Basis selection for SOS programs via facial reduction and polyhedral approximations

Frank Permenter[1] Pablo A. Parrilo[2]

***Abstract*— We develop a monomial basis selection procedure for sum-of-squares (SOS) programs based on facial reduction. Using linear programming and polyhedral approximations, the proposed technique finds a face of the SOS cone containing the feasible set of a given SOS program. The identified face in turn identifies a set of monomials that can be used to convert the SOS program into a semidefinite program (SDP). The technique can be viewed as a generalization of standard parsing algorithms for monomial basis selection. As we illustrate with examples, the proposed method can lead to smaller SDPs that are simpler to solve.**

I. Introduction

Proving non-negativity of multivariate polynomials plays a crucial role in many areas. For instance, in Lyapunov analysis, one can prove stability of a polynomial dynamical system $\dot{x} = f(x)$ by showing a polynomial Lyapunov function $V(x)$ satisfies

$$V(x) \geq 0$$
$$-\dot{V}(x) = -\nabla V^T(x) f(x) \geq 0$$

for all x. Unfortunately, deciding if a polynomial is globally non-negative is NP-hard. A popular alternative is to instead show $V(x)$ and $-\dot{V}(x)$ are sums-of-squares, a sufficient condition for non-negativity more easily checked.

To demonstrate a polynomial is a sum-of-squares, one can solve a *sum-of-squares program* (SOS program)—a convex optimization problem over the cone of polynomials that are sums-of-squares. To solve an SOS program, one first converts it into a semidefinite program (SDP) using an appropriately-specified polynomial basis and then solves the resulting SDP using, for instance, interior point methods (see, for example, Chapter 3 of [1]). The number of polynomials in the basis determines the size of the resulting SDP; hence, cleverly selecting the basis using available problem structure is crucial for solving the SOS program efficiently.

In this paper, we develop a new algorithm for basis selection that generalizes and improves existing approaches, possibly leading to smaller SDPs. While existing algorithms (implemented in parsers such as [14], [6]) leverage polynomial sparsity *explicit* in the problem description, our technique goes further, leveraging additional sparsity *implied* by the sum-of-squares constraint. For instance, given the problem of finding $(c_1, c_2) \in \mathbb{R}^2$ such that

$$g(x) = c_1 x_1^4 + c_2 x_1^2 x_2^2 - 3c_1 x_2^4$$

is a sum-of-squares, our procedure deduces that $g(x)$ is necessarily of the form

$$g(x) = c_2 x_1^2 x_2^2.$$

To detect implicit sparsity, our procedure introduces a specific polyhedral approximation of the SOS cone and performs an iteration of *facial reduction*, an iterative technique for reducing the dimensionality of a given conic optimization problem [2]. The cost of this iteration is the solution of a modestly-sized linear program related to the approximation. Repeating these steps yields our procedure.

This paper is organized as follows. Section II gives background information on SOS programming and a simple facial reduction algorithm (Algorithm 1). Section III adapts this algorithm to the problem of basis selection (Algorithm 2), and contains a few relatively elementary (but to our knowledge new) results on faces of the SOS cone. Section IV compares our algorithm to existing approaches and Section V concludes with examples arising in stability analysis.

A. Prior work

Techniques for efficiently converting SOS programs into SDPs have been studied by many authors. Standard approaches involve picking a basis of monomials. An algorithm for monomial basis selection which exploits polynomial sparsity, based on Newton polytopes, is described in [10]. Other methods for exploiting sparsity are given in [5], [16], [17] and [9]. An algorithm for simultaneously exploiting sparsity and ideal structure was developed by the authors in [12]. A method for leveraging symmetry is discussed in [4]. Practical implementations of these types of techniques are discussed in [7].

As discussed in [19], existing basis selection algorithms can be interpreted as facial reduction of a particular SDP formulated to solve a given SOS program. While our work is of similar flavor, we use facial reduction to not only interpret but to also generalize these algorithms. We will also work directly at the level of polynomials and the sums-of-squares cone. In other words, we perform facial reduction *before* the SOS program is converted into an SDP.

Finally, the systematic use of approximations in facial reduction algorithms for SDP is explored extensively in [13]. This paper extends these ideas to SOS programs.

[1]F. Permenter is with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA 02139. fperment@mit.edu

[2]P. A. Parrilo is with the Laboratory For Information and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA 02139. parrilo@mit.edu

II. Preliminaries

In this section, we define our notation, formally define SOS programs, and derive a simple facial reduction algorithm we will adapt in the next section to the problem of basis selection.

A. Notation

Let $\mathbb{R}[x]_{n,2d}$ denote the vector space of polynomials in n variables with real coefficients of degree $2d$ or less, and let $\mathbb{R}[x]^*_{n,2d}$ denote the corresponding dual space. We use multi-index notation to write elements of $\mathbb{R}[x]_{n,2d}$ as a summation over a set $N \subset \mathbb{N}^n$:

$$f(x) = \sum_{\alpha \in N} c_\alpha x^\alpha, \tag{1}$$

where c_α is a real-valued coefficient and x^α is shorthand for the monomial $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$. For a polynomial $f(x)$ of the form (1), let $\mathcal{N}(f)$ denote its *Newton polytope*, defined as the convex hull of exponents in N for which c_α is non-zero. Let $\mathbf{e}_\alpha$ denote the element of $\mathbb{R}[x]^*_{n,2d}$ for which $\langle \mathbf{e}_\alpha, f \rangle$ returns the coefficient of monomial x^α for any polynomial f in $\mathbb{R}[x]_{x,2d}$. For example, if $f = \sum_\alpha c_\alpha x^\alpha$, then $\langle \mathbf{e}_\alpha, f \rangle = c_\alpha$. Finally, if $f = \sum_i f_i(x)^2$, let $c_{i,\beta}$ denote the coefficient of $f_i(x)$ corresponding to x^β.

