Improving Efficiency and Scalability of Sum of Squares Optimization: Recent Advances and Limitations

Amir Ali Ahmadi[1], Georgina Hall[1], Antonis Papachristodoulou[2], James Saunderson[3], and Yang Zheng[2]

***Abstract*— It is well-known that any sum of squares (SOS) program can be cast as a semidefinite program (SDP) of a particular structure and that therein lies the computational bottleneck for SOS programs, as the SDPs generated by this procedure are large and costly to solve when the polynomials involved in the SOS programs have a large number of variables and degree. In this paper, we review SOS optimization techniques and present two new methods for improving their computational efficiency. The first method leverages the sparsity of the underlying SDP to obtain computational speed-ups. Further improvements can be obtained if the coefficients of the polynomials that describe the problem have a particular sparsity pattern, called *chordal sparsity*. The second method bypasses semidefinite programming altogether and relies instead on solving a sequence of more tractable convex programs, namely linear and second order cone programs. This opens up the question as to how well one can approximate the cone of SOS polynomials by second order representable cones. In the last part of the paper, we present some recent negative results related to this question.**

I. Introduction and sum of squares review

Polynomial optimization is the problem of minimizing a (multivariate) polynomial function on a basic semialgebraic set; *i.e.*, a subset of the Euclidean space defined by polynomial equations and inequalities. This is an extremely broad class of optimization problems, with high-impact application areas throughout engineering and applied mathematics, which until not too long ago was believed to be hopelessly intractable to solve computationally. In recent years, however, a fundamental and exciting interplay between *convex optimization* and *algebraic geometry* has allowed for the solution or approximation of a large class of (nonconvex) polynomial optimization problems.

Amazingly, the success of this area stems from the ability to work around a single central question which is very simple to state: how can one test if a polynomial

$$p(x) := p(x_1, \ldots, x_n)$$

is *nonnegative*, *i.e.*, satisfies $p(x) \geq 0$ for all $x \in \mathbb{R}^n$?

This is an invited tutorial paper for the 2017 IEEE International Conference on Decision and Control. Most details are omitted and can be found in the relevant references.

[1]A. A. Ahmadi and G. Hall are with the department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08540, USA. Emails: a_a_a@princeton.edu; gh4@princeton.edu

[2]Y. Zheng and A. Papachristodoulou are with the Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, U.K. Emails: yang.zheng@eng.ox.ac.uk; antonis@eng.ox.ac.uk

[3]J. Saunderson is with the Department of Electrical and Computer Systems Engineering, Monash University, VIC 3800, Australia. Email: james.saunderson@monash.edu

Unfortunately, answering this question is NP-hard already when $p(x)$ has degree 4. A powerful and more tractable sufficient condition for nonnegativity of $p(x)$, however, is for it to be a *sum of squares* polynomial. A sum of squares (SOS) polynomial $p(x)$ is a polynomial that can be written as

$$p(x) = \sum_{i=1}^{r} f_i^2(x)$$

for some other polynomials $f_i(x), i = 1, \ldots, r$. The question as to whether a nonnegative polynomial can always be written as a sum of squares has a celebrated history, dating back to Hilbert's 17th problem [1] around the year 1900. What has caused a lot of recent excitement, however, is the discovery that the task of testing the SOS property and finding a sum of squares decomposition can be fully automated. This is a consequence of the following characterization of the set of SOS polynomials: a polynomial $p(x)$ of degree $2d$ is SOS if there exists a positive semidefinite matrix Q (usually called the *Gram matrix* of p) such that

$$p(x) = z(x)^T Q z(x), \tag{1}$$

where $z(x) = [1, x_1, \ldots, x_n, \ldots, x_n^d]$ is the standard vector of monomials of degree d [2]. Hence, testing whether a polynomial is a sum of squares amounts to solving a *semidefinite program* (SDP), a class of convex optimization problem for which numerical solution methods are available.

This simple but fundamental discovery forms the basis of a modern subfield of mathematical programming called *"sum of squares optimization"*. An *SOS program* is an optimization problem of the following form:

$$\begin{aligned} \min_{p} \quad & C(p) \\ \text{s.t.} \quad & A(p) = b \\ & p \text{ is SOS}, \end{aligned} \tag{2}$$

where the decision variables are the coefficients of the polynomial p, the objective $C(p)$ is some linear function of the coefficients of p, and $A(p)$ are affine constraints in the coefficients of p. As a consequence of the aforementioned characterization of SOS polynomials, this program can be recast as the following semidefinite program:

$$\begin{aligned} \min_{p,Q} \quad & C(p) \\ \text{s.t.} \quad & A(p) = b \\ & p(x) = z(x)^T Q z(x), \ \forall x \\ & Q \succeq 0. \end{aligned} \tag{3}$$

The most direct consequence of SOS optimization is for polynomial optimization: Under mild assumptions, the (global) minimum of a polynomial $p(x)$ on a basic algebraic set K turns out to be equal to the largest scalar γ such that $p(x) - \gamma$ is certified to be nonnegative on K with a sum of squares proof [2], [3]. The power of this statement stems from the fact that *no convexity assumption* is placed on the polynomial optimization problem and yet the search for the sum of squares certificates is a convex (in fact semidefinite) program.

Aside from polynomial optimization, numerous other areas of computational mathematics have been impacted by sum of squares techniques. These include approximation algorithms for NP-hard combinatorial optimization problems [4], [5], equilibrium analysis of games [6], robust and stochastic optimization [7], statistics and machine learning [8], [9], software verification [10], [11], filter design [12], quantum computation [13], automated theorem proving [14], and fault diagnosis and verification of hypersonic aircraft [15], [16], [17], among many others.

Despite the enormous impact of sum of squares optimization on polynomial optimization and related areas, the applicability of this methodology has always been limited by a single fundamental challenge, which is *scalability*. Indeed, when $p(x)$ has n variables and is of degree $2d$, the size of Q in (3) is $\binom{n+d}{d} \times \binom{n+d}{d}$. Poor scaling with problem dimension is not the only difficulty here: even when their size is not too large, SDPs are arguably the most expensive class of convex optimization problems to solve. This has led the optimization community to sometimes perceive semidefinite programming as a powerful theoretical tool, but not a practical one.

Outline. In this paper, we review two new techniques that aim to make SOS programs more scalable. In Section II, we present two efficient first-order methods based on the alternating direction method of multipliers (ADMM) to solve SDPs arising from sum of squares programming. Both methods exploit sparsity to increase the computational efficiency of SOS programs: the first method exploits the inherent sparsity of the SDPs obtained from SOS programs and can be used for any SOS program; the second one requires additional problem structure, namely that the polynomials at hand be *chordally sparse*. In Section III, we present techniques that do away with semidefinite programming altogether. Instead, the semidefinite program that we wish to solve is replaced by a sequence of linear or second order cone programs, which are much more tractable than SDPs. Generating these sequences amounts to constructing a series of linear and second-order cone programming-representable cones which inner approximate the set of SOS polynomials. This leads to the following conceptual question: how well can second order cone programming based techniques perform for inner approximating the set of SOS polynomials? Could we maybe even exactly represent the SOS cone using second order-representable cones? In Section IV, we review recent negative results by Fawzi which show that an exact representation is not possible, in a case as basic as the case of univariate quartics.

