled by variables $x_1, \ldots, x_n$ or field elements
- all other nodes labeled by $(+)$ or $(\times)$.

An algebraic formula is a circuit, where the DAG is in fact a tree. The size of a circuit/formula is the number of nodes (alternatively, the number of edges, but this is at most quadratic in the number of nodes).

We interpret the circuit C as computing a polynomial in the natural inductive way, where a $(+)$ gate with inputs computing f and g computes the polynomial $f+g$, and a $(\times)$ gate with inputs computing f and g computes the polynomial $f \cdot g$.

An algebraic circuit can be interpreted in two ways: (i) variables $\{x_i\}$ commute, i.e. x_1x_2 and x_2x_1 are the same monomial and (ii) variables $\{x_i\}$ don't commute, i.e. x_1x_2 and x_2x_1 are different monomials.

While proving super-polynomial lower bounds for commutative circuits or even formulas is open, Nisan [Nis91] proved $\exp(n)$ lower bounds for non-commutative formulas (more strongly, non-commutative algebraic branching program) for an explicit polynomial (in particular, also for the permanent). While we have had this progress for 26 years, we still don't have super polynomial lower bounds for non-commutative circuits.

Hrubeš and Wigderson [HW15] considered non-commutative circuits with division $(\div)$ gates. While this makes proving lower bounds only harder, the hope was that such circuits might add more structure and can give new insights. In the commutative world, note that when we allow $(\div)$, each gate computes a rational function instead of a polynomial. Here, we require that the circuit never divides by the zero polynomial. Strassen [Str73] showed that over infinite fields, $(\div)$ gates do not add power to the model of circuits/formulas. That is, any circuit (or formula) with $(\div)$ gates, can be converted, with only a polynomial blowup, to another circuit (resp. formula) without $(\div)$ gates computing the same polynomial. This result was extended to finite fields as well [HY11]. However, the situation is quite tricky in the non-commutative world! Indeed, one of the main questions studied in [HW15] is:

Question: Can division be efficiently eliminated in non-commutative circuits and formulas?

To understand this, we need to know about Skew-fields and Invariant Theory, which we will get to shortly. It is known since recently in the work of Derksen-Makam [DM15], that $(\div)$ gates can be eliminated in non-commutative formulas. It is still open if it can be eliminated in non-commutative circuits.

Before we get into the theory of skew-fields, etc., we provide some context for the questions we are going to discuss.

2.2.1 Word Problems

Word problem is a general type of problem , pervasive in many fields. In many settings, we have an interesting class of objects where elements may not have a unique/canonical description. In particular, there is an equivalence relation over the space of possible descriptions Ω. And the basic word problem is to decide if two a priori distinct descriptions actually refer to the same object.

WORD PROBLEM: Consider a universe Ω, endowed with an equivalence relation $\cong$. Given $x, y \in \Omega$, is it the case that $x \cong y$?

This abstract problem captures many well known problems. For example, word problem for groups is an example of a word problem. This has been important since the early days of computability theory. In particular, it is known that given a description of a group, and an element of the group, it is undecidable to check whether the element is equal to identity.

Polynomial Identity Testing (PIT) also falls under this category of word problems, and so does Non-commutative PIT and non-commutative singularity.

2.2.2 Completeness of Determinant [Val79]

Before we go into the theory of skew-fields, etc. we describe Valiant's fundamental result of completeness of the determinant for formulas.

Theorem 2.8 ([Val79])**.** *If $F(\mathbf{x})$ is an arithmetic formula with $|F| = s$, then there exists $L_F(\mathbf{x})$ (symbolic matrix), such that, $F = \det(L_F)$. Moreover, L_F is $2s \times 2s$.*

In other words, computing a formula reduces to computing a determinant.

"Determinants pervade all branches of mathematics. Textbooks are filled with determinants. It is certainly the most popular polynomial in the history of mankind. Why? This is why!"

Proof Sketch of Theorem 2.8. The proof is obviously by induction. A formula has an inherent inductive structure. There are three cases:

- F is constant or a variable.
- $F = G \times H$.
- $F = G + H$.

The first two cases are easy, but addition is tricky. So we strengthen our inductive hypothesis to say that L_F has a special structure as follows,

$$\begin{bmatrix} * & * & * & * \\ 1 & * & * & * \\ 0 & 1 & * & * \\ 0 & 0 & 1 & * \end{bmatrix}$$

That is, L_F has 1's below the diagonal and all 0's below it.

- F is constant c or variable x_i : Easy!

$$\begin{bmatrix} 1 & 0 \\ 1 & c \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 \\ 1 & x_i \end{bmatrix}$$

- $F = G \times H$, then we let,

$$L_F = \begin{bmatrix} * & * & * & 0 & 0 & 0 \\ 1 & * & * & 0 & 0 & 0 \\ 0 & 1 & * & 0 & 0 & 0 \\ 0 & 0 & 1 & * & * & * \\ 0 & 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 1 & * \end{bmatrix} \quad \text{where,} \quad L_G = \begin{bmatrix} * & * & * \\ 1 & * & * \\ 0 & 1 & * \end{bmatrix} \text{ and } L_H = \begin{bmatrix} * & * & * \\ 1 & * & * \\ 0 & 1 & * \end{bmatrix}$$

- $F = G + H$,

$$L_F = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & * & * & * & 0 & 0 & 0 \\ 0 & 1 & * & * & 0 & 0 & 0 \\ 0 & 0 & 1 & * & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * \end{bmatrix} \quad \text{where,} \quad L_G = \begin{bmatrix} * & * & * \\ 1 & * & * \\ 0 & 1 & * \end{bmatrix} \quad \text{and} \quad L_H = \begin{bmatrix} * & * & * \\ 1 & * & * \\ 0 & 1 & * \end{bmatrix}$$

If we expand by the first column, and we take the first 1, then we are forced to take all the 1's below the diagonal in the L_G matrix and also the first 1 in the first row, and this gives us $\det(L_H)$ from the remaining rows/columns. Similarly, if we start with the second 1 in the first column, then we get $\det(L_G)$.

But this is not in the inductive form we wanted. However, that's easy to fix. We swap the top most row and the first row of L_H, to get,

$$L_F = \begin{bmatrix} 1 & 0 & 0 & 0 & * & * & * \\ 1 & * & * & * & 0 & 0 & 0 \\ 0 & 1 & * & * & 0 & 0 & 0 \\ 0 & 0 & 1 & * & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 0 & 1 & * \end{bmatrix}$$

where, we can multiply the first row by -1 if needed to fix the overall sign.

While this proof seems like magic, there is a systematic way to come up with this proof.
Hint: Algebraic Branching Programs! □

Thus, as a consequence of Theorem 2.8, the PIT question for commutative formula F simply reduces to the problem of determining membership of the symbolic matrix L_F in SING.

Exercise 2.9. Construct a poly(n) size commutative circuit (not formula) for computing the determinant of an $n \times n$ matrix of variables (but with only with $(+)$ and $(\times)$.) [*Hint:* Berkowitz algorithm, following "Samuelson's law"]

2.2.3 Free-Skew fields

In the commutative case, we obtain $\mathbb{F}(\mathbf{x})$, the field of rational functions, by simply adding the inverses to $\mathbb{F}[\mathbf{x}]$, the polynomial ring. Note that every element in $\mathbb{F}(\mathbf{x})$ is of the form $p \cdot q^{-1}$ where $p, q \in \mathbb{F}[\mathbf{x}]$.

The non-commutative case is quite tricky! Usually, for any algebraic object, there is a canonical description (even leaving aside complexity issues). For example, elements of $\mathbb{F}[\mathbf{x}]$ have a canonical expression as vector of coefficients, or elements of $\mathbb{F}(\mathbf{x})$ can be canonically written as $p \cdot q^{-1}$ in the most reduced form.

$\mathbb{F}\langle x\rangle$, the space of non-commutative polynomials, does have a canonical representation as a vector of coefficients. However, when we try to extend it to the free skew field, there seems to be no other way to describe these objects than through computation. That is, we start with constants of $\mathbb{F}$ and the variables $\mathbf{x}$ and produce new elements by $(+)$, $(\times)$ and $(\cdot)^{-1}$ operations. The free skew field $\mathbb{F}\langle\!\langle \mathbf{x}\rangle\!\rangle$ was thus defined[3] this way by Amitsur [Ami66].

Definition 2.10. $\mathbb{F}\langle\!\langle \mathbf{x}\rangle\!\rangle = \left\{ \text{formula with } (+), (\times), (\cdot)^{-1} \text{ over constants in } \mathbb{F} \text{ and variables in } \mathbf{x} \right\}$

We run into complications very quickly when dealing with $\mathbb{F}\langle\!\langle \mathbf{x}\rangle\!\rangle$. For instance, $x^{-1}+y^{-1}$ cannot be expressed as $p\cdot q^{-1}$ or $p^{-1}\cdot q$ for any $p,q \in \mathbb{F}\langle x,y\rangle$. Things get even more complicated. For example, the formula $(x+yz^{-1}w)^{-1}$ has a *nested inversion*. In the commutative world, we can eliminate nested inversions and have only one inversion at the top. However, the formula $(x+yz^{-1}w)^{-1}$ has an *inversion height* 2.

Definition 2.11 (Inversion Height)**.** Given an element $P \in \mathbb{F}\langle\!\langle \mathbf{x}\rangle\!\rangle$, we define the *inversion height of* P as the smallest depth of inversions needed in any non-commutative formula that is "equivalent" to P.

Remark 2.12. In the above definition, "equivalent" needs to be carefully defined. For now, we say that two formulas P and Q are equivalent (i.e., $P \equiv Q$) if we can obtain one from the other by using field axioms, such as, multiplying by 1, writing 1 as $x\cdot x^{-1}$, etc.

Actually, the notion of "equivalence" is defined differently (see Definition 2.15), and it is a theorem of Cohn and Reutenauer [CR99] that these two notions of "equivalence" are in fact equivalent!

Exercise 2.13. Prove that $(x+yz^{-1}w)^{-1}$ has inversion height of 2.

More strongly, Reutenauer [Reu96] showed that for all n, there exists a formula F_n with inversion-height $(F_n) \geq n$.[4] Determining inversion height of a rational function can in fact be tricky.[5] For instance, consider Hua's identity, which says that $(x+xy^{-1}x)^{-1} \equiv x^{-1}-(x+y)^{-1}$. The expression on the left has two nested inversions, but it has inversion-height of 1, as is evidenced by the formula representation on the right.

We will see that a consequence of the main result of [GGOW16] is a polynomial time algorithm for rational identity testing (RIT) of non-commutative formulas. We don't know this for commutative formulas!

Theorem 2.14 (RIT for free skew field)**.** *Given formulas $p,q \in \mathbb{F}\langle\!\langle \mathbf{x}\rangle\!\rangle$, it is possible to determine, in polynomial time, whether $p(\mathbf{x}) \equiv q(\mathbf{x})$ or not (i.e. polynomial in the size of formula size of p and q).*

[3]The definition given here is actually not accurate. A proper definition of the free skew field is given as follows: the free skew field is the set of equivalence classes of non-commutative (valid) formulas with inversion gates, where the equivalence relation is given by the one stated in Remark 2.12 or equivalently Definition 2.15.

[4]This result is morally same as and inspired by Kleene's result that there exist regular languages with arbitrarily large star-depth. Indeed, $(+)$, $(\times)$ and $(\cdot)^{-1}$ operations in the formula world correspond to concatenation, union and star operations in the regular expression world.

[5]The problem of computing the inversion height of a rational function is known to be decidable, and it is an open problem to give efficient algorithms to compute the inversion height.

We now turn to formally defining the equivalence between two non-commutative formulas. Intuitively, the definition should be that the two formulas have the same evaluations on all inputs coming from any ring. However, the following definition says that two formulas are equivalent if they agree when we replace the inputs by matrices with arbitrarily large dimension. This is because zero/non-zero behaviour on matrices of arbitrarily large dimension captures zero/non-zero behaviour on all division rings [Ami66].

Definition 2.15 (Equivalence of non-commutative formulas [Ami66])**.** Given $R \in \mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$, we define $R \equiv 0$ if for all $d \in \mathbb{Z}_{>0}$ and $D = (D_1, \ldots, D_m) \in \mathcal{M}_d(\mathbb{F})^m$, it holds that $R(D) = 0$ (whenever $R(D)$ is defined). More generally, given $P, Q \in \mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$, we say that $P \equiv Q$ iff $(P - Q) \equiv 0$.