Let $\Sigma_{n,2d} \subseteq \mathbb{R}[x]_{n,2d}$ denote the convex cone of polynomials that are sums-of-squares, i.e. $\Sigma_{n,2d}$ equals all polynomials in $\mathbb{R}[x]_{n,2d}$ that can be written in the form $\sum_i f_i^2$ for polynomials f_i. If f is a sum-of-squares, we will sometimes say simply that f *is sos*.

We also define notation for a more abstract setting useful, in part, for deriving the facial reduction algorithm. Let $\mathcal{E}$ be a finite-dimensional vector space with inner product $\langle \cdot, \cdot \rangle$. For a subset S of $\mathcal{E}$, let S^* denote its dual cone, i.e. the set of linear functionals that are non-negative on S. Let $S^\perp$ denote the orthogonal complement of the linear span of S. Let $\operatorname{convhull} S$ and $\operatorname{aff} S$ denote the convex and affine hulls of S. For an element $s \in \mathcal{E}$, let $s^\perp$ denote $\{s\}^\perp$. Finally, call a convex subset $\mathcal{F}$ of a convex cone $\mathcal{K} \subseteq \mathcal{E}$ a *face* if it satisfies

$$\frac{a+b}{2} \in \mathcal{F} \text{ and } a, b \in \mathcal{K} \Rightarrow a, b \in \mathcal{F}.$$

B. Sums-of-squares programs and basis selection

An SOS program optimizes a linear function over an affine subspace $\mathcal{A}$ of polynomials intersected with the cone of sums-of-squares. It can be written as follows:

$$\begin{array}{ll} \underset{u}{\text{minimize}} & \sum_{i=1}^m w_i u_i \\ \text{subject to} & f = g_0(x) + \sum_{i=1}^m u_i g_i(x) \qquad (f \in \mathcal{A}) \\ & f \text{ is sos} \qquad\qquad\qquad\quad (f \in \Sigma_{n,2d}), \end{array}$$

where $u \in \mathbb{R}^m$ is the decision variable, $g_i(x) \in \mathbb{R}[x]_{n,2d}$ are fixed polynomials generating $\mathcal{A}$, and $w \in \mathbb{R}^m$ defines a linear objective function.

This optimization problem can be converted into an SDP by specifying a *monomial basis* for the polynomials f_i appearing in a sum-of-squares decomposition

$$f = \sum_i f_i^2.$$

Representing this basis using a finite set of monomial exponents $M \subset \mathbb{N}^n$ and introducing an $|M| \times |M|$ (symmetric) matrix decision variable Q yields the following SDP:

$$\begin{array}{ll} \underset{u,Q}{\text{minimize}} & \sum_{i=1}^m w_i u_i \\ \text{subject to} & f = g_0(x) + \sum_{i=1}^m u_i g_i(x) \quad (f \in \mathcal{A}) \\ & f = \sum\limits_{\alpha,\beta \in M} q_{\alpha\beta} x^\alpha x^\beta \\ & Q \succeq 0, \end{array} \tag{2}$$

where the entries $q_{\alpha\beta}$ of Q are indexed by exponents in M and the last two constraints imply that $f = \sum_i f_i^2$ for polynomials f_i of the form $\sum_{\beta \in M} c_{i,\beta} x^\beta$ (as can be seen by Cholesky-factorizing Q).

Picking the set M (preferably, of minimum cardinality) so that the set of feasible f equals $\mathcal{A} \cap \Sigma_{n,2d}$ is the job of the parsing algorithms we seek to improve upon. Standard techniques exploit the sparsity pattern of f assuming non-zero u_i. We will improve these techniques using the theory of facial reduction, which we discuss in the next section.

C. Cone programming and facial reduction

Consider the general cone program

$$\text{minimize } \langle w, x \rangle \text{ subject to } x \in \mathcal{A} \cap \mathcal{K},$$

where $\mathcal{K} \subseteq \mathcal{E}$ is a convex cone and $\mathcal{A} \subseteq \mathcal{E}$ is an affine subspace. This problem is said to be *feasible* if $\mathcal{A} \cap \mathcal{K}$ is non-empty and *strictly feasible* if $\mathcal{A} \cap \operatorname{relint} \mathcal{K}$ is non-empty. If this problem is feasible but not strictly feasible, an element of the dual cone $\mathcal{K}^*$ certifies this fact and can be used to formulate an equivalent problem over a lower dimensional face of $\mathcal{K}$. To see this, first consider the following lemma (a proof of which can be found in [13]):

Lemma 1: Let $\mathcal{K}$ be a convex cone and $\mathcal{A}$ be an affine subspace for which $\mathcal{A} \cap \mathcal{K}$ is non-empty. The following statements are equivalent.

1) $\mathcal{A} \cap \operatorname{relint} \mathcal{K}$ is empty,
2) There exists $s \in \mathcal{K}^* \setminus \mathcal{K}^\perp$ for which $s^\perp$ contains $\mathcal{A}$.

