How do exponential size solutions arise in semidefinite programming?

Gábor Pataki, Aleksandr Touzov *

June 30, 2021

Abstract

Semidefinite programs (SDPs) are some of the most popular and broadly applicable optimization problems to emerge in the last thirty years. A curious pathology of SDPs, illustrated by a classical example of Khachiyan, is that their solutions may need exponential space to even write down. Exponential size solutions are the main obstacle to solve a long standing open problem: can we decide feasibility of SDPs in polynomial time?

The consensus seems to be that large size solutions in SDPs are rare. Here we prove that they are actually quite common: a linear change of variables transforms every strictly feasible SDP into a Khachiyan type SDP, in which the leading variables are large. As to "how large", that depends on the singularity degree of a dual problem. Further, we present some SDPs in which large solutions appear naturally, without any change of variables. We also partially answer the question: how do we represent such large solutions in polynomial space?

Key words: semidefinite programming; exponential size solutions; Khachiyan's example; facial reduction; singularity degree

MSC 2010 subject classification: Primary: 90C22, 49N15; secondary: 52A40

OR/MS subject classification: Primary: convexity; secondary: programming-nonlinear-theory

1 Introduction

Contents

*Department of Statistics and Operations Research, University of North Carolina at Chapel Hill

Linear programs and polynomial size solutions The classical linear programming (LP) feasibility problem asks whether a system of linear inequalities

$$Ax \geq b$$

has a solution, where the matrix A and the vector b both have integer entries. When the answer is "yes", then by a classical argument a feasible rational x has size at most $2n^2 \log n$ times the size of (A, b), where n is the number of variables. When the answer is "no", there is a certificate of infeasibility whose size is similarly bounded. Here and in the sequel by "size" of a matrix or vector we mean the number of bits necessary to describe it.

Semidefinite programs and exponential size solutions Semidefinite programs (SDPs) are a far reaching generalization of linear programs, and they have attracted widespread attention in the last few decades. An SDP feasibility problem can be formulated as

$$x_1 A_1 + \cdots + x_m A_m + B \succeq 0, \tag{P}$$

where the A_i and B are symmetric matrices with integer entries and $S \succeq 0$ means that the symmetric matrix S is positive semidefinite.

In a stark contrast to a linear program, the solutions of (P) may have exponential size in the size of the input. This surprising fact is illustrated by a classical example of Khachiyan:

$$x_1 \geq x_2^2,\ x_2 \geq x_3^2, \ldots, x_{m-1} \geq x_m^2,\ x_m \geq 2. \tag{\textit{Khachiyan}}$$

We can formulate (*Khachiyan*) as an SDP, if we write its quadratic constraints as

$$\begin{pmatrix} x_i & x_{i+1} \\ x_{i+1} & 1 \end{pmatrix} \succeq 0 \text{ for } i = 1, \ldots, m-1. \tag{1.1}$$

We see that $x_1 \geq 2^{2^{m-1}}$, hence the size of x_1, and of any feasible solution of (*Khachiyan*), is at least 2^{m-1}.

We show the feasible set of (*Khachiyan*) with $m = 3$ on the left in Figure 1. For better visibility, we replaced the constraint $x_3 \geq 2$ by $2 \geq x_3 \geq 0$ and made x_3 increase from right to left.

Exponential size solutions in SDPs are mathematically intriguing, and greatly complicate approaches to the following fundamental open problem:

Can we decide feasibility of (P) in polynomial time?

Indeed, algorithms that ascertain feasibility of (P) in polynomial time must assume that a polynomial size solution actually exists: see a detailed exposition in [8]. In contrast, the algorithm in [23] that achieves the best known complexity bound for SDP feasibility uses results from the first order theory of reals [26], and is not polynomial.

We know of few papers that deal directly with the complexity of SDP. However, several works study the complexity of a related problem, optimizing a polynomial subject to polynomial inequality constraints. In fixed dimension some polynomial optimization problems are polynomial time solvable

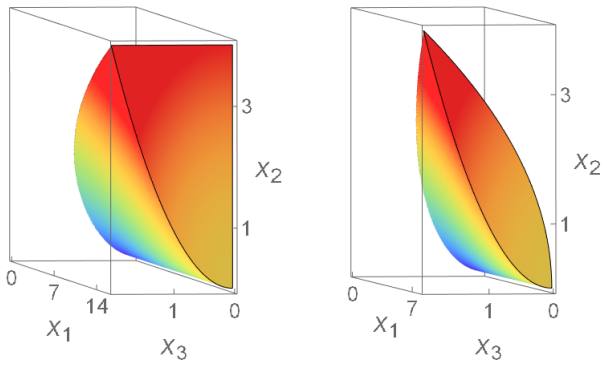


Figure 1: Feasible sets of (*Khachiyan*) (on the left) and of the quadratic inequalities (2.13) derived from (*Mild-SDP*) (on the right)

[4, 5, 3] and polynomial size solutions exist in special cases [30]. On the other hand, several fundamental problems in polynomial optimization are NP-hard, see for example, [5, 18, 1, 2].

Khachiyan's example naturally leads us to the following questions:

$$\text{Can we represent exponential size solutions of SDPs in polynomial space?} \tag{1.2}$$

$$\text{Are such large solutions common?} \tag{1.3}$$

To question (1.2) we have hope to get a "yes" answer. After all, to convince ourselves that $x_1 := 2^{2^{m-1}}$ (with a suitable $x_2, \ldots, x_m$) is feasible in (*Khachiyan*), we do not need to explicity write it down, a symbolic computation suffices. Still, question (1.2) seems to be open.

The answer to (1.3), however, seems to be a definite "no", for some of the following reasons:

- Exponential size solutions do not appear in typical SDPs in the literature.
- They can be eliminated even in (*Khachiyan*) by a fairly simple change. For example, let us add a new variable x_{m+1}, and change the last constraint to $x_m \geq 2 + x_{m+1}$; afterwards x_1 does not have to be large anymore.

  Or, let us replace x by Gx, where G is a random, dense matrix; afterwards (*Khachiyan*) will be quite messy, and will have no variables that are obviously larger than others.

Contributions It turns out that, despite these obstacles, we can still prove general results about large solutions in SDPs. One of our tools is the technique of facial reduction [6, 20, 21, 7, 32].

We assume that (*P*) has a *strictly feasible* solution x for which $\sum_{i=1}^{m} x_i A_i + B$ is positive definite. We fix a nonnegative integer parameter k, the *singularity degree* of a dual problem. We precisely define k soon, but for now we only need to know that $k \leq 1$ holds when (*P*) is a linear program. An informal version of our main result follows.

Informal Theorem 1 *After a linear change of variables $x \leftarrow Mx$, where M is a suitable invertible matrix, the leading k variables in strictly feasible solutions of (P) obey a Khachiyan type hierarchy. Namely, the inequalities*

$$x_1 \geq d_2 x_2^{\alpha_2},\ x_2 \geq d_3 x_3^{\alpha_3}, \ldots,\ x_{k-1} \geq d_k x_k^{\alpha_k} \tag{1.4}$$

hold, where

$$2 \geq \alpha_j \geq 1 + \frac{1}{k-j+1} \quad for j = 2, \ldots, k. \tag{1.5}$$

Here the d_j and α_j are positive constants that depend on the A_i, on B, and the last $m-k$ variables, that we consider fixed. □

Hence, if k is large and $x_k > 1$, then x_1 is larger than x_k.

How much larger? In the worst case, when $\alpha_j = 2$ for all j, like in (*Khachiyan*), x_1 is at least constant times $x_k^{2^{k-1}}$. In the best case, when $\alpha_j = 1 + \frac{1}{k-j+1}$ for all j, by an elementary calculation x_1 is at least constant times x_k^k. So even in this best case the magnitude of x_1 is exponentially larger than that of x_k.

Our assumptions are minimal. We assumed that (P) has a strictly feasible solution, and indeed there are semidefinite programs without strictly feasible solutions, with large singularity degree, and without large solutions: we discuss such an SDP after Example 1. Further, we need to focus on just a subset of variables and allow a linear change of variables [1]. Nevertheless, we show that in SDPs coming from minimizing a univariate polynomial, large variables appear naturally, without any change of variables.

We also partially answer the representation question (1.2). We show that in strictly feasible SDPs, after the change of variables $x \leftarrow Mx$, we can verify that a strictly feasible x exists, without even computing the values of the "large" variables $x_1, \ldots, x_k$. The same is true of SDPs coming from minimizing a univariate polynomial; in the latter SDPs we do not even need a change of variables.

Related work Linear programs can be solved in polynomial time, as it was first proved by Khachiyan [10]; see Grötschel, Lovász, and Schrijver [8] for an exposition that handles important details like the necessary accuracy. Other landmark polynomial time algorithms for linear programming were given by Karmarkar [9], Renegar [25], and Kojima et al [11].

On the other hand, to solve SDPs in polynomial time, one must assume that there is a polynomial size solution. We refer to [8] for an algorithm based on the ellipsoid method, and Nesterov and Nemirovskii [16] for foundational interior point methods to solve SDPs with an objective function. We also refer to Renegar [27] for a very clean exposition of interior point methods for convex optimization.

The complexity of SDP is closely related to the complexity of optimizing a polynomial subject to polynomial inequality constraints. To explain how, first consider a system of convex quadratic inequalities

$$x^\top Q_i x + b_i^\top x + c_i \leq 0 \ (i = 1, \ldots, m) \tag{1.6}$$

where the Q_i are fixed symmetric psd matrices, and $x \in \mathbb{R}^n$ is the vector of variables. The question whether we can decide feasibility of (1.6) in polynomial time is also fundamental, open, and, in a sense, easier than the question of deciding feasibility of (P) in polynomial time. The reason is that (1.6) can be represented as an instance of (P) by choosing suitable A_i and B matrices.

On the other hand, we can formulate semidefiniteness of a symmetric matrix variable by requiring the principal minors (which are polynomials) to be nonnegative. Among positive results in polynomial optimization, Bienstock [4] proved that such problems can be solved in polynomial time, if the number of constraints is fixed, the constraints and objective are quadratic, and at least one constraint is strictly convex. The work of [4] builds on Barvinok's fundamental result [3] that proved we can verify in polynomial time whether a system of a fixed number of quadratic equations is feasible. It also builds

[1] Since a change of variables, say $x \leftarrow Gx$ makes a mess even out of the nicely structured (*Khachiyan*), we may need to perform the inverse operation $x \leftarrow G^{-1}x$ to undo the mess.

on early work of Vavasis [30] which proved that a system with linear constraints and one quadratic constraint has a solution of polynomial size. In other important early work, Vavasis and Zippel [31] proved we can solve indefinite quadratic optimization problems with a ball constraint, in polynomial time.