More strongly, it was shown by Amitsur [Ami66] that we could replace the condition $R(D) = 0$ by a seemingly much weaker condition, and still have the same notion of equivalence. Namely,

Theorem 2.16 ([Ami66])**.** *$R(\mathbf{x}) \equiv 0$ iff for all $d \in \mathbb{Z}_{>0}$ and for all $D_1, D_2, \ldots, D_m \in \mathcal{M}_d(\mathbb{F})$, it holds that $\det(R(D)) = 0$ (whenever $R(D)$ is defined).*

A natural question, given Valiant's Theorem 2.8, is whether there is a canonical problem that one can have, like the symbolic determinant identity testing in the commutative case, which captures the word problem for the free skew field. And that is what Cohn [Coh71] did, which is a generalization of Valiant's theorem and even predates it! (It also works in the commutative setting.)

Theorem 2.17 ([Coh71])**.** *If $F(\mathbf{x}) \in \mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$ is a non-commutative formula with $|F| = s$, then there exists $L_F(\mathbf{x})$ (symbolic matrix over non-commutative variables), such that, $F \equiv (L^{-1})_{1,1}$. Moreover, L_F is $3s \times 3s$. (Note $(L^{-1})_{ij} \in \mathbb{F}\langle\!\langle \mathbf{x} \rangle\!\rangle$.)*

Indeed, the proof is also by induction, although slightly harder as we have to deal with inversion. But there are several proofs of this theorem by now. One of course is by Cohn [Coh71]. There are other proofs in [Mal78] and [HW15].

Exercise 2.18. Construct a poly(n) size non-commutative circuit for computing inverse of an $n \times n$ matrix of variables. [*Hint:* Solve the 2×2 matrix case and then recurse. Or see [HW15].]

From Theorem 2.16, we can obtain the following important corollary.

Corollary 2.19. *$L = (A_1, \ldots, A_m) \in \text{NC-SING}$ iff for all $d \in \mathbb{Z}_{>0}$ and for all $D_1, D_2, \ldots, D_m \in \mathcal{M}_d(\mathbb{F})$, it holds that $\det(\sum A_i \otimes D_i) = 0$.*

Exercise 2.20. Prove Corollary 2.19. [*Hint:* In one direction, use $L \in \text{NC-SING}$ implies that there is a factorization over the free skew field. In other direction, use $L \notin \text{NC-SING}$ implies that L has an inverse over the free skew field (along with Theorem 2.16).]

Note that a corresponding statement holds true in the commutative case, as in the following remark (again, we do need that $\mathbb{F}$ is large enough).

Remark 2.21. $L = (A_1, \ldots, A_m) \in \text{SING}$ iff $d \in \mathbb{Z}_{>0}$ and for all $\delta_1, \delta_2, \ldots, \delta_m \in \mathbb{F}$, it holds that $\det(\sum A_i \cdot \delta_i) = 0$.

An important question here is whether we can impose a bound on d in Corollary 2.19. In particular, is there an explicit $d^*(n)$, such that, we could replace the "for all $d \in \mathbb{Z}_{>0}$ by $d \leq d^*(n)$? What kind of bounds can we prove on $d^*(n)$?

Interestingly, in [HW15] it is shown that $(\div)$ gates can be eliminated from non-commutative formulas with a quasi-polynomial blowup if $d^*(n) = \mathrm{poly}(n)$.

Note that the question of "$L \in \text{NC-SING}$?" doesn't change when we pre-multiply or post-multiply all the A_i's in L by non-singular matrices. This is a group action, and should immediately remind us of invariant theory.

2.3 Invariant Theory

"Invariant theory studies symmetries of group actions. All modern physics relies on Invariant theory, because all forces of nature are group actions."

2.3.1 Historical detour

Invariant Theory is attributed to Cayley's 1845 paper "On the Theory of Linear Transformations." We illustrate the ideas of Invariant theory from problems in Euclidean geometry.

Given two (not necessarily convex) polygons S and T in $\mathbb{R}^2$, when can we cut S into pieces and rearrange it to get T? It turns out that this can be done iff $\mathrm{area}(S) = \mathrm{area}(T)$. Clearly, this is a necessary condition, but it also turns out to be sufficient!

Exercise 2.22. Prove that it is possible to "transform" a polygon S in $\mathbb{R}^2$ into another polygon T in $\mathbb{R}^2$ iff $\mathrm{area}(S) = \mathrm{area}(T)$. Here, the operations allowed in "transform" are cutting the polygon using straight cuts and rearrangement of the pieces.

The main question in Invariant Theory in genereal is to characterize all the invariants.

What about polytopes in $\mathbb{R}^3$? Of course, volume is an invariant. But are there any other invariants? Indeed, this was the 3rd question in Hilbert's list of 23 questions. This was the first of Hilbert's questions that was answered. It was answered by his student Max Dehn, who introduced what is now known as Dehn's invariant! This invariant is related to angles in the polytope, sizes of edges, etc. In particular, it follows from this invariant that we cannot transform a cube in to a regular tetrahedron of the same volume.

2.3.2 Invariants of Linear Group Actions

While the definitions make sense for any group G and over any field $\mathbb{F}$, the theory is nicest when $\mathbb{F}$ is algebraically closed of characteristic zero (e.g. $\mathbb{F} = \mathbb{C}$) and the group G is reductive and algebraic (e.g. finite groups, $SL_n(\mathbb{C})$, $\mathrm{GL}_n(\mathbb{C})$, etc.). We will implicitly assume this throughout this lecture.

Let G be a group that acts on Ω, denoted by $G \curvearrowright \Omega$. The action is specified by a function $A \colon G \times \Omega \to \Omega$. That is, $g\omega \in \Omega$ for all $g \in G$ and $\omega \in \Omega$. Group action must satisfy all the natural properties of the group. For example, it must be associative, that is, $(gh)(\omega) = g(h(\omega))$.

We are going to consider linear group actions on linear spaces. If $\Omega = \mathbb{F}^k$, then all "linear" group actions are simply matrices. A linear group $G \subseteq \mathrm{GL}_k(\mathbb{F})$ acts by sending points in $\mathbb{F}^k$ to points in $\mathbb{F}^k$ using invertible linear transformations. We can also extend this group action to act on the polynomial ring $\mathbb{F}[z_1, \ldots, z_k]$. That is, we apply the linear transformation on the symbolic vector of variables $(z_1, \ldots, z_k)$.

We are going to look for polynomial invariants of a group action. Note that, not all invariants are polynomial; namely, if p is an invariant, then so is e^p or $\log p$. While for actions of finite groups, all invariants are captured by polynomial invariants (i.e. orbits of two elements intersect iff they have the same value on all polynomial invariants), this is not necessarily true for actions of infinite groups because of the distinction between orbit-closures and orbits. More on this later.

Definition 2.23. Let $V = \mathbb{F}^k$. The *invariant ring* (or the ring of invariant polynomials) is defined as

$$\mathbb{F}[V]^G := \left\{ p \in \mathbb{F}[\mathbf{z}] \, : \, gp = p \, \forall g \in G \right\}$$

We now give some examples illustrating the definition above of invariant rings.

Example 1. $V = \mathbb{F}^k$ and $G = S_k$ is the permutation group on $[k]$. In this case, $\mathbb{F}[V]^G$ is the set of all symmetric polynomials, which are generated by the set of elementary symmetric polynomials, where $\mathsf{ESYM}_k^r(\mathbf{z}) := \sum_{\substack{S \subseteq [k] \\ |S| = r}} \prod_{i \in S} z_i$.

$$\mathbb{F}[V]^G = \{\text{symm polys}\} = \left\langle \mathsf{ESYM}_k^r \right\rangle, 1 \leq r \leq k$$

Note that, here $\langle g_1, \ldots, g_m \rangle$ is the ring generated by $g_1, \ldots, g_m$. In other words, it is the set of polynomials $\left\{ P(g_1, \ldots, g_m) \, : \, P \in \mathbb{F}[y_1, \ldots, y_m] \right\}$.

Example 2. $V = \mathbb{F}^k$ where $k = n^2$ and we interpret the variables as forming an $n \times n$ matrix variable Z. The group $G = SL_n(\mathbb{F})^2$ acts on $V = M_n(\mathbb{F})$ by pre and post multiplication. That is for any $g = (R, C) \in G$, the action of g on $Z \in V$ is given by $gZ = R \cdot Z \cdot C$.

In this case, $\mathbb{F}[V]^G = \langle \det(Z) \rangle$. In other words, the determinant polynomial generates all the invariants!

Example 3. $V = \mathbb{F}^k$ where $k = mn^2$, and we interpret the variables as an m-tuple of $n \times n$ matrix variables $Z = (Z_1, \ldots, Z_m)$. The group $G = \mathrm{SL}_n(\mathbb{F})^2$ acts on V by simultaneous pre and post multiplication. That is, for any $g = (R, C) \in G$, the action of g on element $(Z_1, \ldots, Z_m)$ as, $gZ = (R \cdot Z_1 \cdot C, \ldots, R \cdot Z_m \cdot C)$.

This was open for a long time until 2000, when several groups [DW00, DZ01, SdB01] independently discovered the set of generators of the invariant ring. We summarize this result in the theorem below.

Theorem 2.24. *Let $V = \mathbb{F}^k$ where $k = mn^2$, and we interpret the variables as an m-tuple of $n \times n$ matrix variables $Z = (Z_1, \ldots, Z_m)$. The group action of $G = SL_n(\mathbb{F})^2$ on V is given by $gZ = (R \cdot Z_1 \cdot C, \ldots, R \cdot Z_m \cdot C)$ for $g = (R, C)$. Then,*

$$V^G = \left\langle \det\left(\sum Z_i \otimes D_i \right) \, : \, \forall d, \, \forall D_1, \ldots, D_m \in M_d(\mathbb{F}) \right\rangle$$

Unlike Examples 1 and 2, the invariant ring in Example 3 (which is relevant for our application) is specified by an infinite set of polynomials. We can ask if the invariant ring is finitely generated. In fact, Hilbert [Hil93] showed that V^G is always finitely generated for actions of reductive groups (we won't define this but as mentioned before finite groups, $\mathrm{GL}_n(\mathbb{C})$, $SL_n(\mathbb{C})$ are examples of reductive groups). Indeed, this is the same paper which proves the Nullstellensatz, but this application to Invariant Theory was really what Hilbert was after!

It was shown much later by Popov [Pop82], that in the nice setting (groups are reductive and algebraic, field is $\mathbb{C}$, etc.) in fact the invariant ring is generated by some polynomials with degree at most $\exp(\exp(n,k))$ (recall that n is the dimension of the group G and k is the dimension of vector space V). This was improved by Derksen [Der01] who showed that the invariant ring is generated by polynomials with degrees at most $\exp(\mathrm{poly}(n,k))$. In fact, for the specific action of pre- and post- multiplying by $\mathrm{SL}_n(\mathbb{F})$ matrices, as mentioned in Theorem 2.24, there is even a $\mathrm{poly}(n,k)$ upper bound on the degree of generators. We will see this in Lecture 3 (see Theorem 3.14, proven in [IQS15]). But for the purposes of proving Theorem 2.1, an $\exp(\mathrm{poly}(n,k))$ bound will suffice. We summarize Derksen's result in the following theorem.

Theorem 2.25 ([Hil93, Pop82, Der01])**.** *Let $V = \mathbb{F}^k$ be a vector space. Let $G \subseteq \mathrm{GL}_n(\mathbb{F})$ be a "nice" (as explained above) group action on V. Then, $\mathbb{F}[V]^G$ is generated by a set of polynomials of degree at most* $\exp(\mathrm{poly}(n,k))$.

As a corollary, we get a bound on the d^* for the specific action of simultaneous pre and post multiplication. The main difference between the following corollary and Theorem 2.24 is that we only need to consider $d \le d^*$.

Corollary 2.26. *For any m, n, there exists $d^* = d^*(m,n) \le \exp(\mathrm{poly}(n,m))$[6] s.t. the following holds:*

Let $V = \mathbb{F}^k$ where $k = mn^2$, and we interpret the variables as an m-tuple of $n \times n$ matrix variables $Z = (Z_1,\dots,Z_m)$. The group action of $G = SL_n(\mathbb{F})^2$ on V is given by $gZ = (R\cdot Z_1 \cdot C,\dots,R\cdot Z_m \cdot C)$ for $g = (R,C)$. Then,

$$\mathbb{F}[V]^G = \left\langle \det\left(\sum Z_i \otimes D_i\right) \;:\; \forall d \le d^*,\ \forall D_1,\dots,D_m \in M_d(\mathbb{F}) \right\rangle$$

Using this corollary, we will now move to proving Theorem 2.1, which implies polynomial time convergence of Algorithm 2! As an aside, note that this bound is not useful for application of division elimination in [HW15]. However, the much better bound of [DM15] actually shows that inversion can be eliminated for non-commutative formulas.