The vector s in statement (2) of the above is called a *reducing certificate* for $\mathcal{A} \cap \mathcal{K}$. Since intersection with $s^\perp$ leaves $\mathcal{A} \cap \mathcal{K}$ unchanged, s defines an equivalent optimization problem with feasible set $\mathcal{A} \cap \mathcal{K} \cap s^\perp$. Since s is in $\mathcal{K}^*$, $\mathcal{K} \cap s^\perp$ has special structure. In particular, it is a face of $\mathcal{K}$. Since faces of convex cones are also convex cones, taking $\mathcal{F} = \mathcal{K} \cap s^\perp$ yields an equivalent cone program:

$$\text{minimize } \langle w, x \rangle \text{ subject to } \quad x \in \mathcal{A} \cap \mathcal{F} \ .$$

Note these steps can be repeated if one can find a reducing certificate for $\mathcal{A} \cap \mathcal{F}$. Indeed, it may be possible to find a chain of nested faces $\mathcal{F}_i$

$$\mathcal{K} = \mathcal{F}_0 \supset \mathcal{F}_1 \supset \cdots \supset \mathcal{F}_{n-1} \supset \mathcal{F}_n,$$

where each $\mathcal{F}_i$ contains $\mathcal{A} \cap \mathcal{K}$. This is done by taking $\mathcal{F}_{i+1} = \mathcal{F}_i \cap s_i^\perp$ for an $s_i \in \mathcal{F}_i^* \setminus \mathcal{F}_i^\perp$ that is orthogonal to $\mathcal{A}$. Producing this chain is called *facial reduction*. A simple algorithm for facial reduction is given in Algorithm 1 (which closely resembles algorithms of [11] and [18]). In the next section, we adapt this algorithm to the problem of selecting a monomial basis.

Algorithm 1: Facial reduction algorithm for a cone program with feasible set $\mathcal{A} \cap \mathcal{K}$.

Initialize: $\mathcal{F}_0 \leftarrow \mathcal{K}, i = 0$

repeat

1) Find $s_i \in \mathcal{F}_i^* \setminus \mathcal{F}_i^\perp$ for which $s_i^\perp$ contains $\mathcal{A}$ $\quad (\star)$
2) Set $\mathcal{F}_{i+1} = \mathcal{F}_i \cap s_i^\perp$
3) Increment counter i

until $(\star)$ is infeasible;

III. Basis selection via facial reduction

In this section, we modify Algorithm 1 to compute monomial bases for SOS programs. Given an affine subspace $\mathcal{A}$ of polynomials, the modified algorithm finds a chain of faces containing $\mathcal{A} \cap \Sigma_{n,2d}$, where each face can be represented (in a specific way defined below) using a set of monomials M_i. Letting $\Sigma(M_i)$ denote the i^{th} face, the algorithm introduces a specific polyhedral approximation $\hat{\Sigma}(M_i)$ to $\Sigma(M_i)$ and finds a reducing certificate s_i by solving a linear program. From this reducing certificate, a new set of monomials M_{i+1} is computed representing the face $\Sigma(M_{i+1}) := \Sigma(M_i) \cap s_i^\perp$. The resulting procedure is explicitly given by Algorithm 2.

Algorithm 2: Monomial selection algorithm for an SOS program with feasible set $\mathcal{A} \cap \Sigma_{n,2d}$, where $\mathcal{A} := \left\{ g_0(x) + \sum_{j=1}^m u_j g_j(x) : u \in \mathbb{R}^m \right\}$.

Input: Generators $g_j(x)$ of an affine subspace $\mathcal{A}$, an initial set of monomials M_0 for which $\Sigma(M_0)$ contains $\mathcal{A} \cap \Sigma_{n,2d}$

Output: A reduced set of monomials M_i for which $\Sigma(M_i)$ contains $\mathcal{A} \cap \Sigma_{n,2d}$

Initialize: $i = 0$

repeat

1) Find $s_i \in \hat{\Sigma}(M_i)^* \setminus \hat{\Sigma}(M_i)^\perp$ for which $s_i^\perp$ contains $\mathcal{A}$ by solving the LP:

$$\begin{aligned} &\text{Find } s_i, \lambda_{\geq 0}, p \in \hat{\Sigma}(M_i)^\perp \\ &\text{subject to} \\ &\textstyle\sum_{\alpha \in M_i^+} \lambda_{2\alpha} = 1 \qquad (\star) \\ &s_i = p + \textstyle\sum_{\alpha \in M_i^+} \lambda_{2\alpha} \mathbf{e}_{2\alpha} \\ &\langle s_i, g_j(x) \rangle = 0 \;\; j \in \{0, \ldots, m\} \end{aligned}$$

2) Set $M_{i+1} = M_i \setminus \{\alpha : \lambda_{2\alpha} > 0\}$
3) Increment counter i

until $(\star)$ is infeasible;

To explain the behavior of Algorithm 2, we begin by defining types of faces that can be represented using monomial exponents. We then show how the steps of Algorithm 2 relate to the steps of Algorithm 1.

A. Types of faces

We consider two types of faces that can be represented using a set M of monomial exponents. These faces are sets of polynomials $\sum_i f_i^2$ that arise by fixing a monomial basis for the f_i, i.e. they are sets of the following form:

Definition 1: For a subset M of $\mathbb{N}^n$, let

$$\Sigma(M) := \left\{ \sum_i f_i^2 : f_i = \sum_{\alpha \in M} c_{\alpha,i} x^\alpha, c_{\alpha,i} \in \mathbb{R} \right\}.$$

We now give two conditions for which $\Sigma(M)$ is a face of $\Sigma_{n,2d}$, yielding two types of faces.