On the flip side, there are many hardness results. For example, Bienstock, del Pia, and Hildebrand [5] proved it is NP-hard to test whether a system of quadratic inequalities has a polynomial size rational solution, even if we know that the system has a rational solution. Pardalos and Vavasis [18] proved the fundamental problem of minimizing a (nonconvex) quadratic function subject to linear constraints is also NP-hard. The following problems are also classical and NP-hard: testing convexity of a polynomial, see Ahmadi et al [1] (this problem was open since 1992); and testing whether a polynomial optimization problem attains its optimal value, see Ahmadi and Zhang [2].

One of the tools we use is an elementary facial reduction algorithm. These algorithms originated in the paper of Borwein and Wolkowicz [6], then simpler variants were given, for example, by Waki and Muramatsu [32] and in [20, 21]. For a recent comprehensive survey of facial reduction and its applications, see Drusvyatskiy and Wolkowicz [7].

In other related work, O' Donnell [17] presented an SDP that certifies nonnegativity of a polynomial via the sum-of-squares proof system, and is essentially equivalent to (*Khachiyan*). Previously it was thought that sum-of-squares proofs, a popular tool in theoretical computer science, can be found in polynomial time. However, due to this work, it has become clear that this is not obviously the case.

The plan of the paper In Subsection 1.1 we review preliminaries. In Subsection 2.1 we formally state Theorem 1 and illustrate it via two extreme examples. In Subsection 2.2 we prove it in a sequence of lemmas. In particular, in Lemma 5 we give a recursive formula, akin to a continued fractions formula, to compute the α_j exponents in (1.4). As an alternative, in Subsection 2.3 we show how to compute the α_j using the classical Fourier-Motzkin elimination for linear inequalities; this is an interesting contrast with SDPs being highly nonlinear. In Section 3 we cover the case of SDPs coming from polynomial optimization and also revisit the example from [17]. Section 4 concludes with a discusion.

Our proofs are fairly elementary. We use Proposition 1, a convex analysis argument about positive semidefinite matrices and linear subspaces, but other than that, we only rely on basic linear algebra, and on manipulating quadratic polynomials.

1.1 Notation and preliminaries

Matrices Given a matrix $M \in \mathbb{R}^{n\times n}$ and $R, S \subseteq \{1,\ldots,n\}$ we denote the submatrix of M corresponding to rows in R and columns in S by $M(R,S)$. We write $M(R)$ to abbreviate $M(R,R)$.

We let $\mathcal{S}^n$ be the set of $n\times n$ symmetric matrices and $\mathcal{S}^n_+$ be the set of $n\times n$ symmetric positive semidefinite (psd) matrices. The notation $S \succ 0$ means that the symmetric matrix S is positive definite. The inner product of symmetric matrices S and T is defined as $S \bullet T := \mathrm{trace}(ST)$.

Definition 1. *We say that $(C_1,\ldots,C_\ell)$ is a* regular facial reduction sequence for $\mathcal{S}^n_+$ *if each C_i is in $\mathcal{S}^n$ and of the form*

$$C_1 = \begin{pmatrix} \overbrace{I}^{r_1} & \overbrace{0}^{n-r_1} \\ 0 & 0 \end{pmatrix}, \ldots, C_i = \begin{pmatrix} \overbrace{\times}^{r_1+\ldots+r_{i-1}} & \overbrace{\times}^{r_i} & \overbrace{\times}^{n-r_1-\ldots-r_i} \\ \times & I & 0 \\ \times & 0 & 0 \end{pmatrix}$$

for $i = 1, \ldots, \ell$, where the r_i are nonnegative integers, and the $\times$ symbols correspond to blocks with arbitrary elements.

These sequences appear in facial reduction algorithms. The term "facial reduction" reflects the following: given a psd matrix Y which has zero $\bullet$ product with $C_1, \ldots, C_\ell$ we see that the first r_1 diagonal elements of Y are zero (by $C_1 \bullet Y = 0$), hence the first r_1 rows and columns are zero. Continuing, we deduce that the first $r_1 + \cdots + r_\ell$ rows and columns of Y are zero, so overall Y is reduced to live in a face of $\mathcal{S}^n_+$. [2]

Next we formalize what we mean by "replacing x by Mx for some invertible matrix M in (P)."

Definition 2. *We say that we* reformulate *(P) if we apply to it some of the following operations (in any order):*

(1) Exchange A_i and A_j, where i and j are distinct indices in $\{1, \ldots, m\}$.

(2) Replace A_i by $\lambda A_i + \mu A_j$, where λ and μ are reals, and $\lambda \neq 0$.

(3) Replace all A_i by $T^\top A_i T$ and B by $T^\top B T$, where T is a suitably chosen invertible matrix.

We also say that by reformulating (P) we obtain a reformulation.

We see that operations (1) and (2) amount to performing elementary row operations on a dual type system, say, on

$$A_i \bullet Y = 0 \text{ for } i = 1, \ldots, m.$$

Since operations (1) and (2) can be encoded by an invertible matrix, they amount to replacing the variable x by Mx in (P), where M is some invertible matrix. As to operation (3), it does not influence the magnitude of the x_i and we only use it to put (P) into a more convenient looking form.

Reformulations were previously used to study various pathologies in SDPs, for example, unattained optimal values and duality gaps [22]; and infeasibility [13]. In this work we show that they help us understand another classical pathology, exponential size solutions.

We will rely on the following statement about the connection of $\mathcal{S}^n_+$ and a linear subspace.

Proposition 1. *Suppose L is a linear subspace of $\mathcal{S}^n$. Then exactly one of the followig two alternatives is true:*

(1) There is a nonzero positive semidefinite matrix in L.

(2) There is a positive definite matrix in $L^\perp$.

2 Main results and proofs

2.1 Reformulating (P) and statement of Theorem 1

In our first lemma we present an algorithm to reformulate (P) into a more convenient looking form. The algorithm is a simplified version of the algorithm in [13].

[2] A convex subset F of $\mathcal{S}^n_+$ is a face, if for any $X, Y \in \mathcal{S}^n_+$ if the open line segment $\{\lambda X + (1-\lambda)Y : 0 < \lambda < 1\}$ intersects F, then both X and Y must be in F.

Lemma 1. *The problem (P) has a reformulation*

$$x_1A'_1 + \cdots + x_kA'_k + x_{k+1}A'_{k+1} + \cdots + x_mA'_m + B' \succeq 0 \qquad (P')$$

with the following properties:

- *k is a nonnegative integer, and $(A'_1, \ldots, A'_k)$ is a regular facial reduction sequence.*
- *If $r_1, \ldots, r_k$ is the size of the identity block in $A'_1, \ldots, A'_k$, respectively, then $n - r_1 - \cdots - r_k$ is the maximum rank of a matrix in*

$$\{\, Y \succeq 0 \,|\, A_i \bullet Y = 0 \text{ for } i = 1, \ldots, m \,\}. \qquad (2.7)$$

Proof Let L be the linear span of $A_1, \ldots, A_m$ and apply Proposition 1. If item (2) holds we let $k = 0$, $A'_i = A_i$ for all i, $B' = B$ and stop.

If item (1) holds, we choose a nonzero psd matrix $V = \sum_{i=1}^m \lambda_i A_i$ in L and assume $\lambda_1 \neq 0$ without loss of generality. We then choose a T invertible matrix so that

$$T^\top V T = \begin{pmatrix} I_{r_1} & 0 \\ 0 & 0 \end{pmatrix},$$

where r_1 is the rank of V. We let $A'_1 := T^\top V T, A'_i := T^\top A_i T$ for $i \geq 2$, and $B' = T^\top B T$.

Let r be the maximum rank of a psd matrix in $L^\perp$ (i.e., in (2.7)). Also, let L_{new} be the linear span of $A'_1, \ldots, A'_m$. We claim that r is also the maximum rank of a psd matrix in $L^\perp_{new}$. For that, suppose $Y \succeq 0$ is in $L^\perp$. Then

$$A_i \bullet Y \;=\; T^\top A_i T \bullet T^{-1} Y T^{-\top} \;=\; 0,$$

so $T^{-1}YT^{-\top}$ is in $L^\perp_{new}$ and has the same rank as Y. Similarly, from any psd matrix in $L^\perp_{new}$ we can construct a psd matrix in $L^\perp$ with the same rank. This proves our claim.

We see that if $Y \in L^\perp_{\rm new} \cap \mathcal{S}^n_+$ then $A'_1 \bullet Y = 0$ so the sum of the first r_1 diagonal elements of Y is zero, hence the first r_1 rows and columns of Y are zero.

We next construct an SDP

$$\sum_{i=2}^m x_i F_i + G \succeq 0,$$

where F_i is obtained from A'_i by deleting the first r_1 rows and columns for $i = 2, \ldots, m$ and G is obtained from B' in the same manner. By the above argument the maximum rank of a matrix in $\{Z \succeq 0 : F_i \bullet Z = 0\,(i = 2, \ldots, m)\}$ is also r, so we can proceed in a similar manner with this smaller SDP. When our process stops, we have the required reformulation. □

From now on we assume that

k is the smallest integer that satisfies the requirements of Lemma 1.

Using the terminology of facial reduction, k is the *singularity degree of* the system (2.7). This concept was originally introduced by Sturm in [29] and used to derive error bounds, namely, bounds on the distance of a point from the feasible set of an SDP. For a broad generalization of Sturm's result to conic systems over so-called amenable cones, see a recent result by Lourenço [14].

Definition 3. *We say that* $(\bar{x}_{k+1}, \ldots, \bar{x}_m)$ *is* partially strictly feasible *in* (P') *if there is* $(x_1, \ldots, x_k)$ *such that* $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ *is strictly feasible in it.*

For the rest of the paper we fix

$$\boxed{(\bar{x}_{k+1}, \ldots, \bar{x}_m) \text{ a partially strictly feasible solution in } (P').}$$

From now on we will say that a number is a *constant*, if it depends only on the $\bar{x}_i$, A_i and B. Theorem 1 will rely on such constants.

We now formally state our main result.

