Combining SOS and Moment Relaxations with Branch and Bound to Extract Solutions to Global Polynomial Optimization Problems

Hesameddin Mohammadi, Matthew M. Peet

Abstract— In this paper, we present a branch and bound algorithm for extracting approximate solutions to Global Polynomial Optimization (GPO) problems with bounded feasible sets. The algorithm is based on a combination of SOS/Moment relaxations and successively bisecting a hyper-rectangle containing the feasible set of the GPO problem. At each iteration, the algorithm makes a comparison between the volume of the hyper-rectangles and their associated lower bounds to the GPO problem obtained by SOS/Moment relaxations to choose and subdivide an existing hyper-rectangle. For any desired accuracy, if we use sufficiently large order of SOS/Moment relaxations, then the algorithm is guaranteed to return a suboptimal point in a certain sense. For a fixed order of SOS/Moment relaxations, the complexity of the algorithm is linear in the number of iterations and polynomial in the number of constrains. We illustrate the effectiveness of the algorithm for a 6-variable, 5-constraint GPO problem for which the ideal generated by the equality constraints is not zero-dimensional - a case where the existing Moment-based approach for extracting the global minimizer might fail.

I. INTRODUCTION

Global Polynomial Optimization (GPO) is defined as optimization of the form

$$f^* := \min_{x \in \mathbb{R}^n} . \quad f(x) \tag{1}$$

$$\text{subject to} \quad g_i(x) \geq 0 \quad \text{for } i = 1, \cdots, s$$
$$h_j(x) = 0 \quad \text{for } j = 1, \cdots, t,$$

where f, g_i, and h_i are real-valued polynomials in decision variables x. As defined in Eq. 1, the GPO problem encompasses many well-studied sub-classes including Linear Programming (LP), Quadratic Programming (QP), Integer Programming (IP), Semidefinite Programming (SDP) and Mixed-Integer Nonlinear Programming (MINLP) [1]. Because of its generalized form, almost any optimization problem can be cast or approximately cast as a GPO, including certain NP-hard problems from economic dispatch [2], optimal power flow [3] and optimal decentralized control [4]. As applied to control theory, GPO can be used for stability analysis of polynomial dynamical systems by, e.g., verifying polytopic invariants as in [5].

Although GPO is NP-hard, there exist a number of heuristics and algorithms that can efficiently solve special cases of GPO. For example, LP [6], [7], QP [8], and SDP all have associated polynomial-time algorithms. More broadly, if the feasible set is convex and the objective function is convex, then barrier functions and descent methods will typically yield a computationally tractable algorithm. When the problem is not convex, there also exist special cases in which the GPO problem is solvable. For example, in [9] the unconstrained problem was solved by parameterizing the

critical points of the objective function via Groebner bases. In the special case of $x \in \mathbb{R}^1$, the problem was solved in [10], [11]. In addition, there exist several widely used heuristics which often yield reasonably suboptimal and approximately or exactly feasible solutions to the GPO problem (e.g. [12], [13]), but which we will not discuss here in depth.

If we expand our definition of algorithm to include those with combinatorial complexity but finite termination time, then if the feasible set of the GPO problem is compact and the ideal generated by the equality constraints is radical and zero dimensional (typical for integer programming [14]), then one may use the moment approach to solving sequential Greatest Lower Bound (GLB) problems to obtain an algorithm with finite termination time. Unfortunately, however, it has been shown that the class of problems for which these methods terminate is a strict subset of the general class of GPO problems [15] and furthermore, there are no tractable conditions verifying if the algorithm will terminate or bounds on computational complexity in the case of finite termination. In this paper, however, we take the idea of generating Greatest Lower Bounds and propose an alternative method for extracting approximate solutions that does not require finite convergence and, hence, has polynomial-time complexity.

Let $\quad S := \{x \in \mathbb{R}^n : g_i(x) \geq 0, \ h_j(x) = 0\} \tag{2}$

be the feasible set and x^* be an optimal solution of GPO Problem (1). The Greatest Lower Bound (GLB) problem associated to GPO Problem (1) is defined as

$$\lambda^* := \max_{\lambda \in \mathbb{R}} . \quad \lambda \tag{3}$$

$$\text{subject to} \quad f(x) - \lambda > 0 \quad , \forall x \in S.$$

The GLB and GPO problems are closely related, but are not equivalent. For example, it is clear that $\lambda^* = f^* = f(x^*)$, where f^* is as defined in Eq. (1). Furthermore, as discussed in Section V, an algorithm which solves the GLB problem with complexity $O(k)$ can be combined with Branch and Bound to approximately solve the GPO for any desired level of accuracy ε with complexity $O(\log(1/\varepsilon)k)$, where we define an approximate solution to the GPO as a point $x \in \mathbb{R}^n$ such that $|x - x^*| \leq \varepsilon$ and $|f(x) - f(x^*)| < \varepsilon$.

Many convex approaches have been applied to solving the GLB problem, all of which are based on parameterizing the cone of polynomials which are positive over the feasible set of the corresponding GPO problem. The most well-known of these approaches are Sum of Squares (SOS) programming [16], and its dual Moment-relaxation problem [17]. Both these approaches are well-studied and

have implementations as Matlab toolboxes, including SOS-TOOLS [18] and Gloptipoly [19]. These approaches both yield a hierarchy of primal/dual semidefinite programs with an increasing associated sequence of optimal values $\{p_k^*\}_{k\in\mathbb{N}}$ (SOS) and $\{d_k^*\}_{k\in\mathbb{N}}$ (Moment) such that if we denote the OBV of the corresponding GPO problem by f^*, then under mild conditions, both sequences satisfy $p_k^* \leq d_k^* \leq f^*$ and $\lim_{k\to\infty} p_k^* = \lim_{k\to\infty} d_k^* = f^*$ [20]. Moreover, if the feasible set of GPO Problem (1), S, as defined in (2), is nonempty and compact, then there exist bounds on the error of SOS/Moment relaxations which scale as $|p_k^* - f^*| \cong \frac{c_2}{c_1\sqrt{\log(k)}}$ for constants c_1 and c_2 that are functions of polynomials f, g_i and h_j [21]. The goal of this paper, then, is to combine the SOS and Moment approaches with a branch and bound methodology to create an algorithm for solving the GPO problem and extracting a solution.

Specifically, in Sec. VI, we propose a sequence of branch and bound algorithms, denoted by E_k, such that for any $k \in \mathbb{N}$ and for a given GPO problem of Form (1) with a bounded feasible set, Algorithm E_k in polynomial-time returns a point $x_k \in \mathbb{R}^n$ that is sub-optimal to the GPO problem in the following sense. If x_k is the sequence of proposed solutions produced by the sequence of algorithms E_k, we show that if the feasible set, S, of Problem (1) is bounded, then there exist a sequence of feasible point $y_k \in S$ such that $\lim_{k\to\infty}||x_k - y_k|| = 0$ and $\lim_{k\to\infty} f(y_k) = \lim_{k\to\infty} f(x_k) = f^*$, where f^* is the OBV of the GPO problem.