Notation. Unless otherwise specified, we will be considering throughout polynomials p of degree $2d$ and in n variables. We write:

$$p(x) = \sum_{\alpha \in \mathbb{N}^n_{2d}} p_\alpha x^\alpha, \quad p_\alpha \in \mathbb{R}, \tag{4}$$

where $x^\alpha = x_1^{\alpha_1} \dots x_n^{\alpha_n}$ is a monomial of degree $|\alpha| = \sum_i \alpha_i$ and $\mathbb{N}^n_{2d} = \{\alpha \in \mathbb{N}^n \mid |\alpha| \leq 2d\}$. We denote by $PSD_{n,2d}$ (resp. $SOS_{n,2d}$) the set of nonnegative (resp. sum of squares polynomials) in n variables and of degree $2d$. We further denote by $\mathbb{S}^k$ the cone of $k \times k$ symmetric matrices, and by $\mathbb{S}^k_+$ the cone of $k \times k$ positive semidefinite matrices.

II. Exploiting Sparsity in SOS Programs

In this section, we introduce two strategies that exploit sparsity to increase the computational efficiency of SOS programs. The first strategy exploits sparsity in the coefficient matching conditions arising from SOS programs for general polynomials, and the second one takes advantage of chordal sparsity for sparse polynomials. Both of them use a first-order operator splitting algorithm, known as the alternating direction method of multipliers (ADMM) [18], to efficiently compute a solution of the SDP from SOS programs at the cost of reduced accuracy.

A. Sparsity in the coefficient matching conditions

Consider a real polynomial $p(x)$ of degree $2d$ in (4). As mentioned in (1), if $p(x)$ is SOS, then we have

$$p(x) = \sum_{i=1}^{r} f_i^2(x) = \sum_{i=1}^{r} \left(q_i^T z(x)\right)^2 = z^T(x) Q z(x), \tag{5}$$

where $Q \succeq 0$, and $z(x)$ is a monomial basis. Generally, $z(x)$ is the vector of all monomials of degree no greater than d:

$$z(x) = [1, x_1, x_2, \dots, x_n, x_1^2, x_1 x_2, \dots, x_n^d]^T. \tag{6}$$

Let A_α be the indicator matrix for the monomials x^α in the rank-one matrix $z(x)z(x)^T$. The SOS constraint (5) can then be reformulated as

$$p(x) = \langle z(x)z(x)^T, Q \rangle = \sum_{\alpha \in \mathbb{N}^n_{2d}} \langle A_\alpha, Q \rangle x^\alpha.$$

Matching the coefficients of the left- and right-hand sides gives the equality constraints

$$\langle A_\alpha, Q \rangle = p_\alpha \quad \forall\, \alpha \in \mathbb{N}^n_{2d}. \tag{7}$$

These equalities (7) are referred to as *coefficient matching conditions* [19]. Then, the existence of an SOS decomposition for $p(x)$ can be checked by solving the feasibility SDP [20]

$$\begin{aligned} \text{find} \quad & Q \\ \text{subject to} \quad & \langle A_\alpha, Q \rangle = p_\alpha, \quad \alpha \in \mathbb{N}^n_{2d}, \\ & Q \succeq 0. \end{aligned} \tag{8}$$

As mentioned in Section I, the dimension of the positive semidefinite variable Q in (8) is $\binom{n+d}{d} \times \binom{n+d}{d}$, which grows quickly as n or d increases. Note that this number may

be reduced by taking advantage of the structural properties of $p(x)$ to eliminate redundant monomials in $z(x)$; well-known techniques include Newton polytope [21], diagonal inconsistency [22], and symmetry property [23]. Also, the size of the underlying SDP was investigated for some classes of matrix polynomials with sparsity in [24].

One important feature of (8) is that the coefficient matching conditions are sparse, in the sense that each equality constraint in (8) only involves a small subset of entries of Q [19], since only a small subset of entries of the product $z(x)z(x)^T$ are equal to a given monomial x^α. In particular, we re-index the constraint matching conditions (7) using integer indices $i = 1, \ldots, m$, where $m = \binom{n+2d}{2d}$. Let $\mathrm{vec} : \mathbb{S}^N \to \mathbb{R}^{N^2}$ be the operator mapping a matrix to the stack of its columns, and define

$$A = \left[\mathrm{vec}(A_1), \cdots, \mathrm{vec}(A_m)\right]^T.$$

Then, the equality constraints in (8) can be rewritten as

$$A \cdot \mathrm{vec}(Q) = b, \tag{9}$$

where $b \in \mathbb{R}^m$ is a vector collecting the coefficients p_i of p. We have the following result.

Theorem 1 (Sparsity of constraints [19]): Let A be the coefficient matrix of the equality constraints for (8). The density of nonzero elements in A is $\mathcal{O}(\frac{1}{n^{2d}})$.

Note that this result holds for dense polynomials. Also, the density decreases quickly as n or d increases, which means the SDP (8) becomes very sparse for large-scale (dense or not) polynomials. In Table I, we list the density of nonzero elements in typical cases. Therefore, it is desirable to exploit this sparsity to improve the computational efficiency of (8). Let us represent $A = [a_1, a_2, \ldots, a_m]^T$, so that each vector a_i is a row of A, and let $H_i, i = 1, \ldots, m$ be "entry-selector" matrices of 1's and 0's that select the nonzero elements of a_i. We have the following equivalence.

$$A \cdot \mathrm{vec}(Q) = b \Leftrightarrow \begin{cases} (H_i a_i)^T z_i = b_i \\ z_i = H_i \cdot \mathrm{vec}(Q) \end{cases}, i = 1, \ldots, m, \tag{10}$$

where z_i is a copy of the non-zero elements of $\mathrm{vec}(Q)$ in the i-th equality constraint. Note that only the non-zero elements are involved in (10). Also, the equalities in (10) are enforced *individually* by z_i, not *simultaneously* as in (9).

Together with equivalence (10) that exploits the sparsity in A, we can apply ADMM to (8), resulting in an efficient algorithm that is free of any matrix inversion. Each iteration of the resulting ADMM algorithm consists of one conic projection and multiple quadratic programs with closed-form solutions; we refer the interested reader to [19] for details.

B. Chordal sparsity in sparse polynomials

The strategy that exploits the sparsity in the coefficient matching conditions works for general (dense or not) polynomials. However, the dimension of Q in (8) is unchanged, which may still require extensive computation for large-scale instances. If the polynomial $p(x)$ has structured sparsity, the large cone constraint $Q \succeq 0$ can be replaced by a set of smaller cone constraints [25]–[27]. Specifically, we assume the polynomial $p(x)$ has *correlative sparsity*, first introduced by Waki *et al.* [26], which is a higher level of view of the sparsity of a polynomial in terms of the interaction of variables x_i. Given a polynomial $p(x)$ in (4), the correlative sparsity pattern is represented by a matrix $R \in \mathbb{S}^n$,

$$R_{ij} = \begin{cases} 1, & \text{if } i = j \text{ or } \alpha_i, \alpha_j \geq 1, p_\alpha \neq 0 \\ 0, & \text{otherwise} \end{cases}.$$

Further, we can associate an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{1, \ldots, n\}$ and $\mathcal{E} = \{(i, j) \mid R_{ij} = 1, i \leq j\}$. Then, Waki *et al.* proposed that multiple sets of monomial basis could be used to construct an SOS polynomial [26], *i.e.*,

$$p(x) = \sum_{i=1}^{q} z_i^T(x) Q_i z_i(x), \tag{11}$$

where $z_i(x)$ is a monomial basis and $Q_i \succeq 0$. The choice of $z_i(x)$ depends on the maximal clique (see the precise definition below) of a *chordal extension* of the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, and the dimension of Q_i becomes small if the size of the largest maximal clique is small (please refer to [26] for further information).