2.4 Proof of Capacity Lower Bound

We put together the theorems developed so far to prove Theorem 2.1.

Proof Sketch of Theorem 2.1. Let $L_n = L = (A_1,\dots,A_m)$ with $A_i \in M_n(\mathbb{F})$, with $C(L) = I$. We wish to show that if $\mathrm{cap}(L) > 0$ then in fact $\mathrm{cap}(L) > n^{-O(n^c)}$.

[6] as mentioned earlier, this bound has been improved to $d^* \le n+1$ in Theorem 3.14 proven in [IQS15].

For any $K_d = K = (D_1, \ldots, D_m)$ for $D_i \in M_d(\mathbb{F})$, define $L\tilde{\otimes}K := (A_1 \otimes D_1, \ldots, A_m \otimes D_m)$, where each $A_i \otimes D_j \in M_{nd}(\mathbb{F})$.

From Theorem 2.6, we know that if L is commutatively non-singular, then we do have the desired theorem. In other words, if there exist $\delta_1, \ldots, \delta_m \in \mathbb{F}$ such that $\det(\sum_i A_i \delta_i) \neq 0$ then $\text{cap}(L) > n^{-O(n^c)}$.

If L is non-commutatively non-singular, that is, $L \notin$ NC-SING, then from Corollary 2.26 and from Corollary 2.19 we know that for $d^* \leq \exp(\text{poly}(n, m))$ there exist $D_1, \ldots, D_m \in \mathcal{M}_{d^*}(\mathbb{F})$ such that, $\det(\sum_i A_i \otimes D_i) \neq 0$. We can replace the entries of D_i's by formal commutative variables, to get that the polynomial $\det(\sum_i A_i \otimes D_i)$ is not identically zero. Note that the this polynomial has degree at most $n \cdot d^* \leq \exp(\text{poly}(n, m))$. Thus, by the Schwartz-Zippel lemma, we have that there exists a choice of D_i's with integer entries no larger than $\text{poly}(n \cdot d^*)$.

From Theorem 2.6, we have that $\text{cap}(L\tilde{\otimes}K) > (nd^*)^{-nd^*}$. We wish to use this to lower bound $\text{cap}(L)$, towards which, we use the following lemma.

Lemma 2.27. $\text{cap}(L\tilde{\otimes}K) \leq \text{cap}(L)^{d^*} \cdot \text{cap}(K)^n$.
Alternately, if we define $\overline{\text{cap}}(L) = \text{cap}(L)^{1/n}$, *then,* $\overline{\text{cap}}(L\tilde{\otimes}K) \leq \overline{\text{cap}}(L) \cdot \overline{\text{cap}}(K)$.

Exercise 2.28. Prove Lemma 2.27. [*Hint:* Consider $P > 0$ that is a tensor product of a $n \times n$ matrix and a $d^* \times d^*$ matrix. Moreover, consider the $d \times d$ matrix to be identity.]

Using the above lemma, we have that,

$$(nd^*)^{-nd^*} \leq \text{cap}(L \otimes K) \leq \text{cap}(L)^{d^*} \cdot \text{cap}(K)^n \,. \tag{2.1}$$

But observe that

$$\text{cap}(K) \leq (nd^*)^{O(n)} \,. \tag{2.2}$$

This is because $\text{cap}(K) \leq \frac{\det(K(I))}{\det(I)} \leq \text{poly}(nd^*)^n$.

Thus, combining Equation 2.1 and Equation 2.2, we get

$$\text{cap}(L) \geq (nd^*)^{-O(n)} \geq \exp(-\text{poly}(n, m)) \,.$$

since $d^* \leq \exp(\text{poly}(n, m))$.

To summarize the proof sketch,

- We used ideas from non-commutative algebra to get Corollary 2.19, which was a criterion for non-commutative singularity.
- We used Invariant Theory to obtain upper bound on $d^*(n)$, i.e. Corollary 2.26.
- Finally, we combined this with Gurvits' solution to the commutatively non-singular case, i.e. Theorem 2.6.

With hindsight, in the way we have presented the proof in these lectures, it seems that we did not really have to go into non-commutative algebra at all, since formally Corollary 2.19 is also a consequence of Corollary 2.26. There are several explanations why this "detour" was needed. The obvious one is that this is how the proof was found. A better one is that Cohn's characterizations of non-commutative rank are still used in the proof, as well as Amitsur's rigidity theorem which is used in the bound on $d^*(n)$. A third reason is that the algorithm has an important corollary to non-commutative algebra, namely a polynomial time algorithm for the word problem over free skew fields. □

2.5 Exercises

Definition 2.29 (Hall Blocker)**.** Let R be a ring (not necessarily commutative), $M_n(R)$ be the space of $n \times n$ matrices with entries in R and $r \in \mathbb{N}$ be such that $0 \le r < n$. We say that a matrix $A \in M_n(R)$ has an *r-Hall blocker* (or simply a Hall blocker) if A has a zero submatrix of dimensions $i \times j$ s.t. $i + j \ge 2n - r$.

The weak algorithm (e.g. Chapter 2 in [Coh06]) is a generalization of the division algorithm for univariate polynomials to multivariate non-commutative polynomials. No such analogue exists for multivariate commutative polynomials.

Problem 2.30 (Weak algorithm)**.** Let $p, q_1, \dots, q_n \in \mathbb{F}\langle \mathbf{x} \rangle$ be polynomials in the non-commutative variables $\mathbf{x} = (x_1, \dots, x_m)$. p is said to be right degree-dependent on $q_1, \dots, q_n$ if $p = 0$ or there exist polynomials $r_1, \dots, r_n$ s.t.

$$\deg\left(p - \sum_{i=1}^{n} q_i r_i\right) < \deg(p)$$

and

$$\deg(q_i) + \deg(r_i) \le \deg(p)$$

for all i. Polynomials $q_1, \dots, q_n$ are said to be right degree-dependent if there exist polynomials $r_1, \dots, r_n$ s.t.

$$\deg\left(\sum_{i=1}^{n} q_i r_i\right) < \max_i \{\deg(q_i) + \deg(r_i)\}$$

Prove that if $q_1, \dots, q_n$ are right degree-dependent, then there exists one of them which is right degree-dependent on rest of them.

Problem 2.31 (Commutative vs. Non-commutative rank - lower bound)**.** The 3×3 (generic) skew-symmetric matrix gives a gap of $3/2$. Construct an example with gap $> 3/2$.

Recently, [DM16], following [LO15], constructed examples with gap approaching arbitrarily close to 2.

Problem 2.32 (Hua's identity)**.** Prove that for non-commuting variables x, y,

$$\left(x + xy^{-1}x\right)^{-1} + (x + y)^{-1} = x^{-1}$$

Problem 2.33 (Higman's trick)**.** Given a matrix $M \in M_n(\mathbb{F}[\mathbf{x}])$ (or $\in M_n(\mathbb{F}\langle \mathbf{x} \rangle)$), find a map that linearizes the matrix M, i.e. the entries of the new matrix should be affine forms, and preserves its co-crank (or co-ncrank). If the entries of M have small formula size, then the map shouldn't blow up the dimension of M by a lot.

Problem 2.34. One of Cohn's theorems [Coh95] says that for a linear matrix (whose entries are affine forms) $L \in M_n(\mathbb{F}\langle \mathbf{x} \rangle)$, $\text{ncrank}(L) \le r$ iff there exist $n \times r$ and $r \times n$ linear matrices s.t. $L = L_1 L_2$. Use this theorem to give an exponential time algorithm for computing $\text{ncrank}(L)$.

Problem 2.35 (Proof of Cohn's theorem)**.** In this exercise, we will prove Cohn's theorem[7] that $\text{ncrank}(L) \le r$ iff there exist invertible matrices $B, C \in M_n(\mathbb{F})$ s.t. BLC contains an r-Hall blocker. Our starting point will the non-trivial theorem[8] that $\text{ncrank}(L) \le r$ iff there exist matrices $K_1 \in M_{n\times r}(\mathbb{F}\langle \mathbf{x} \rangle)$ and $K_2 \in M_{r\times n}(\mathbb{F}\langle \mathbf{x} \rangle)$ (not necessarily linear) s.t. $L = K_1 K_2$.

1. Suppose $p_1, \dots, p_n$ and $q_1, \dots, q_n$ are non-commutative polynomials such that
$$\sum_{i=1}^{n} p_i q_i = f \tag{2.3}$$
Then there exists a matrix $E \in \text{GL}_n(\mathbb{F}\langle \mathbf{x} \rangle)$ (the entries of E^{-1} will also have polynomial entries) which "kills all cancellations" in equation (2.3). Formally, if
$$\begin{bmatrix} p'_1 & p'_2 & \dots & p'_n \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & \dots & p_n \end{bmatrix} E$$
and
$$\begin{bmatrix} q'_1 \\ q'_2 \\ \vdots \\ q'_n \end{bmatrix} = E^{-1} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}$$
then
$$\sum_{i=1}^{n} p'_i q'_i = f$$
and for each i, either one of p'_i or q'_i is 0, or $\deg(p'_i) + \deg(q'_i) \le \deg(f)$. (*Hint*: use the weak algorithm).
2. If there exist K_1, K_2 as above s.t. $L = K_1 K_2$, then prove that there exist linear L_1, L_2 with the same dimensions s.t. $L = L_1 L_2$. Furthermore, for every $1 \le i \le r$, either the ith column of L_1 has only elements from $\mathbb{F}$ or the ith row of L_2 does. Note that this does not mean that $L_1 \in M_{n\times r}(\mathbb{F})$ (think of a counterexample). (*Hint*: Reduce to the above lemma by introducing auxilliary non-commuting variables)
3. Use the above to construct a Hall blocker for L.

The following connection between non-commutative rank and tensor rank is due to [DM16].

Problem 2.36 (Non-commutative rank and tensor rank)**.** Let the linear matrix $L = \sum_{i=1}^{n} x_i A_i \in M_n(\mathbb{F}\langle \mathbf{x} \rangle)$ be s.t. $\text{crank}(L) = \alpha n$ and $\text{ncrank}(L) = n$. Suppose $B_1, \dots, B_n$ be matrices in $M_n(\mathbb{F})$ s.t. $\sum_{i=1}^{n} A_i \otimes B_i$ is full rank (they exist due to the recent bounds of [DM15, IQS15]). Let B be the $n \times n \times n$ tensor whose slices are $B_1, \dots, B_n$. Then prove that the tensor rank of B is at least n/α.

[7]Technically Cohn only proved it for $r = n - 1$ and the full theorem appeared in [FR04]

[8]Again due to Cohn

The same bound also applies for border rank. With the examples in [DM16], this gives the best known lower bounds for border rank. These examples were discovered by Landsberg-Ottaviani [LO15] to prove lower bounds on border rank of the matrix multiplication tensor (they don't talk about non-commutative rank though and this connection was discovered by [DM16]).

Given an $n \times n$ unitary matrix U, let $u_1, \dots, u_n$ denote its column vectors. Given a completely positive operator L, consider the non-negative matrix $A_{U,V}$ whose $(i, j)^{\text{th}}$ entry is given by $\operatorname{tr}\left[L(u_i u_i^\dagger) v_j v_j^\dagger\right]$. So this describes a reduction from operators to non-negative matrices after fixing a basis.

Problem 2.37. Use the above described reduction to prove that $\operatorname{cap}(L) > 0$ iff L is rank non-decreasing.

This is quite common in geometric invariant theory, fixing basis and compactness arguments reduce the non-commutative group actions to the setting of commutative group actions. However this reduction is highly non-algorithmic and finding the "right basis" is one of the main challenges in the algorithmic setting.

Problem 2.38. Prove that the problem of computing the rank of a linear matroid (or of the intersection of two linear matroids) can be reduced to the problem of determining whether a completely positive operator is rank non-decreasing.

