SIAM J. OPTIM.
Vol. 31, No. 3, pp. 2368–2395

A LAGRANGE MULTIPLIER EXPRESSION METHOD FOR BILEVEL POLYNOMIAL OPTIMIZATION[*]

JIAWANG NIE[†], LI WANG[‡], JANE J. YE[§], AND SUHAN ZHONG[†]

Abstract. This paper studies bilevel polynomial optimization. We propose a method to solve it globally by using polynomial optimization relaxations. Each relaxation is obtained from the Karush–Kuhn–Tucker (KKT) conditions for the lower level optimization and the exchange technique for semi-infinite programming. For KKT conditions, Lagrange multipliers are represented as polynomial or rational functions. The Moment–sum-of-squares relaxations are used to solve the polynomial optimization relaxations. Under some general assumptions, we prove the convergence of the algorithm for solving bilevel polynomial optimization problems. Numerical experiments are presented to show the efficiency of the method.

Key words. bilevel optimization, polynomial, Lagrange multiplier, Moment-SOS relaxation, semidefinite program

AMS subject classifications. 65K05, 90C22, 90C26, 90C34

DOI. 10.1137/20M1352375

1. Introduction.

This paper considers the bilevel optimization problem in the form

$$
(1.1)\qquad \left\{\begin{array}{rl} F^* := \min\limits_{x\in\mathbb{R}^n, y\in\mathbb{R}^p} & F(x,y) \\ \text{s.t.} & h_i(x,y) = 0\,(i\in\mathcal{E}_1), \\ & h_j(x,y) \ge 0\,(j\in\mathcal{I}_1), \\ & y\in S(x), \end{array}\right.
$$

where $S(x)$ is the set of optimizer(s) of the lower level problem

$$
(P_x)\qquad \left\{\begin{array}{rl} \min\limits_{z\in\mathbb{R}^p} & f(x,z) \\ \text{s.t.} & g_i(x,z) = 0\,(i\in\mathcal{E}_2), \\ & g_j(x,z) \ge 0\,(j\in\mathcal{I}_2). \end{array}\right.
$$

In the above, $F(x,y)$ is the upper level objective function and $h_i(x,y), h_j(x,y)$ are the upper level constraints, while $f(x,z)$ is the lower level objective function and $g_i(x,z), g_j(x,z)$ are the lower level constraints. Here $\mathcal{E}_1, \mathcal{I}_1, \mathcal{E}_2, \mathcal{I}_2$ are finite index sets (some or all of them are possibly empty). For convenience, we denote the feasible set of the lower level problem by

$$
(1.2)\qquad Z(x) := \big\{ z\in\mathbb{R}^p \mid g_i(x,z) = 0\ (i\in\mathcal{E}_2),\ g_j(x,z)\ge 0\,(j\in\mathcal{I}_2)\big\}.
$$

We call (1.1) a *simple* bilevel optimization problem (SBOP) if $Z(x)\equiv Z$ is independent of x, and call (1.1) a *general* bilevel optimization problem (GBOP) if $Z(x)$

[*]Received by the editors July 15, 2020; accepted for publication (in revised form) June 8, 2021; published electronically September 28, 2021.
https://doi.org/10.1137/20M1352375

[†]Department of Mathematics, University of California, La Jolla, CA 92093 USA (njw@math.ucsd.edu, suzhong@ucsd.edu).
[‡]Department of Mathematics, University of Texas at Arlington, Arlington, TX 76019 USA (li.wang@uta.edu).
[§]Department of Mathematics and Statistics, University of Victoria, Victoria, B.C., V8W 2Y2 Canada (janeye@uvic.ca).

depends on x. When all defining functions are polynomials, we call (1.1) a bilevel polynomial optimization problem. Throughout the paper, we assume that the solution set $S(x)$ of the lower level problem (P_x) is nonempty for all feasible x.

Bilevel optimization has broad applications, e.g., the moral hazard model of the principal-agent problem in economics [36], electricity markets and networks [7], facility location and production problem [8], meta learning and hyperparameter selection in machine learning [18, 25, 32]. More applications can be found in the monographs [4, 13, 16, 48] and the surveys on bilevel optimization [12, 17] and the references therein.