As we have seen that an SOS program can always be transformed into a special SDP (see Section I), we focus in the following on a general sparse SDP and introduce an ADMM algorithm that exploits the inherent chordal sparsity. We consider the following primal standard SDP.

$$\begin{aligned} \min_X \quad & \langle C, X \rangle \\ \text{subject to} \quad & \langle A_i, X \rangle = b_i, i = 1, \ldots, m, \\ & X \succeq 0. \end{aligned} \tag{12}$$

For the sake of completeness, we first introduce several graph-theoretic concepts. Given an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a subset of vertices $\mathcal{C} \subseteq \mathcal{V}$ is called a clique if $(i, j) \in \mathcal{E}, \forall\, i, j \in \mathcal{C}, i \neq j$. The clique is called maximal if it is not a subset of any other clique. An undirected graph $\mathcal{G}$ is called *chordal* if every cycle of length greater than three has at least one chord. Note that if $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is not chordal, it can be *chordal extended*, *i.e.*, we can construct a chordal graph $\mathcal{G}'(\mathcal{V}, \mathcal{E}')$ by adding additional edges to $\mathcal{E}$. More details can be found in [28].

Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be an undirected graph with self-loops. We say that X is a partial symmetric matrix defined by $\mathcal{G}$ if $X_{ij} = X_{ji}$ are given when $(i, j) \in \mathcal{E}$, and arbitrary otherwise. We define the following sparse cones.

$$\begin{aligned} \mathbb{S}^n(\mathcal{E}, ?) =& \{X \in \mathbb{S}^n \mid X_{ij} = X_{ji} \text{ given if } (i, j) \in \mathcal{E}\}, \\ \mathbb{S}^n_+(\mathcal{E}, ?) =& \{X \in \mathbb{S}^n(\mathcal{E}, ?) \mid \\ & \exists M \succeq 0,\, M_{ij} = X_{ij}, \forall (i, j) \in \mathcal{E}\}. \end{aligned}$$

Given a clique $\mathcal{C}_k$ of $\mathcal{G}$, we let $E_{\mathcal{C}_k} \in \mathbb{R}^{|\mathcal{C}_k| \times n}$ be the matrix with $(E_{\mathcal{C}_k})_{ij} = 1$ if $\mathcal{C}_k(i) = j$ and zero otherwise, where $\mathcal{C}_k(i)$ is the i-th vertex in $\mathcal{C}_k$, sorted in the natural ordering. Then, we have the following result.

Theorem 2 (Grone's theorem [29]): Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a chordal graph with a set of maximal cliques $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_p\}$.

TABLE I

DENSITY OF NONZERO ELEMENTS IN THE EQUALITY CONSTRAINTS OF SDP (8)

n	4	6	8	10	12	14	16
$2d=4$	1.42×10^{-2}	4.76×10^{-3}	2.02×10^{-3}	9.99×10^{-4}	5.49×10^{-4}	3.27×10^{-4}	2.06×10^{-4}
$2d=6$	4.76×10^{-3}	1.08×10^{-3}	3.33×10^{-4}	1.25×10^{-4}	5.39×10^{-5}	2.58×10^{-5}	1.34×10^{-5}
$2d=8$	2.02×10^{-3}	3.33×10^{-4}	7.77×10^{-5}	2.29×10^{-5}	7.94×10^{-6}	3.13×10^{-6}	1.36×10^{-6}

Then, $X \in \mathbb{S}^n_+(\mathcal{E},?)$ if and only if $X_k = E_{\mathcal{C}_k} X E_{\mathcal{C}_k}^T \in \mathbb{S}_+^{|\mathcal{C}_k|}$ for all $k = 1, \ldots, p$.

Remark 1: This theorem allows us to equivalently replace $\mathbb{S}^n_+(\mathcal{E},?)$ with a set of coupled but smaller convex cones. A dual result can be found in [30]. These results have been exploited in interior-point methods for SDPs [25]; also, see recent applications in stability analysis and controller synthesis of large-scale linear systems [31]–[33].

We assume that (12) is sparse with an *aggregate sparsity pattern* described by $\mathcal{G}(\mathcal{V},\mathcal{E})$, meaning that $(i,j) \in \mathcal{E}$ if and only if the entry ij of at least one of the matrices C, $A_1, \ldots, A_m$ is nonzero. Also, it is assumed that $\mathcal{G}$ is chordal with a set of maximal cliques $\mathcal{C}_1, \ldots, \mathcal{C}_p$. In (12), only the entries of X corresponding to the edges $\mathcal{E}$ appear in the cost and constraint functions. Therefore, the constraint $X \in \mathbb{S}^n_+$ can be replaced by $X \in \mathbb{S}^n_+(\mathcal{E},?)$. Using Theorem 2, we can reformulate (12) as

$$\begin{aligned} \min_{X, X_1, \ldots, X_p} \quad & \langle C, X\rangle \\ \text{subject to} \quad & \langle A_i, X\rangle = b_i, && i = 1, \ldots, m \\ & X_k - E_k X E_k^T = 0, && k = 1, \ldots, p, \\ & X_k \in \mathbb{S}_+^{|\mathcal{C}_k|}, && k = 1, \ldots, p. \end{aligned} \tag{13}$$

In other words, the original large semidefinite cone is decomposed into multiple smaller cones at the cost of introducing a set of consensus constraints between the variables. Together with this reformulation that exploits the aggregate sparsity pattern to reduce the cone dimension, we can apply ADMM to (13), which results in an efficient algorithm that works with smaller positive semidefinite cones; see [27] for details. Similar decompositions are available for the dual standard SDP and the homogeneous self-dual embedding of sparse SDPs [34], [35].

C. Numerical results

The two strategies have been implemented in the MATLAB packages SOSADMM and CDCS [36], respectively. These two packages are available from https://github.com/oxfordcontrol/SOSADMM and https://github.com/oxfordcontrol/CDCS. This section presents numerical tests of SOSADMM and CDCS on the random unconstrained polynomial optimization problems (more numerical results can refer to [19], [34]). In the experiments, we set the termination tolerance to 10^{-4}, and the maximum number of iterations to 2×10^3 for SOSADMM and CDCS. The primal method in CDCS was used, and SeDuMi [37] was used as a benchmark solver. Consider the polynomial minimization problem $\min_{x\in\mathbb{R}^n} p(x)$, where $p(x)$ is a given polynomial.

TABLE II

CPU TIME (S) TO SOLVE THE SDP RELAXATIONS (14). N IS THE SIZE OF THE PSD CONE, m IS THE NUMBER OF CONSTRAINTS.