Chapter 3

Invariant Theory (continued) and Brascamp Lieb Inequalities

Scribed by: Chin Ho Lee (chlee@ccs.neu.edu) & Aditya Potukuchi (ap1311@cs.rutgers.edu)

Abstract

Today, I will discuss invariant theory in more depth than in the previous lecture: I will describe some of the main objects studied in the theory of invariants (nullcone, orbits and orbit closures) and show how these objects relate to some areas of complexity theory (such as GCT (Geometric Complexity Theory) and the graph isomorphism problem). Then, I will give another way to upper bound the degree of the polynomials that generate the invariant ring of the action of $\mathrm{GL}_k(\mathbb{F})^2$ on symbolic matrices with non commuting variables, the action that we saw in the last lecture.

After discussing invariant theory, I will shift gears and discuss a recent application of Operator Scaling to analysis. I will introduce, with many examples, the Brascamp–Lieb inequalities, a broad family which includes numerous famous inequalities throughout analysis, probability, information theory and geometry. I will discuss their rich structural theory and their basic computational problems. I will then describe an efficient algorithm for proving them. While the algorithm is derived from the operator scaling one, I will give a self-contained description naturally suggested from their structure theory. It should become clear why this algorithm efficiently solves large family of linear programs of exponential size, and I will give some examples.

3.1 Reintroduction to Invariant Theory

Recall from previous lecture that we will implicitly assume that $\mathbb{F}$ is an algebraically closed field of characteristic zero (think of $\mathbb{F} = \mathbb{C}$) and G to be an algebraic and reductive group (think of finite groups, $\mathrm{GL}_n(\mathbb{C})$, $SL_n(\mathbb{C})$).

We were talking about the action of a group $G = \mathrm{GL}_n(\mathbb{F})$ on the vector space $V = \mathbb{F}^k$ for some field $\mathbb{F}$. This action naturally induces a group action of G on the ring of polynomials $\mathbb{F}[z_1, \ldots, z_k]$. Under such an action, one important object to study is the ring of the polynomials that are invariant under this action, more specifically

$$\mathbb{F}[V]^G := \left\{ P \in \mathbb{F}[z_1, \ldots, z_k] : \; gP = P \; \forall g \in G \right\}$$

Hilbert, in [Hil93] showed that $\mathbb{F}[V]^G$ is finitely generated. Derksen, in [Der01] showed that $\mathbb{F}[V]^G$ is generated by polynomials of degree at most $exp(\text{poly}(n, k))$. In fact, these are answers to the two fundamental problems when studying any group action in Invariant Theory.

Today, we will go back to our usual action of the group $G = \text{SL}_n(\mathbb{F})^2$ on an operator $(A_1 \ldots, A_m)$, where every $A_i \in \mathscr{M}_n(\mathbb{F})$. We will be able to prove much better bounds on the degree of the polynomials that generate $\mathbb{F}[V]^G$.

3.2 Some problems regarding orbits

3.2.1 Orbit Containment Problem

For a group G acting on a vector space V, and for a element $u \in V$, the *orbit* of v is defined as the set of all points that G maps it to. Formally,

Definition 3.1 (Orbit)**.** For a group G acting on a vector space V, and an element $v \in V$, the orbit of v , denoted by Gv is given by:

$$Gv := \left\{ gv : \; g \in G \right\}.$$

Let us look at the following simple to state problem which proves extremely general:

ORBIT CONTAINMENT: Given a group G which acts on a vector space V, and two elements $u, v \in V$ answer

"Does $v \in Gu$?"

Indeed, this is quite a general problem, for a simple example, take V to be the space of all $n \times n$ matrices (think of symmetric $0-1$ matrices as adjacency matrices of graphs), and take G to be S_n that acts on V by permuting the rows and columns of a matrix (induced by the permutation of vertices of the underlying graph). Then orbit containment for this action captures the graph isomorphism problem.

3.2.2 Orbit Closure Containment

Since we are working over $\mathbb{C}$, and have a topology it is much more natural to ask whether a point is contained not in the orbit, but in the *closure* of the orbit of another element. This is particularly natural in the context of *Geometric Invariant Theory*.

Definition 3.2 (Orbit Closure)**.** For a group G acting on a vector space V, and an element $v \in V$, the orbit closure of v , denoted by $\overline{Gv}$ is the subset of V which contains the limit points of all convergent sequences whose elements are taken in Gv.

In the above definition, convergence is in ℓ_2 norm. We have the problem definition:

ORBIT CLOSURE CONTAINMENT: Given a group G which acts on a vector space V, and two elements $u, v \in V$ answer

"Does $v \in \overline{Gu}$?"

This is a slightly relaxed problem as it is sufficient to find a sequence $g_1, g_2, \ldots \in G$ such that $(g_i u)_i$ gets closer to v in the ℓ_2 norm, as i grows. Viewed through this lens, the *approximate* operator scaling also seems like a very natural problem. Successively applying the appropriate LR action makes our operator converge to a doubly stochastic one.

3.2.3 Orbit Closure Intersection

Here, we generalize all the above problems even further in a natural way. We now want to know whether the closed orbits of two elements of our space intersect.

ORBIT CLOSURE INTERSECTION: Given a reductive algebraic group G which acts on a vector space V, and two elements $u, v \in V$ answer

"Is $\overline{Gu} \cap \overline{Gu} \neq \emptyset$?"

This problem has a really convenient characterization due to Mumford [Mum65]:

Theorem 3.3 (Mumford)**.** *For G, u, v as given above, we have that $\overline{Gu} \cap \overline{Gv} \neq \emptyset$ if and only if $P(u) = P(v)$ for every $P \in \mathbb{F}[V]^G$.*

Clearly, one direction is trivial, if $\overline{Gu} \cap \overline{Gv} \neq \emptyset$, then one has that the polynomials in $\mathbb{F}[V]^G$ agree on u and v. The other direction is non trivial.

One interesting group action of $\mathrm{GL}_n(\mathbb{F})$ on a tuple of $n \times n$ matrices that has been studied in the literature of Algebraic Complexity Theory is the *Simultaneous Conjugation.*

Definition 3.4 (Simultaneous Conjugation)**.** We say that the group $\mathrm{GL}_n(\mathbb{F})$ acts on a symbolic matrix $L = (A_1, \ldots, A_m)$ by Simultaneous Conjugation if for every $B \in \mathrm{GL}_n(\mathbb{F})$, we have:

$$B(L) = (BA_1B^{-1}, \ldots, BA_mB^{-1})$$

A deterministic polynomial time algorithm for the orbit-closure intersection problem for this action follows from the work of Raz-Shpilka [RS05] as observed in Forbes-Shpilka [FS13] (who solve a more general problem called Noether Normalization Lemma for this action in deterministic quasipolynomial time).

Geometric Complexity Theory

Speaking in this language of orbits can help us capture some extremely important and difficult problems. This is one of key points of view in Geometric Complexity Theory, started by Mulmuley and Sohoni [MS01] . This program to prove arithmetic lower bounds (and then perhaps Boolean ones) is to use the symmetries of the functions whose complexity is studied. In particular, the determinant and permanent polynomials are actually characterized by their symmetries under natural group actions, and together with their completeness properties allows to cast the lower bound problem in terms of of orbits and orbit-closures under these actions. We would like to know whether a "permanent-like" polynomial is contained in the orbit closure of "determinant-like" polynomial under the action of the group $G = \mathrm{GL}_m(\mathbb{F})$.

More specifically, let us use perm_n and $\det_n$ to denote the permanent and determinant polynomials of an $n \times n$ matrix of variables. We would like to know if the polynomial $X_1^{m-n}\,\mathrm{perm}_n$ is contained in $\overline{G\det_m}$, where m is comparable to n. Here, the action of G on the space of symbolic matrices is just the usual linear transformations. Formally,

MAIN PROBLEM OF GCT: For m^2 variables $X_1, \dots, X_{m^2}$, and an integer n such that $m = O(n^c)$ for some constant c,

"Does $\overline{G\det_m} \ni X_1^{m-n}\,\mathrm{perm}_n$?"

This question lies at the heart of Geometric Complexity theory program. If the answer to the above question is NO (as is conjectured), then it implies that "VP $\neq$ VNP". Unsurprisingly, geometric invariant theoretic tools play an important role in the program.

Null Cone problem

This is a special case of the above problem when we are just looking for the 0 point in the orbit closure of another point

NULL CONE PROBLEM: Given a group G which acts on a vector space V, and an element $u \in V$ answer

"Is $0 \in \overline{Gu}$?"

Indeed, NCRANK is a special case of this problem where the vector space is the space of $n \times n$ symbolic matrices $L = (A_1, \dots, A_m)$, the group is $\mathrm{SL}_n(\mathbb{F})^2$, and the action of an element $(R, C) \in \mathrm{SL}_n(\mathbb{F})^2$ is given by the usual LR action

$$(R, C)(L) = (RA_1C, \dots, RA_mC)$$

Indeed, for every homogeneous polynomial P in the invariant ring $\mathbb{F}[V]^G$, $P(0) = 0$, and we would like to know if the same is true for L, i.e, whether for every d, and matrices $D_1, \dots, D_m \in \mathcal{M}_d(\mathbb{F})$ we have

$$\det\left(\sum_{i \in [m]} A_i \otimes D_i\right) = 0.$$

Recall that these are exactly the generators of the invariant ring under this action. Hence the main result of [GGOW16], by Theorem 3.3, proves that for this particular group action, NULL-CONE-PROBLEM has a polynomial time algorithm.

Remark 3.5. The NULL CONE PROBLEM is very interesting for other group actions as well! Since the group action partitions the vector space into orbits, we would like to have a geometric understanding of these orbits, and in particular, answer questions about containment, etc.

3.3 Upper bounding the degree of our invariant ring generators

In this section, $\mathbb{F}$ will be a large enough field (not necessarily algebraically closed or characteristic zero).

3.3.1 The noncommutative rank of symbolic matrices

Let us review the notion of commutative rank of a symbolic matrix, in order to contrast, and relate it with the noncommutative rank. For a symbolic matrix $L = \sum_{i\in[m]} A_i x_i$, the commutative rank (denoted crank(L)) is defined as:

$$\operatorname{crank}(L) := \min\{\, r : \exists M \in \mathcal{M}_{n\times r}(\mathbb{F}(\mathbf{x})), N \in \mathcal{M}_{r\times n}(\mathbb{F}(\mathbf{x})),\ L = MN\,\}$$

The entries of M and N come from $\mathbb{F}(\mathbf{x})$, the field of rational functions over $\mathbb{F}$.

Now we describe a few equivalent ways to define the noncommutative rank of a symbolic matrix

Factoring the matrix over the free skew field

This is similar to the above definition of commutative rank. Here, instead of the matrices M and N being over the field of rational functions $\mathbb{F}(\mathbf{x})$, they are over the free skew field $\mathbb{F}\langle\!(\mathbf{x})\!\rangle$.

$$\operatorname{ncrank}(L) = \min\{\, r : \exists M \in \mathcal{M}_{n\times r}(\mathbb{F}\langle\!(\mathbf{x})\!\rangle), N \in \mathcal{M}_{r\times n}(\mathbb{F}\langle\!(\mathbf{x})\!\rangle),\ L = MN\,\}$$

A really interesting and useful result due to Cohn [Coh95] states that the entries of the factors may be assumed to be affine forms over $\mathbb{F}\langle\mathbf{x}\rangle$.

This is interesting and useful because now, at least this shows that the problem of finding the noncommutative rank is decidable. In fact, this gives an exponential time algorithm to find the rank. This is done by setting up a system of quardratic equations in $2n(m+1)r$ variables, where the indeterminates are the coefficients of the affine forms. Then we can solve this using, say, any Gröbner basis solver to obtain the factorization.

Although [GGOW16] tests only the singularity of noncommutative symbolic matrices in polynomial time, a recent result by [IQS15] finds the factors efficiently as well. Moreover, it works over every field (in particular, deciding nonsingularity works over every field).

Finding a large zero submatrix

Another result by Cohn [Coh95] (and in full generality, by [FR04]) that will be useful, is the following:

Theorem 3.6. *The noncommutative rank of a symbolic matrix L is the minimum r such that there are matrices $R, C \in \mathrm{GL}_n(\mathbb{F})$ such that RLC satisfies the following:*

1. *There are i, and j such that the submatrix given by the rows indexed by $[n] \setminus [i]$, and columns indexed by $[n] \setminus [j]$ is the zero matrix*
2. *$i + j = r$*

Pictorially, RLC looks as follows:

$$RLC = \begin{array}{cc} & i \\ \left[\begin{array}{c|c} M_{1,1} & M_{1,2} \\ \hline M_{2,1} & 0 \end{array}\right] & j \end{array}$$

Here $M_{1,1}$ is an $i \times j$ submatrix with $i + j = r$.