Theorem 1 (Type I Face): Let M be a finite subset of $\mathbb{N}^n$ and let $d := \max\{\sum_{i=1}^n z_i : z \in M\}$. If $\text{convhull}(M) \cap \mathbb{N}^n = M$, then $\Sigma(M)$ is a face of $\Sigma_{n,2d}$.

Definition 2: For a subset M of $\mathbb{N}^n$, let M^+ be the subset of points that cannot be written as the midpoint of distinct points in M, i.e.

$$M^+ := M \setminus \left\{ \frac{\alpha + \beta}{2} : \alpha, \beta \in M, \alpha \neq \beta \right\}.$$

Theorem 2 (Type II Face): Let M be a finite subset of $\mathbb{N}^n$ and let $\beta \in M^+$. If $\Sigma(M)$ is a face of $\Sigma_{n,2d}$, then $\Sigma(M \setminus \beta)$ is a face of $\Sigma_{n,2d}$.

We see that Type I faces arise by imposing conditions on M and can hence be used to *initialize* Algorithm 2. A Type II face, on the other hand, is defined recursively; if $\Sigma(M)$ is a face then $\Sigma(M \setminus \beta)$ is a Type II face for any β in the set M^+. A Type II face will be the output of each iteration of Algorithm 2. We now prove correctness of these theorems.

1) Proof of Theorem 1: To prove Theorem 1, we first recall a well-known result of [15] relating the Newton polytope of a polynomial $f = \sum_i f_i^2$ to the Newton polytopes of the f_i:

Lemma 2: If $f(x) = \sum_i f_i(x)^2$, then $\mathcal{N}(f_i) \subseteq \frac{1}{2}\mathcal{N}(f)$.

From this result, we have the following lemma:

Lemma 3: Let N be a finite subset of $\mathbb{N}^n$ and let $2d := \max\{\sum_{i=1}^n z_i : z \in N\}$. The set

$$\{f : f \text{ is sos}, \mathcal{N}(f) \subseteq \text{convhull}\, N\}$$

is a face of $\Sigma_{n,2d}$.

Proof: Suppose f is in the mentioned set and also suppose that $f = g + h$ for polynomials $g = \sum_i g_i(x)^2$

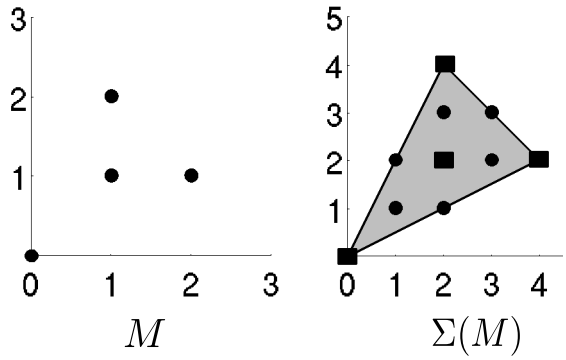

Fig. 1: Example of a Type I face $\Sigma(M)$. The affine span of $\Sigma(M)$ contains all polynomials with Newton polytope contained in the shaded region. The square markers corresponding to the set $2M^+$ and label non-negative coefficients.

and $h = \sum_i h_i(x)^2$. Since $f = \sum_i g_i(x)^2 + \sum_i h_i(x)^2$, we must have, per Lemma 2, that $\mathcal{N}(g_i)$ and $\mathcal{N}(h_i)$ are subsets of $\frac{1}{2}\mathcal{N}(f)$. But this implies that $\mathcal{N}(g)$ and $\mathcal{N}(h)$ are both subsets of $\mathcal{N}(f)$, which shows g and h are in the mentioned set, proving the claim. ∎

We are now ready to prove Theorem 1. To do this, it suffices to show that $\Sigma(M)$ equals the set of sums-of-squares polynomials f satisfying $\mathcal{N}(f) \subseteq \text{convhull}(M+M)$ when M satisfies the hypothesis of Theorem 1. This is demonstrated by the following.

Lemma 4: Let M be a finite subset of $\mathbb{N}^n$. If $\text{convhull}(M) \cap \mathbb{N}^n = M$, then

$$\Sigma(M) = \{f : f \text{ is sos}, \mathcal{N}(f) \subseteq \text{convhull}(M+M)\}.$$

Proof: We first note that $\text{convhull}(M) = \frac{1}{2}\text{convhull}(M+M)$ for any finite set M. Hence, our hypothesis implies that $\frac{1}{2}\text{convhull}(M+M) \cap \mathbb{N}^n = M$.

Now, suppose for some polynomial $f = \sum_i f_i^2$ that $\mathcal{N}(f) \subseteq \text{convhull}(M+M)$. This implies that $\mathcal{N}(f_i) \subseteq \frac{1}{2}\text{convhull}(M+M)$. But if $\frac{1}{2}\text{convhull}(M+M) \cap \mathbb{N}^n = M$, then f is in $\Sigma(M)$. In other words, $\Sigma(M)$ contains $\{f : f \text{ is sos}, \mathcal{N}(f) \subseteq \text{convhull}\, M+M\}$. The reverse containment is obvious. ∎

2) Proof of Theorem 2: To prove Theorem 2, we show $\Sigma(M \setminus \beta)$ equals the set $\Sigma(M) \cap s^\perp$ for a particular $s \in \Sigma(M)^*$ for any $\beta \in M^+$. This shows that $\Sigma(M \setminus \beta)$ is a face of $\Sigma(M)$. Since "is a face of" is a *transitive* relation, it will follow that $\Sigma(M \setminus \beta)$ is a face of $\Sigma_{n,2d}$ whenever $\Sigma(M)$ is a face of $\Sigma_{n,2d}$, proving the result.