Bilevel optimization is challenging both theoretically and computationally, because of the optimality constraint $y \in S(x)$. The classical (or the first order) approach is to relax this constraint by the first order optimality condition for the lower level problem. But solving the resulting single-level problem may not even recover a stationary point of the original bilevel optimization problem if the lower level problem is nonconvex; see [36] and Example 6.1 for counterexamples. Moreover, even for the case that the lower level optimization is convex, the resulting single-level problem may not be equivalent to the original bilevel optimization problem if local optimality is considered and the lower level multiplier set is not a singleton (see [14]).

For each $y \in Z(x)$, it is easy to see the following equivalence (without any assumptions about the lower level optimization, e.g., convexity),

$$y \in S(x) \Longleftrightarrow f(x,y) - v(x) \le 0 \Longleftrightarrow f(x,z) - f(x,y) \ge 0 \quad \forall z \in Z(x), \tag{1.3}$$

where $v(x) := \inf_{z \in Z(x)} f(x,z)$ is the so-called value function for the lower level problem. We call any reformulation using the first equivalence in (1.3) the value function reformulation, while those using the second equivalence in (1.3) the semi-infinite programming (SIP) reformulation. Using the value function reformulation results in an intrinsically nonsmooth optimization problem which never satisfies the usual constraint qualification ([55]). Despite these difficulties, recent progress have been made on constraint qualifications and optimality conditions for bilevel optimization problems, where the lower level optimization is not assumed to be convex; see the works [3, 17, 51, 54, 56] and the references therein.

Solving bilevel optimization problems numerically is extremely hard, since even when all defining functions are linear, the computational complexity is already NP-hard [5]. Most prior methods in the literature are for mathematical programs with equilibrium constraints (MPECs) [34, 45] and hence can be used only to solve the reformulation of bilevel optimization by the first order approach. Recently, some methods for solving bilevel programs that are not formulated as MPECs were proposed in [26, 31, 37, 43, 50, 52, 53].

When all defining functions are polynomials, an optimization problem can be solved globally by the Lasserre type Moment-sum of squares (SOS) relaxations [27, 28, 30, 40]. This motivates the usage of polynomial optimization techniques for solving bilevel optimization problems globally [23, 42].

Contributions. Denote the set containing all upper and lower level constraints:

$$\mathcal{U} := \left\{ (x,y) \in \mathbb{R}^n \times \mathbb{R}^p \;\middle|\; \begin{array}{l} h_i(x,y) = 0\,(i \in \mathcal{E}_1),\, g_i(x,y) = 0\,(i \in \mathcal{E}_2), \\ h_j(x,y) \ge 0\,(j \in \mathcal{I}_1),\, g_j(x,y) \ge 0\,(j \in \mathcal{I}_2) \end{array} \right\}. \tag{1.4}$$

Based on the second equivalence in (1.3), the bilevel optimization (1.1) is equivalent to the following single-level optimization problem:

$$(P) \qquad \left\{ \begin{array}{ll} \min\limits_{x,y} & F(x,y) \\ \text{s.t.} & (x,y) \in \mathcal{U}, \quad f(x,z) - f(x,y) \ge 0 \quad \forall\, z \in Z(x). \end{array} \right.$$

Problem (P) belongs to the class of the so-called generalized semi-infinite programs since the set $Z(x)$ is typically infinite and depends on x.

To solve (P), one could construct a sequence of polynomial optimization relaxations (P_k) which have the same objective function as (P) and have the feasible set $\mathcal{U}_k$ satisfying the nesting containment:

$$\mathcal{F} \subsetneqq \cdots \subsetneqq \mathcal{U}_k \subsetneqq \cdots \subsetneqq \mathcal{U}_1 \subsetneqq \mathcal{U}_0 \subsetneqq \mathcal{U},$$

where $\mathcal{F}$ is the feasible set of (P). Let $(x^{(k)}, y^{(k)})$ be a global minimizer of (P_k). If $y^{(k)} \in S(x^{(k)})$, then $(x^{(k)}, y^{(k)})$ is also a global minimizer of (P). Otherwise, we can add new constraints to get a tighter relaxation (P_{k+1}). For the sequence $\{(x^{(k)}, y^{(k)})\}_{k=1}^{\infty}$ produced this way, we expect that its limit or accumulation point is a global minimizer of (P). This is a kind of exchange technique from SIP [22]. In each relaxation, if we only relax the infinitely many constraints