Remark 3.7. This is to be understood as a linear algebraic analogue of Hall's Theorem. In fact, this $(n-i) \times (n-j)$ zero submatrix is called the *Hall blocker*. This is further expounded upon, in Section 3.3.1. The main theorem there is to be thought of as the analogue of 'neighbourhood shrinkage' that Hall's Theorem states, as a condition for no perfect matching. See Problem 1.9 in the tutorial homework.

Aside: Commutative rank of symbolic matrices

The statement of Theorem 3.6, in form, looks similar to a theorem that says something similar about commutative rank, due to Flanders [Fla62]

Theorem 3.8. *Suppose the commutative rank of a symbolic matrix L over a large enough field $\mathbb{F}$ is at most r. Then there are matrices $R, C \in \mathrm{GL}_n(\mathbb{F})$ such that RLC satisfies the following:*

1. *There are i, and j such that the submatrix given by the rows indexed by $[n] \setminus [i]$, and columns indexed by $[n] \setminus [j]$ is the zero matrix*
2. *$i = j = r$*

Pictorially, RLC looks as follows:

$$RLC = \begin{array}{cc} & r \\ \left[\begin{array}{c|c} M_{1,1} & M_{1,2} \\ \hline M_{2,1} & 0 \end{array}\right] & r \end{array}$$

Proof. Let us look at L as a space of matrices. If $\mathrm{rank}(L) = r$, there must be some matrix of rank r. By linear transformations, assume it is the block matrix with I_r in the top left, and zero elsewhere. Call this matrix A.

$$A = \left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right]$$

For any other matrix B, consider $At + B$ for a variable t

$$At + B = \left[\begin{array}{c|c} B_{1,1} + tI_r & B_{1,2} \\ \hline B_{2,1} & B_{2,2} \end{array}\right]$$

Let us denote the columns of $B_{1,2}$ as $u_{r+1}, \ldots, u_n$, and the rows of $B_{2,1}$ as $v_{r+1}, \ldots, v_n$. For any $r+1 \times r+1$ submatrix, the determinant is zero so we have, so in particular, for the $[r] \cup \{i\} \times [r] \cup \{j\}$ submatrix, we have the determinant:

$$\det(B_{1,1} + tI_r) b_{i,j} - \langle v_i, u_j \rangle = 0$$

If $|\mathbb{F}|$ is large enough, then this can only happen when $b_{i,j} = 0$ (and $\langle v_i, u_j \rangle = 0$). This implies that $B_{2,2} = 0$. □

This gives us that for every symbolic matrix L, we have

$$\operatorname{crank}(L) \leq \operatorname{ncrank}(L) \leq 2 \operatorname{crank}(L)$$

So in particular, computing the noncommutative rank gives a deterministic factor 2 approximation for the commutative rank of a space of matrices over a large enough field. Later, Blaser, et al. in [BJP16] gave a deterministic FPTAS for the same problem.

Shrunk subspaces

Another way to say if the matrix is nonsingular is if it does not decrease the dimension of any subspace. Formally,

Theorem 3.9. *A symbolic $n \times n$ matrix $L = \sum_{i \in [m]} A_i x_i$ in noncommuting variables is singular if and only if there exist subspaces $W, U \leq \mathbb{F}^n$ with $\dim(W) < \dim(U)$ and*

$$A_i U \subseteq W \tag{3.1}$$

For every $i \in [m]$.

In fact, this theorem is implied by the above characterization due to Cohn, and further shows the connection to Hall's Theorem (see Remark 3.7).

Proof of implication. We may assume that the operator L, and therefore, all the A_i's are already of the form. Since we have that $i + j < n$, we have that $i < n - j$.

Now that we this, it is clear to see that every A_i takes every vector that is supported on the last $n - j$ coordinates to vectors whose support is contained in the first i coordinates. Setting U to be the subspace of vectors supported on the last $n - j$ coordinates, and V to be the subspace of vectors supported on the first i coordinates, we have our required claim. □

Rank Decreasing

In the *rank decreasing* notion of Gurvits [Gur04], we have (here the underlying field $\mathbb{F} = \mathbb{C}$):

Theorem 3.10 (Gurvits, [Gur04])**.** *A symbolic matrix $L = \sum_{i\in[m]} A_i x_i$ in noncommuting variables (recall this can also be thought of as an operator $L(P) = \sum_{i=1}^{m} A_i P A_i^{\dagger}$) is singular if and only if there exists a matrix $P \succeq 0$ such that*

$$\operatorname{rank}(L(P)) < \operatorname{rank}(P) \tag{3.2}$$

This can be proved by showing that rank decreasing property is equivalent to the subspace shrinkage property.

Exercise 3.11. For a tuple of matrices $L = (A_1, \ldots, A_m)$, condition 3.1 holds if and only if condition 3.2 holds.

3.3.2 Blow-up and degree upper bounds

Recall that $d^*(n)$ was used to denote the upper bound on the degree of the generators of the invariant ring V_G where $G = \mathrm{SL}_n(\mathbb{F})^2$ acts on a space V of symbolic matrices by the LR action. For a nonsingular $n \times n$ symbolic matrix $L = \sum_{i\in[m]} A_i x_i$, we have a related quantity (note that $\sigma_L(n) \le d^*(n)$):

$$\sigma_L(n) = \min\left\{ d \,:\, \exists D_1, \ldots, D_m \in \mathcal{M}_d(\mathbb{F}),\ \det\left(\sum_{i\in[m]} A_i \otimes D_i\right) \neq 0 \right\}$$

By a very general result of Derksen [Der01],[1] it is enough to bound $\sigma_L(n)$, as in that paper Derksen proved that $d^*(n) \le n^3 \cdot \sigma_L(n)^2$.

We have used an upper bound on this in the proof of [GGOW16], but now we will see a different and simple proof of much stronger bound for this group action due to [IQS15].

The proof goes by introducing several intermediate quantities $\operatorname{rank}_d(L)$ between $\operatorname{rank}(L)$ and $\operatorname{ncrank}(L)$; defined as

$$\operatorname{rank}_d(L) := \max\left\{ \operatorname{rank}\left(\sum_{i\in[m]} A_i \otimes D_i\right) \,:\, D_1, \ldots, D_m \in \mathcal{M}_d(\mathbb{F}) \right\}$$

Clearly, we have $\operatorname{rank}_d(L) \le \operatorname{ncrank}(L)$ for every d, by definition. Moreover we also have $\operatorname{rank}(L) = \operatorname{rank}_1(L)$. We will need some lemmas relating quantities $\operatorname{rank}_d(L)$ and $\operatorname{rank}_{d-1}(L)$. The first is a simple one that allows us to 'go from' $\operatorname{rank}(L)$ to $\operatorname{ncrank}(L)$, via a sequence of $\operatorname{rank}_d(L)$'s.

Lemma 3.12. *We have that* $\operatorname{rank}_d(L) \le \operatorname{rank}_{d-1}(L) + 2n$.

Proof. Indeed, for any $d \times d$ matrix K, whose rows and columns indexed by $[d]$, let $K^{(d-1)} := K([d-1],[d-1])$ be a submatrix of K whose rows and columns are indexed by $[d-1]$.

We have that $\sum_{i\in[m]} A_i \otimes D_i^{(d-1)}$ is a submatrix of $\sum_{i\in[m]} A_i \otimes D_i$ with n fewer rows and n fewer columns, and hence has rank at least $\operatorname{rank}\left(\sum_{i\in[m]} A_i \otimes D_i\right) - 2n$. $\square$

[1] Derksen's result holds for any general reductive group action. We will not state his result here in full generality as we only need what is stated above, which is pertinent to the group action which we are studying.

Another very important lemma from For every d, L, we will need the following result from [IQS15], which was inspired by the ideas in [Ami66]. This, combined with Lemma 3.12, will give us much more control when 'going from' rank(·) to ncrank(·).

Lemma 3.13. *For every integer d, and symbolic matrix L, we have*

$$d \,\big|\, \mathrm{rank}_d(L)$$

Theorem 3.14. *For an $n \times n$ nonsingular symbolic matrix $L = \sum_{i\in[m]} A_i x_i$, we have*

$$\sigma_L(n) \le n+1$$

Proof. By the definition of $\sigma = \sigma_L(n)$, we have that $\mathrm{rank}_{\sigma-1}(L) < (\sigma-1)n$. But Lemma 3.13 tells us that this means

$$\mathrm{rank}_{\sigma-1}(L) \le (\sigma-1)(n-1).$$

On the other hand, we have from Lemma 3.12 that

$$\mathrm{rank}_{\sigma-1}(L) \ge \mathrm{rank}_{\sigma}(L) - 2n = n\sigma - 2n.$$

Combining, we have $\sigma \le n+1$. □

3.4 Brascamp–Lieb inequalities

We will now move on to a completely different topic, in the field of Analysis.

While PIT is the problem of proving algebraic identities, the analytic analog is proving inequalities. I will now introduce the Brascamp–Lieb inequalities, discuss their rich structural theory and their basic computational problems. I will then describe a scaling algorithm for proving them and give an application of the Brascamp–Lieb inequalities to optimization. From their structure theory, it will become clear that our scaling algorithm efficiently solves large family of linear programs of exponential size, and I will give some examples. This part of the lecture is based mainly on [GGOW17].

3.4.1 Introduction

Brascamp–Lieb inequalities (and their reverse version) are a vast generalization of lots of inequalities that are important in various themes in Mathematics. For example, Hölder's inequality, the Brunn–Minkowski inequality, Nelson's Hypercontractivity, the Prékopa–Leindler inequality, the Loomis–Whitney inequality, and many more.

We will start by seeing a series of examples of some of them, starting with simple ones and getting more complex, to illustrate their general structure. In particular, we will point out some features of these inequalities, and then we will see how to generalize them, and what are the main questions that arise when we study them.

Throughout this lecture, we will look at general functions $f : \mathbb{R}^d \to \mathbb{R}_{\ge 0}$ that are integrable. We will also need a notion of norm. We will work with the standard norms, but will depart slightly from the conventions by considering $p \in (0,1)$ and writing p in the denominator (the reason for this will become clear). So our "p-norm" of f is

$$\|f\|_{1/p} = \left(\int_x f(x)^{1/p}\right)^p.$$

Let us start with something that we all know – the Cauchy–Schwarz inequality. If we have two functions $f_1, f_2 \colon \mathbb{R} \to \mathbb{R}_{\geq 0}$, then

$$\int_{x\in\mathbb{R}} f_1(x) f_2(x) \leq \|f_1\|_2 \|f_2\|_2.$$

In this case our norms p_1 and p_2 on the R.H.S. are $\boxed{p_1 = p_2 = 1/2}$.

Once we have seen an inequality like this, an immediate natural question is whether we can generalize it. One way to do so is to ask if the norms that appear on the right hand side can be different, as we would always try to estimate the integral of the product of some functions.

Of course we can, because of Hölder's inequality, which says that we can actually use p_1 and p_2 norms, as long as $\boxed{p_1 + p_2 = 1}$. That is,

$$\int_{x\in\mathbb{R}} f_1(x) f_2(x) \leq \|f_1\|_{1/p_1} \|f_2\|_{1/p_2}.$$

An even simpler example

Now let us move to an even simpler example. But in this example we are going to blow up the dimension. In the previous example, they are functions from $\mathbb{R}$ to $\mathbb{R}_{\geq 0}$. Now we have to look at functions on $\mathbb{R}^2$. We will denote $x \in \mathbb{R}^2$ by $x = (x_1, x_2)$, and will write two functions f_1, f_2 as before.

Suppose we are given some, convex or nice, shape S in $\mathbb{R}^2$, and we want to estimate the area of this shape. One simple thing we can do is to project it to both directions (See Figure 3.1). So the projections will be of some lengths ℓ_1 and ℓ_2, and we know that

$$\text{Area}(S) \leq \ell_1 \cdot \ell_2.$$

This is a consequence of a trivial inequality. In fact in our example it will even be an identity. If we look at the integral

$$\int_{x\in\mathbb{R}^2} f_1(x_1) f_2(x_2),$$

because x can be separated into (x_1, x_2), this exactly equals to $\|f_1\|_1 \|f_2\|_1$. This should be compared with the entropy inequality (this holds for discrete as well as continuous entropy but the continuous one is more relevant here):

$$H(X_1, X_2) \leq H(X_1) + H(X_2).$$

If we wanted to write it as an inequality by replacing the two norms, we couldn't. The only thing we can get even with an inequality is when $\boxed{p_1 = p_2 = 1}$.