We begin by constructing the $s \in \Sigma(M)^*$ of interest. To do this, we need the following basic result, which essentially follows from Proposition 3.7 of [3]:

Lemma 5: Let M be a finite subset of $\mathbb{N}^n$ and let $\beta \in \mathbb{N}^n$ be an element of M^+. If $f = \sum_i f_i^2$ is in $\Sigma(M)$, then f can be written as

$$f = \textstyle\sum_i (c_{i,\beta})^2 x^{2\beta} + \text{terms involving } (M+M) \setminus 2\beta,$$

where $f_i = \sum_{\alpha \in M} c_{i,\alpha} x^\alpha$.

Using this lemma, we see that for any $\beta \in M^+$ and $f \in \Sigma(M)$,

$$\langle \mathbf{e}_{2\beta}, f \rangle = \sum_i (c_{i,\beta})^2 \geq 0.$$

In other words, $\mathbf{e}_{2\beta}$ is in $\Sigma(M)^*$, demonstrating $\Sigma(M) \cap \mathbf{e}_{2\beta}^\perp$ is a face of $\Sigma(M)$. We now show $\Sigma(M \setminus \beta)$ equals this face, which proves Theorem 2.

Lemma 6: Let M be a finite subset of $\mathbb{N}^n$ and let $\beta \in \mathbb{N}^n$ be an element of M^+. The following relationship holds:

$$\Sigma(M) \cap \mathbf{e}_{2\beta}^\perp = \Sigma(M \setminus \beta).$$

Proof: If f is in $\Sigma(M \setminus \beta)$ for β in M^+, then f does not have a term involving $x^{2\beta}$ by definition of M^+, i.e. $\langle \mathbf{e}_{2\beta}, f \rangle = 0$. To see the reverse inclusion, suppose f is in $\Sigma(M) \cap \mathbf{e}_{2\beta}^\perp$, i.e.

$$\langle \mathbf{e}_{2\beta}, f \rangle = \sum_i (c_{i,\beta})^2 = 0.$$

This shows $c_{i,\beta}$ vanishes for all i which demonstrates f is in $\Sigma(M \setminus \beta)$. ∎

B. Step #1: Find reducing certificate

At each iteration i, the first step of Algorithm 1 finds a reducing certificate $s_i \in \mathcal{F}_i^*$ orthogonal to $\mathcal{A}$, which requires solving a cone program over $\mathcal{F}_i^*$. Despite the special structure of $\Sigma(M_i)$, optimizing over $\Sigma(M_i)^*$ is expensive—hence, it is impractical for a basis selection algorithm to implement this step directly. In addition, the set $\Sigma(M_i) \cap s_i^\perp$ is not in general a set of the form $\Sigma(M_{i+1})$—i.e. for arbitrary s_i, the set $\Sigma(M_i) \cap s_i^\perp$ cannot be represented using a set of monomials M_{i+1}. To resolve these issues, the first step of Algorithm 2 searches for s_i using an *inner approximation* of $\Sigma(M_i)^*$. This subset is defined as the dual cone of a polyhedral *outer approximation* of $\Sigma(M_i)$ given by the following.

Lemma 7: Let M be a finite subset of $\mathbb{N}^n$. The following polyhedral cone

$$\hat{\Sigma}(M) := \left\{ f \in \text{aff}(\Sigma(M)) : \langle \mathbf{e}_{2\alpha}, f \rangle \geq 0 \ \ \forall \alpha \in M^+ \right\}$$

has dual cone

$$\hat{\Sigma}(M)^* = \left\{ \sum_{\alpha \in M^+} \lambda_{2\alpha} \mathbf{e}_{2\alpha} : \lambda_{2\alpha} \geq 0 \right\} + \Sigma(M)^\perp,$$

and the following inclusions hold:

$$\Sigma(M) \subseteq \hat{\Sigma}(M) \qquad \hat{\Sigma}(M)^* \subseteq \Sigma(M)^*.$$

Replacing $\Sigma(M)^*$ with $\hat{\Sigma}(M)^*$ reduces the search for reducing certificates to a *linear program*, given explicitly as LP $(\star)$ in Algorithm 2. Supposing $\mathcal{A} \cap \hat{\Sigma}(M)$ is non-empty, it is easy to see (using Lemma 1) that this LP is feasible if and only if $\mathcal{A} \cap \text{relint}\, \hat{\Sigma}(M)$ is empty–i.e. if the inequalities defining $\hat{\Sigma}(M)$ cannot be strictly satisfied. In other words, Algorithm 2 finds a reducing certificate if a specific relaxation of the SOS program is not strictly feasible.

C. Step #2: Compute new face

The second step of Algorithm 1 computes a new face via the intersection $\mathcal{F}_i \cap s_i^\perp$. It turns out for a face of the form $\Sigma(M_i)$ and reducing certificate $s_i \in \hat{\Sigma}(M_i)^*$, this intersection equals $\Sigma(M_i \setminus \Lambda)$ for some $\Lambda \subseteq M_i$. In other words, this step is implemented in Algorithm 2 by removing particular elements of M_i. Consider the following.