The sequence of algorithms can be briefly summarized as follows. Algorithm E_k initializes with hyper-rectangle $C_1 := \{x \mid (x_i - \underline{c}_{i,1})(\bar{c}_{i,1} - x_i) \geq 0, i = 1, \cdots, n\}$ such that $S \subset C_1$. Then, at iteration m, the algorithm forms two new hyper-rectangles by bisecting C_m by its longest edge $(\bar{c}_i - \underline{c}_i)$ and intersecting each of the new hyper-rectangles with S to create two new GPO problems. Then, the algorithm computes a GLB estimate of the OBV of the new GPO problem by solving the corresponding k'th-order SOS/Moment relaxations. If λ_m^* denotes the best lower bound to f^*, obtained up to iteration m, then C_{m+1} is determined to be the existing hyper-rectangle with a smallest volume subject to the constraint that the corresponding GLB is less that $\frac{m\eta}{l} + \lambda_m^*$, where $\eta > 0$ and $l \in \mathbb{N}$ are the design parameters. Finally, after certain number of iterations, the algorithm terminates by returning the centroid of the last hyper-rectangle.

In Sec. VII we discuss the complexity of the proposed algorithm by first showing that for any $k \geq 2$, the feasible set of the k'th order SOS relaxation associated to a branch is completely contained in that of each of its subdivided branches - implying that the lower bounds obtained by both the SOS and Moment relaxation (due to the duality between SOS and Moment methods) are increasing. Of course, at those branches with $S \cap C_m = \emptyset$, these bounds will approach $+\infty$. Next, we show that if the error of the k'th order SOS/Moment relaxations associated to the hyper-rectangles obtained through Algorithm E_k is bounded by $\frac{c_2}{c_1\sqrt{\log(k)}} \leq \eta/(l+1)$, then each of the bisected hyper-rectangles is

guaranteed to contain a feasible point that is η-suboptimal. Therefore, the feasible set will be reduced in volume as $\frac{1}{2^m}$, up to iteration m. In other words, the number of iterations necessary to achieve a hyper-rectangle with a longest edge of the length ε is logarithmic in $1/\varepsilon$. Finally, since at each iteration the number of constraints is fixed and the complexity is linear in the number of iterations, we conclude that algorithm E_k is polynomial-time.

In Sec. VIII, we illustrate the effectiveness of the proposed algorithm by applying it to an example problem wherein the existing Moment based approach fails to extract a solution. Finally we conclude in Sec. IX.

II. NOTATION

Let $\mathbb{N}^n$ be the set of n-tuples of natural numbers. We use $\mathbb{S}^n$ and $\mathbb{S}^{n+}$ to denote the symmetric matrices and cone of positive semidefinite matrices of size $n \times n$, respectively. For any $a, b \in \mathbb{R}^n$, we denote by $C(a,b)$ the hyper-rectangle $\{x \in \mathbb{R}^n \mid a \leq x \leq b\}$, where $y \geq 0$ is defined by the positive orthant. We use ℓ_∞ to denote the set of bounded infinite sequences. For any $k, n \in \mathbb{N}$, let $\mathbb{N}_{(k)}^n := \{b \in \mathbb{N}^n : |b|_{\ell_1} \leq k\}$, where $|b|_{\ell_1} := \sum_{i=1}^n |b_i|$. Finally, we denote the ring of multivariate polynomials with real coefficients as $\mathbb{R}[x]$.

III. PROBLEM STATEMENT

In this paper, we consider simplified GPO problems of the form:
$$f^* := \min_{x\in\mathbb{R}^n}. \quad f(x) \tag{4}$$
$$\text{subject to} \quad g_i(x) \geq 0 \quad \text{for } i = 0, \cdots, s$$

where $f, g_i \in \mathbb{R}[x]$. The class of problems in (4) is equivalent to that in (1), where we have simply replaced every $h_i(x) = 0$ constraint with some $g_1(x) = h(x) \geq 0$ and $g_2(x) = h(x) \leq 0$. For every problem of Form (4), we define the associated feasible set $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0\}$.

In this paper, we assume $S \neq \emptyset$. Note that given g_i, one may use SOS optimization combined with Positivstellensatz results [22] to determine feasibility of S.

Proposed Algorithm In this paper, we propose a GLB and Branch and Bound-based algorithm which, for any given $\varepsilon > 0$, will return some $x \in \mathbb{R}^n$ for which there exists a point $y \in S$ such that:
$$f(y) - f^* \leq \varepsilon, \quad \text{and} \quad ||y - x|| < \varepsilon. \tag{5}$$

Furthermore x itself is ε-suboptimal in the sense that $|f(x) - f^*| \leq \varepsilon$ and $g_i(x) \geq -\varepsilon$.

Before defining this algorithm, however, in the following section, we describe some background on the dual SOS and Moment algorithms for generating approximate solutions of the GLB Problem.

IV. BACKGROUND ON SEMIDEFINITE REPRESENTATIONS OF SOS/MOMENT RELAXATIONS

In this section, we describe two well-known asymptotic algorithms which are known to generate sequences of increasingly accurate suboptimal solutions to the GLB problem - namely the SOS and Moment approaches. Both these methods use Positivstellensatz results which parameterize

the set of polynomials which are positive over a given semialgebraic set.

A. Sum-of-Squares Polynomials

In this subsection, we briefly define and denote sets of sums of squares of polynomials.

We denote monomials in variables $x \in \mathbb{R}^n$ as $x^\alpha := \prod_{i=1}^n x_i^{\alpha_i}$ where $\alpha \in \mathbb{N}^n$. Monomials can be ordered using various orderings on $\mathbb{N}^n$. In this paper, we use the graded lexicographical ordering. This ordering is defined inductively as follows. For $a, b \in \mathbb{N}^n$, $a \le b$ if $\sum_{i=1}^n a_i < \sum_{i=1}^n b_i$, or $a_1 = b_1$ and $[a_2, \cdots, a_n] \le [b_2, \cdots, b_n]$. Denote by $Z(x)$ the infinite ordered vector of all monomials, where $x^\alpha < x^\beta$ if $\alpha < \beta$. Because we have used the graded lexicographical ordering, if we restrict ourselves to the first $\binom{d+n}{d}$ elements of Z, then this is the vector of all monomials of degree d or less. We denote this truncated vector as $Z_d(x)$ and the length of Z_d as $\Lambda(d) := \binom{d+n}{d}$. Using this definition, it is clear that any polynomial can be represented as $p(x) = c^T Z_d(x)$ for some $c \in \mathbb{R}^{\Lambda(d)}$, where d is the degree of p.

The vector of monomials can also be combined with positive matrices to completely parameterize the cone of sums-of-squares polynomials. Formally, we can denote the subset of polynomials which are the sum of squares of polynomials as

$$\Sigma_S := \{s \in \mathbb{R}[x] : s(x) = \sum_{i=1}^l p_i^2(x), \, p_i \in \mathbb{R}[x], l \in \mathbb{N}\}. \quad (6)$$

Clearly any element of Σ_S is a nonnegative polynomial. Furthermore, if $p \in \Sigma_S$ and is of degree $2d$, then there exists a positive semidefinite matrix $\Omega \in \mathbb{S}^{\Lambda(d)+}$ such that

$$p(x) = Z_d(x)^T \Omega Z_d(x).$$

Conversely, any polynomial of this form, with $\Omega \ge 0$, is SOS. This parametrization of SOS polynomials using positive matrices will allow us to convert the GLB problem to an LMI. However, before defining this LMI approach, we must examine the question of positivity on semialgebraic subsets of $\mathbb{R}^n$.