Dimensions			CPU time (s)		
n	N	m	SeDuMi	CDCS (primal)	SOS-ADMM
2	6	14	0.108	0.163	0.041
6	28	209	0.295	0.212	0.093
10	66	1000	4.197	0.340	0.294
14	120	3059	53.68	0.575	0.490
18	190	7314	621.2	1.696	1.339
20	231	10625	1806.6	4.694	2.362

As described in Section I, we can obtain an SDP relaxation as

$$\begin{aligned} \max \quad & \gamma \\ \text{subject to} \quad & p(x) - \gamma \quad \text{is SOS.} \end{aligned} \tag{14}$$

We generated $p(x)$ according to $p(x) = p_0(x) + \sum_{i=1}^n x_i^{2d}$, where $p_0(x)$ is a random polynomial with normally distributed coefficients of degree strictly less than $2d$. We used GloptiPoly [38] to generate the examples. Table II compares the CPU time (in seconds) required to solve the SOS relaxation as the number of variables was increased n with $d = 2$. Both SOSADMM and CDCS-primal were faster than SeDuMi on these examples. Also, the optimal value returned by SOSADMM was within 0.05% of the high-accuracy value returned by SeDuMi. Note that SOSADMM was faster than CDCS-primal in the experiments and this is expected since the random polynomials were dense; major computational improvements have been achieved by SOSADMM. For brevity, the interested reader is referred to [34] for more numerical results of CDCS on sparse SDPs.

III. LINEAR AND SECOND-ORDER PROGRAMMING-BASED ALTERNATIVES TO SOS PROGRAMS

In this section, we focus on another class of methods that enables us to increase the computational efficiency of SOS programs. These consist in replacing the semidefinite program that underlies any SOS program (see Section I) by more tractable convex programs such as linear or second-order cone programs.

A. DSOS and SDSOS programs

Recall from Section I that a polynomial $p(x)$ of degree $2d$ and in n variables is SOS if and only if there exists a positive semidefinite matrix Q such that $p(x) = z(x)^T Q z(x)$, where $z(x) = [1, x_1, x_2, \ldots, x_n^d]$ is the vector of monomials of degree d. As a consequence, solving any SOS program amounts to solving a semidefinite program where the variable

Q is of size $\binom{n+d}{d}$. As the number of variables and degree of the polynomial $p(x)$ at hand increase, the cost of solving such a semidefinite program can quickly become prohibitive. The idea proposed in [39] is to replace the condition that the Gram matrix Q be positive semidefinite with stronger but cheaper conditions in the hope of obtaining more efficient inner approximations to the cone $SOS_{n,2d}$. Two such conditions come from the concepts of *diagonally dominant* and *scaled diagonally dominant* matrices in linear algebra. We recall these definitions below.

Definition 1: A symmetric matrix A is *diagonally dominant* (dd) if $a_{ii} \geq \sum_{j \neq i} |a_{ij}|$ for all i. We say that A is *scaled diagonally dominant* (sdd) if there exists a diagonal matrix D, with positive diagonal entries, which makes DAD diagonally dominant.

We refer to the set of $n \times n$ dd (resp. sdd) matrices as DD_n (resp. SDD_n). The following inclusions are a consequence of Gershgorin's circle theorem [40]:

$$DD_n \subseteq SDD_n \subseteq \mathbb{S}^n_+.$$

Definition 2 ([39]): A polynomial $p(x)$ of degree $2d$ is said to be *diagonally-dominant-sum-of-squares* (DSOS) if it admits a representation as $p(x) = z^T(x)Qz(x)$, where Q is a dd matrix and $z(x)$ is the standard vector of monomials of degree d. A polynomial $p(x)$ of degree $2d$ is said to be *scaled-diagonally-dominant-sum-of-squares* (SDSOS) if it admits a representation as $p(x) = z^T(x)Qz(x)$, where Q is an sdd matrix and $z(x)$ is the standard vector of monomials of degree d.

Let us denote the cone of polynomials in n variables and degree $2d$ that are DSOS and SDSOS by $DSOS_{n,2d}$, $SDSOS_{n,2d}$. The following inclusion relations are straightforward:

$$DSOS_{n,2d} \subseteq SDSOS_{n,d} \subseteq SOS_{n,2d} \subseteq PSD_{n,2d}.$$

An illustration of how sections of these different cones compare on an example is given in Figure 1, taken from [39]. We consider a parametric family of polynomials parameterized by a and b,

$$p_{a,b}(x_1, x_2) = 2x_1^4 + 2x_2^4 + ax_1^3x_2 + (1-a)x_1^2x_2^2 + bx_1x_2^3,$$

and plot in the figure the values of (a, b) for which the polynomial is DSOS (innermost set), SDSOS (set containing the DSOS set), and SOS (or equivalentally nonnegative in this case) (outermost set).

These definitions give rise to the notion of DSOS and SDOS programs. With similar notation to (2), we say that a DSOS (resp. SDSOS) program is an optimization problem of the following form:

$$\begin{aligned} \min_{p \in \mathbb{R}_{n,2d}[x]} \quad & C(p) \\ \text{s.t.} \quad & A(p) = b \\ & p \text{ is DSOS (resp. SDSOS)} \end{aligned} \tag{15}$$

Note that (15) provides upperbounds on the optimization problem given in (2) as the set of DSOS and SDSOS polynomials is a subset of the set of SOS polynomials. This loss in solution accuracy is compensated by gains in terms of scalability and solving-time. This is a consequence of the following theorem.

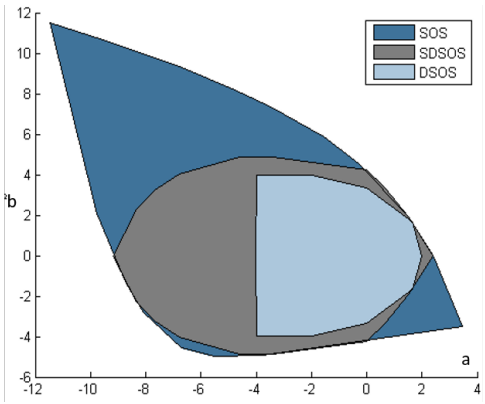

Fig. 1. A comparison of the DSOS/SDSOS/SOS cones on an example

Theorem 3 ([39]): For any fixed d, solving a DSOS (resp. SDSOS) program can be done with linear programming (resp. second order cone programming) of size polynomial in n.

The "LP part" of this theorem is not hard to see. The equality $p(x) = z(x)^T Qz(x)$ gives rise to linear equality constraints between the coefficients of p and the entries of the matrix Q. The requirement of diagonal dominance on the matrix Q can also be described by linear inequality constraints on Q. The "SOCP part" of the statement is not as straightforward and its proof can be found in [39].

We illustrate the gains that one can make using these methods in Table III taken from [39]. We have reported the time and bounds obtained when mininimizing a random polynomial of degree $d = 4$ and with a varying number of variables n over the unit sphere, using a DSOS, SDSOS and SOS program (see [39] for the precise formulations). Note that when n is small, the SOS program returns a better bound in slightly longer times than the DSOS/SDSOS programs. However, when n gets large, the SOS program cannot be solved due to memory issues whereas both the DSOS and SDSOS programs run in the order of seconds. These results were obtained on a 3.4 GHz Windows computer with 16 GB of memory.