Theorem 1. *There is* $(x_1, \ldots, x_k)$ *such that* $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ *is strictly feasible in* (P') *and* x_k *is arbitrarily large.*

Further, if x_k *is sufficiently large and* $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ *is strictly feasible in* (P') *then*

$$x_j \quad \geq \quad d_{j+1} x_{j+1}^{\alpha_{j+1}} \text{ for } j = 1, \ldots, k-1, \tag{2.8}$$

where

$$2 \geq \alpha_{j+1} \geq 1 + \frac{1}{k-j} \quad \text{for} j = 1, \ldots, k-1. \tag{2.9}$$

Here the d_j *and* α_j *are positive constants.*

□

The proof of Theorem 1 has three main parts. First, in Lemma 2 we prove the first statement, that in strictly feasible solutions of (P') we can have arbitrarily large x_k. Lemma 3 is a technical statement about a certain parameter, the *tail-index* of the A'_j.

In the second part, Lemma 4 deduces from (P') a set of quadratic polynomial inequalities. These are typically "messy", namely they look like

$$(x_1 + x_2 + x_3)(x_4 + 10x_5) > (x_2 - 3x_4)^2.$$

Third, in Lemma 5 from these messy inequalities we first derive "cleaned up" versions, such as

$$x_1 x_4 > \text{constant}\, x_2^2,$$

then from these cleaned up inequalities we deduce the inequalities (2.8) and a recursive formula to compute the α_j. Next, Lemma 6 proves that the α_j exponents are a monotone function of the tail-index of the A'_j. Finally, Lemma 7 shows that a minimal tail-index gives the smallest possible exponent α_j. Combining all lemmas gives us Theorem 1.

Before we get to the proof, we illlustrate Theorem 1 via two extreme examples. Although Theorem 1 is about strictly feasible solutions, the examples are simple, and in all of them we just look at feasible solutions.

Example 1. *(Khachiyan SDP) Consider the SDP*

$$\begin{pmatrix} x_1 & & & & x_2 \\ & x_2 & & & x_3 \\ & & x_3 & & x_4 \\ & & & x_4 & \\ x_2 & x_3 & x_4 & & 1 \end{pmatrix} \succeq 0, \tag{Kh-SDP}$$

which can be written in the form of (P') *with the* A'_i *matrices given below and* B' *the matrix whose lower right corner is* 1 *and the remaining elements are zero:*

$$\underbrace{\begin{pmatrix} 1 & & & & & \\ & 0 & & & & \\ & & 0 & & & \\ & & & 0 & & \\ & & & & 0 & \\ & & & & & 0 \end{pmatrix}}_{A'_1}, \underbrace{\begin{pmatrix} 0 & & & & & 1 \\ & 1 & & & & \\ & & 0 & & & \\ & & & 0 & & \\ & & & & 0 & \\ 1 & & & & & 0 \end{pmatrix}}_{A'_2}, \underbrace{\begin{pmatrix} 0 & & & & & \\ & 0 & & & & 1 \\ & & 1 & & & \\ & & & 0 & & \\ & & & & 0 & \\ & 1 & & & & 0 \end{pmatrix}}_{A'_3}, \underbrace{\begin{pmatrix} 0 & & & & & \\ & 0 & & & & \\ & & 0 & & & 1 \\ & & & 1 & & \\ & & & & 0 & \\ & & 1 & & & 0 \end{pmatrix}}_{A'_4}.$$

The subdeterminants in (Kh-SDP) *with three red, three blue, and three green corners, respectively, give the inequalities*

$$x_1 \geq x_2^2,\ x_2 \geq x_3^2,\ x_3 \geq x_4^2 \tag{2.10}$$

that appear in ($Khachiyan$). *(For simplicity we left out the inequality* $x_4 \geq 2$*).*

We note in passing that the feasible sets of (Kh-SDP) and of the derived quadratic inequalities (2.10) are not equal. For example $x = (256, 16, 4, 2)$ is not feasible in (Kh-SDP), but is feasible in (2.10). However, we can easily construct an SDP that exactly represents ($Khachiyan$), as follows. We define a $(2m-1) \times (2m-1)$ matrix whose first $m-1$ order two principal minors are of the form (1.1) and the last order one principal minor represents the constraint $x_m \geq 2$. Permuting rows and columns puts this exact SDP into the regular form of (P').

We next note that the singularity degree of $\{Y \succeq 0 : A'_i \bullet Y = 0 \text{ for } i = 1, \ldots, 4\}$ is four. Indeed, consider a regular facial reduction sequence, say $\hat{A}_1, \hat{A}_2, \ldots$ whose members are in the linear span of the A'_i. Suppose without loss of generality that $\hat{A}_1 \neq 0$. Then $\hat{A}_1 = A'_1$, since A'_1 is the only nonzero psd matrix in the linear span of the A'_i. Similarly, asume without loss of generality that the lower right 3×3 block of $\hat{A}_2$ is nonzero. Then $\hat{A}_2 = A'_2$, and so on.

We finally discuss whether we need to assume that a strictly feasible solution exists, in order to derive Theorem 1. On the one hand, there are semidefinite programs which have no strictly feasible solutions, nor do they exhibit the hierarchy among the leading variables seen in (1.5). Suppose indeed that in (Kh-SDP) we change x_1 to $x_1 + 1$ and the 1 entry in the bottom right corner to 0. This change does not affect the parameter k. Further, the new SDP is no longer strictly feasible, and $x_2 = x_3 = x_4 = 0$ holds in any feasible solution, but x_1 can be -1 [3].

On the other hand, there are SDPs with no strictly feasible solution, which, however, have large size solutions: to produce such a problem, we simply take any SDP that has large solutions, and add all-zero rows and columns.

Example 2. *(Mild SDP) As a counterpoint to* (Kh-SDP) *we next consider a mild SDP (we will see soon why we call it "mild")*

$$\begin{pmatrix} \boldsymbol{x_1} & & \boldsymbol{x_2} & & \\ & \boldsymbol{x_2} & & \boldsymbol{x_3} & \\ \boldsymbol{x_2} & & \boldsymbol{x_3} & & \boldsymbol{x_4} \\ & \boldsymbol{x_3} & & \mathbf{x_4} & \\ & & \boldsymbol{x_4} & & \mathbf{1} \end{pmatrix} \succeq 0. \tag{$Mild$-SDP}$$

We write ($Mild$-SDP) *in the form of* (P') *with the* A'_i *matrices shown below and* B' *the matrix whose*

[3] We can similarly create such an SDP with any number of variables.

lower right corner is 1 *and the remaining elements are zero:*

$$\underbrace{\begin{pmatrix} 1 & & & & & \\ & 0 & & & & \\ & & 0 & & & \\ & & & 0 & & \\ & & & & 0 & \\ & & & & & 0 \end{pmatrix}}_{A'_1}, \underbrace{\begin{pmatrix} 0 & & 1 & & & \\ & 1 & & & & \\ 1 & & 0 & & & \\ & & & 0 & & \\ & & & & 0 & \\ & & & & & 0 \end{pmatrix}}_{A'_2}, \underbrace{\begin{pmatrix} 0 & & & & & \\ & 0 & & 1 & & \\ & & 1 & & & \\ & 1 & & 0 & & \\ & & & & 0 & \\ & & & & & 0 \end{pmatrix}}_{A'_3}, \underbrace{\begin{pmatrix} 0 & & & & & \\ & 0 & & & & \\ & & 0 & & 1 & \\ & & & 1 & & \\ & & 1 & & 0 & \\ & & & & & 0 \end{pmatrix}}_{A'_4}.$$

In (Mild-SDP) the subdeterminants with three red, three blue, and three green corners, respectively, yield the inequalities

$$x_1 x_3 \geq x_2^2,\ x_2 x_4 \geq x_3^2,\ x_3 \geq x_4^2. \tag{2.11}$$

Next from (2.11) we derive the inequalities

$$x_1 \geq x_2^{4/3},\ x_2 \geq x_3^{3/2},\ x_3 \geq x_4^2 \tag{2.12}$$

as follows. The last inequality $x_3 \geq x_4^2$ is copied from (2.11) to (2.12) only for completeness. Next we plug $x_3^{1/2} \geq x_4$ into the middle inequality in (2.11) to get $x_2 \geq x_3^{3/2}$. We finally use this last inequality in the first one in (2.11) and deduce $x_1 \geq x_2^{4/3}$.

To summarize, the exponents in the derived inequalities (2.12) are the smallest permitted by our bounds (2.9).

To illustrate the difference between (*Khachiyan*) and the inequalities derived from (*Mild-SDP*), we show the set defined by the inequalities

$$x_1 x_3 \geq x_2^2,\ x_2 \geq x_3^2,\ 2 \geq x_3 \geq 0 \tag{2.13}$$

on the right in Figure 1. Note that the set defined by (2.13) is a three dimensional version of the set given in (2.11), normalized by adding upper and lower bounds on x_3.

2.2 Proof of Theorem 1

In Lemmas 2–4 we will use the following notation:

$$\begin{array}{rcl} r_j & = & \text{size of the identity block in } A'_j \text{ for } j = 1, \ldots, k, \\ \mathcal{I}_1 & := & \{1, \ldots, r_1\}, \\ \mathcal{I}_2 & := & \{r_1 + 1, \ldots, r_1 + r_2\}, \\ & \vdots & \\ \mathcal{I}_k & := & \{r_1 + \cdots + r_{k-1} + 1, \ldots, r_1 + \cdots + r_k\}, \\ \mathcal{I}_{k+1} & := & \{r_1 + \cdots + r_k + 1, \ldots, n\}. \end{array} \tag{2.14}$$

Lemma 2. *There is $(x_1, \ldots, x_k)$ such that x_k is arbitrarily large, and $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P').*

Proof Let

$$Z := \sum_{i=k+1}^{m} \bar{x}_i A'_i + B'.$$

Since there is $x_1, \ldots, x_k$ such that $\sum_{i=1}^{k} x_i A_i' + Z \succ 0$, and $A_i'(\mathcal{I}_{k+1}) = 0$ for $i = 1, \ldots, k$ we see that

$$Z(\mathcal{I}_{k+1}) \succ 0.$$

By the definition of positive definiteness (G is positive definite if $x^\top G x > 0$ for all nonzero x), and by the shape of A_k', we see that the $\mathcal{I}_k \cup \mathcal{I}_{k+1}$ diagonal block of $x_k A_k' + Z$ is positive definite when x_k is large enough. For any such x_k there is x_{k-1} so the $\mathcal{I}_{k-1} \cup \mathcal{I}_k \cup \mathcal{I}_{k+1}$ diagonal block of $x_{k-1} A_{k-1}' + x_k A_k' + Z$ is positive definite. We construct $x_{k-2}, \ldots, x_1$ in a similar manner. □

The proof of Lemma 2 partially answers the representation question (1.2). In particular, for the moment let us ignore the requirement that we need to choose x_k to be large and just focus on completing $(\bar{x}_{k+1}, \ldots, \bar{x}_m)$ to a strictly feasible solution. The proof that the required $(x_1, \ldots, x_k)$ *could be* computed is fairly simple, and it is illustrated on Figure 2, where the red blocks stand for the larger and larger blocks that we make positive definite. So we can convince ourselves that $(x_1, \ldots, x_k)$ *exist*, even without computing their actual values.