Lemma 8: Let M be a finite subset of $\mathbb{N}^n$ and take Λ to be any subset of M^+. If for $\lambda_{2\alpha} > 0$ and $p \in \Sigma(M)^\perp$ we have that

$$s = p + \sum_{\alpha \in \Lambda} \lambda_{2\alpha} \mathbf{e}_{2\alpha},$$

then

$$\Sigma(M) \cap s^\perp = \Sigma(M \setminus \Lambda).$$

Proof: If f is in $\Sigma(M)$ then $\langle s, f \rangle = 0$ if and only if $\langle \mathbf{e}_{2\alpha}, f \rangle = 0$ for each $\alpha \in \Lambda$. This follows since $\mathbf{e}_{2\alpha}$ is in $\Sigma(M)^*$ for each $\alpha \in \Lambda$. In other words,

$$f \in \Sigma(M) \cap s^\perp \Rightarrow f \in \Sigma(M) \cap \left(\cap_{\alpha \in \Lambda} \mathbf{e}_{2\alpha}^\perp \right).$$

Using Lemma 6 to compute the intersection on the right shows that f is in $\Sigma(M \setminus \Lambda)$. The reverse inclusion follows since Λ is a subset of M^+. ∎

D. Extension to multiple SOS constraints

It is common for an SOS program to contain several, perhaps coupled, SOS constraints. In this setting, one is interested in finding $u \in \mathbb{R}^m$ such that for $j \in \{1, \ldots, q\}$ the polynomials $p_j(x) := g_{j0}(x) + \sum_{k=1}^{m} u_k g_{jk}(x)$ are sums-of-squares. Notice as u varies over $\mathbb{R}^m$, the polynomials $g_{jk}(x)$ generate an affine subspace $\mathcal{A}$ of the Cartesian product $\mathbb{R}[x]_{n,2d_1} \times \cdots \times \mathbb{R}[x]_{n,2d_q}$. The set of feasible u is described by the intersection of $\mathcal{A} \cap \mathcal{K}$, where $\mathcal{K}$ is the product of SOS cones $\Sigma_{n_1,2d_1} \times \cdots \times \Sigma_{n_q,2d_q}$.

With these observations, generalizing our algorithm is straightforward. Given initial sets of monomial exponents $M_0^1, \ldots, M_0^q$ for each SOS constraint (where $M_0^j \subset \mathbb{N}^n$), one can compute a chain of faces

$$\mathcal{F}_i := \Sigma\left(M_i^1\right) \times \cdots \times \Sigma\left(M_i^q\right) \subseteq \mathcal{K}$$

by finding s_i (orthogonal to $\mathcal{A}$) in the Cartesian product $\hat{\Sigma}\left(M_i^1\right)^* \times \cdots \times \hat{\Sigma}\left(M_i^q\right)^*$. Given s_i, a new face $\mathcal{F}_{i+1}$ is then computed by projecting s_i onto $\hat{\Sigma}\left(M_i^j\right)^*$ and computing M_{i+1}^j via Lemma 8 for each j.

IV. Comparison to previous work

In this section, we study monomial selection algorithms described in [10], [5], [7], and [16]. As we show, these methods can be viewed as a variant of our proposed algorithm in a precise sense: the set of monomials they identify equals the set of monomials identified by Algorithm 2 if it is modified to search over particular subsets of $\hat{\Sigma}\left(M_i\right)^*$. In Section IV-A, we define this subset for the Newton polytope-based algorithm given in [10]. In Section IV-B, we do the same for algorithms based on the notion of *diagonal consistency* described in [5], [7], and [16].

We perform this analysis imposing the following conditions on the subspace $\mathcal{A}$ and the initial set of monomials M_0:

- Condition 1. The affine subspace consists of a single polynomial $g_0(x) \in \mathbb{R}[x]_{n,2d}$, i.e. $\mathcal{A} := \{g_0(x)\}$, where the vertices of $\mathcal{N}(g_0)$ are vectors of even integers.
- Condition 2. The initial set of monomial exponents M_0 contains all monomials of degree d or less.

A. Monomial selection via Newton polytopes

Suppose Condition 1 holds. In this case, the monomial selection algorithm in [10] picks a set of monomials M equal to $\frac{1}{2}\mathcal{N}(g_0) \cap \mathbb{N}^n$. Algorithm 2 will output the same set of monomials if it is restricted to a subset of $\hat{\Sigma}\left(M_i\right)^*$ corresponding to vertices of $\operatorname{convhull} M_i$. Consider the following.

Proposition 1: Suppose Conditions 1-2 hold and that the search step $(\star)$ of Algorithm 2 is modified to

$$\text{Find } s_i \in R(M_i)_{\mathcal{N}} \text{ for which } s_i^{\perp} \text{ contains } \mathcal{A} \qquad (\star),$$

where

$$R(M)_{\mathcal{N}} := \left\{\mathbf{e}_{2\alpha} : \alpha \text{ is a vertex of } \operatorname{convhull} M\right\}.$$

Then, the output of Algorithm 2 equals $\frac{1}{2}\mathcal{N}(g_0) \cap \mathbb{N}^n$.