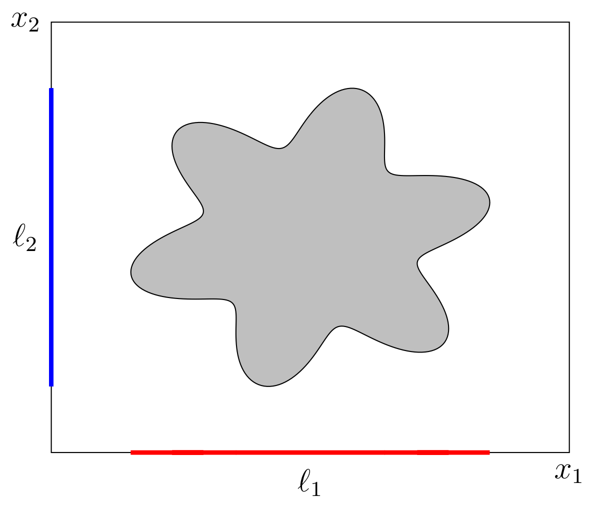

Figure 3.1: Projecting a shape $S \subseteq \mathbb{R}^2$ onto the lines x_1 and x_2.

The Loomis–Whitney inequality

But we can still do a generalization of this, which is the Loomis–Whitney inequality. In fact it is a special case of the Loomis–Whitney inequality, that was discovered many times and in many contexts.

A good way to think about it is to generalize the above to three dimensions. How do we think of it as a generalization to three dimensions? We have some block lying in $\mathbb{R}^3$, and again we want to estimate its volume. How do we estimate the volume? We can again project it to the x_i's like before, but it is not so interesting. Suppose we project it now, to planes, (x_1, x_2), and we get some area here. We can do the same for (x_2, x_3) and (x_1, x_3), and we get some other areas (See Figure 3.2). Suppose we know this block is some set $S \subseteq \mathbb{R}^3$, and its projections' areas are A_1, A_2, A_3. What do we know? Suppose we want to bound the volume of S in terms of a function of the areas. We can have

$$\mathrm{Vol}(S) \le \sqrt{\mathrm{Area}(A_1)\mathrm{Area}(A_2)\mathrm{Area}(A_3)}.$$

Many people have seen this inequality in many contexts. It is easy to prove. People may have seen, for example, for any graph G,

$$\text{\# of triangles in } G \le (\text{\# of edges in } G)^{3/2}.$$

As an entropy inequality, it is equivalent to:

$$H(X_1, X_2, X_3) \le \frac{1}{2}\Big(H(X_1, X_2) + H(X_2, X_3) + H(X_1, X_3)\Big).$$

They are all the same and they come from inequalities of the above kind, that talk about projections. Now we are in $\mathbb{R}^3$, we are going to have three functions, $f_1, f_2, f_3 \colon \mathbb{R}^2 \to \mathbb{R}_+$. And we have

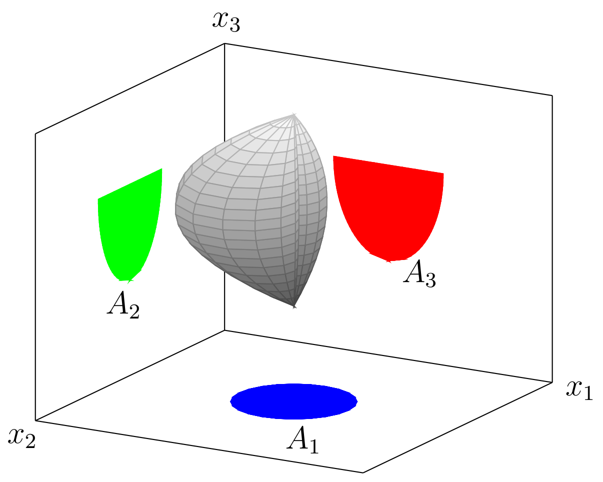

Figure 3.2: Projecting a body $S \subseteq \mathbb{R}^3$ onto planes.

the following inequality for their integral,

$$\int_{x=(x_1,x_2,x_3)\in\mathbb{R}^3} f_1(\pi_{\{2,3\}}(x))\, f_2(\pi_{\{1,3\}}(x))\, f_3(\pi_{\{1,2\}}(x)) \le \|f_1\|_2\|f_2\|_2\|f_3\|_2.$$

We see here that $\boxed{p_1 = p_2 = p_3 = 1/2}$. Again we cannot replace p_1, p_2, p_3 with other numbers.

Remark 3.15. An extension of the Loomis–Whitney inequality, that is called the *Shearer inequality*, also has a similar form.

The sharp Young inequality

Let us do one more example. Let us go back to two dimensions. Recall that we are looking at some estimation problem. We are given some projections and we want to understand the area. Again we have some shape S in $\mathbb{R}^2$. Like before we want to estimate its area. So we are given some projections onto x_1 and x_2. Suppose you are given more information, maybe a projection on a third direction, $x_1 - x_2$, that has length ℓ_3 (See Figure 3.3). Is there a better estimate of the area using these 3 projections? It turns out that

$$\text{Area}(S) \le \frac{\sqrt{3}}{2}(\ell_1\ell_2\ell_3)^{2/3}.$$

Remark 3.16. For this inequality to hold, the angle of the third projection is important.

For the first time we see there is a constant $\frac{\sqrt{3}}{2}$, in the other inequalities, the constant is 1. In fact $\sqrt{3}/2$ is the optimal constant for this inequality to hold.

More generally, suppose you have three functions $f_1, f_2, f_3 \colon \mathbb{R}^2 \to \mathbb{R}_{\ge}$. The *sharp Young inequality* says that

$$\int_{x=(x_1,x_2)} f_1(\pi_{x_1}(x))\, f_2(\pi_{x_2}(x))\, f_3(\pi_{x_1-x_2}(x)) \le \frac{\sqrt{3}}{2}\|f_1\|_{3/2}\|f_2\|_{3/2}\|f_3\|_{3/2}.$$

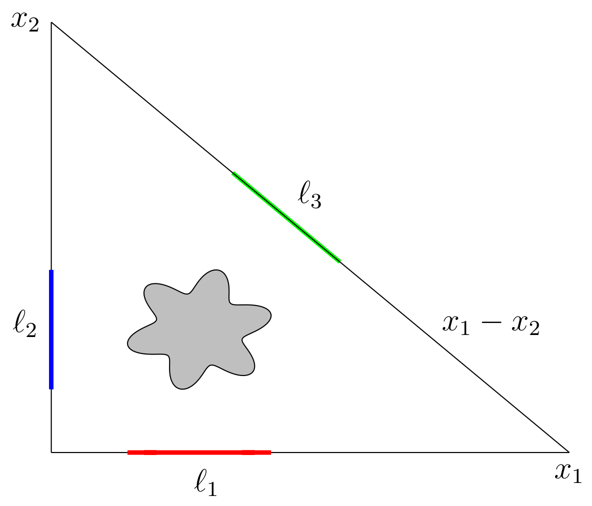

Figure 3.3: Projecting a shape $S \subseteq \mathbb{R}^2$ onto the lines x_1, x_2 and $x_1 - x_2$.

Again we ask the same question. Are these the only norms we can put? It turns out that you can replace the norms, but to replace the norms you also have to replace the constant. In general, we can replace the $2/3$ by p_1, p_2, p_3, where $\boxed{p_1 + p_2 + p_3 = 2 \text{ and } 0 \le p_i \le 1}$, and the constant $\sqrt{3}/2$ will become

$$\left(\frac{\prod_i (1-p_i)^{(1-p_i)}}{\prod_i {p_i}^{p_i}} \right)^{1/2}.$$

3.4.2 General setting of Brascamp–Lieb inequalities

Now let us state the general setting where all these inequalities fit in [BL76, Lie90].

First we are going have some input data. We have some dimension $\mathbb{R}^n$, and we have some *linear* maps $B_j \colon \mathbb{R}^n \to \mathbb{R}^{n_j}$ from $\mathbb{R}^n$ to lower dimensions. They can be represented by $n_j \times n$ matrices. We also have some $p_j \ge 0$. So we have input $B = (B_1, \ldots, B_m)$ and $p = (p_1, \ldots, p_m)$, which will be our input data.

The nature of our inequality looks like the following. For any $f_j \colon \mathbb{R}^n \to \mathbb{R}_{\ge 0}$ that are integrable,

$$\int_{x \in \mathbb{R}^n} \prod_{i=1}^{m} f_j(B_j(x)) \le C \prod_{j=1}^{m} \|f_j\|_{1/p_j}, \tag{3.3}$$

where $C \in (0, \infty]$.

This should capture all the inequalities that have been written. Of course, we expect some of them to hold ($C < \infty$) and some of them to not. So given this form of potential inequalities, we can ask which of them are true and which are not. The basic questions here are:

1. Is $C < \infty$?
2. If C is finite, what is the *optimal* C? Here optimal is smallness. This optimal C is called the *Brascamp–Lieb constant*, denoted by $\mathrm{BL}(B, p)$.

3. Continuity of $\mathrm{BL}(B,p)$? What happens to the constant if we perturb B a little bit? This question is crucial for the non-linear versions of the BL inequality, which are important because they are related to number theory, analysis, etc.

Basically the upshot of this lecture is that whatever structure is known, we can actually do efficiently in polynomial time. Finding the answer to 1 and 2 is in P [GGOW17]; namely, we can in polynomial time test whether this constant is finite, and compute (to arbitrary precision) the optimal constant. And because we are doing it using a similar method to our operator scaling algorithm, it is continuous. We actually give explicit bounds that were not known because all previous arguments used compactness. We will not talk about 3 in this lecture.

Structure of Brascamp–Lieb inequality

Let us first see what was known structurally. There are two sets of results. One is in [Lie90], where Lieb actually characterizes the optimal constant. And it is the following characterization which should look familiar to you. He proved that

Theorem 3.17 ([Lie90])**.**

$$\mathrm{BL}(B,p)^2 = \sup_{A_j \succ 0} \frac{\prod_{j=1}^m \det(A_j)^{p_j}}{\det(\sum_{j=1}^m p_j B_j^T A_j B_j)}.$$

What does the R.H.S. look like? It looks like **capacity** (Definition 1.14)! There are various but not too complicated differences. For example, it is a supremum but we can invert it. It is a non-convex optimization in terms of the A_i's. So you can think of given B and p, compute $\mathrm{BL}(B,p)$. And [GGOW17] show that computing this (approximately) is in P. It turns out that

$$\mathrm{BL}(B,p)^2 = 1/\operatorname{cap}(L_{B,p}) = \left(\inf_{P \succ 0} \frac{\det(L_{B,p}(P))}{\det(P)}\right)^{-1}$$

for some operator $L_{B,p}$ that is going to depend on B and p. We are not going to define it here. That's one way to do it and in fact we give a reduction in the paper (See [GGOW17, Construction 4.2]).

But today we will give a self-contained algorithm. The analysis here reduces to the analysis in the operator scaling algorithm. But one can also do it directly using the same invariant-theoretic proof and the same Derksen bound.

Let us explain how the algorithm suggests itself, and should at least have suggested to the people who characterize it. Remember the obvious definition (i.e. Equation (3.3)) of the optimal constant says it takes the supremum of all functions. So why do these determinants in the characterization show up?

They come from Gaussian functions. What Lieb [Lie90] actually showed is that the worst functions one can take for such inequalities are Gaussian functions. So the optimal $f_j(x_j)$ will look something like $\exp(-\pi x_j A_j x_j)$, and its integral is the reciprocal of the square root of the determinant, i.e.,

$$\int_{x_j} f_j(x_j) = \frac{1}{\sqrt{\det(A_j)}}.$$

So the question is which of the correlation matrices A_j that make it the hardest. It is an optimization problem similar to the one that arose in operator scaling. The domain is convex, but the function itself is not. Nevertheless we can prove that it is in P.

And then comes the beautiful paper of [BCCT08]. they give a characterization for when the constant is finite.