B. Improving on DSOS and SDSOS programming

As mentioned previously, the advantages of substituting an SOS program with a DSOS or SDSOS program are scalability and computational efficiency of the program obtained. This comes at the cost of solution accuracy. In this section, we present two methods for mitigating the loss in accuracy that we observe. These methods involve constructing a sequence of iterative linear or second-order cone programs where the first iteration consists in solving the DSOS/SDSOS program given in (15). Our goal throughout will be to solve the optimization problem given in (2).

1) Column generation method [41]: For simplicity, we present here the linear programming-based version of the

TABLE III

LOWER BOUNDS OBTAINED USING S/D/SOS PROGRAMS TO COMPUTE THE MINIMUM OF A QUARTIC FORM ON THE SPHERE FOR VARYING n, ALONG WITH RUN TIMES (IN SECS).

	$n=15$		$n=20$		$n=25$		$n=30$		$n=40$	
	bd	t(s)	bd	t(s)	bd	t(s)	bd	t(s)	bd	t(s)
DSOS	-10.96	0.38	-18.012	0.74	-26.45	15.51	-36.85	7.88	-62.30	10.68
SDSOS	-10.43	0.53	-17.33	1.06	-25.79	8.72	-36.04	5.65	-61.25	18.66
SOS	-3.26	5.60	-3.58	82.22	-3.71	1068.66	NA	NA	NA	NA

algorithm. An analogous method based on second-order cone programming can be found in [41]. To understand this method, the following characterization of diagonally dominant matrices is needed [44]: A symmetric matrix M is diagonally dominant if and only if it can be written as

$$M = \sum_{i=1}^{n^2} \alpha_i v_i v_i^T, \alpha_i \geq 0,$$

where $\{v_i\}$ is the set of all nonzero vectors in $\mathbb{R}^n$ with at most 2 nonzero components, each equal to ± 1. The first iteration of our algorithm (i.e., solving (15)) then amounts to solving:

$$\begin{aligned} \min_{p \in \mathbb{R}_{n,2d}[x], \alpha} \quad & C(p) \\ \text{s.t.} \quad & A(p) = b \\ & p(x) = z(x)^T Q z(x), \forall x, \\ & Q = \sum_i \alpha_i v_i v_i^T, \alpha_i \geq 0, \end{aligned} \tag{16}$$

where $\{v_i\}$ are fixed. At each iteration, one adds a new "column" v to the set $\{v_i\}$ and the problem is then solved again. This leads to Algorithm 1.

Algorithm 1 Column generation algorithm

1: **initialize:** Solve (16). Obtain α and p.
2: **repeat**
3: Find a vector v using the dual as described below
4: Solve

$$\begin{gathered} \min_{\alpha,p} C(p) \\ \text{s.t. } A(p) = b \\ p(x) = z(x)^T Q z(x), \forall x, \\ Q = \textstyle\sum_i \alpha_i v_i v_i^T + \alpha v v^T \\ \alpha_i \geq 0 \ \forall i, \alpha \geq 0 \end{gathered}$$

5: **until** Termination Condition is met
6: **return** $C(p)$ and p

Note that at each iteration, this algorithm can only improve: indeed, by taking $\alpha = 0$, one recovers the solution to the previous iteration. To obtain strict improvement, one needs to carefully pick the vector v that we add to our set of columns. One way of doing this is via the dual of the problem given in (16). The constraint $Q = \sum_i \alpha_i v_i v_i^T$ in the primal gives rise to a constraint of the type $v_i^T X v_i \geq 0, \ \forall i$ in the dual, where X is a dual variable. A good choice of a vector v is then given by any vector v such that $v^T X v < 0$, see [41] for more information regarding the choice of v.

One can choose to terminate the algorithm under different conditions, e.g., lack of improvement of the optimal value/solution, or expiration of the time/computational budget associated to the task of solving the SDP.

An illustration of the performance of this technique is given in Table IV taken from [41]. The setting is analogous to the one used to obtain Table III: we report the time and bounds obtained when minimizing a random degree $d = 4$ homogeneous polynomial over the unit sphere. The experiments were run on a 2.33 GHz Linux machine with 32 GB of memory. In each iteration, we add an appropriate new vector v to the sequence $\{v_i\}$ that has at most three nonzero elements, each equaling 1 or -1. We stop the algorithm when either of the two following conditions are met: lack of improvement in the optimal value or time budget of 600s exceeded. Note that one can significantly improve on the initial approximations provided in Table III within a reasonable 10 min time lapse.

2) Sum of squares basis pursuit [42]: The idea behind this algorithm is the following. Let us assume that one could solve the problem given in (2) and obtain the Gram matrix Q^* associated to the optimal polynomial p^*. If we changed the monomial basis $z(x)$ in (3) to the monomial basis $L^* z(x)$, where L^* is the Cholesky decomposition (or square root) of Q^*, then we could use linear programming to recover the optimal solution of (2). Indeed, in this case, the optimal Gram matrix would be given by the identity matrix, which is diagonal, and searching for a diagonal Gram matrix can be done via linear programming. In our case, we do not have access to L^*, but the idea is to work towards such a basis. This procedure is detailed in Algorithm 2.

The algorithm will terminate under similar conditions to the ones given in Section III-B.1. Note that this algorithm is guaranteed to converge: at each iteration, the optimal value of the problem decreases (indeed, setting $Q = I$ enables us to recover the solution at the previous iteration), and is lowerbounded by the optimal solution to the SDP in (2).

We finally point the reader to very recent work [43], which goes beyond DSOS and SDSOS optimization and produces a converging hierarchy for the polynomial optimization prob-

TABLE IV

LOWER BOUNDS OBTAINED USING COLUMN GENERATION TO COMPUTE THE MINIMUM OF A QUARTIC FORM ON THE SPHERE FOR VARYING n, ALONG WITH RUN TIMES (IN SECS).

	$n=15$		$n=20$		$n=25$		$n=30$		$n=40$	
	bd	t(s)	bd	t(s)	bd	t(s)	bd	t(s)	bd	t(s)
Col Gen	-5.57	31.19	-9.02	471.39	-20.08	600	-32.28	600	-35.144	600

Algorithm 2 Sum of squares basis pursuit algorithm

1: **initialize:** Solve (15). Obtain the Gram matrix Q associated to the optimal p. Compute $L = chol(Q)$.
2: **repeat**
3: Solve

$$\begin{gathered}\min_{\alpha,p} C(p)\\ \text{s.t. } A(p) = b\\ p(x) = z(x)^T L^T Q L z(x), \forall x,\\ Q \text{ is d.d./s.d.d.}\end{gathered}$$

4: Obtain the Gram matrix Q associated to the optimal p. Set $L \leftarrow chol(Q)L$.
5: **until** Termination Condition is met
6: **return** $C(p)$ and p

lem that does not even require the use of linear or second order cone programming. This hierarchy only involves multiplying certain polynomials together and checking whether the coefficients of the product are nonnegative.

IV. LIMITATIONS ON REPRESENTING SOS CONES WITH BOUNDED SIZE PSD BLOCKS

In Section III-A we discussed methods to certify nonnegativity of polynomials by showing that they are SDSOS. Checking that a polynomial is SDSOS involves solving a second-order cone program. It is well-known that any second-order cone program can be written as a semidefinite program in which all blocks have size 2×2 and hence testing whether a polynomial is SDSOS simply amounts to solving an SDP involving only 2×2 blocks. Restricting to SDPs with only small blocks is attractive because such problems can be solved more efficiently than general SDPs of the same size.