B. Putinar's Positivstellensatz, Quadratic Modules and the Archimedean Property

Sum-of-Squares polynomials are globally non-negative. In this section, we briefly review Putinar's positivstellensatz which gives necessary conditions for a polynomial to be positive on the semiaglebraic set $S := \{x \in \mathbb{R}^n : g_i(x) \ge 0, i = 1, \ldots, s\}$, where $S \ne \emptyset$ and is compact.

Putinar's Positivstellensatz uses the g_i which define S to deduce a cone of polynomials which are non-negative on S. This cone is the quadratic module which we define as follows.

Definition 1: Given a finite collection of polynomials $g_i \in \mathbb{R}[x]$, we define the quadratic module as

$$M := \{p \mid p = \sigma_0 + \sum_{i=1}^s \sigma_i g_i \quad \sigma_i \in \Sigma_S\},$$

and the degree-k bounded quadratic module as

$$M^{(k)} := \{p \mid p = \sigma_0 + \sum_{i=1}^s \sigma_i g_i \quad \sigma_i \in \Sigma_S \quad \deg(\sigma_i g_i) \le k\}.$$

Clearly, any polynomial in M is non-negative on S. Furthermore, since Σ_S parameterizes M and positive matrices parameterize Σ_S, the constraint $p \in M_k$ can be represented as an LMI. Furthermore, if the module satisfies the Archimedean property, then Putinar's Positivstellensatz states that any polynomial which is positive on S is an element of M. That is, M parameterizes the cone of polynomials positive on S.

A quadratic module M is said to be Archimedean if there exists some $p \in M$ and $R \ne 0$ such that $p(x) = R^2 - \sum_{i=1}^n x_i^2$. We say that $\{g_i\}$ is an Archimedean representation of S if the associated quadratic module is Archimedean. Note that the Archimedean property is a property of the functions g_i which then define the quadratic module and not a property of S. Specifically, if S is compact, then there always exists an Archimedean representation of S. Specifically, in this case, there exists an $R > 0$ such that $\|x\| \le R$ for all $x \in S$. Now define $g_{s+1}(x) = R^2 - \sum_{i=1}^n x_i^2$.

C. SOS approach to solving the GLB problem

In this subsection, we briefly describe the use of SOS programming to define a hierarchy of GLB problems.

Consider the GPO Problem (4) where $\{g_i\}$ is an Archimedean representation of the feasible set, S, with associated quadratic module M. We now define the degree-unbounded version of the SOS GLB problem.

$$f^* = \lambda^* := \max_{\lambda \in \mathbb{R}} . \quad \lambda \quad (7)$$
$$\text{subject to} \quad f(x) - \lambda \in M.$$

Since M is Archimedian, it follows that $\lambda^* = f^*$ (where f^* is as defined in (4)). Although Problem (7) is convex, for practical implementation we must restrict the degree of the SOS polynomials which parameterize M - meaning, we must restrict ourselves to optimization on $M^{(k)}$. This defines a new sequence of GLB problems as

$$p_k^* := \max_{\lambda \in \mathbb{R}} . \quad \lambda \quad (8)$$
$$\text{subject to} \quad f(x) - \lambda \in M^{(k)}.$$

Clearly, $p_i^* \le p_j^* \le \lambda^*$ for any $i < j$. Additionally, it was shown in [16] that $\lim_{k \to \infty} p_k^* = p^*$. Furthermore, it was shown in [20], [21] that bounds on the convergence rate of $p_k^* \to \lambda^*$ exist as a function of g_i, f and k. Finally, the computational complexity of p_k is equivalent to that of a semidefinite program with order $(s+1)\Lambda(\lceil \frac{k}{2} \rceil)^2$ scalar variables.

D. Moment approach to solving the GLB problem

In this subsection, we briefly describe the Moment approach to solving the GLB problem.

Let $\mathbb{K}$ denote the set of Borel subsets of $\mathbb{R}^n$ and let $\mathcal{M}(\mathbb{K})$ be the set of finite and signed Borel measures on $\mathbb{K}$. The following lemma uses the set of probability measures with support on S to provide a necessary and sufficient condition for a polynomial to be positive over S.

Lemma 1: Given $S \in \mathbb{K}$ and $f : \mathbb{R}^n \to \mathbb{R}$ integrable over S, a polynomial $f(x)$ is nonnegative on S if and only if $\int_S f(x) \, d\mu \ge 0$ for all $\mu \in \mathcal{M}(\mathbb{K})$ such that $\mu(S) = 1$ and $\mu(\mathbb{R}^n / S) = 0$.

Again, consider GPO Problem (4) with the feasible set S. Now, if we define the moment optimization problem

$$\lambda^* := \max_{\mu \in \mathscr{M}(\mathbb{K}), \lambda \in \mathbb{R}} \lambda \qquad (9)$$

$$\text{subject to} \quad \int_S (f - \lambda) \, d\mu \geq 0,$$

$$\mu(S) = 1 \quad \text{and} \quad \mu(\mathbb{R}^n/S) = 0,$$

then Lemma 1 implies, using a duality argument, that $\lambda^* = f^*$ where f^* is as defined in Problem (4) [17].

Unfortunately, Problem (9) requires us to optimize over the space of measures $\mu \in \mathscr{M}(\mathbb{K})$. However, as yet we have no way of parameterizing these measures or imposing the constraints $\mu(S) = 1$ and $\mu(\mathbb{R}^n/S) = 0$. Fortunately, we find that a measure can be parameterized effectively using the moments generated by the measure. That is, any measure $\mu \in \mathscr{M}(\mathbb{K})$ has an associated ordered vector of moments, indexed by $\alpha \in \mathbb{N}^n$ using monomial x^α as

$$\psi(\mu)_\alpha := \int_S x^\alpha \, d\mu.$$

Furthermore, if $\mu(S) = 1$ and $\mu(\mathbb{R}^n/S) = 0$, then $\psi(\mu)^T c \geq 0$ for all $c \in C_g$, where we define $C_g := \{c : c^T Z(x) \in M\} \subset \ell_\infty$, where M is the module defined by the g_i in $S := \{x : g_i(x) \geq 0\}$. More significantly, the converse is also true. This means that if we add the constraint $\psi_1 = 1$ to ensure $\mu(S) = 1$, then we may replace the measure variable μ by the moment variable ψ, as described. However, this requires us to enforce the constraint $\psi^T c \geq 0$ for all c such that $c^T Z(x) \in M$. This constraint, however, can be represented using semidefinite programming [20], [17]. Finally, if $f(x) = d^T Z(x)$, we can enforce the integral constraint of optimization problem (1) as

$$\int_S f(x) - \lambda \, d\mu = d^T \psi - \lambda \geq 0.$$

This allows us to formulate the equivalent GLB problem as

$$d^* = \max_{\mathbf{y} \in \ell_\infty, \lambda \in \mathbb{R}} \lambda \qquad (10)$$

$$\text{subject to} \quad \mathbf{y}^T d - \lambda \geq 0,$$

$$\mathbf{y}^T c \geq 0 \quad , \forall \, c \in C_g,$$

$$\mathbf{y}_0 = 1.$$

Then $d^* = \lambda^*$ and as for the SOS GLB problem, we define a sequence of truncations of the moment GLB problem

$$d_k^* := \max_{\mathbf{y} \in \mathbb{R}^q, \lambda \in \mathbb{R}} \lambda \qquad (11)$$

$$\text{subject to} \quad \mathbf{y}^T d - \lambda \geq 0,$$

$$\mathbf{y}^T c \geq 0 \quad , \forall \, c \in C_g^k,$$

$$\mathbf{y}_0 = 1,$$

where $q = \Lambda(k)$ and $C_g^k := \{c \, : \, c^T Z_k(x) \in M^{(k)}\}$.