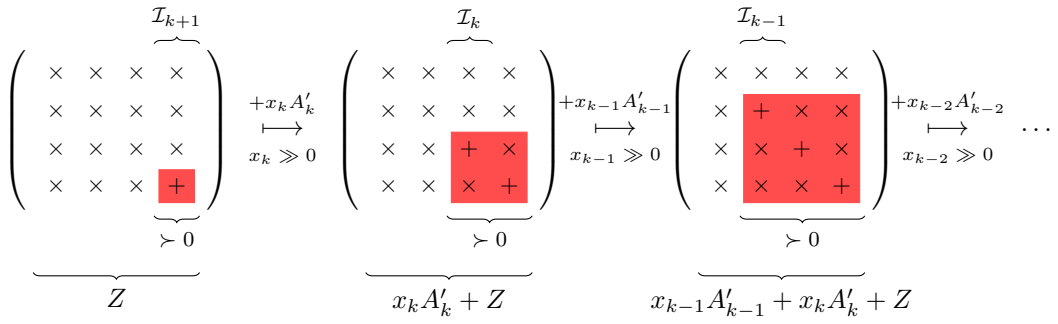


Figure 2: Verifying that $x_1, \ldots, x_k$ exist, without computing them

From now on we will assume

$$r_1 + \cdots + r_k < n, \tag{2.15}$$

and we claim that we can do so without loss of generality. Indeed, suppose that the sum of the r_j is n. Then an argument like in the proof of Lemma 2 proves that $A_1', \ldots, A_k'$ have a positive definite linear combination. Hence the singularity degree of (2.7) is actually just 1, so Theorem 1 holds vacuously.

By (2.15) we see that $\mathcal{I}_{k+1} \neq \emptyset$.

To motivate our next definition we compare our two extreme examples from two viewpoints. From the first viewpoint we see that in (*Kh-SDP*) the x_j variables in the offdiagonal positions are more to the right than in (*Mild-SDP*). From the second viewpoint, in the inequalities (2.10) derived from (*Kh-SDP*) the exponents are larger than in the inequalities (2.12) derived from (*Mild-SDP*). We will see that these two facts are closely connected, so in the next definition we capture "how far to the right the x_j are in off-diagonal positions."

Definition 4. *The* tail-index *of* A_{j+1}' *is*

$$t_{j+1} := \max\{\, t \,:\, A_{j+1}'(\mathcal{I}_j, \mathcal{I}_t) \neq 0\} \text{ for } j = 1, \ldots, k-1. \tag{2.16}$$

In words, t_{j+1} is the index of the rightmost nonzero block of columns "directly above" the identity block in A_{j+1}'. We illustrate the tail-index on Figure 3. Here and in later figures the $\bullet$ blocks are nonzero, and we separate the columns indexed by $\mathcal{I}_{k+1}$ from the other columns by double vertical lines.

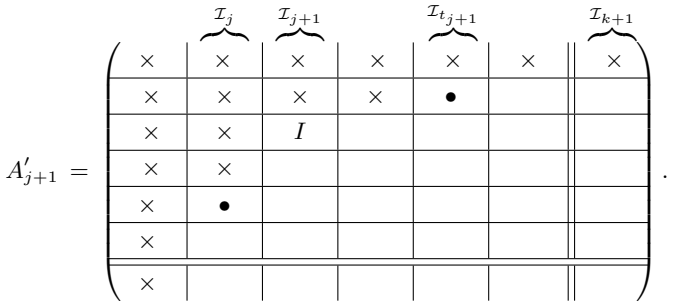


Figure 3: The tail-index of A'_{j+1}

Continuing our examples, we see that $t_2 = t_3 = t_4 = 5$ in (*Kh-SDP*), whereas $t_2 = 3$, $t_3 = 4$, $t_4 = 5$ in (*Mild-SDP*).

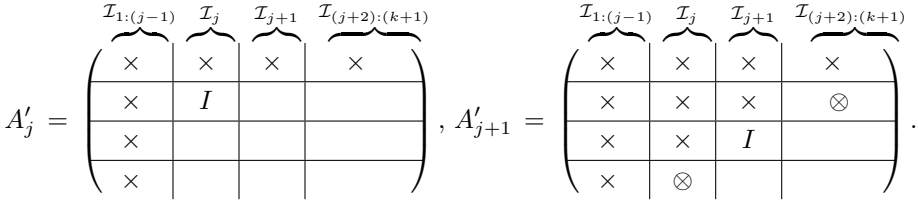
Lemma 3.

$$t_{j+1} > j + 1 \text{ for } j = 1, \ldots, k-1.$$

Proof We will use the following notation: for $r, s \in \{1, \ldots, k+1\}$ such that $r \leq s$ we let

$$\mathcal{I}_{r:s} := \mathcal{I}_r \cup \cdots \cup \mathcal{I}_s. \tag{2.17}$$

Let $j \in \{1, \ldots, k-1\}$ be arbitrary. To help with the proof, we picture A'_j and A'_{j+1} in equation (2.18). As always, the empty blocks are zero, and the $\times$ blocks are arbitrary. The blocks marked by $\otimes$ are $A'_{j+1}(\mathcal{I}_j, \mathcal{I}_{(j+2):(k+1)})$ and its symmetric counterpart. We will prove that these blocks are nonzero.

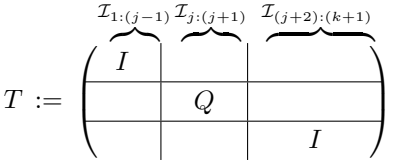


(2.18)

For that, suppose the $\otimes$ blocks are zero and let $A'_j := \lambda A'_j + A'_{j+1}$ for some large $\lambda > 0$. Then by the definition of positive definiteness (G is positive definite if $x^T G x > 0$ for all nonzero x) we find

$$A'_j(\mathcal{I}_{j:(j+1)}) \succ 0.$$

Let Q be a matrix of suitable scaled eigenvectors of $A'_j(\mathcal{I}_{j:(j+1)})$, define



and let

$$A'_i := T^\top A_i T \text{ for } i = 1, \ldots, j, j+2, \ldots, k.$$

Then an elementary calculation shows that $(A'_1, \ldots, A'_j, A'_{j+2}, \ldots, A'_k)$ is a length $k-1$ regular facial reduction sequence that satisfies the requirements of Lemma 1. However, we assumed that the shortest such sequence has length k. This contradiction completes the proof. □

In Lemma 4 we construct a sequence of polynomial inequalities that must be satisfied by any $(x_1, \ldots, x_k)$ that complete $(\bar{x}_{k+1}, \ldots, \bar{x}_m)$ to a strictly feasible solution. We need some more notation. Given a strictly feasible solution

$$(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$$

we will write δ_j for an affine combination of the "x" and "$\bar{x}$" terms with indices larger than j, in other words,

$$\delta_j = \gamma_{j+1}x_{j+1} + \cdots + \gamma_k x_k + \gamma_{k+1}\bar{x}_{k+1} + \cdots + \gamma_m \bar{x}_m + \gamma_{m+1} \tag{2.19}$$

where the γ_i are constants.

We will actually slightly abuse this notation. We will write δ_j more than once, but we may mean a different affine combination each time. For example, if $k = m = 4$, then we may write $\delta_2 = 2x_3 + 3x_4 + 5$ on one line, and $\delta_2 = x_3 - 2x_4 - 3$ on another. Given that $\bar{x}_{k+1}, \ldots, \bar{x}_m$ are fixed, δ_k will always denote a constant.

Lemma 4. *Suppose that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P'). Then*

$$p_j(x_1, \ldots, x_k) > 0 \text{ for } j = 1, \ldots, k-1, \tag{2.20}$$

for some p_j polynomials defined as follows:

- *if $t_{j+1} \leq k$, then we choose p_j as*

$$p_j(x_1, \ldots, x_k) = (x_j + \delta_j)(x_{t_{j+1}} + \delta_{t_{j+1}}) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2 \tag{2.21}$$

  where β_{j+1} is a nonzero constant. In this case we call p_j a type 1 polynomial.

- *if $t_{j+1} = k+1$, then we choose p_j as*

$$p_j(x_1, \ldots, x_k) = (x_j + \delta_j) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2, \tag{2.22}$$

  where β_{j+1} is a nonzero constant. In this case we call p_j a type 2 polynomial.

□

Before we prove it, we discuss Lemma 4. First we note that by Lemma 3 we have $t_k = k+1$ so p_{k-1} is always type 2.

In Khachiyan's example (*Khachiyan*) all inequalities come from type 2 polynomials, namely from $x_j - x_{j+1}^2$ for $j = 1, \ldots, k-1$. In contrast, among the inequalities (2.11) derived from (*Mild-SDP*) the first two come from type 1 polynomials and the last one from a type 2 polynomial.