On the one hand, in Section III-A we saw that approximating SOS cones with SDSOS cones (and variations on this idea) are empirically very powerful. On the other hand, in general the SOS cone strictly contains the SDSOS cone. Beyond the SDSOS cone, there are potentially many other ways to certify non-negativity of a family of polynomials using SDPs with only 2×2 blocks. In this section we consider recent results illustrating the *limitations* of modeling with linear matrix inequalities (LMIs) having blocks of bounded size (particularly 2×2 blocks). Concretely, we consider the following question:

Q1 Is it possible to *exactly represent* SOS cones using LMIs that only involve 2×2 blocks, (or, more generally, blocks of bounded size)?

Closely related, and more refined, is the following approximation version of the question:

Q2 Given a positive integer p, how well can we *approximate* SOS cones with convex cones that can be described using LMIs that involve at most p, 2×2 blocks?

The focus of this section is to discuss recent results of Fawzi [45] showing that the answer to Q1 is already negative for non-negative univariate quartics. Addressing Q2 remains a research challenge. In presenting Fawzi's result, we briefly describe recently developed ideas and tools for reasoning about all possible LMI descriptions of a convex set (with fixed block sizes). These are based on the idea of *cone ranks* (introduced by Gouveia, Parrilo, and Thomas [46]) of certain non-negative matrices associated with the convex set.

A. General PSD lifts with fixed block size

As mentioned previously, we use the notation $\mathbb{S}^k_+$ for the cone of $k\times k$ positive semidefinite matrices, and the notation $(\mathbb{S}^k_+)^p$ for the Cartesian product of p copies of $\mathbb{S}^k_+$.

With this notation, we can now formalize the idea of a representation of a convex set with LMIs involving only 2×2 blocks. The following definition is a special case of a definition due to Gouveia, Parrilo, and Thomas [46].

Definition 3: A convex cone $C \subseteq \mathbb{R}^n$ has a *proper* $(\mathbb{S}^2_+)^p$-*lift*[1] if there is a subspace L of $(\mathbb{S}^2)^p$ and a linear map $\pi : (\mathbb{S}^2)^p \to \mathbb{R}^p$ such that

$$C = \pi\left[(\mathbb{S}^2_+)^p \cap L\right]$$

and L meets the interior of $(\mathbb{S}^2_+)^p$.
More concretely, C has a $(\mathbb{S}^2_+)^p$-*lift* if and only if it can be expressed in the form

$$C = \left\{ x \in \mathbb{R}^n : \exists y \in \mathbb{R}^m \text{ s.t. } \sum_{i=1}^n A_i x_i + \sum_{j=1}^m B_j y_j \succeq 0 \right\}$$

where the A_i and the B_j are $2p \times 2p$ symmetric matrices that are all block diagonal (with the same block structure) consisting of p blocks, each of size 2×2.

Question Q1 can now be expressed more concisely as:

Q1' For which (n, d) does there exist a finite p such that $SOS_{n,d}$ has a $(\mathbb{S}^2_+)^p$-lift?

Answering questions like this in the negative, i.e., showing that lifts do *not* exist, has proven very challenging. The work of Gouveia, Parrilo, and Thomas [46] (building on ideas of

[1]Clearly there is an analogous definition for a proper $(\mathbb{S}^k_+)^p$-lift for any fixed k. All the definitions and results in Section IV-A extend to that case.

Yannakakis [47] in the case of linear programming), made a connection between the existence of lifts and a certain generalization of the non-negative rank of an entry-wise non-negative matrix.

Definition 4: If S is an $a \times b$ matrix with non-negative entries, the $\mathbb{S}^2_+$*-rank* of S is the smallest p such that

$$S_{ij} = \sum_{k=1}^{p} \langle A_{ik}, B_{jk} \rangle$$

where $A_{ik}, B_{jk} \in \mathbb{S}^2_+$ for all $1 \leq i \leq a$, $1 \leq j \leq b$ and $1 \leq k \leq p$.

There is a connection between $(\mathbb{S}^2_+)^p$ lifts of a convex set C and the $\mathbb{S}^2_+$-rank of various non-negative matrices associated with C (so-called *slack matrices* of C, see [46]).

The following result is a special case of [46, Theorem 1] expressed in the conic setting.

Theorem 4: Let $v_1, \ldots, v_b \in C$ and $\ell_1, \ldots, \ell_a \in C^* = \{\ell \in \mathbb{R}^n : \langle \ell, v \rangle \geq 0 \quad \text{for all } v \in C\}$. If C has a proper $(\mathbb{S}^2_+)^p$ lift, then the non-negative matrix with entries $S_{ij} = \langle \ell_i, v_j \rangle$ has $\mathbb{S}^2_+$-rank at most p.

One implication of this result, is that a lower bounds on the $\mathbb{S}^2_+$-rank of any S constructed as in Theorem 4 gives a lower bound on the smallest p for which C has an $(\mathbb{S}^2_+)^p$-lift. To show that C has no $(\mathbb{S}^2_+)^p$-lift for any p, it is enough to find a sequence of points $v_1, v_2, \ldots \in C$ and $\ell_1, \ell_2, \ldots \in C^*$, such that the corresponding sequence of non-negative matrices S (of growing dimensions) also has growing $\mathbb{S}^2_+$-rank.

B. Limitations of $(\mathbb{S}^2_+)^p$-lifts

We now state the main results of [45], establishing fundamental limitations on the convex sets that can be exactly expressed using LMIs with only 2×2 blocks.

Theorem 5: There is no finite p such that $PSD_{1,4}$, the cone of non-negative univariate quartic polynomials, has an $(\mathbb{S}^2_+)^p$-lift.

Since $PSD_{1,4} = SOS_{1,4}$ is the image of $\mathbb{S}^3_+$ under a linear map (this follows from (1) as Q is of size 3×3), the following is a direct consequence of Theorem 5.

Corollary 1 ([45, Theorem 1]): There is no finite p such that $\mathbb{S}^3_+$, the cone of 3×3 positive semidefinite matrices, has an $(\mathbb{S}^2_+)^p$-lift.

Providing the proof of these results is well beyond the scope of this article. We will, however, sketch some of the ingredients.

First, we note that $PSD^*_{1,4} = \text{cone}\{(1, t, t^2, t^3, t^4) : t \in \mathbb{R}\}$. This essentially follows directly from the definition of a non-negative polynomial and the definition of the dual cone. Define a sequence of points $\ell_j := (1, j, j^2, j^3, j^4) \in PSD^*_{1,4}$ for $j = 1, 2, \ldots$ and a collection of non-negative quartic polynomials

$$v_{\{i_1,i_2\}}(t) = \left[(i_1 - i_2)(i_1 - t)(i_2 - t)\right]^2,$$

indexed by pairs of positive integers $\{i_1, i_2\}$. Then define a sequence of $S^{(1)}, S^{(2)}, \ldots, S^{(k)}, \ldots$ of $\binom{k}{2} \times k$ non-negative matrices of the form

$$S^{(k)}_{\{i_1,i_2\},j} = v_{\{i_1,i_2\}}(j) = \left[(i_1 - i_2)(i_1 - j)(i_2 - j)\right]^2$$

where $1 \leq i_1 < i_2 \leq k$ and $1 \leq j \leq k$. The aim is to show that the $\mathbb{S}^2_+$-rank of the $S^{(k)}$ grows with k. It then follows from the discussion following the statement of Theorem 4 that $PSD_{1,4} = SOS_{1,4}$ does not have a $(\mathbb{S}^2_+)^p$-lift for any positive integer p.