It has been shown that $p_k^* \leq d_k^*$ for all k and furthermore $\lim_{k \to \infty} p_k^* = \lim_{k \to \infty} d_k^* = \lambda^*$. Moreover, if the interior of S is not empty, then $p_k^* = d_k^*$ [20]. Note that $\mathbf{y}^T c \geq 0 \quad , \forall \, c \in C_g^k$ is an SDP constraint and the number of decision variables is $\Lambda(k)$. This implies that the computational complexity of d_k and p_k are similar.

In the following section, we combine the GLB problems defined by the SOS/Moment approach with a Branch and Bound sequence to approximately solve the GPO problem.

V. Solving the GPO Problem using SOS, Moments and Branch and Bound

In this section, we show that the following algorithm can be used to solve the GPO problem given a solution to the GLB problem.

The Ideal Branch and Bound Algorithm

At every iteration, we have a hyper-rectangle $A_i = [a_i, b_i]$;
1) Initialize the algorithm;
2) Bisect $A = [a_i, b_i] = [a', b'] \cup [a'', b''] = A_1 \cup A_2$;
3) Compute the Greatest Lower Bound of

$$\lambda_i^* := \max_{\lambda \in \mathbb{R}} \lambda \qquad (12)$$

$$\text{subject to} \quad f(x) - \lambda > 0 \quad , \forall x \in S \cap A_i;$$

4) If $\lambda_1^* > \lambda_2^*$, set $A = A_1$, otherwise $A = A_2$;
5) Goto 2 ;

At termination, we choose any $x \in A$, which will be accurate within $|x - x^*| \leq r 2^{-k/n}$.

Let us examine these steps in more detail.

Initialize the algorithm Since the set S is compact, there exists some $r > 0$ such that $S \subset B_r(0)$. We may then initialize $A = [-r\mathbf{1}, r\mathbf{1}]$, where $\mathbf{1}$ is the vector of all 1's.

Bisect Bisection of the hypercube occurs along the longest edge. Thus, after n iterations, we are guaranteed a two-fold increase in accuracy. As a result, the largest edge of the hypercube diminishes as $2^{-k/n}$.

Compute the Greatest Lower bound We assume that our solution to the GLB problem is exact. In this case, we are guaranteed that an optimizing x will always lie in A_i.

A. Complexity of the Ideal Branch and Bound Algorithm

In this subsection, we show that any exact solution to the GLB can be used to solve the GPO problem with arbitrary accuracy in a logarithmic number of steps using the Ideal Branch and Bound algorithm.

Suppose G is a GLB Problem of the Form (3) with solution λ^*. Define the algorithm $H : G \mapsto \lambda^*$ as $\lambda^* = H(G)$. Further suppose H has time-complexity $O(k)$, where k is a measure for the size of G. In the Ideal Branch and Bound algorithm, if we use H to perform Step (3), it is straightforward to show that for any $\varepsilon > 0$, after $m = 2n(c_1 + \log \frac{1}{\varepsilon})$ iterations, if $A_m = [a, b]$, then $\max_i |b_i - a_i| \leq \varepsilon$ and GPO Problem (4) has a minimizer $x^* \in C(a, b)$, where c_1 depends on the size of S and n is the number of variables. Now since all the GLB problems defined in Step (3) of the algorithm are of equal size, k, the complexity of the Ideal Branch and Bound algorithm for a given ε, is $O(2nk(c_1 + \log \frac{1}{\varepsilon}))$.

In this section, we considered the ideal case when the GLB can be solved exactly. In the following section we adapt this algorithm to the case when sequential algorithms such as SOS/Moment problems are used to solve the GLB problem. In this case, our approach will also be defined as a sequence of approximation algorithms.

VI. MODIFIED BRANCH AND BOUND ALGORITHM

In this section, we present a slightly modified branch and bound algorithm that combined with SOS/Moment relaxations, can approximate the solution to the GPO problem to any desired accuracy, in a certain sense.

The Modified Branch and Bound Algorithm At every iteration, we have an active hyper-rectangle $A = [a, b]$ and a set of feasible rectangles $Z = \{[a_i, b_i]\}_i$ each with associated GLB λ_i.

1) Initialize the algorithm
2) Bisect $A = [a, b] = [a', b'] \cup [a'', b''] = A_1 \cup A_2$
3) Compute the Greatest Lower Bound of

$$\lambda_i^* := \max_{\lambda \in \mathbb{R}}. \quad \lambda \tag{13}$$
$$\text{subject to} \quad f(x) - \lambda > 0 \quad , \forall x \in S \cap A_i.$$

4) If $\lambda_i^* \leq \lambda^* + \varepsilon$, add A_i to Z.
5) Set $A = Z_i$ where Z_i is the smallest element of Z.
6) Goto 2

At termination, we choose any $x \in A$, which will be accurate within $|x - x^*| \leq r 2^{-k/n}$.

A. Problem Definition and SOS/Moment Subroutine

Consider GPO Problem (4) and suppose the corresponding feasible set $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0\}$, is nonempty and compact with $S \subset C(a, b)$, for some $a, b \in \mathbb{R}^n$ with associated Archimedean quadratic module M.

Before defining the main sequential algorithm E_k, we will define the kth-order SOS/Moment GLB subroutine, denoted B_k, which calculates the GLB in Step (3) of the Modified Branch and Bound Algorithm.

SOS/Moment Subroutine $\lambda_k^* = \mathbf{B_k[a, b]}$

Given $A = [a, b]$, define the polynomials $w_i(x) := (b_i - x_i)(x_i - a_i)$. These polynomials are then used to define the modified feasible set $S \cap A$ as

$$S_{ab} := \{x \in \mathbb{R}^n : g_i(x) \geq 0, \forall i : 1 \leq i \leq s, \tag{14}$$
$$w_j(x) \geq 0, \forall j : 1 \leq j \leq n\},$$

and the corresponding modified degree-k bounded quadratic module as

$$M_{ab}^{(k)} := \Big\{ p : p = \sum_{i=0}^{s} \sigma_i g_i + \sum_{i=s+1}^{s+n} \sigma_i w_i, \quad \sigma_i \in \Sigma_S, \tag{15}$$
$$\deg(\sigma_i g_i) \leq k, \ \deg(\sigma_i w_i) \leq k \Big\},$$

where $g_0(x) = 1$. This allows us to formulate and solve the modified k-th order SOS GLB problem

$$p_k^* := \max_{\lambda \in \mathbb{R}}. \quad \lambda \tag{16}$$
$$\text{subject to} \quad f(\mathbf{x}) - \lambda \in M_{ab}^{(k)}$$

and the corresponding dual GLB moment problem as described in the preceeding section. The subroutine returns the value $\lambda_k^* = p_k^*$.

B. Formal Definition of the Modified Branch and Bound Algorithm, E_k

We now define a sequence of Algorithms E_k such that for any $k \in \mathbb{N}$, E_k takes GPO Problem (4) and returns an estimated feasible point x^*.