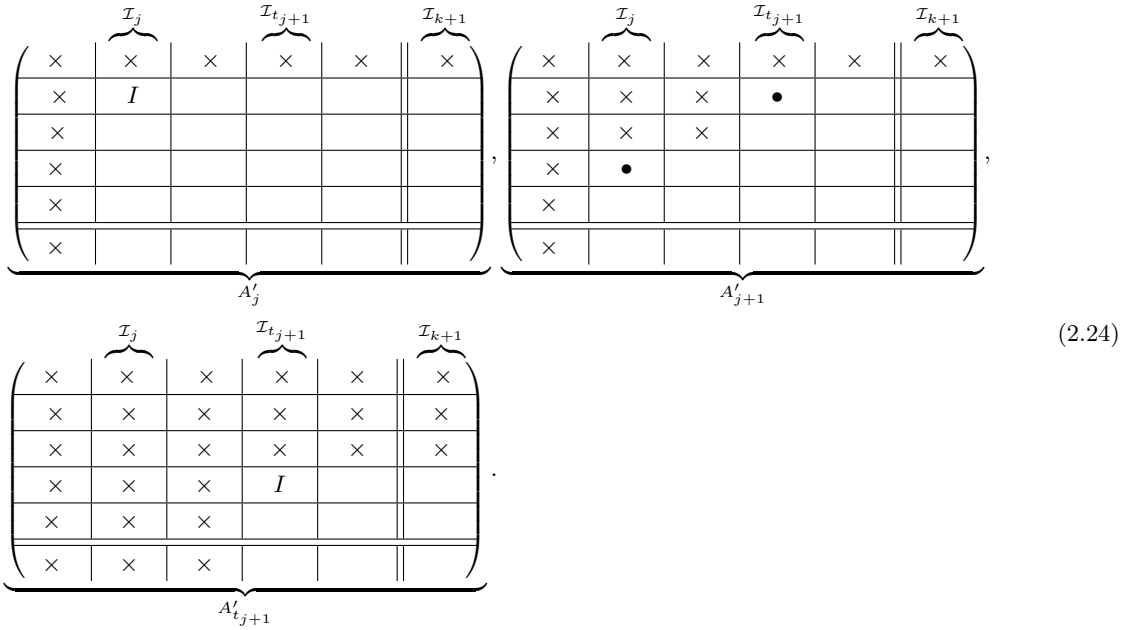
Proof of Lemma 4 Fix $j \in \{1, \ldots, k-1\}$. Let $\ell_1 \in \mathcal{I}_j$ and $\ell_2 \in \mathcal{I}_{t_{j+1}}$ such that $(A'_{j+1})_{\ell_1,\ell_2} \neq 0$.

As stated, suppose that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P'). For brevity, define

$$S := \sum_{i=1}^{k} x_i A'_i + \sum_{i=k+1}^{m} \bar{x}_i A'_i + B'. \tag{2.23}$$

We distinguish two cases.

Case 1: Suppose $t_{j+1} \leq k$. Below we show the matrices that will be important in defining p_j :



(2.24)

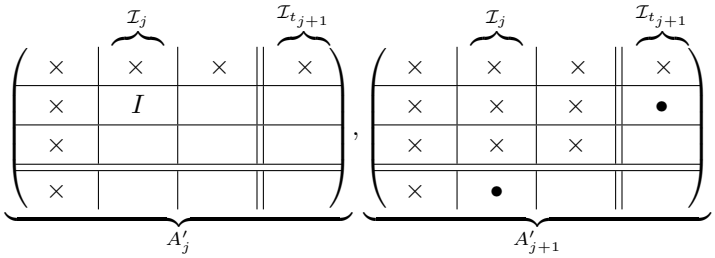
As usual, the empty blocks are zero, the $\times$ blocks may have arbitrary elements, and the $\bullet$ block is nonzero. (More precisely, $A'_{j+1}(\mathcal{I}_{j+1}) = I$, but we do not indicate this in equation (2.24), since the other entries suffice to derive the p_j polynomial.)

Define $\beta_{j+1} := (A'_{j+1})_{\ell_1,\ell_2}$. Let S' be the submatrix of S that contains rows and columns indexed by ℓ_1 and ℓ_2, then

$$S' = \begin{pmatrix} x_j + \delta_j & \beta_{j+1}x_{j+1} + \delta_{j+1} \\ \beta_{j+1}x_{j+1} + \delta_{j+1} & x_{t_{j+1}} + \delta_{t_{j+1}} \end{pmatrix}.$$

We define $p_j(x_1, \ldots, x_k)$ as the determinant of S', then p_j is a type 1 polynomial in the form (2.21). Since $S' \succ 0$, we see that $p_j(x_1, \ldots, x_k) > 0$ and the proof in this case is complete.

Case 2: Suppose $t_{j+1} = k + 1$. Now p_j will mainly depend on two matrices that we show below:



(2.25)

Again, the $\bullet$ blocks are nonzero. We again let S' be the submatrix of S that contains rows and columns indexed by ℓ_1 and ℓ_2.

Define $\lambda := (A'_{j+1})_{\ell_1,\ell_2}$, $\mu := S'_{\ell_2,\ell_2}$. Observe that μ depends only on $\bar{x}_{k+1}, \ldots, \bar{x}_m$, the A'_i and B', in other words it is a constant. Then S' looks like

$$S' = \begin{pmatrix} x_j + \delta_j & \lambda x_{j+1} + \delta_{j+1} \\ \lambda x_{j+1} + \delta_{j+1} & \mu \end{pmatrix}.$$

Define

$$p_j(x_1, \ldots, x_k) := \frac{1}{\mu} \det S'.$$

Since $S' \succ 0$ we have $\mu > 0$. So $p_j(x_1, \ldots, x_k) > 0$ and $p_j(x_1, \ldots, x_k)$ is a type 2 polynomial in the form required in (2.22) with $\beta_{j+1} = \lambda/\sqrt{\mu}$ (note that according to our definition of δ_j, if we divide it by a constant, the result is still δ_j). The proof in this case is now complete.

□

Lemma 5. *Suppose that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P') and x_k is sufficiently large. Then*

$$x_j \geq d_{j+1} x_{j+1}^{\alpha_{j+1}} \text{ for } j = 1, \ldots, k-1, \tag{2.26}$$

where the d_{j+1} are positive constants and the α_{j+1} can be computed by the recursion

$$\alpha_{j+1} = \begin{cases} 2 - \dfrac{1}{\alpha_{j+2} \ldots \alpha_{t_{j+1}}} & \text{if } t_{j+1} \leq k \\ 2 & \text{if } t_{j+1} = k+1 \end{cases} \tag{2.27}$$

for $j = 1, \ldots, k-1$.

□

Before proving it, we discuss Lemma 5. We have $t_k = k+1$ (by Lemma 3) hence Lemma 5 implies $\alpha_k = 2$. Hence, by induction the recursion (2.27) implies that $\alpha_j \in (1, 2]$ holds for all j. Thus, if x_k is large enough, then $x_j > 0$ for $j = 1, \ldots, k$.

It is also interesting to note that formula (2.27) is reminiscent of a continued fractions formula.

To illustrate Lemma 5 we show how from (*Mild-SDP*) we can deduce the inequalities (2.12) much more quickly than we did before. Precisely, we compute the exponents by the recursion (2.27) as

$$\begin{array}{lcll} \alpha_4 & = & 2 & (\text{since } t_4 = 5) \\ \alpha_3 & = & 2 - 1/\alpha_4 \; = \; 3/2 & (\text{since } t_3 = 4) \\ \alpha_2 & = & 2 - 1/\alpha_3 \; = \; 4/3 & (\text{since } t_2 = 3). \end{array} \tag{2.28}$$

The proof of Lemma 5 has two ingredients. First, from the messy looking type 1 inequalities (2.21) we deduce cleaned up versions

$$x_j x_{t_{j+1}} \geq \text{constant} \, x_{j+1}^2,$$

and we similarly clean up the type 2 inequalities (2.22). Then from the cleaned up inequalities we derive the required inequalities (2.26) and the recursion (2.27).

Since the proof of Lemma 5 is somewhat technical, we illustrate the cleaning up step with an example.

Example 3. *(Perturbed Khachiyan) Consider the SDP*

$$\left(\begin{array}{c|c|c|c} x_1 - 2x_2 & & & x_2 - x_3 \\ \hline & x_2 + x_3 & & x_3 \\ \hline & & x_3 & \\ \hline x_2 - x_3 & x_3 & & 1 \end{array}\right) \succeq 0. \tag{2.29}$$

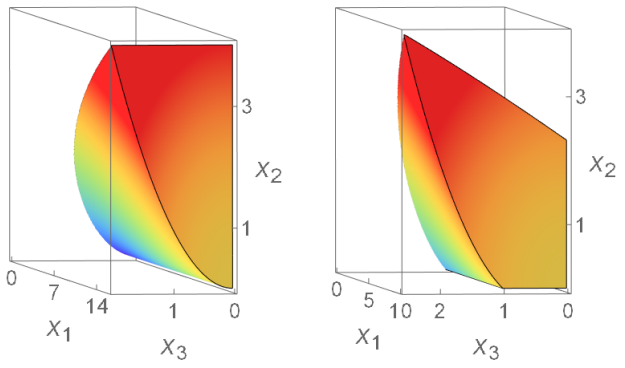


Figure 4: Feasible sets of (*Khachiyan*) (on left) and of inequalities derived from the perturbed Khachiyan SDP (2.29) (on the right)

From its principal minors we deduce the inequalities

$$x_1 - 2x_2 \geq (x_2 - x_3)^2 \tag{2.30}$$
$$x_2 + x_3 \geq x_3^2. \tag{2.31}$$

These two inequalities are a "perturbed" version of the inequalities in (Khachiyan), since we obtain them by replacing x_1 by $x_1 - 2x_2$ and x_2 by $x_2 \pm x_3$.

Suppose (x_1, x_2, x_3) is feasible in (2.29). Then the inequalities $x_1 \geq x_2^2$ and $x_2 \geq x_3^2$ no longer hold. However, assuming $x_3 \geq 10$, we claim that the following inequalities do:

$$x_1 \geq \frac{1}{2}x_2^2 \tag{2.32}$$
$$x_2 \geq \frac{1}{2}x_3^2. \tag{2.33}$$

Indeed, (2.33) follows from (2.31) and $x_3 \geq 10$ directly. Using (2.33) we see that x_3 is lower order than x_2. A straightforward calculation shows

$$(x_2 - x_3)^2 \geq \frac{1}{2}x_2^2. \tag{2.34}$$

Hence from (2.30) we get

$$\begin{aligned} 0 &\leq x_1 - 2x_2 - (x_2 - x_3)^2 \\ &\leq x_1 - (x_2 - x_3)^2 \\ &\leq x_1 - \frac{1}{2}x_2^2, \end{aligned} \tag{2.35}$$

where the last inequality follows from using (2.34). Thus (2.32) follows, and the proof is complete.

We show the feasible set of (*Khachiyan*) (when $m = 3$) and the feasible set described by the inequalities (2.30) on Figure 4. From (*Khachiyan*) we removed the inequality $x_3 \geq 2$ and we normalized both sets by suitable bounds on x_3. Note that x_3 increases from right to left for better visibility.

Proof of Lemma 5 We use an argument analogous to the one in Example 3. We use induction and show how to suppress the "δ" terms in the type 1 and type 2 polynomials at the cost of making x_k large and choosing suitable d_j constants.