Proof: Let M denote the output. We first show M is a superset of $\frac{1}{2}\mathcal{N}(g_0) \cap \mathbb{N}^n$. We proceed by induction on M_i, the sets of monomials computed at iteration i. Condition 2 implies the base case holds, i.e. M_0 is a superset. Suppose M_i is a superset and let α be the vertex of $\operatorname{convhull} M_i$ removed at iteration i, i.e. $\mathbf{e}_{2\alpha}$ is orthogonal to $\mathcal{A}$ and $M_{i+1} = M_i \setminus \alpha$. Since the vertices of $\mathcal{N}(g_0)$ are even, $\operatorname{convhull} M_i$ contains $\frac{1}{2}\mathcal{N}(g_0)$. This implies α is either a vertex of $\frac{1}{2}\mathcal{N}(g_0)$ or not in $\frac{1}{2}\mathcal{N}(g_0)$. But if α is a vertex of $\frac{1}{2}\mathcal{N}(g_0)$, then 2α is a vertex of $\mathcal{N}(g_0)$ and $\langle \mathbf{e}_{2\alpha}, g_0 \rangle \neq 0$, i.e. $\mathbf{e}_{2\alpha}$ is not orthogonal to $\mathcal{A}$. Hence, α is not in $\frac{1}{2}\mathcal{N}(g_0)$ and M_{i+1} is also a superset of $\frac{1}{2}\mathcal{N}(g_0) \cap \mathbb{N}^n$.

We now show that M cannot be a strict superset of $\frac{1}{2}\mathcal{N}(g_0) \cap \mathbb{N}^n$. If this were the case, then at least one vertex β of $\operatorname{convhull} M$ must lie outside of $\frac{1}{2}\mathcal{N}(g_0)$. But this implies that 2β lies outside of $\mathcal{N}(g_0)$ which implies $\langle \mathbf{e}_{2\beta}, g_0 \rangle = 0$, i.e. an element of $R(M)_{\mathcal{N}}$ is orthogonal to $\mathcal{A}$, contradicting the termination criteria of the algorithm. ∎

B. Monomial selection via diagonal consistency

Other algorithms for monomial selection given in [5] [16] [7] are essentially based on the following observation: if a diagonal element of a positive semidefinite matrix is zero, then so is the entire corresponding row and column. Hence, if the equality constraints in the SDP (2) require that a diagonal entry of the matrix Q vanishes, i.e. if (assuming Condition 1) the implication

$$\sum_{\alpha,\beta \in M} q_{\alpha\beta} x^{\alpha} x^{\beta} = g_0(x) \Rightarrow q_{\zeta\zeta} = 0 \tag{3}$$

holds for some $\zeta \in M$, then ζ can be removed from M.

Following the terminology of Löfberg [7], we say ζ is *diagonally-inconsistent* when (3) holds. Notice that two things must happen for this implication to hold. First, ζ must be in M^+, otherwise matching coefficients for the term $x^{2\zeta}$ does not completely determine the value of $q_{\zeta\zeta}$. Two, $\langle \mathbf{e}_{2\zeta}, g_0(x) \rangle$ must vanish, i.e. $\mathbf{e}_{2\zeta}$ must be orthogonal to $\mathcal{A} = \{g_0(x)\}$. With these two observations, the following is immediate:

Proposition 2: Suppose Conditions 1-2 hold and that the search step $(\star)$ of Algorithm 2 is modified to

$$\text{Find } s_i \in R(M_i)_{\mathcal{C}} \text{ for which } s_i^{\perp} \text{ contains } \mathcal{A} \qquad (\star),$$

where

$$R(M)_{\mathcal{C}} := \left\{\mathbf{e}_{2\alpha} : \alpha \in M^+\right\}.$$

Then, the output of Algorithm 2 contains no diagonally-inconsistent monomials.

C. Comparison and simple example

We now give a concrete and extremely simple example comparing Algorithm 2 with the aforementioned techniques. Consider the SOS program:

$$\begin{aligned} &\text{Find } u \in \mathbb{R} \\ &\text{subject to} \quad f(x) = x_1^2 x_2^4 + u \cdot (1 - x_1^4 x_2^2) \text{ is sos.} \end{aligned}$$

Let $M_{\mathcal{N}}$, $M_{\mathcal{C}}$ and $M_{proposed}$ denote $\frac{1}{2}\mathcal{N}(f)$, the set of diagonally consistent monomials, and the output of Algorithm 2, respectively. The sets $M_{\mathcal{N}}$ and $M_{\mathcal{C}}$ (which we compute supposing u is non-zero) equal

$$M_{\mathcal{N}} = \{(0,0),(1,1),(1,2),(2,1)\}$$
$$M_{\mathcal{C}} = \{(0,0),(1,2),(2,1)\}$$

whereas Algorithm 2 outputs

$$M_{proposed} = \{(1,2)\},$$

leveraging the fact u vanishes if f is a sum-of-squares.

V. Examples

We conclude with examples from stability analysis, showing the practical utility of our technique. The sets $M_{\mathcal{N}}$, $M_{\mathcal{C}}$ and $M_{proposed}$ are defined as in the previous simple example.

A. Stability of Van Der Pol Oscillator

Consider a quadratic Lyapunov function

$$V(x) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}^T \begin{pmatrix} 15 & -5 \\ -5 & 10 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

and the polynomial vector field

$$\begin{pmatrix} \dot{x_1} \\ \dot{x_2} \end{pmatrix} = \begin{pmatrix} -x_2 \\ x_2 x_1^2 + x_1 - x_2 \end{pmatrix}$$

corresponding to a time reversed Van Der Pol oscillator. This system has an equilibrium point at the origin. The ρ-sublevel set of V is contained in the region of attraction if the following SOS program is feasible.

Find $\lambda(x)$
subject to $f(x) = \lambda(x)\dot{V}(x) + (x^T x)(V(x) - \rho)$ is sos.