The Sequence of Algorithms E_k:

In the following, we use the notation $a \leftarrow b$ to indicate that the algorithm takes value b and assigns it to a. That is, $a = b$. In addition, $0 < \eta < 1$ and $l \in \mathbb{N}$ are design parameters and may be chosen arbitrarily. Some guidance on the choice of η and l is given in the section on complexity analysis.

The inputs to the following algorithm E_k are the functions $\{g_i\}$ and f, the initial hyper-rectangle such that $S \subset [a, b]$, and the design parameters. The output is the estimated feasible point, x.

Algorithm E_k:

```
input: η > 0, l ∈ ℕ, a, b ∈ ℝⁿ, f, g₁, ..., gₛ ∈ ℝ[x].
output: x ∈ ℝⁿ (as an approximate solution to GPO
```
Problem (4)).
```
Initialize:
   a(0) ← a; b(0) ← b; m ← 0; λ(0) ← Bₖ(a(0), b(0));
While (m < l): {
```

$$j^* \leftarrow \underset{j \in \{0, \ldots, m\}}{\arg\min} \lambda(j);$$

$$i^* \leftarrow \underset{j \in \{0, \ldots, m\}}{\arg\min} \prod_{i=1}^{n} (b(j)_i - a(j)_i)$$
$$\text{subject to} \quad \lambda(j) \leq \lambda(j^*) + \frac{m\eta}{1+l}; \tag{17}$$

$$a^* \leftarrow a(i^*); \qquad b^* \leftarrow b(i^*);$$
$$r^* \leftarrow \underset{j \in \{1, \ldots, n\}}{\arg\max} (b_j^* - a_j^*); \quad \tilde{a} \leftarrow a^*; \quad \hat{b} \leftarrow b^*;$$

```
For r from 1 to n: {
```
$$\tilde{b}_r \leftarrow \begin{cases} \frac{b_r^* + a_r^*}{2} & \text{if } r = r^* \\ b_r^* & \text{otherwise} \end{cases}; \quad \hat{a}_r \leftarrow \begin{cases} \frac{b_r^* + a_r^*}{2} & \text{if } r = r^* \\ a_r^* & \text{otherwise,} \end{cases};\}$$

$$\tilde{\lambda} \leftarrow B_k(\tilde{a}, \tilde{b}); \qquad \hat{\lambda} \leftarrow B_k(\hat{a}, \hat{b}); \qquad m \leftarrow m + 1;$$
$$a(i^*) \leftarrow \tilde{a}; \qquad b(i^*) \leftarrow \tilde{b}; \qquad \lambda(i^*) \leftarrow \tilde{\lambda};$$
$$a(m) \leftarrow \hat{a}; \qquad b(m) \leftarrow \hat{b}; \qquad \lambda(m) \leftarrow \hat{\lambda};\}$$

```
Return x := (a(l)+b(l))/2;
```

In the following section we will discuss the complexity and accuracy of the sequence of Algorithms E_k.

VII. CONVERGENCE AND COMPLEXITY OF E_k

In this section, we first show that for any $k \in \mathbb{N}$, the lower bounds obtained through Algorithm E_k are increasing. Next we show that for a sufficiently large k, Algorithm E_k can be designed to return an accurate suboptimal point, in a certain sense.

In the following lemma, we use Lemma 3 from the Appendix to show that for any $a_1, a_2, b_1, b_2 \in \mathbb{R}^n$ such that $a_1 \leq a_2$ and $b_2 \leq b_1 \in \mathbb{R}^n$, the feasible set of the SOS

problem solved in Subroutine $B_k(a_1, b_1)$ is contained in that of Subroutine $B_k(a_2, b_2)$.

Lemma 2: For any $k \in \mathbb{N}$ and $a \leq b \in \mathbb{R}^n$, let $M_{ab}^{(k)}$ be the modified degree-k bounded quadratic module associated to polynomials $g_1, \ldots, g_s$, as defined in (15). If $\gamma \leq \alpha < \beta \leq \delta \in \mathbb{R}^n$, then $M_{\gamma\delta}^{(k)} \subset M_{\alpha\beta}^{(k)}$, for all $k \geq 2$.

Proof:

For any $j = 1, \ldots, n$, let $w_{j,1}(x) := (\beta_j - x_j)(x_j - \alpha_j)$, and $w_{j,2}(x) := (\delta_j - x_j)(x_j - \gamma_j)$. Since $\gamma_j \leq \alpha_j < \beta_j \leq \delta_j$, then it is followed from Lemma 3 in the Appendix that there exist $p_j, q_j, r_j \in \mathbb{R}$ such that

$$w_{j,2}(x) = p_j^2(x)\, w_{j,1}(x) + q_j^2(x)\, (x_j + r_j)^2.$$

Now, Let $h \in M_{\gamma\delta}^{(k)}$, we will show that $h \in M_{\alpha\beta}^{(k)}$. Clearly, there exist $\sigma_i, \omega_{j,2} \in \Sigma_S$ such that $h = \sum_{i=0}^{s} \sigma_i g_i + \sum_{j=1}^{n} \omega_j w_{j,2}$, where $g_0(x) = 1$. Hence, we can plug in the expression for $w_{j,2}$ to get

$$h = \sum_{i=0}^{s} \sigma_i \cdot g_i + \sum_{j=1}^{n} \omega_j \cdot (p_j^2 \cdot w_{j,1} + q_j^2 \cdot (x_j + r_j)^2)$$

$$= \underbrace{(\sigma_0 + \sum_{j=1}^{n} q_j^2 \cdot \omega_j \cdot (x_j + r_j)^2)}_{\sigma_{0\,\text{new}}} + \sum_{i=1}^{s} \sigma_i \cdot g_i + \sum_{j=1}^{n} \underbrace{p_j^2 \omega_j}_{\omega_{j\,\text{new}}} \cdot w_{j,1}.$$

Then, since $k \geq 2$, it can be shown that there exist $\sigma_{0\,\text{new}}, \omega_{j\,\text{new}} \in \Sigma_S$ where $\deg(\sigma_{0\,\text{new}}) \leq k$, and $\deg(\omega_{j\,\text{new}} \cdot w_{j,1}) \leq k$ which implies that $h \in M_{\alpha\beta}^{(k)}$. ∎

If d is the maximum degree of polynomials g_i, then for any $k \geq d + 2$ and for any hyper-rectangles $C(c,d) \subset C(a,b)$, if $\lambda_{(a,b)}$ and $\lambda_{(c,d)}$ be the solutions obtained by Subroutines $B_k(a,b)$ and $B_k(c,d)$ applied to GPO Problem (4), then Lemma 2 shows that $\lambda_{(a,b)} \leq \lambda_{(c,d)}$. Now, for a fixed $k \in \mathbb{N}$, let η and l be the design parameters of Algorithm E_k applied to GPO Problem (1). For $m = 0, \ldots, l$, let $(\lambda^*)_m := \lambda(j^*)$, where j^* is defined as in iteration m of Algorithm E_k. Using Lemma 2, it is straightforward to show that $(\lambda^*)_m \leq (\lambda^*)_{m+1}$ for $\forall m \leq l - 1$.