Suppose that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P'). Then by Lemma 4 the inequalities $p_j(x_1, \ldots, x_k) > 0$ hold for $j = 1, \ldots, k-1$.

When $j = k-1$ we have $p_{k-1}(x_1, \ldots, x_k) = (x_{k-1} + \delta_{k-1}) - (\beta_k x_k + \delta_k)^2 > 0$. Using the definition of δ_{k-1} we get

$$(x_{k-1} + \gamma_k x_k + \delta_k) - (\beta_k x_k + \delta_k)^2 > 0,$$

where γ_k is a constant (possibly zero) and $\beta_k \neq 0$. Then for a suitable positive d_k we have $x_{k-1} \geq d_k x_k^2$ if x_k is sufficiently large.

For the inductive step we will adapt the O, Θ and o notation from theoretical computer science. Given functions $f, g : \mathbb{R}^k \to \mathbb{R}_+$ we say that

(1) $f = O(g)$ (in words, f is big-Oh of g) if there are positive constants C_1 and C_2 such that for all $(x_1, \ldots, x_k)$ such that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P') and $x_k \geq C_1$ we have

$$f(x_1, \ldots, x_k) \leq C_2 g(x_1, \ldots, x_k).$$

(2) $f = \Theta(g)$ (in words, f is big-Theta of g) if $f = O(g)$ and $g = O(f)$.

(3) $f = o(g)$ (in words, f is little-oh of g) if for all $\epsilon > 0$ there is $\delta > 0$ such that if $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P') and $x_k \geq \delta$ then

$$f(x_1, \ldots, x_k) \leq \epsilon g(x_1, \ldots, x_k).$$

The usual calculus of O, Θ and o carries over verbatim. For example

$$\begin{array}{rcl} f = o(g) & \Rightarrow & f = O(g) \\ f = O(g) \text{ and } h = O(g) & \Rightarrow & f + h = O(g). \end{array} \tag{2.36}$$

Suppose next that $j + 1 \leq k - 1$ and we have proved the following: there are positive constants $d_{j+1}, \ldots, d_k$ and $\alpha_{j+2}, \ldots, \alpha_k$ derived from the recursion (2.27) such that the inequalities

$$\begin{array}{rcl} x_{j+1} & \geq & d_{j+2} x_{j+2}^{\alpha_{j+2}} \\ x_{j+2} & \geq & d_{j+3} x_{j+3}^{\alpha_{j+3}} \\ & \vdots & \\ x_{k-1} & \geq & d_k x_k^{\alpha_k} \end{array} \tag{2.37}$$

hold for all $x_1, \ldots, x_k$ such that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P') and x_k is large enough.

We will construct positive constants $d_{j+1},$ and α_{j+1} according to the recursion (2.27) such that

$$x_j \geq d_{j+1} x_{j+1}^{\alpha_{j+1}} \tag{2.38}$$

holds for all $x_1, \ldots, x_k$ such that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P') and x_k is large enough.

We first observe that the recursion (2.27) implies $\alpha_{j+2}, \ldots, \alpha_k \in (1, 2]$, so by the inequalities (2.37) we have

$$x_s = o(x_\ell) \text{ when } s > \ell > j. \tag{2.39}$$

Assume that $(x_1, \ldots, x_k, \bar{x}_{k+1}, \ldots, \bar{x}_m)$ is strictly feasible in (P'). We distinguish two cases.

Case 1: First suppose that $t_{j+1} \leq k$, in other words, the quadratic polynomial p_j is type 1 (see Lemma 4).

Then the inequality $p_j(x_1, \ldots, x_k) > 0$ implies

$$\begin{aligned} 0 &< (x_j + \delta_j)(x_{t_{j+1}} + \delta_{t_{j+1}}) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2 \\ &= (x_j + \gamma_{j+1}x_{j+1} + \delta_{j+1})(x_{t_{j+1}} + \delta_{t_{j+1}}) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2 \\ &= (x_j + \gamma_{j+1}x_{j+1} + o(x_{j+1}))(x_{t_{j+1}} + o(x_{t_{j+1}})) - (\beta_{j+1}x_{j+1} + o(x_{j+1}))^2 \\ &\leq (x_j + \Theta(x_{j+1}))\Theta(x_{t_{j+1}}) - \Theta(x_{j+1})^2 \end{aligned} \tag{2.40}$$

where γ_{j+1} is a constant (which may be zero). The first equality follows from the definition of δ_j (see 2.19). The second equality follows since by (2.39) and by $t_{j+1} > j+1$ we have

$$|\delta_{j+1}| = o(x_{j+1}),\ |\delta_{t_{j+1}}| = o(x_{t_{j+1}}). \tag{2.41}$$

The last inequality follows from $\beta_{j+1} \neq 0$ and the calculus rules (2.36). We now continue (2.40):

$$\begin{aligned} 0 &< (x_j + \Theta(x_{j+1}))\Theta(x_{t_{j+1}}) - \Theta(x_{j+1})^2 \\ &= x_j\Theta(x_{t_{j+1}}) + \Theta(x_{j+1}x_{t_{j+1}}) - \Theta(x_{j+1})^2 \\ &\leq x_j\Theta(x_{t_{j+1}}) + o(x_{j+1}^2) - \Theta(x_{j+1})^2 \\ &\leq x_j\Theta(x_{t_{j+1}}) - \Theta(x_{j+1})^2, \end{aligned} \tag{2.42}$$

where the second inequality follows, since $t_{j+1} > j+1$ hence by (2.39) we have $x_{t_{j+1}} = o(x_{j+1})$. The last inequality follows from the calculus rules (2.36).

Next from the inequalities (2.37), using $t_{j+1} > j+1$ we learn that

$$x_{t_{j+1}}^{\alpha} = O(x_{j+1})$$

where $\alpha = \alpha_{j+2}\alpha_{j+3}\ldots\alpha_{t_{j+1}}$. Hence $x_{t_{j+1}} = O(x_{j+1}^{1/\alpha})$.

We plug this last estimate into the last inequality in (2.42) and deduce

$$0 < x_j\Theta(x_{j+1}^{1/\alpha}) - \Theta(x_{j+1}^2).$$

Dividing by $x_{j+1}^{1/\alpha}$ and a constant, we see that $\alpha_{j+1} := 2 - 1/\alpha$ satisfies the recursion (2.27), as required.

Case 2 Suppose that $t_{j+1} = k+1$, in other words, the quadratic polynomial p_j is type 2. Then

$$\begin{aligned} p_j(x_1, \ldots, x_k) &= (x_j + \delta_j) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2 \\ &= (x_j + \gamma_{j+1}x_{j+1} + \delta_{j+1}) - (\beta_{j+1}x_{j+1} + \delta_{j+1})^2 \end{aligned}$$

for some $\beta_{j+1} \neq 0$ and γ_{j+1} constants, where γ_{j+1} may be zero. By (2.39) we have $\delta_{j+1} = o(x_{j+1})$, hence

$$x_j \geq \Theta(x_{j+1}^2).$$

So we can set $\alpha_{j+1} = 2$, thereby completing the proof. □

As a prelude to Lemma 6, in Figure 5 we show three SDPs (for brevity we left out the $\succeq$ symbols). The first is (*Mild-SDP*). The second and third arise from it by shifting x_2 in the offdiagonal position to the right. Underneath we show the vector of the $\alpha = (\alpha_2, \alpha_3, \alpha_4)$ exponents in the inequalities derived by the recursion (2.27).

We see that α_2 increases from left to right and Lemma 6 presents a general result of this kind.

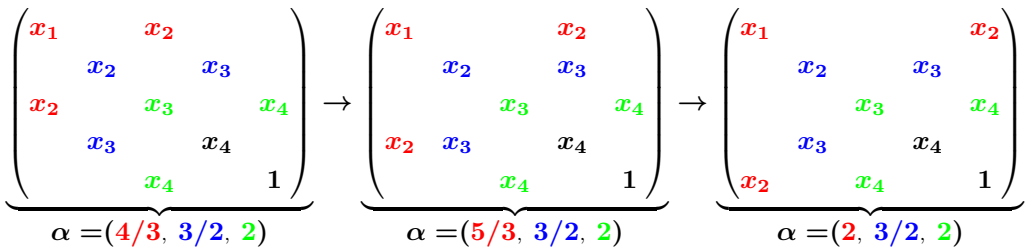


Figure 5: Shifting x_2 to the right increases α_2

Lemma 6. *The α_j exponents in (2.9) are strictly increasing functions of the t_{j+1} tail-indices defined in Definition 4.*

Precisely, suppose we derived the inequalities

$$x_\ell \geq d_{\ell+1} x_{\ell+1}^{\alpha_{\ell+1}} \text{ for } \ell = 1, \ldots, k-1 \tag{2.43}$$

from (P') using the recursion (2.27).

Suppose also that $j \in \{1, \ldots, k-1\}$, $t_{j+1} \leq k$, and we change A'_{j+1} so that t_{j+1} increases by 1. After the change we derive inequalities

$$x_\ell \geq f_{\ell+1} x_{\ell+1}^{\omega_{\ell+1}} \text{ for } \ell = 1, \ldots, k-1, \tag{2.44}$$

using the recursion (2.27). Here $f_{\ell+1}$ is a positive constant for all ℓ.

Then

$$\omega_{\ell+1} \begin{cases} = \alpha_{\ell+1} & \text{if } \ell > j \\ > \alpha_{\ell+1} & \text{if } \ell = j \\ \geq \alpha_{\ell+1} & \text{if } \ell < j. \end{cases} \tag{2.45}$$

Proof Recall from the proof of Lemma 4 that A'_{j+1} affects only polynomial p_j. Also recall from the proof of Lemma 5 that p_j does not affect inequalities (2.43) for $\ell > j$. So we conclude that $\omega_{\ell+1} = \alpha_{\ell+1}$ for all $\ell > j$.

We next prove $\omega_{j+1} > \alpha_{j+1}$. For brevity, let $s = t_{j+1}$ and recall that

$$\alpha_{j+1} \quad = \quad 2 - \frac{1}{\alpha},$$

where $\alpha = \alpha_{j+2} \cdot \cdots \cdot \alpha_s$.

We distinguish two cases. If $s < k$, then formula (2.27) implies $\omega_{j+1} = 2 - 1/(\alpha \cdot \alpha_{s+1})$, hence $\omega_{j+1} > \alpha_{j+1}$, as wanted. If $s = k$, then by the same formula $2 = \omega_{j+1}$ and $2 > \alpha_{j+1}$ so $\omega_{j+1} > \alpha_{j+1}$ again follows.