If you now go back to all the boxes, they all look like linear problems. They are linear inequalities in the $p_j's$ (this linearity explains why we wrote p in the denominator in the definition of norm). They realized that this works in general. They proved that

$$\mathrm{BL}(B,p) < \infty \text{ if and only if } p \text{ lies in some polytope } P_B \text{ defined by } B.$$

P_B is a very simple polytope. There are two conditions. The first one is a normalization condition, and the second one is a list of inequalities:

Definition 3.18 (Brascamp–Lieb Polytope)**.** A vector $p = (p_1, \dots, p_m)$ lies in the *Brascamp–Lieb polytope P_B* if and only if

1. $\sum_j p_j n_j = n$;
2. for every vector space $V \le \mathbb{R}^n$, $\dim(V) \le \sum_j p_j \dim(B_j V)$.

So P_B is a convex set. In fact when you look at it, it is not clear why it is a polytope. Because there are uncountable many inequalities. But if you think about it, what numbers could appear in the inequalities? The dimension is a number between 1 and n. So there could be only exponentially many discrete inequalities, and P_B is a polytope. Of course, knowing which of these possible inequalities are actually facets of the polytope is not obvious, and that's a major question: We are given B and p, and we ask if $p \in P_B$. Again [GGOW17] can solve this in P (in the case when p_i's have "small" (poly(n)) common denominator).

BL-polytopes are a new class of polytopes that have exponentially many facets. We have an efficient membership oracle, as well as a separation oracle for them. We can now solve all of these in polynomial time (with the caveat that p_i's should have small common denominator). So that's a connection to combinatorial optimization that we wanted to talk about. At the end we hope to give some examples of known optimization problems which give rise to such polytopes. However, we have no new applications yet, and this remains an interesting challenge.

Before we tell you the algorithm, let us explain why you should have known the algorithm once you know about the operator scaling algorithm.

Geometric Brascamp–Lieb inequalities

There are two more things that were known in this theory. One is *geometric Brascamp–Lieb inequalities.* These were introduced by Ball [Bal89] and Barthe[Bar98]. These are some special cases of of BL-inequalities, where some nice things happen to the data.

Definition 3.19 (Geometric Brascamp–Lieb data)**.** A BL-data (B, p) is *geometric* if it satisfies

1. **Projection property:** $B_j B_j^T = I_{n_j}$ for every j. That is, B_j's are actually projection matrices.

2. **Isotropy:** $\sum_j p_j B_j^T B_j = I_n$. That is, the matrices are sort of well spread when put together with weights p_j.

What they proved is that, if we are in the geometric case – if we are given a geometric data, then $\mathrm{BL}(B,p) = 1$.

Theorem 3.20 ([Bal89, Bar98])**.** *If* (B,p) *is geometric, then* $\mathrm{BL}(B,p) = 1$.

Remark 3.21. One can prove that if either (1) or (2) is satisified, then $BL(B,p) \geq 1$.

So we understand perfectly well in the geometric condition. There is an inequality and equality holds with a constant 1.

The characterization of $\mathrm{BL}(B,p)$ reminds you of capacity. What do conditions (1) and (2) remind you of? **Double stochasticity** (Definition 1.11)! There is a natural group action on these things. We have all these B_j's. We can change basis in all of the spaces $\mathbb{R}^{n_1},\dots,\mathbb{R}^{n_m}$. This corresponds to multiplying on the left by appropriate (invertible) matrices $C_1,\dots,C_m$. We can also multiply on the right to change simultaneously the basis in $\mathbb{R}^n$. So we transform

$$(B_1,\dots,B_m) \mapsto (C_1B_1C,\dots,C_mB_mC) =: \tilde{B}.$$

There is a simply way for which the BL constant is updated. It updates multiplicatively:

Theorem 3.22 ([BCCT08])**.**
$$\frac{BL(\tilde{B},p)}{BL(B,p)} = \frac{1}{\det(C)\prod_j \det(C_j)^{p_j}}.$$

It is a very simple action. Since we care about finding the BL constant, it is clear what we should do. Take an arbitrary BL-data. Let us bring it to the geometric position by these linear transformations. After we bring it to geometric position, we know what the constant is. It is 1. We also know how to go back using the updates.

Scaling algorithm for approximating $\mathrm{BL}(B,p)$

Now the question is how to impose these two conditions together. They conflict with each other. But they conflict with each other the same way we saw before. So now you should realize what we should do in the algorithm:

Algorithm 3 Algorithm for $(1+\varepsilon)$-approximating $\mathrm{BL}(B,p)$

```
Require: $(B,p)$, a BL-data
1: for $t = \mathrm{poly}(n,d,1/\varepsilon)$ steps do (here $d$ denotes the common denominator of $p_i$'s)
2:     Enforce Projection property: For each $j$, let $C_j = B_jB_j^T$, then $B_j \leftarrow C_j^{-1/2}B_j$.
3:     Enforce Isotropy: Let $C = \sum_j p_jB_j^TB_j$, then $B_j \leftarrow B_jC^{-1/2}$
4: end for
```

And this will converge for the same reason like we said about Algorithm 2. We can prove it directly, or we can prove it by reduction. We can take a BL-data and construct from it a symbolic matrix $L_{B,p}$, and run the algorithm to compute the capacity of $L_{B,p}$ and and this will give us $\mathrm{BL}(B,p)$.

This scaling viewpoint also gives the first explicit bounds on $\mathrm{BL}(B,p)$ when it is finite. Then here is another question [BB10, BBFL15] had to deal with in the non-linear case. Say I have the BL-constant for B and p. Now I perturb B a little bit, by ε in some norm. Let's say it remains finite, how much different can it be. Again, the scaling viewpoint gives first explicit bounds here.

This connection to operator scaling also gives another proof of the [BCCT08]'s characterization. $\mathrm{BL}(B,p)$ is similar to capacity. The two conditions for geometric BL-data are doubly stochasticity. What does [BCCT08]'s characterization (Definition 3.18) remind you of in the operator scaling world? The inequality

$$\dim(V) \le \sum_j p_j \dim(B_j V)$$

should remind you of **shrunk subspaces**. Indeed, it follows directly, after the reduction to Operator Scaling, by Fact 2.3.

There is also the *Reverse BL inequality* [Bar98]. It is not as clean, but we can state similar things for the reverse just because Lieb's characterization holds there as well [Bar98] and gives rise to the same optimization problem.

3.4.3 Summary and some applications

We talked about an application of the operator scaling result. Until now it was all algebraic, most of the problems come from algebra. It turns out there is this completely different theory in analysis, which is the Brascamp–Lieb inequality. In this world, there is a question that is completely analagous to the question we asked in the algebraic world. In the algebraic world, we hypothesize, conjecture an algebraic identity and we want to prove it and think about the complexity of this class. In the analytic world, we are in the same business with inequalities. For this large family of inequalities, it turns out the technique in operating scaling does suggest itself. So we give computational answer, efficient answer of the same nature. We get polynomial time algorithms to answer the important questions in this theory. Some answers to the basic questions follow from this operator scaling algorithm. We did not show the connection to operator scaling directly. But we think everything was suggestive. The objects that appear here, and the structure theory are very similar to the objects that appear in the operator scaling world. Moreover, this suggestion immediately suggests an algorithm. It is an alternating minimization algorithm applied to the data of the BL inequality. So because of this, we have another algorithm in the analytic world that solves this basic question. Moreover, we give different proofs of some of the structural results.

Let us now give two examples of BL-polytopes which we know. We do not have new combinatorial optimization problem that we can optimize over this membership oracle. But we know some old ones. Also the membership and separation oracles we give have bad dependence on the common denominator of p_i's.

Matroid. Here the input is an $n \times m$ full-dimension matrix $M = [\, v_1 \cdots v_m \,]$. So $v_1, \ldots, v_m$ are independent. Here the polytope of M, P_M is simply the convex hull of the characteristic vectors

$$\{\, \mathbb{1}_I : \ I \subseteq [m], \dim(M_I) = n \,\}.$$

In other words, we look at the full dimension subsets, and we take the convex hull of these subsets. It turns out this is a BL-polytope. It is easy to see that this is P_B, where $B_j x = \langle v_j, x \rangle$. All the projections are one-dimensional. We must stress that even the one-dimension is extremely interesting. The scaling is used in Forster's theorem [For02] about sign rank. Forster analyzed the same scaling algorithm that we mentioned above in the one dimensional case (although he just proved convergence which sufficed for him). Also, the problem of putting a matrix to an isotropic position [HM13] is also a special case of this algorithm. There are other applications which I did not talk about. For higher-dimension there are applications to combinatorial geometry [DGOS16].

Matroid Intersection. It is a more complex combinatorial optimization problem. It is a theory that generalizes bipartite matching. We are given two matrices, $M_1 = [\, v_1 \cdots v_m \,]$ and $M_2 = [\, u_1 \cdots u_m \,]$. Here we only care about sets I that are full-dimension in both M_1 and M_2. So the polytope P_{M_1, M_2} is simply the convex hull of

$$\left\{ \mathbb{1}_I : \ I \subseteq [m], \dim({M_1}_I) = n, \dim({M_2}_I) = n \right\}.$$

Here's a simple exercise for the students.

Exercise 3.23. Show that $P_{M_1, M_2} = P_B$, where $B_j(x) = \langle v_j, x \rangle, \langle u_j, x \rangle$, that is, B_j is a projection onto two dimensions.

Even though there are exponentially many vertices and exponentially many facets in both cases, it is well known in the optimization community on how to test membership and compute separation oracle in polynomial time.

One open question is to encode general matching as BL-polytope. Finding applications or lower bounds, to show what BL-polytopes cannot do, would be interesting.

Chapter 4

Additional exercises

To conclude these notes, we have collected below some extra exercises, which will help the interested reader to deepen their understanding of different parts of this material.

Problem 4.1 (Wallace-Bolyai-Gerwien theorem)**.** Prove that a polygon in 2 dimensions can be transformed into another by cutting (along straight lines) into finite number of pieces and recomposing them by rotations and translations iff they have the same area. (*Hint*: Using triangulation, prove that a polygon of area A has in its orbit a rectangle of side lengths 1 and A.)

Problem 4.2 (Hilbert's third problem)**.** Hilbert conjectured that the same cut and paste strategy does not work for polyhedra in 3 dimensions of equal volume. This was proved by Max Dehn in 1900. Challenge yourself to prove that a regular tetrahedron cannot be transformed into a cube of equal volume. (*Hint*: construct an invariant other than volume using angles and edge-lengths!)

Problem 4.3 (Invariant rings)**.** Let $G \leqslant \mathrm{GL}_m(\mathbb{F})$ be a linear group that acts on $\mathbb{F}^m$, which after fixing a basis for $\mathbb{F}^m$ can be thought of as acting on variables $\mathbf{x} = (x_1, \dots, x_m)$. Prove that the set of invariant polynomials $\mathbb{F}[\mathbf{x}]^G$ is a subring of $\mathbb{F}[\mathbf{x}]$.

Problem 4.4 (Symmetric polynomials)**.** Hilbert in 1890 proved his famous theorem that invariant rings of linearly reductive group actions (includes all group actions one will ever encounter!) are finitely generated. Prove this theorem for the natural action of the symmetric group S_m on $\mathbb{F}^m$. In other words, prove that the ring of symmetric polynomials is generated by the m elementary symmetric ones.

Problem 4.5. What are the generators for the invariant ring of the following group action: $SL_n(\mathbb{C}) \times SL_n(\mathbb{C})$ acts on $M_n(\mathbb{C})$ by left-right multiplication i.e. $(A, B) \colon X \to AXB$.

Let G act on a vector space V. Nullcone of the group action is defined as the set of common zeroes of all homogeneous invariant polynomials. Orbit of a vector $v \in V$ is the set $\{g \cdot v : g \in G\}$. Orbit-closure of v is the closure of the orbit in the Zariski topology. When the field is $\mathbb{C}$ and the group G is an algebraic group (like $\mathrm{GL}_n(\mathbb{C})$), this coincides with closure in the Euclidean topology.

Problem 4.6. What is the nullcone of the above left-right action? When are two matrices in the same orbit? When does one lie in the orbit-closure of the other? When do two orbit-closures intersect?

Problem 4.7 (Orbits for finite groups)**.** Let G be a finite group that acts on a vector space V. Prove that two vectors $v_1, v_2 \in V$ are in the same orbit iff $p(v_1) = p(v_2)$ for all invariant polynomials $p \in \mathbb{F}[V]^G$. (*Hint*: Use Hilbert's Nullstellansatz.)

Graph isomorphism is also an orbit problem and it is an excellent open question whether the above invariant theory approach can lead to efficient algorithms for it by giving a succinct description of the generating invariants.