In the next theorem, we will show that for any given $\varepsilon > 0$, there exist $k \in \mathbb{N}$ such that Algorithm E_k applied to GPO Problem (4) can be designed to provide a point $x \in \mathbb{R}^n$ satisfying (5).

Theorem 1: Suppose GPO Problem (4) has a nonempty and compact feasible set S. Choose $a, b \in \mathbb{R}^n$ such that $S \subset C(a,b)$. For any desired accuracy, $\varepsilon > 0$, there exist $k, l \in \mathbb{N}$, $\eta \in \mathbb{R}$ such that if $x = E_k(\eta, l, a, b, f, g_i)$ then there exists a feasible point $y \in S$ such that if f^* is the OBV of Problem (4), then $f(y) - f^* \leq \varepsilon$ and $\|y - x\| < \varepsilon$, where f^* is the OBV of GPO Problem (4). Furthermore, there exists a bound $c > 0$, depending only on f and g_i such that we may restrict the number of branches, l, in $E_k(\eta, l, a, b, f, g_i)$ as $l < c \log(1/\varepsilon)$.

Proof: Choose η such that $0 < \eta < \min\{\varepsilon, 1\}$. For any $l \in \mathbb{N}$, define $\mathscr{P}_l$ as the set of all hyper-rectangles generated through Algorithms E_k, $\forall k \in \mathbb{N}$ with the two first inputs η and l. The vertices of all elements of $\mathscr{P}_l$, clearly, lie on a grid with spacings $\frac{|a_i, b_i|}{2^l}$ for $i = 1, \ldots, n$. Therefore, $|\mathscr{P}_l|$ is finite. Let L be a longest edge of $C(a,b)$. Define l_η as

the smallest natural number l subject to $\frac{L\sqrt{n}}{2^{\lfloor l/n \rfloor}} \leq \eta$. It can be shown that for any $C_\alpha := C(e,f) \in \mathscr{P}_{l_\eta}$, there exists a $k_\alpha \in \mathbb{N}$ such that for any $k' \geq k_\alpha$, the solution of Subroutine $B_{k'}(e,f)$ is accurate with the error tolerance $\frac{\eta}{1+l_\eta}$. Define $k_\eta := \max\{k_\alpha : |C_\alpha \in \mathscr{P}_{l_\eta}\}$.

We now claim that for the design parameters $l := l_\eta$ and η, $\forall k \geq k_\eta$, Algorithm E_k returns the desired point. Let $k \geq k_\eta$. First, we show that Algorithm E_k with design parameters l_η and η generates exactly l_η nested hyper-rectangles. Proof by induction on m.

For $m = 0, \ldots, l_\eta - 1$, let $(a)_m := a(i^*)$, $(b)_m := b(i^*)$, $(C)_m := C((a)_m, (b)_m)$, $(\lambda)_m := B_k((a)_m, (b)_m)$, $(\tilde{a})_m := \tilde{a}$, $(\tilde{b})_m := \tilde{b}$, $(\hat{a})_m := \hat{a}$, $(\hat{b})_m := \hat{b}$, $(\tilde{C})_m := C((\tilde{a})_m, (\tilde{b})_m)$, $(\hat{C})_m := C((\hat{a})_m, (\hat{b})_m)$ and $(\lambda^*)_m := \lambda(j^*)$ where i^*, j^*, $\tilde{a}$, $\tilde{b}$, $\hat{a}$ and $\hat{b}$ are defined as in iteration m of Algorithm E_k.

We use induction on m to show that for all $m \leq l_\eta$:

$$(C)_m \subset (C)_{m-1}.$$

The base case $m = 0$ is trivial. For the inductive step, first note that $(\lambda^*)_m \leq f^*$ for all $m \leq l_\eta$ and $(\lambda^*)_1 \leq \cdots \leq (\lambda^*)_{l_\eta}$. The latter is obtained from Lemma 2 and the former is because at each iteration, $S \subset \bigcup_{i=0}^{m} C(a(i), b(i))$. Constraint (17) at iteration m, implies that

$$B_k((a)_m, (b)_m) \leq (\lambda^*)_m + \frac{m\eta}{l_\eta + 1}. \tag{18}$$

Now we will show that again, Constraint (17) at iteration $m+1$ is satisfied at least by one of $(\tilde{C})_m$ and $(\hat{C})_m$. Suppose this is not true. Then we can write

$$(\lambda^*)_{m+1} < B_k((\tilde{a})_m, (\tilde{b})_m) - \frac{(m+1)\eta}{1+l_\eta}, \tag{19}$$

$$(\lambda^*)_{m+1} < B_k((\hat{a})_m, (\hat{b})_m) - \frac{(m+1)\eta}{1+l_\eta}.$$

Now, since $(\lambda^*)_{m+1} \geq (\lambda^*)_m$, Eq. (19) implies

$$(\lambda^*)_m < B_k((\tilde{a})_m, (\tilde{b})_m) - \frac{(m+1)\eta}{1+l_\eta}, \tag{20}$$

$$(\lambda^*)_m < B_k((\hat{a})_m, (\hat{b})_m) - \frac{(m+1)\eta}{1+l_\eta}.$$

Using Eq. (20) and Eq. (18) one can write

$$B_k((a)_m, (b)_m) < B_k((\tilde{b})_m, (\tilde{b})_m) - \eta/l_\eta, \tag{21}$$

$$B_k((a)_m, (b)_m) < B_k((\hat{a})_m, (\hat{b})_m) - \eta/l_\eta.$$

This contradicts with the fact that all $B_k((\tilde{a})_m, (\tilde{b})_m)$, $B_k((\hat{a})_m, (\hat{b})_m)$ and $B_k((a)_m, (b)_m)$ have accuracy higher than $\eta/(1+l_\eta)$. Therefore, it is clear that both $(\tilde{C})_m$ and $(\hat{C})_m$ can be possible choices to be bisected at iteration $m+1$. This fact, together with the induction hypothesis which certifies that $(C)_m$ possesses the smallest volume between all the hyper-rectangles obtained up to that iteration, guaranteeing that either $(\tilde{C})_m$ or $(\hat{C})_m$, will be branched at the next iteration. Therefore, the algorithm will generate l_η nested hyper-rectangles.

Now, $(\lambda^*)_0 \in [f^* - \eta/l_\eta, f^*]$ implies that

$$(\lambda^*)_m \in [f^* - \eta/l_\eta, f^*], \quad \text{for all } m = 1, \ldots, l_\eta. \tag{22}$$

Eq. (18) and Eq.(22) together with the fact that $(\lambda)_m \geq (\lambda^*)_m$ imply

$$(\lambda)_m \in [f^* - \eta/l_\eta, f^* + \frac{m\eta}{l_\eta + 1}], \ \forall \, m = 1, \ldots, l_\eta. \quad (23)$$

Finally, as a special case $m = l_\eta$, one can write:

$$f^* - \frac{\eta}{1 + l_\eta} \leq B_k((a)_l, (b)_l) \leq f^* + \frac{l_\eta \eta}{l_\eta + 1}.$$

Now, note that the $\eta/(l_\eta + 1)$-accuracy of $B_k((a)_{l_\eta}, (b)_{l_\eta})$ implies that $(C)_{l_\eta}$ is feasible. It also can be implied that $(C)_{l_\eta} \cap S$ contains y such that $f(y) \geq B_k((a)_{l_\eta}, (b)_{l_\eta}) \geq f(y) - \eta/(l_\eta + 1)$. Therefore, $|f(y) - f^*| \leq \eta \leq \varepsilon$.