The remaining inequalities in (2.43) follow by induction using the recursion formula (2.27).

□

Lemma 7. *Suppose that $t_{j+1} = j+2$ for $j = 1, \ldots, k-1$, in other words, t_{j+1} is the smallest possible. Then in the inequalities (2.37) we have*

$$\alpha_{j+1} = 1 + \frac{1}{k-j} \text{ for } j = 1, \ldots, k-1. \tag{2.46}$$

Proof We use induction. First suppose $j = k-1$. Since p_{k-1} is of type 2, we see $\alpha_{j+1} = \alpha_k = 2$, as wanted. Next assume that $1 \le j < k-1$ and

$$\alpha_{j+2} = 1 + \frac{1}{k-j-1}.$$

By the recursion (2.27) we get

$$\alpha_{j+1} = 2 - \frac{1}{\alpha_{j+2}} = 1 + \frac{1}{k-j},$$

completing the proof. □

Proof of Theorem 1 The result follows from Lemmas 2 through 7. Precisely, by Lemma 2 variable x_k can be arbitrarily large in a strictly feasible solution of (P'). By Lemma 4 we derive the polynomial inequalities (2.20). From these in Lemma 5 we derive the clean inequalities (2.26) via the recursion (2.27).

From the recursion (2.27) it directly follows that all α_{j+1} are at most 2. The lower bound on the α_{j+1} is proved as follows: by Lemma 6 the α_{j+1} are monotone functions of the tail-indices t_{j+1}. On the other hand $t_{j+1} \ge j+2$ for all j by Lemma 3 and when $t_{j+1} = j+2$ for all j, then by Lemma 7 we have $\alpha_{j+1} = 1 + 1/(k-j)$. The proof is now complete. □

2.3 Computing the exponents by Fourier-Motzkin elimination

The recursion (2.27) gives a convenient way to compute the α_j exponents. Equivalently, we can compute the α_j via the well known Fourier-Motzkin elimination algorithm, designed for linear inequalities; this is an interesting contrast, since SDPs are highly nonlinear.

We do this as follows. If polynomial p_j is of type 1, then we suppress the lower order terms to get

$$x_j x_{t_{j+1}} \ge \text{constant}\, x_{j+1}^2, \tag{2.47}$$

see the last inequality in (2.42). If polynomial p_j is of type 2, then we similarly suppress the lower order terms to deduce

$$x_j \ge \text{constant}\, x_{j+1}^2. \tag{2.48}$$

After this, since $x_1, \ldots, x_k$ are all positive, we rewrite the inequalities in terms of $y_j := \log x_j$ for all j, then eliminate variables. For example, from the inequalites (2.11) we deduce

$$\begin{aligned} y_1 + y_3 &\ge 2y_2 \\ y_2 + y_4 &\ge 2y_3 \\ y_3 &\ge 2y_4. \end{aligned} \tag{2.49}$$

We add $\frac{1}{2}$ times the last inequality in (2.49) to the middle one to get

$$y_2 \ge \frac{3}{2} y_3. \tag{2.50}$$

We then add $\frac{2}{3}$ times (2.50) to the first inequality in (2.49) to get

$$y_1 \ge \frac{4}{3} y_2. \tag{2.51}$$

Finally, (2.50), (2.51) and the last inequality in (2.49) translate back to the inequalities (2.12).

3 When we do not even need a change of variables

As we previously discussed, the linear change of variables $x \leftarrow Mx$ is necessary to obtain a Khachiyan type hierarchy among the variables. Nevertheless, in this section we show a natural SDP in which large variables occur even without a change of variables; more precisely, the SDP is in the form of (P'). For completeness, we also revisit the example from [17], and show that the SDP therein is also in the regular form of (P').

Given a univariate polynomial of even degree $f(x) = \sum_{i=0}^{2n} a_i x^i$ we consider the problem of minimizing f over $\mathbb{R}$. We write this problem as

$$\begin{array}{lll} \sup & \lambda & \\ s.t. & f - \lambda & \geq \quad 0. \end{array} \tag{3.52}$$

We will show that in the natural SDP formulation of (3.52) exponentially large variables appear naturally, although here by "exponentially large" we only mean in magnitude, not in size.

It is known that $f - \lambda$ is nonnegative iff it is a sum of squares (SOS), that is, $f = \sum_{i=1}^{t} g_i^2$ for a positive integer t and polynomials g_i.

Define the vector of monomials

$$z = (1, x, x^2, \ldots, x^n)^\top.$$

Then $f - \lambda$ is SOS if and only if (see [12, 15, 19, 28]) $f - \lambda = zz^\top \bullet Q$ for some $Q \succeq 0$. Matching monomials in $f - \lambda$ and $zz^\top \bullet Q$ we translate (3.52) into the SDP

$$\begin{array}{lrcl} \max & -A_0 \bullet Q & & \\ s.t. & A_i \bullet Q & = & a_i \text{ for } i = 1, \ldots, 2n \\ & Q & \in & \mathcal{S}_+^{n+1}. \end{array} \tag{3.53}$$

Here for all $i \in \{0, 1, \ldots, 2n\}$ the (k, ℓ) element of the matrix A_i is 1 if $k + \ell = i + 2$ for some $k, \ell \in \{1, \ldots, n+1\}$ and all other entries of A_i are zero.

The dual problem of (3.53) is

$$\begin{array}{lll} \min & \sum_{i=1}^{2n} a_i y_i & \\ s.t. & \sum_{i=1}^{2n} y_i A_i + A_0 & \succeq \quad 0, \end{array} \tag{3.54}$$

whose constraints can be written as

$$\begin{pmatrix} 1 & y_1 & y_2 & \cdots & y_n \\ y_1 & y_2 & & \cdots & y_{n+1} \\ y_2 & & & \cdots & y_{n+2} \\ \vdots & & & \ddots & \vdots \\ y_n & y_{n+1} & y_{n+2} & \cdots & y_{2n} \end{pmatrix} \succeq 0.$$

Permuting rows and columns, this is equivalent to

$$\begin{pmatrix} y_{2n} & y_{2n-1} & y_{2n-2} & \cdots & y_n \\ y_{2n-1} & y_{2n-2} & & \cdots & y_{n-1} \\ y_{2n-2} & & & \cdots & y_{n-2} \\ \vdots & & & \ddots & \vdots \\ y_n & y_{n-1} & y_{n-2} & \cdots & 1 \end{pmatrix} \succeq 0. \tag{3.55}$$

Let us rename the variables so the even numbered ones come first, and the rest come afterwards, as

$$\begin{array}{rclrclcrcl} x_1 & = & y_{2n}, & x_2 & = & y_{2n-2}, & \dots & x_n & = & y_2; \\ x_{n+1} & = & y_{2n-1}, & x_{n+2} & = & y_{2n-3}, & \dots & x_{2n} & = & y_1. \end{array}$$

Then the constraints (3.55) become

$$\sum_{i=1}^{2n} x_i A'_i + B' \succeq 0. \tag{3.56}$$

Here the A'_i for $i = 1, \dots, n$ and B' are defined as follows. In A'_i the (k, ℓ) and (ℓ, k) entry is 1, if $k + \ell = 2i$ and all other entries are zero. and the rest are zero. In B' the lower right corner is 1 and the other elements are zero.

For example, when $n = 3$ the constraints (3.56) look like

$$x_1 \begin{pmatrix} 1 & & \\ & 0 & \\ & & 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 & & 1 \\ & 1 & \\ 1 & & 0 \end{pmatrix} + x_3 \begin{pmatrix} 0 & & \\ & 0 & 1 \\ & 1 & 0 \end{pmatrix} + \sum_{i=4}^{6} x_i A'_i + B' \succeq 0.$$

Thus $(A'_1, A'_2, \dots, A'_n)$ is a regular facial reduction sequence and (3.56) is in the form of (P'). The tail-indices (cf. Definition 4) are $t_{j+1} = j + 2$ for $j = 1, \dots, n - 1$, hence we can derive the following inequalities, just like we did in Lemma 4:

$$x_j x_{j+2} \geq x_{j+1}^2 \text{ for } j = 1, \dots, n-2; \text{ and } x_{n-1} \geq x_n^2.$$

Note that now the "δ" terms that appear in Lemma 4 are all zero, so we do not have to worry about "making x_k large." Hence by Lemma 7 we deduce that

$$x_j \geq x_{j+1}^{\alpha_{j+1}} \text{ for } j = 1, \dots, n-1$$

hold, where $\alpha_{j+1} = 1 + 1/(n - j)$ for all j.

We translate these inequalities back to the original y_j variables, and obtain the following result:

Theorem 2. *Suppose that $y \in \mathbb{R}^{2n}$ is feasible in (3.54). Then*

$$y_{2(n-j+1)} \geq y_{2(n-j)}^{1+1/(n-j)} \text{ for } j = 1, \dots, n-1.$$

□

Combining these inequalities we obtain

$$y_{2n} \geq y_2^n.$$

Theorem 2 complements a result of Lasserre [12, Theorem 3.2], which states the following: if $\bar{x}$ minimizes the polynomial $f(x)$ then

$$(y_1, y_2, \dots, y_{2n}) = (\bar{x}, \bar{x}^2, \dots, \bar{x}^{2n})$$

is optimal in (3.54). On the one hand, Theorem 2 states bounds on all feasible solutions, on the other hand, it does not specify an optimal solution.

For completeness, we next revisit an example of O' Donnell in [17], and show how the SDP that arises in there is in the regular form of (P').

Example 4. *We are given the polynomial with $2n$ variables*

$$p(x,y) = p(x_1,\ldots,x_n,y_1,\ldots,y_n) = x_1 + \cdots + x_n - 2y_1$$

and the set K defined as

$$\begin{array}{rclcrclcrcl} 2x_1y_1 & = & y_1, & & 2x_2y_2 & = & y_2 & \ldots & 2x_ny_n & = & y_n \\ x_1^2 & = & x_1, & & x_2^2 & = & x_2 & \ldots & x_n^2 & = & x_n \\ y_1^2 & = & y_2, & & y_2^2 & = & y_3 & \ldots & y_n^2 & = & 0. \end{array}$$

Note that in the description of K the very last constraint $y_n^2 = 0$ breaks the pattern seen in the previous $n-1$ columns. We ask the following question:

- *Is $p(x,y) \geq 0$ for all $(x,y) \in K$?*

The answer is clearly yes, since for all $(x,y) \in K$ we have $x_1,\ldots,x_n \in \{0,1\}$ and $y_1 = \cdots = y_n = 0$.