The determinantal complexity of the $n \times n$ permanent is defined as the least m s.t. there is an affine map from the $n \times n$ matrix Y to an $m \times m$ matrix $A(Y)$ s.t. $\text{perm}(Y) = \det(A(Y))$. The least such m is denoted by $\text{dc}(n)$. The VP vs. VNP question is roughly equivalent to proving super-polynomial lower bounds on $\text{dc}(n)$. The best known lower bound (over the complex numbers) is $\text{dc}(n) \geq n^2/2$, due to [MR04].

If a group G acts on a vector space V, then this induces an action on polynomials $\mathbb{F}[V]$ as well as the degree d part $\mathbb{F}[V]_d$ via $g \cdot p(v) := p(g^{-1} \cdot v)$. The next problem considers such an action of $\text{GL}_m(\mathbb{C})$ on the space of polynomials over $M_m(\mathbb{C})$ induced by the action of $\text{GL}_m(\mathbb{C})$ on $M_m(\mathbb{C})$ by left multiplication. Let $X_{[n],[n]}$ denote top left $n \times n$ minor of X.

Problem 4.8 (Permanent vs. Determinant)**.** Suppose the determinantal complexity of the $n \times n$ permanent is at most m. Then prove that the polynomial $X_{1,1}^{m-n}\,\text{perm}(X_{[n],[n]})$ lies in the orbit-closure of the polynomial $\det(X)$.

Problem 4.9. Prove that if $p \notin P_B$, then $\text{BL}(B, p) = \infty$.

Problem 4.10. Using Lieb's description, prove that the BL constant is at least 1 if the BL datum satisfies either the projection or isotropy property.

Problem 4.11 (BL constant for geometric datum)**.** Using Lieb's description of the BL constant, prove that the BL constant of a geometric BL datum is always 1.

Bibliography

[ALOW] Zeyuan Allen Zhu, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Personal Communication. 3

[ALOW17] Zeyuan Allen Zhu, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Much faster algorithms for matrix scaling. *CoRR*, abs/1704.02315, 2017. 1.2.1

[Ami66] S. A. Amitsur. Rational identities and applications to algebra and geometry. *J. Algebra*, 3:304–359, 1966. 2.2.3, 2.2.3, 2.15, 2.2.3, 2.16, 3.3.2

[Bal89] Keith Ball. Volumes of sections of cubes and related problems. In *Geometric aspects of functional analysis (1987–88)*, volume 1376 of *Lecture Notes in Math.*, pages 251–260. Springer, Berlin, 1989. 3.4.2, 3.20

[Bar98] Franck Barthe. On a reverse form of the Brascamp-Lieb inequality. *Invent. Math.*, 134(2):335–361, 1998. 3.4.2, 3.20, 3.4.2

[BB10] Jonathan Bennett and Neal Bez. Some nonlinear Brascamp-Lieb inequalities and applications to harmonic analysis. *J. Funct. Anal.*, 259(10):2520–2556, 2010. 3.4.2

[BBFL15] J. Bennett, N. Bez, T. C. Flock, and S. Lee. Stability of the Brascamp-Lieb constant and applications. *ArXiv e-prints*, August 2015. 3.4.2

[BCCT08] Jonathan Bennett, Anthony Carbery, Michael Christ, and Terence Tao. The Brascamp-Lieb inequalities: finiteness, structure and extremals. *Geom. Funct. Anal.*, 17(5):1343–1415, 2008. 3.4.2, 3.22, 3.4.2

[BJP16] Markus Bläser, Gorav Jindal, and Anurag Pandey. Greedy strikes again: A deterministic ptas for commutative rank of matrix spaces. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:145, 2016. 3.3.1

[BL76] Herm Jan Brascamp and Elliott H. Lieb. Best constants in Young's inequality, its converse, and its generalization to more than three functions. *Advances in Math.*, 20(2):151–173, 1976. 3.4.2

[CMTV17] Michael B. Cohen, Aleksander Madry, Dimitris Tsipras, and Adrian Vladu. Matrix scaling and balancing via box constrained newton's method and interior point methods. *CoRR*, abs/1704.02310, 2017. 1.2.1

[Coh71] Paul Moritz Cohn. *Free rings and their relations.* Academic Press London, New York, 1971. 2.2.3, 2.17, 2.2.3

[Coh73] Paul Moritz Cohn. The word problem for free fields. *J. Symb. Log.*, 38(2):309–314, 1973. 1.1

[Coh75] Paul Moritz Cohn. The word problem for free fields: A correction and an addendum. *J. Symb. Log.*, 40(1):69–74, 1975. 1.1

[Coh95] Paul Moritz Cohn. *Skew Fields, Theory of General Division Rings.* Cambridge University Press, 1995. 2.34, 3.3.1, 3.3.1

[Coh06] Paul Cohn. *Free Ideal Rings and Localization in General Rings.* Cambridge University Press, 2006. 2.5

[CR99] Paul Moritz Cohn and Christophe Reutenauer. On the construction of the free field. *IJAC*, 9(3-4):307–324, 1999. 1.1, 2.12

[Der01] Harm Derksen. Polynomial bounds for rings of invariants. *Proceedings of the American Mathematical Society*, 129(4):955–963, 2001. 2.3.2, 2.25, 3.1, 3.3.2

[DGOS16] Z. Dvir, A. Garg, R. Oliveira, and J. Solymosi. Rank bounds for design matrices with block entries and geometric applications. *ArXiv e-prints*, October 2016. 3.4.3

[DM15] H. Derksen and V. Makam. Polynomial degree bounds for matrix semi-invariants. *ArXiv e-prints*, December 2015. 1.1, 1.2, 2.2, 2.3.2, 2.36

[DM16] Harm Derksen and Visu Makam. On non-commutative rank and tensor rank. *arXiv preprint arXiv:1606.06701*, 2016. 2.5, 2.5, 2.5

[DW00] Harm Derksen and Jerzy Weyman. Semi-invariants of quivers and saturation for littlewood-richardson coefficients. *Journal of the American Mathematical Society*, 13(3):467–479, 2000. 2.3.2

[DZ01] Mátyás Domokos and A. N. Zubkov. Semi-invariants of quivers as determinants. *Transformation Groups*, 6(1):9–24, 2001. 2.3.2

[Edm67] Jack Edmonds. Systems of distinct representatives and linear algebra. *J. Res. Nat. Bur. Standards Sect. B*, 71:241–245, 1967. 1.1, 1.2

[Fla62] H. Flanders. On spaces of linear transformations with bounded rank. *Journal of the London Mathematical Society*, s1-37(1):10–16, 1962. 3.3.1

[For02] Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *J. Comput. System Sci.*, 65(4):612–625, 2002. Special issue on complexity, 2001 (Chicago, IL). 3.4.3

[FR04] Marc Fortin and Christophe Reutenauer. Commutative/noncommutative rank of linear matrices and subspaces of matrices of low rank. In *Séminaire Lotharingien de Combinatoire*, volume 52, pages Art. B52f, 12, 7 2004. 7, 3.3.1

[FS13] Michael A. Forbes and Amir Shpilka. Explicit noether normalization for simultaneous conjugation via polynomial identity testing. *CoRR*, abs/1303.0084, 2013. 3.2.3

[GGOW16] Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 109–117, Oct 2016. (document), 1.1, 1.1, 1.3, 1.15, 1.19, 2.1, 1, 2.7, 2.1, 2.2.3, 3.2.3, 3.3.1, 3.3.2

[GGOW17] Ankit Garg, Leonid Gurvits, Rafael Oliveira, and Avi Wigderson. Algorithmic and optimization aspects of brascamp-lieb inequalities, via operator scaling. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 397–409. ACM, 2017. 3.4, 3.4.2, 3.4.2, 3.4.2

[GGRW02] I. Gelfand, S. Gelfand, V. Retakh, and R. Wilson. Quasideterminants. *http://arxiv.org/abs/math/0208146*, 2002. 1.1

[Gur04] Leonid Gurvits. Classical complexity and quantum entanglement. *J. Comput. Syst. Sci.*, 69(3):448–484, 2004. (document), 1.3, 1.16, 1.3, 2.1, 2.6, 2.7, 3.3.1, 3.10

[GY98] Leonid Gurvits and Peter N. Yianilos. The deflation-inflation method for certain semidefinite programming and maximum determinant completion problems (extended abstract). Technical report, NECI, 1998. 1.2.1, 1.8

[Hil93] David Hilbert. Ueber die vollen invariantensysteme. *Mathematische Annalen*, 42(3):313–373, Sep 1893. 2.3.2, 2.25, 3.1

[HM13] Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In Shai Shalev-Shwartz and Ingo Steinwart, editors, *COLT*, volume 30 of *JMLR Workshop and Conference Proceedings*, pages 354–375. JMLR.org, 2013. 3.4.3

[HW15] Pavel Hrubeš and Avi Wigderson. Non-commutative arithmetic circuits with division. *Theory of Computing*, 11:357–393, 2015. 2.2, 2.2.3, 2.18, 2.2.3, 2.3.2

[HY11] Pavel Hrubeš and Amir Yehudayoff. Arithmetic complexity in ring extensions. *Theory of Computing*, 7(8):119–129, 2011. 2.2

[Ide16] Martin Idel. A review of matrix scaling and Sinkhorn's normal form for matrices and positive maps. *CoRR*, abs/1609.06349, 2016. 1.2.1

[IQS15] Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive noncommutative rank computation in deterministic polynomial time over fields of arbitrary characteristics. *CoRR*, abs/1512.03531, 2015. 1.1, 1.1, 2.3.2, 6, 2.36, 3.3.1, 3.3.2, 3.3.2

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. 1.1

[KK96] Bahman Kalantari and Leonid Khachiyan. On the complexity of nonnegative-matrix scaling. *Linear Algebra and its Applications*, 240:87 – 103, 1996. 1.2.1

[Lie90] Elliott H. Lieb. Gaussian kernels have only Gaussian maximizers. *Invent. Math.*, 102(1):179–208, 1990. 3.4.2, 3.4.2, 3.17, 3.4.2

[LO15] J. M. Landsberg and Giorgio Ottaviani. New lower bounds for the border rank of matrix multiplication. *Theory of Computing*, 2015. 2.5, 2.5

[Lov79] László Lovász. On determinants, matchings, and random algorithms. In *FCT*, pages 565–574, 1979. 1.1

[LSW00] Nathan Linial, Alex Samorodnitsky, and Avi Wigderson. A deterministic strongly polynomial algorithm for matrix scaling and approximate permanents. *Combinatorica*, 20(4):545–568, Apr 2000. (document), 1.2.1, 1.2.1, 1.2.1

[Mal78] P. Malcolmson. A prime matrix ideal yields a skew field. *Journal of the London Mathematical Society*, 18(221-233), 1978. 2.2.3

[MR04] T. Mignon and N. Ressayre. A quadratic bound for the determinant and permanent problem. *International Mathematics Research Notices*, 2004. 4

[MS01] Ketan D. Mulmuley and Milind Sohoni. Geometric Complexity Theory I: An Approach to the P vs. NP and Related Problems. *SIAM Journal on Computing*, 31(2):496–526, 2001. 3.2.3

[Mum65] David Mumford. *Geometric invariant theory*. Ergebnisse der Mathematik und ihrer Grenzgebiete, Neue Folge, Band 34. Springer-Verlag, Berlin-New York, 1965. 3.2.3

[Nis91] Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC), May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418, 1991. 2.2

[Pop82] V L Popov. The constructive theory of invariants. *Mathematics of the USSR-Izvestiya*, 19(2):359, 1982. 2.3.2, 2.25

[Reu96] C. Reutenauer. Inversion height in free fields. *Selecta Mathematica*, 2(1):93–109, Sep 1996. 2.2.3

[RS05] Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *computational complexity*, 14(1):1–19, Apr 2005. 3.2.3

[SdB01] Aidan Schofield and Michel Van den Bergh. Semi-invariants of quivers for arbitrary dimension vectors. *Indagationes Mathematicae*, 12(1):125–138, 2001. 2.3.2

[Sin64] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964. (document), 1.2.1, 1.2.1

[Str73] Volker Strassen. Vermeidung von divisionen. *Journal für die reine und angewandte Mathematik*, 264:184–202, 1973. 2.2

[SY10] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010. 1.1

[Val79] L. G. Valiant. Completeness classes in algebra. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '79, pages 249–261, New York, NY, USA, 1979. ACM. (document), 1.1, 2.2.2, 2.8