Finally, if x be the point return by Algorithm E_k, then based on the definition of l_η, it is implied that after the last iteration $m = l_\eta - 1$, the largest diagonal of the branched hyper-rectangle is less than $\eta \leq \varepsilon$, hence $\|y - x\|_2 \leq \varepsilon$, as desired. ∎

Theorem 1 ensures that for any accuracy $\varepsilon > 0$ there exists a $k \in \mathbb{N}$ such that Algorithm E_k, in logarithmic number of iteration, achieves approximate solutions to the GPO problem in a certain sense. The following corollary shows that these approximate solutions can approximately satisfy the constraints as follows.

Corollary 1: Let GPO Problem (4) have nonempty and compact feasible set that is contained in $C(a, b)$ for some $a, b \in \mathbb{R}^n$. For any given $\delta > 0$, there exists $\varepsilon > 0$ such that if ε and $x = E(\eta, l, a, b, f, g_i)$ satisfy the conditions in Theorem 1, then

$$|f(x) - f^*| \leq \delta \text{ and } \quad g_i(x) \geq -\delta, \ \forall i = 1, \ldots, s. \quad (24)$$

Proof: Let L be such that any polynomial $h \in \{f, g_1, \ldots, g_s\}$ satisfies $|h(c) - h(d)| \leq L|c - d|_2, \ \forall c, d \in C(a, b)$. (Existence of L follows from the Lipschitz continuity of polynomials on compact sets.) Choose ε such that $\varepsilon \leq \delta/L$. Let ε and x satisfy the conditions in Thoerem 1. It is straightforward to show that x satisfies Eq. (24). ∎

Unfortunately, however, Theorem 1 does not provide any bound on the size of this k. Now, since all the SOS problems that are solved in Subroutine B_k, differ at only $2n$ parameters, (which is basically shown in the proof of Lemma 2) one may wish to use SOS bounds $\|p_k^* - f^*\| \cong \frac{1}{\sqrt[c]{\log(k)}}$ introduced in [21] to obtain a bound on such k as a function of polynomials g_i. However, despite the effort that led to Lemma 2, the authors were not able to find such a unique bound for all the B_k problems, used in Algorithm E_k. This is remained unsolved and will be a topic for the future research. Finally, since obtaining a bound on k for Algorithm E_k is computationally difficult, for practical applications, given $\varepsilon > 0$, we first choose a k and run the algorithm until it saturates. By saturation we mean it does not generate nested hyper-rectangles effectively and as a result approximations do not get improved effectively.

For small $k \in \mathbb{N}$, Algorithm E_k may yet yield good approximations of an optimal point. Indeed, it may not generate only nested hyper-rectangles, however, it can be shown that for a fixed k, by increasing the number of iterations we can eventually obtain any number of nested hyper-rectangles at a cost of higher complexity.

VIII. NUMERICAL RESULTS

Consider the following GPO problem.

$$\min_{x \in \mathbb{R}^6} \quad f(x) = 7x_1 x_5^3 + 6x_1 x_5^2 x_6 + 9x_2 x_4^3 + 4x_2 x_4 x_5 +$$
$$3x_2 x_5 x_6 + x_3 x_4 x_5$$

subject to
$$g_1(x) = 100 - (x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2) \geq 0$$
$$g_2(x) = x_1^3 + x_2^2 x_4 + x_3 x_5^2 \geq 0$$
$$g_3(x) = x_2^2 x_1 + x_5^3 + x_4 x_1 x_2 \geq 0$$
$$h_1(x) = x_1 + x_2^2 - x_3^2 + x_4 x_5 = 0$$
$$h_2(x) = x_5 x_1 - x_4^2 = 0$$

In this example we have 6 variables, an objective function of degree 4 and several equality and inequality constraints of degree 4 or less. The ideal generated by equality constraints is not zero dimensional, hence the existing Moment method of extracting solutions might fail. We applied Algorithm E_5 with design parameters $\eta = 0.005$ and $l = 200$ to this example using Sedumi to solve the SDP related to both the SOS and Moment problems. As seen in Figure 1, the branch and bound algorithm converges relatively quickly to a certain level of error and then saturates. Iterations pass this point do not significantly improve accuracy of the feasible point. As predicted, this saturation and residual error (blue shaded region) is due the use of a fixed degree bound $k = 5$. As k is decreased, the residual error increases and as k is increased the residual error decreases. For this problem the final iteration returns the point $\hat{x} = [5.1416, 3.9307, 0.7568, -4.6777, 4.2676, -4.1504]$ for which all inequalities are feasible and the equality constraints h_1 and h_2 have errors of 0.0563 and 0.0610, respectively. The objective value is $f(\hat{x}) = -3693.3$.

IX. CONCLUSION

We proposed a hierarchy of Algorithms E_k, $k \in \mathbb{N}$ to extract solutions to the GPO problem based on a combination of Branch and Bound and SOS/Moment relaxations. The computational-complexity of Algorithm E_k is polynomial in k, polynomial in the number of constraints and linear in the number of branches l. Additionally, for any scaler $\varepsilon > 0$, there exist $k \in \mathbb{N}$ such that Algorithms E_k, in $O(\log(1/\varepsilon))$ number of iterations, returns a point that is within the ε-distance of a feasible and ε-suboptimal point. For a fixed degree of semidefinite relaxations, our numerical case study demonstrates convergence to a level of residual error which can then be decreased by increasing the degree. In ongoing work, we seek to bound this residual error as a function of degree using available bounds on the error of SOS/Moment relaxations. This, however requires a more deep understanding of the relationship between these bounds and structure of the semialgebraic sets.
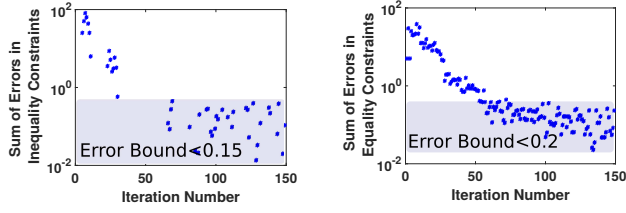
(a) Sum of errors in inequality constraints at center, $\sum_{g_i(\mathbf{x}_m)\le 0}|g_i(\mathbf{x}_m)|$ vs. number of iteration

(b) Sum of errors in equality constraints at center, $\sum_j |h_j(\mathbf{x}_m)|$ vs. number of iteration
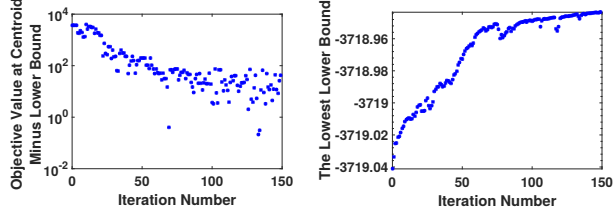
(c) Gap between lower bound and objective value at center point, $|f(\mathbf{x}_m)-l|$ vs. number of iteration

(d) Lower bound vs. number of iteration
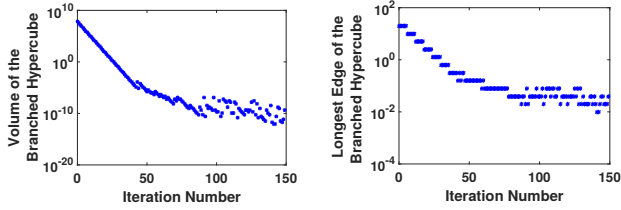
(e) Lower bound vs. number of iteration

(f) Longest edge of the branched hypercube vs. number of iteration
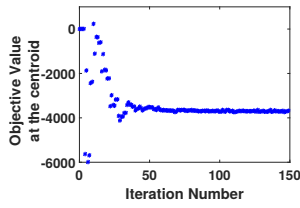
(g) Objective function value at center point vs. number of iteration

Fig. 1. Numerical results of the example

APPENDIX

The following lemma gives an algebraic property of the polynomials of the form $w(x) = (x - a_i)(b_i - x)$ which are used to define the augmented feasible set S_{ab}.