On the other hand, the sum of squares procedure verifies the "yes" answer as follows. Let

$$z = (1, x_1, \ldots, x_n, y_1, \ldots, y_n)^\top$$

be a vector of monomials, and find $\lambda, \mu, \nu \in \mathbb{R}^n$ and $Q \succeq 0$ such that

$$\begin{array}{rcl} p(x,y) = z^\top Q z & + & \lambda_1(2x_1y_1 - y_1) + \mu_1(x_1^2 - x_1) + \nu_1(y_1^2 - y_2) \\ & + & \lambda_2(2x_2y_2 - y_2) + \mu_2(x_2^2 - x_2) + \nu_2(y_2^2 - y_3) \\ & \vdots & \\ & + & \lambda_n(2x_ny_n - y_n) + \mu_n(x_n^2 - x_n) + \nu_n(y_n^2 - 0) \end{array} \tag{3.57}$$

Matching coefficients on the left and right hand side, [17] shows that any Q feasible in (3.57) must be of the form

$$Q = \left(\begin{array}{ccccc|ccccc|c} u_1 & 0 & \ldots & 0 & 0 & -u_2 & \ldots & 0 & 0 & 0 & 0 \\ 0 & u_2 & \ldots & 0 & 0 & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & 0 & \ldots & -u_{n-1} & 0 & 0 & 0 \\ 0 & 0 & \ldots & u_{n-1} & 0 & 0 & \ldots & 0 & -u_n & 0 & 0 \\ 0 & 0 & \ldots & 0 & u_n & 0 & \ldots & 0 & 0 & -2 & 0 \\ \hline -u_2 & \ldots & 0 & 0 & 0 & 1 & \ldots & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & -u_{n-1} & 0 & 0 & 0 & \ldots & 1 & 0 & 0 & 0 \\ 0 & \ldots & 0 & -u_n & 0 & 0 & \ldots & 0 & 1 & 0 & 0 \\ 0 & \ldots & 0 & 0 & -2 & 0 & \ldots & 0 & 0 & 1 & 0 \\ \hline 0 & \ldots & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \end{array}\right) \tag{3.58}$$

for suitable $u_1,\ldots,u_n$. Looking at 2×2 subdeterminants of Q we see that the u_i satisfy

$$4 \leq u_n,\ u_n^2 \leq u_{n-1},\ \ldots,\ u_2^2 \leq u_1, \tag{3.59}$$

which is the same as (Khachiyan), except we replaced the constant 2 by 4.

Let us define $E_{k\ell}$ to be the unit matrix in $\mathcal{S}^{2n+1}$ in which the (k,ℓ) and (ℓ,k) entries are 1 and the rest zero. Define

$$A'_1 = E_{11}, A'_i = E_{ii} - E_{i-1,n+i-1} \text{ for } i = 2,\ldots,n.$$

Then any Q feasible in (3.57) is written as

$$Q = u_1 A'_1 + u_2 A'_2 + \cdots + u_n A'_n + B' \succeq 0, \tag{3.60}$$

for a suitable B' (precisely, $B' = -2E_{n,2n} + \sum_{i=2n+1}^{2n} E_{ii}$).

We see that $(A'_1,\ldots,A'_n)$ is a regular facial reduction sequence, thus the system (3.60) is in the regular form of (P').

We remark that (3.60) arises by concatenating 2×2 psd blocks of the form (1.1), then permuting rows and columns. In other words, (3.60) is the exact representation of (*Khachiyan*) (apart from the constant 2 being replaced by 4), that we discussed after Example 1.

Among followup papers of O' Donnell [17] we should mention the work of Raghavendra and Weitz [24] which gave SDPs which also have a sum-of-squares origin, and exponentially large solutions. It would be interesting to see whether those SDPs are also in the normal form of (P').

4 Conclusion

We showed that large size solutions do not just appear as pathological examples in semidefinite programs, but are quite common: after a linear change of variables, they appear in every strictly feasible SDP. As to "how large" they get, that depends on the singularity degree of a dual problem and the so-called tail-indices of the constraint matrices. Further, large solutions naturally appear in SDPs that come from minimizing a univariate polynomial, without any change of variables.

We also studied how to represent large solutions of SDPs in polynomial space. Our main tool was the regularized semidefinite program (P'). If (P) and (P') are strictly feasible, then in the latter we can verify that a strictly feasible solution exists, without computing the actual values of the "large" variables $x_1,\ldots,x_k$: see Figure 2. Further, SDPs that arise from polynomial optimization (Section 3) and the SDP that represents (*Khachiyan*) are naturally in the form of (P'). Hence in these SDPs we can also certify large solutions without computing their actual values.

Several questions remain open. For example, what can we say about large solutions in semidefinite programs that are not strictly feasible? The discussion after Example 1 shows that we do not have a complete answer.

Also, recall that we transform (P) into (P') by a linear change of variables (equivalent to operations (1) and (2) in Definition 2) and a similarity transformation (operation (3) in Definition 2). The latter has no effect on how large the variables are. We are thus led to the following question: are *all* SDPs with exponentially large solutions in the form of (P') (perhaps after a similarity transformation)? In other words, can we *always* certify large size solutions in SDPs using a regular facial reduction sequence? Answering this question would help us answer the greater question: can we decide feasibility of SDPs in polynomial time?

Acknowledgement The authors are very grateful for helpful discussions to Richard Rimányi, in particular, to suggestions that led to the proof of Lemma 6. We gratefully acknowledge the support of National Science Foundation, award DMS-1817272.

References

[1] Amir Ali Ahmadi, Alex Olshevsky, Pablo A Parrilo, and John N Tsitsiklis. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical Programming*, 137(1):453–476, 2013. 3, 5

[2] Amir Ali Ahmadi and Jeffrey Zhang. On the complexity of testing attainment of the optimal value in nonlinear optimization. *Mathematical Programming*, pages 1–21, 2019. 3, 5

[3] Alexander I Barvinok. Feasibility testing for systems of real quadratic equations. *Discrete & Computational Geometry*, 10(1):1–13, 1993. 3, 4

[4] Daniel Bienstock. A note on polynomial solvability of the cdt problem. *SIAM Journal on Optimization*, 26(1):488–498, 2016. 3, 4

[5] Daniel Bienstock, Alberto Del Pia, and Robert Hildebrand. Complexity, exactness, and rationality in polynomial optimization. *arXiv preprint arXiv:2011.08347*, 2020. 3, 5

[6] Jonathan M. Borwein and Henry Wolkowicz. Regularizing the abstract convex program. *J. Math. Anal. App.*, 83:495–530, 1981. 3, 5

[7] Dmitriy Drusvyatskiy and Henry Wolkowicz. The many faces of degeneracy in conic optimization. *Foundations and Trends® in Optimization*, 3(2):77–170, 2017. 3, 5

[8] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012. 2, 4

[9] N Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984. 4

[10] Leonid Genrikhovich Khachiyan. A polynomial algorithm in linear programming. In *Doklady Akademii Nauk*, volume 244, pages 1093–1096. Russian Academy of Sciences, 1979. 4

[11] Masakazu Kojima, Shinji Mizuno, and Akiko Yoshise. A primal-dual interior point algorithm for linear programming. In *Progress in mathematical programming*, pages 29–47. Springer, 1989. 4

[12] Jean B Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001. 21, 22

[13] Minghui Liu and Gábor Pataki. Exact duality in semidefinite programming based on elementary reformulations. *SIAM J. Opt.*, 25(3):1441–1454, 2015. 6

[14] Bruno F Lourenço. Amenable cones: error bounds without constraint qualifications. *to appear, Mathematical Programming, arXiv preprint arXiv:1712.06221*, 2017. 7

[15] Yurii Nesterov. Squared functional systems and optimization problems. In *High performance optimization*, pages 405–440. Springer, 2000. 21

[16] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994. 4

[17] Ryan O'Donnell. SOS is not obviously automatizable, even approximately. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017. 5, 21, 22, 23, 24

[18] Panos M Pardalos and Stephen A Vavasis. Quadratic programming with one negative eigenvalue is NP-hard. *Journal of Global optimization*, 1(1):15–22, 1991. 3, 5

[19] Pablo A Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical programming*, 96(2):293–320, 2003. 21

[20] Gábor Pataki. A simple derivation of a facial reduction algorithm and extended dual systems. Technical report, Columbia University, 2000. 3, 5

[21] Gábor Pataki. Strong duality in conic linear programming: facial reduction and extended duals. In David Bailey, Heinz H. Bauschke, Frank Garvan, Michel Théra, Jon D. Vanderwerff, and Henry Wolkowicz, editors, *Proceedings of Jonfest: a conference in honour of the 60th birthday of Jon Borwein*. Springer, also available from http://arxiv.org/abs/1301.7717, 2013. 3, 5

[22] Gábor Pataki. Characterizing bad semidefinite programs: normal forms and short proofs. *SIAM Review*, 61(4):839–859, 2019. 6

[23] Lorant Porkolab and Leonid Khachiyan. On the complexity of semidefinite programs. *Journal of Global Optimization*, 10(4):351–365, 1997. 2

[24] Prasad Raghavendra and Benjamin Weitz. On the bit complexity of sum-of-squares proofs. *arXiv preprint arXiv:1702.05139*, 2017. 24

[25] James Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Mathematical programming*, 40(1):59–93, 1988. 4

[26] James Renegar. On the computational complexity and geometry of the first-order theory of the reals. part i: Introduction. preliminaries. the geometry of semi-algebraic sets. the decision problem for the existential theory of the reals. *Journal of symbolic computation*, 13(3):255–299, 1992. 2

[27] James Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, USA, 2001. 4

[28] Naum Z Shor. Class of global minimum bounds of polynomial functions. *Cybernetics*, 23(6):731–734, 1987. 21

[29] Jos Sturm. Error bounds for linear matrix inequalities. *SIAM J. Optim.*, 10:1228–1248, 2000. 7

[30] Stephen A Vavasis. Quadratic programming is in NP. *Information Processing Letters*, 36(2):73–77, 1990. 3, 5

[31] Stephen A Vavasis and Richard Zippel. Proving polynomial-time for sphere-constrained quadratic programming. Technical report, Cornell University, 1990. 5

[32] Hayato Waki and Masakazu Muramatsu. Facial reduction algorithms for conic optimization problems. *J. Optim. Theory Appl.*, 158(1):188–215, 2013. 3, 5