Fawzi's argument makes crucial use of the sparsity pattern of these matrices, and in particular certain relationships between the sparsity patterns of $S^{(k)}$ and $S^{(k')}$ for different values of k and k'. In particular, he shows that a certain combinatorial lower bound on the $\mathbb{S}^2_+$-rank of $S^{(k)}$ must grow with k, and so the $\mathbb{S}^2_+$-rank itself must grow with k.

C. Challenges

We conclude this section with a discussion of some challenges related to understanding the limitations of approximating SOS cones with convex cones having $(\mathbb{S}^2_+)^p$-lifts.

1) Lower bounds on approximation quality: Fawzi's result suggests that if we want to model SOS cones, in general, using cones that have $(\mathbb{S}^2_+)^p$-lifts, approximation is necessary. One approximation strategy uses SDSOS cones, but it is conceivable that a significantly better approach exists. For a given notion of approximation and approximation level ϵ, it would be very interesting to produce lower bounds on p, such that a given SOS cone can be ϵ-approximated by a convex cone having a $(\mathbb{S}^2_+)^p$-lift. Is there any reasonable notion of approximation under which SDSOS is an optimal approximation in this sense?

2) Strategies to construct approximate lifts: On the positive side, how should we go about systematically approximating SOS cones with convex cones having $(\mathbb{S}^2_+)^p$-lifts, with p growing mildly with approximation quality? Are there ideas from classical constructive approximation theory that can be applied in this setting? A concrete question in this direction is the following:

> For a given approximation quality ϵ, how large must p be so that we can ϵ-approximate the cone of univariate non-negative polynomials of degree $d \geq 4$ with a convex cone having a $(\mathbb{S}^2_+)^p$-lift?

As an example of positive results in this broad direction, recent work [48] develops an approach for constructing high-quality approximations, with small $(\mathbb{S}^2_+)^p$-lifts, of relative entropy cones.

3) Techniques to lower bound $\mathbb{S}^2_+$-rank: For a non-negative matrix S there are numerous notions of cone rank. One example is $\mathbb{S}^2_+$-rank, defined in Section IV-A above; another is the *non-negative rank* (equivalent to $\mathbb{S}^1_+$-rank in our notation); another is *PSD rank* (see, e.g., [49], for a definition and survey related to its properties). These have interpretations in terms of modeling convex bodies in terms of second-order cone programs, linear programs, and (general) semidefinite programs, respectively. Recently, there has been considerable interest, in a number of fields, in finding lower bounds on various notions of cone rank such as these. Techniques for bounding the non-negative rank are most developed, with lower bounds based on combinatorial tools [50], ideas from information theory [51], and systematic computational methods [52], being available. On the other

end of the spectrum, lower bounds on the PSD rank of non-negative matrices seem much more challenging (see the survey [49] for a discussion), although there has been recent progress for some very specific matrices related to combinatorial optimization problem [53].

The $\mathbb{S}^2_+$-rank in some ways behaves like the non-negative rank (because of the inherent product structure), and in some ways like the PSD rank. It would be very interesting to see which of the combinatorial tools applicable to non-negative rank can be modified to this setting. On the other hand, developing methods for bounding the $\mathbb{S}^2_+$-rank that are distinct from the approaches used for non-negative rank, may provide a path towards understanding the PSD rank in general.

V. Conclusion

In this paper, we have reviewed two new classes of techniques that aim to improve the scalability of SOS programs. First, two efficient first-order methods based on ADMM were proposed to solve the SDPs arising from SOS programs efficiently. Both of these techniques exploit the underlying sparsity to increase the computational efficiency. Second, we introduce techniques that replace the semidefinite program underlying any SOS program by more tractable convex programs such as linear or second-order cone programs. For this strategy, we first inner approximate the set of positive semidefinite matrices by the set of diagonally dominant matrices (resp. scaled diagonally dominant matrices) which is LP-representable (resp. SOCP representable). We then iteratively improve on these initial approximations while staying in the realm of LP and SOCP-representable sets. Finally, we reviewed recent results relating to how well one can represent the cone of SOS polynomials by SDPs involving only small blocks. We focused on second order representable cones, i.e., cones that involve SDPs with blocks of size bounded by 2. We presented a recent result that states that one cannot even represent the set of nonnegative univariate quartics using these kinds of cones. This leaves open the question of how well one can approximate SOS polynomials with SDPs involving only small blocks.

References

[1] B. Reznick, "Some concrete aspects of Hilbert's 17th problem," in *Contemporary Mathematics*. American Mathematical Society, 2000, vol. 253, pp. 251–272.

[2] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Mathematical Programming*, vol. 96, no. 2, Ser. B, pp. 293–320, 2003.

[3] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 796–817, 2001.

[4] N. Gvozdenović and M. Laurent, "Semidefinite bounds for the stability number of a graph via sums of squares of polynomials," *Mathematical Programming*, vol. 110, no. 1, pp. 145–173, 2007.

[5] B. Barak and D. Steurer, "Sum-of-squares proofs and the quest toward optimal algorithms," in *Proceedings of the Inter. Congress of Mathematicians. arXiv:1404.5236*, 2014.

[6] P. A. Parrilo, "Polynomial games and sum of squares optimization," 2006.

[7] D. Bertsimas, D. A. Iancu, and P. A. Parrilo, "A hierarchy of near-optimal policies for multistage adaptive optimization," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2809–2824, 2011.

[8] B. Barak, J. A. Kelner, and D. Steurer, "Dictionary learning and tensor decomposition via the sum-of-squares method," *arXiv preprint arXiv:1407.1543*, 2014.

[9] A. Magnani, S. Lall, and S. Boyd, "Tractable fitting with convex polynomials via sum of squares," 2005.

[10] M. Roozbehani, "Optimization of Lyapunov invariants in analysis and implementation of safety-critical software systems," Ph.D. dissertation, Massachusetts Institute of Technology, 2008.

[11] M. Roozbehani, A. Megretski, and E. Feron, "Convex optimization proves software correctness," in *American Control Conference, 2005. Proceedings of the 2005*. IEEE, 2005, pp. 1395–1400.

[12] R. Tae, B. Dumitrescu, and L. Vandenberghe, "Multidimensional FIR filter design via trigonometric sum-of-squares optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 641–650, 2007.

[13] A. C. Doherty, P. A. Parrilo, and F. M. Spedalieri, "Distinguishing separable and entangled states," *Physical Review Letters*, vol. 88, no. 18, 2002.

[14] J. Harrison, "Verifying nonlinear real formulas via sums of squares," in *Theorem Proving in Higher Order Logics*. Springer, 2007, pp. 102–118.