Using a basis of all monomials of degree 4 or less for the polynomial $\lambda(x)$ and computing $M_{\mathcal{N}}$ and $M_{\mathcal{C}}$ supposing each coefficient is non-zero gives

$$M_{\mathcal{N}} = M_{\mathcal{C}} = \{(0,1),(0,2),(0,3),(1,0),(1,1),(1,2),(1,3),(2,0),(2,1),(2,2),(3,0),(3,1)\}$$
$$M_{proposed} = \{(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2),(3,1)\}.$$

B. Stable oscillations of the Moore Greitzer model

In [8], Manchester and Slotine study the Moore Greitzer model, a simplified model of the surge-stall dynamics of a jet engine. Using SOS programming, they establish bounds on a perturbation for which the model exhibits stable oscillations. The specific SOS program appears in Section VI-A of [8], and is stated in terms of so-called SOS matrix constraints (which can be converted into standard SOS constraints), polynomial matrices $W(x)$ and $L_i(x)$ and a polynomial $\alpha(x)$. The set of monomial exponents computed for each SOS constraint is summarized in Table I for various degrees of $W(x)$, $L_i(x)$ and $\alpha(x)$, where the column labeled "degree" contains the tuple $(\deg W, \deg L_i, \deg \alpha)$. Formulating these SOS programs using the aid of MATLAB code referenced in [8] and computing $M_{proposed}$ using a variant of our algorithm modified to handle multiple SOS constraints (as discussed in Section III-D) yields the results shown.

degree	$\lvert M\rvert_{\mathcal{N}}$	$\lvert M\rvert_{proposed}$
$(4,2,2)$	$(6,6,6,6,3,20,21)$	$(6,2,6,4,3,14,19)$
$(6,4,4)$	$(12,12,12,12,6,32,33)$	$(12,2,12,9,6,20,30)$
$(8,6,6)$	$(20,20,20,20,10,46,47)$	$(20,6,20,16,10,30,43)$
$(10,8,8)$	$(30,30,30,30,15,62,63)$	$(30,12,30,25,15,42,58)$

TABLE I: Comparison of our method to Newton polytope-based techniques for the Section V-B example, which contains seven SOS constraints.

References

[1] Grigoriy Blekherman, Pablo A. Parrilo, and Rekha R. Thomas. *Semidefinite optimization and convex algebraic geometry*. SIAM, 2013.

[2] Jonathan M Borwein and Henry Wolkowicz. Facial reduction for a cone-convex programming problem. *Journal of the Australian Mathematical Society. Series A: Pure mathematics*, 30:369–380, 1981.

[3] Man-Duen Choi, Tsit Yuen Lam, and Bruce Reznick. Sums of squares of real polynomials. In *Proceedings of Symposia in Pure mathematics*, volume 58, pages 103–126. American Mathematical Society, 1995.

[4] Karin Gatermann and Pablo A. Parrilo. Symmetry groups, semidefinite programs, and sums of squares. *Journal of Pure and Applied Algebra*, 192(1–3):95–128, 2004.

[5] Masakazu Kojima, Sunyoung Kim, and Hayato Waki. Sparsity in sums of squares of polynomials. *Mathematical Programming*, 103(1):45–62, 2005.

[6] Johan Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[7] Johan Löfberg. Pre-and post-processing sum-of-squares programs in practice. *IEEE Transactions on Automatic Control*, 54(5):1007–1011, 2009.

[8] Ian R. Manchester and Jean-Jacques E. Slotine. Transverse contraction criteria for existence, stability, and robustness of a limit cycle. *Systems and Control Letters*, 63:32–38, 2014.

[9] Jiawang Nie and James Demmel. Sparse sos relaxations for minimizing functions that are summations of small polynomials. *SIAM Journal on Optimization*, 19(4):1534–1558, 2008.

[10] Pablo A. Parrilo. Exploiting algebraic structure in sum of squares programs. In Didier Henrion and Andrea Garulli, editors, *Positive Polynomials in Control*, volume 312 of *Lecture Notes in Control and Information Sciences*, pages 580–580. Springer Berlin / Heidelberg, 2005. 10.1007/10997703_11.

[11] Gábor Pataki. Strong duality in conic linear programming: facial reduction and extended duals.

[12] Frank Permenter and Pablo A. Parrilo. Selecting a monomial basis for sums of squares programming over a quotient ring. In *IEEE 51st Annual Conference on Decision and Control (CDC)*, pages 1871–1876. IEEE, 2012.

[13] Frank Permenter and Pablo A. Parrilo. Partial facial reduction: simplified, equivalent SDPs via approximations of the PSD cone. http://arxiv.org/abs/1408.4685, 2014.

[14] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*, 2004.

[15] Bruce Reznick. Extremal psd forms with few terms. *Duke Mathematical Journal*, 45(2):363–374, 1978.

[16] Peter Seiler, Qian Zheng, and Gary Balas. Simplification methods for sum-of-squares programs. *preprint http://arxiv.org/abs/1303.0714*, 2013.

[17] Hayato Waki, Sunyoung Kim, Masakazu Kojima, and Masakazu Muramatsu. Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. *SIAM Journal on Optimization*, 17:218–242, 2006.

[18] Hayato Waki and Masakazu Muramatsu. Facial reduction algorithms for conic optimization problems. *Journal of Optimization Theory and Applications*, pages 1–28.

[19] Hayato Waki and Masakazu Muramatsu. A facial reduction algorithm for finding sparse sos representations. *Operations Research Letters*, 38(5):361–365, 2010.