Lemma 3: Let $a \le c < d \le b \in \mathbb{R}$, $g := (x-a)(b-x)$ and $h := (x-c)(d-x)$. Then, there exist α, β and $\gamma \in \mathbb{R}$, such that

$$g(x) = \alpha h(x) + \beta (x+\gamma)^2, \quad \alpha, \beta \ge 0$$

Proof: Without loss of generality, one can assume that $a = 0$ (consider the change of variable $z := x - a$). Now let $p^2 := c$, $q^2 := d - c$, and $r^2 := b - d$. First, we consider the case where $p^2, r^2 \neq 0$. This leads to two sub-cases:

Case 1 : $r^2 \neq p^2$. Let

$$\gamma = \frac{p^4 + p^2 q^2 - \sqrt{p^2 r^2 (p^2+q^2)(q^2+r^2)}}{r^2 - p^2}, \beta = \frac{p^4 + p^2 q^2}{\gamma^2 - p^4 - p^2 q^2},$$

and $\alpha = \beta + 1$. Verifying the equality $g(x) = \alpha h(x) + \beta (x+\gamma)^2$ is straightforward. To show that $\beta, \alpha \ge 0$, we use the following.

$$\beta \ge 0 \iff \gamma^2 > p^4 + p^2 q^2$$

$$\iff \left(p^4 + p^2 q^2 - \sqrt{p^2 r^2 (p^2+q^2)(q^2+r^2)}\right)^2 > (p^4 + p^2 q^2)(r^2 - p^2)^2$$

$$\iff \underbrace{(p^4+p^2q^2)^2 + p^2 r^2(p^2+q^2)(q^2+r^2) - (p^4+p^2q^2)(r^2-p^2)^2}_{L} >$$

$$\underbrace{2(p^4+p^2q^2)\sqrt{p^2r^2(p^2+q^2)(q^2+r^2)}}_{U} \iff \begin{cases} L > 0 \\ L^2 > U^2 \end{cases}$$

After simplification we have:

$$L^2 - U^2 = p^4 q^4 (p^2+q^2)^2 (p^2 - r^2)^2 > 0, \quad \text{and}$$

$$L = p^2 (p^2 + q^2)(p^2 q^2 + 2 p^2 r^2 + q^2 r^2) > 0$$

which completes the proof for Case 1.

Case 2 : $r^2 = p^2$. In this case, let

$$\gamma = -\frac{2p^2+q^2}{2}, \beta = \frac{4p^2(p^2+q^2)}{q^4}, \alpha = \beta + 1$$

Equality and positivity for this case can then be easily verified. Now, suppose $r^2 = p^2 = 0$. In this case, simply set $\beta = 0$, $\alpha = 1$. If $p^2 = 0, r^2 \neq 0$, set $\beta = \frac{b}{d} - 1, \alpha = \frac{b}{d}, \gamma = 0$. The case $p^2 \neq 0$, $r^2 = 0$ is similar to $p^2 = 0$, $r^2 \neq 0$, through the change of variable $z = b - x$. ∎

REFERENCES

[1] M. Laurent, *Emerging Applications of Algebraic Geometry*, ch. Sums of Squares, Moment Matrices and Optimization Over Polynomials, pp. 157–270. New York, NY: Springer New York, 2009.

[2] J. B. Park, Y. W. Jeong, J. R. Shin, and K. Y. Lee, "An improved particle swarm optimization for nonconvex economic dispatch problems," *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 156–166, 2010.

[3] B. Ghaddar, J. Marecek, and M. Mevissen, "Optimal power flow as a polynomial optimization problem," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 539–546, 2016.

[4] J. Lavaei, "Optimal decentralized control problem as a rank-constrained optimization," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*, pp. 39–45, IEEE, 2013.

[5] M. A. B. Sassi and A. Girard, "Computation of polytopic invariants for polynomial dynamical systems using linear programming," *Automatica*, vol. 48, no. 12, pp. 3114 – 3121, 2012.

[6] L. Khachiyan, "Polynomial algorithms in linear programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 20, no. 1, pp. 53 – 72, 1980.

[7] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395.

[8] Z.-Q. Luo and S. Zhang, "A semidefinite relaxation scheme for multivariate quartic polynomial optimization with quadratic constraints," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1716–1736, 2010.

[9] B. Sturmfels, *Solving systems of polynomial equations*. No. 97, American Mathematical Soc., 2002.

[10] H. D. Sherali and C. H. Tuncbilek, "New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems," *Operations Research Letters*, vol. 21, no. 1, pp. 1 – 9, 1997.

[11] N. Z. Shor, "Quadratic optimization problems," *Soviet Journal of Computer and Systems Sciences*, vol. 25, no. 6, pp. 1–11, 1987.

[12] W. Cook, T. Koch, D. E. Steffy, and K. Wolter, "A hybrid branch-and-bound approach for exact rational mixed-integer programming," *Mathematical Programming Computation*, vol. 5, no. 3, pp. 305–344, 2013.

[13] D. E. Steffy and K. Wolter, "Valid linear programming bounds for exact mixed-integer programming," *INFORMS Journal on Computing*, vol. 25, no. 2, pp. 271–284, 2013.

[14] J. Lasserre, "Polynomials nonnegative on a grid and discrete optimization," *Transactions of the American Mathematical Society*, vol. 354, no. 2, pp. 631–649, 2002.

[15] J. Nie, "Optimality conditions and finite convergence of lasserre's hierarchy," *Mathematical Programming*, vol. 146, no. 1, pp. 97–121, 2013.

[16] P. A. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, Citeseer, 2000.

[17] J. B. Lasserre, *Moments, positive polynomials and their applications*, vol. 1. London: World Scientific, 2009.

[18] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing sostools: A general purpose sum of squares programming solver," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 1, pp. 741–746, IEEE, 2002.

[19] D. Henrion and J.-B. Lasserre, *Positive Polynomials in Control*, ch. Detecting Global Optimality and Extracting Solutions in GloptiPoly, pp. 293–310. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[20] M. Schweighofer, "Optimization of polynomials on compact semialgebraic sets," *SIAM Journal on Optimization*, vol. 15, no. 3, pp. 805–825, 2005.

[21] J. Nie and M. Schweighofer, "On the complexity of putinar's positivstellensatz," *Journal of Complexity*, vol. 23, no. 1, pp. 135 – 150, 2007.

[22] G. Stengle, "A nullstellensatz and a positivstellensatz in semialgebraic geometry," *Mathematische Annalen*, vol. 207, no. 2, pp. 87–97, 1974.