[15] A. Ataei-Esfahani and Q. Wang, "Nonlinear control design of a hypersonic aircraft using sum-of-squares methods," in *Proceedings of the American Control Conference*. IEEE, 2007, pp. 5278–5283.

[16] A. Chakraborty, P. Seiler, and G. J. Balas, "Susceptibility of F/A-18 flight controllers to the falling-leaf mode: Nonlinear analysis," *Journal of guidance, control, and dynamics*, vol. 34, no. 1, pp. 73–85, 2011.

[17] P. Seiler, G. J. Balas, and A. K. Packard, "Assessment of aircraft flight controllers using nonlinear robustness analysis techniques," in *Optimization Based Clearance of Flight Control Laws*. Springer, 2012, pp. 369–397.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[19] Y. Zheng, G. Fantuzzi, and A. Papachristodoulou, "Exploiting sparsity in the coefficient matching conditions in sum-of-squares programming using ADMM," *IEEE Control Systems Letters*, vol. 1, no. 1, pp. 80–85, 2017.

[20] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Math. Program.*, vol. 96, no. 2, pp. 293–320, 2003.

[21] B. Reznick *et al.*, "Extremal PSD forms with few terms," *Duke Math. J.*, vol. 45, no. 2, pp. 363–374, 1978.

[22] J. Löfberg, "Pre-and post-processing sum-of-squares programs in practice," *IEEE Trans. Autom. Control*, vol. 54, no. 5, pp. 1007–1011, 2009.

[23] K. Gatermann and P. A. Parrilo, "Symmetry groups, semidefinite programs, and sums of squares," *J. Pure Appl. Algebra*, vol. 192, no. 1, pp. 95–128, 2004.

[24] G. Chesi, "On the complexity of SOS programming: formulas for general cases and exact reductions," in *Control Systems (SICE ISCS), 2017 SICE International Symposium on*. IEEE, 2017, pp. 1–6.

[25] M. Fukuda, M. Kojima, K. Murota, and K. Nakata, "Exploiting sparsity in semidefinite programming via matrix completion I: General framework," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 647–674, 2001.

[26] H. Waki, S. Kim, M. Kojima, and M. Muramatsu, "Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 218–242, 2006.

[27] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn, "Fast ADMM for semidefinite programs with chordal sparsity," in *American Control Conference (ACC)*. IEEE, 2017, pp. 3335–3340.

[28] J. R. Blair and B. Peyton, "An introduction to chordal graphs and clique trees," in *Graph theory and sparse matrix computation*. Springer, 1993, pp. 1–29.

[29] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz, "Positive definite completions of partial hermitian matrices," *Linear algebra and its applications*, vol. 58, pp. 109–124, 1984.

[30] J. Agler, W. Helton, S. McCullough, and L. Rodman, "Positive semidefinite matrices with a given sparsity pattern," *Linear algebra and its applications*, vol. 107, pp. 101–149, 1988.

[31] R. P. Mason and A. Papachristodoulou, "Chordal sparsity, decomposing SDPs and the Lyapunov equation," in *American Control Conference (ACC), 2014*. IEEE, 2014, pp. 531–537.

[32] Y. Zheng, R. P. Mason, and A. Papachristodoulou, "Scalable design of structured controllers using chordal decomposition," *IEEE Transactions on Automatic Control, to appear*, vol. PP, p. 99, 2017.

[33] ——, "A chordal decomposition approach to scalable design of structured feedback gains over directed graphs," in *IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 6909–6914.

[34] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn, "Chordal decomposition in operator-splitting methods for sparse semidefinite programs," *arXiv preprint arXiv:1707.05058*, 2017.

[35] ——, "Fast ADMM for homogeneous self-dual embeddings of sparse SDPs," in *Proc. 20th World Congr. Int. Fed. Autom. Control*, Toulouse, France, 2017, pp. 8741–8746. [Online]. Available: http://arxiv.org/abs/1611.01828

[36] ——, "CDCS: Cone decomposition conic solver, version 1.1," https://github.com/oxfordcontrol/CDCS, Sept. 2016.

[37] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optim. Methods Softw.*, vol. 11, no. 1-4, pp. 625–653, 1999.

[38] D. Henrion and J.-B. Lasserre, "GloptiPoly: Global optimization over polynomials with MATLAB and SeDuMi," *ACM Tans. Math. Softw. (TOMS)*, vol. 29, no. 2, pp. 165–194, 2003.

[39] A. A. Ahmadi and A. Majumdar, "DSOS and SDSOS: more tractable alternatives to sum of squares and semidefinite optimization," 2017, available on arxiv: https://arxiv.org/abs/1706.02586.

[40] S. A. Gershgorin, "Uber die Abgrenzung der Eigenwerte einer Matrix," *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et na*, no. 6, pp. 749–754, 1931.

[41] A. A. Ahmadi, S. Dash, and G. Hall, "Optimization over structured subsets of positive semidefinite matrices via column generation," *In Press, Discrete Optimization*, 2016.

[42] A. A. Ahmadi and G. Hall, "Sum of squares basis pursuit with linear and second order cone programming," in *Algebraic and Geometric Methods in Discrete Mathematics*. Contemporary Mathematics, 2016.

[43] ——, "On the construction of converging hierarchies for polynomial optimization based on certificates of global positivity," 2017, available on arxiv.

[44] G. Barker and D. Carlson, "Cones of diagonally dominant matrices," *Pacific Journal of Mathematics*, vol. 57, no. 1, pp. 15–32, 1975.

[45] H. Fawzi, "On representing the positive semidefinite cone using the second-order cone," *arXiv preprint arXiv:1610.04901*, 2016.

[46] J. Gouveia, P. A. Parrilo, and R. R. Thomas, "Lifts of convex sets and cone factorizations," *Mathematics of Operations Research*, vol. 38, no. 2, pp. 248–264, 2013.

[47] M. Yannakakis, "Expressing combinatorial optimization problems by linear programs," *Journal of Computer and System Sciences*, vol. 43, no. 3, pp. 441–466, 1991.

[48] H. Fawzi, J. Saunderson, and P. A. Parrilo, "Semidefinite approximations of the matrix logarithm," *arXiv preprint arXiv:1705.00812*, 2017.

[49] H. Fawzi, J. Gouveia, P. A. Parrilo, R. Z. Robinson, and R. R. Thomas, "Positive semidefinite rank," *Mathematical Programming*, vol. 153, no. 1, pp. 133–177, 2015.

[50] S. Fiorini, V. Kaibel, K. Pashkovich, and D. O. Theis, "Combinatorial bounds on nonnegative rank and extended formulations," *Discrete mathematics*, vol. 313, no. 1, pp. 67–83, 2013.

[51] G. Braun, R. Jain, T. Lee, and S. Pokutta, "Information-theoretic approximations of the nonnegative rank," *computational complexity*, pp. 1–51, 2014.

[52] H. Fawzi and P. A. Parrilo, "Self-scaled bounds for atomic cone ranks: applications to nonnegative rank and cp-rank," *Mathematical Programming*, vol. 158, no. 1-2, pp. 417–465, 2016.

[53] J. R. Lee, P. Raghavendra, and D. Steurer, "Lower bounds on the size of semidefinite programming relaxations," in *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*. ACM, 2015, pp. 567–576.