$$f(x, z) - f(x, y) \geq 0 \quad \forall z \in Z(x)$$

by finitely many ones of them, then the convergence would be extremely slow. This is because the set $\mathcal{U}$ typically has a much higher dimension than the feasible set $\mathcal{F}$ of (P), since each $(x, y) \in \mathcal{F}$ additionally satisfies the optimality condition $y \in S(x)$. For instance, when the lower level optimization (P_x) is unconstrained, every point in $\mathcal{F}$ satisfies the first order optimality condition which is a system of p equations, and hence the set $\mathcal{U}$ is generally p-dimensionally higher than $\mathcal{F}$. To fasten the convergence significantly, it was proposed in [42] to add the Jacobian representation for the Fritz John conditions of the lower level problem into each relaxation. However, the usage of a Jacobian representation is typically inconvenient, because it requires one to compute minors of Jacobian matrices. Moreover, the convergence of the method in [42] is only guaranteed for SBOPs. In this paper, we address these difficulties and give an efficient method for solving GBOPs.

The major motivation for our new method is as follows. For each $y \in S(x)$, we assume the Karush–Kuhn–Tucker (KKT) conditions hold,

$$\begin{cases} \nabla_z f(x, y) - \sum_{j \in \mathcal{E}_2 \cup \mathcal{I}_2} \lambda_j \nabla_z g_j(x, y) = 0, \\ \lambda_j \geq 0, \ \lambda_j g_j(x, y) = 0 \, (j \in \mathcal{I}_2), \end{cases}$$

where the λ_j's are Lagrange multipliers. This can be guaranteed if f and all the g_j are linear, or by imposing the LICQ/MFCQ (see section 2.2). In the initial relaxation (P_0), we relax the constraint $y \in S(x)$ to its KKT conditions. However, if we add the KKT conditions to $\mathcal{U}$ and minimize $F(x, y)$ over the original variables (x, y) as well as λ_j's, the number of variables is significantly increased. This is not practical if there are a large number of constraints. By using the technique called *Lagrange multiplier expression* introduced in [41], we express λ_j as a polynomial (or rational) function, say, $\lambda_j(x, y)$. Then, we choose the initial polynomial optimization relaxation to be

$$(P_0) \qquad \begin{cases} \min & F(x, y) \\ \text{s.t.} & h_i(x, y) = 0 \, (i \in \mathcal{E}_1), \ h_j(x, y) \geq 0 \, (j \in \mathcal{I}_1), \\ & g_i(x, y) = 0 \, (i \in \mathcal{E}_2), \ g_j(x, y) \geq 0 \, (j \in \mathcal{I}_2), \\ & \nabla_z f(x, y) - \sum_{j \in \mathcal{E}_2 \cup \mathcal{I}_2} \lambda_j(x, y) \nabla_z g_j(x, y) = 0, \\ & \lambda_j(x, y) \geq 0, \ \lambda_j(x, y) g_j(x, y) = 0 \, (j \in \mathcal{I}_2). \end{cases}$$

Suppose $(x^{(k)}, y^{(k)})$ is a global minimizer of (P_k). If $y^{(k)} \in S(x^{(k)})$, then $(x^{(k)}, y^{(k)})$ must be a global minimizer for (P). Otherwise, we can find a point $z^{(k)} \in Z(x^{(k)})$

such that

$$f(x^{(k)}, z^{(k)}) - f(x^{(k)}, y^{(k)}) < 0.$$

Can we add the following constraint,

$$f(x, z^{(k)}) - f(x, y) \geq 0, \tag{1.5}$$

to $\mathcal{U}_k$ to get a new relaxation (P_{k+1}) whose feasible set is

$$\widetilde{\mathcal{U}}_{k+1} := \{(x, y) \in \mathcal{U}_k \mid f(x, z^{(k)}) - f(x, y) \geq 0\}\,? \tag{1.6}$$

Since the point $(x^{(k)}, y^{(k)}) \notin \widetilde{\mathcal{U}}_{k+1}$, the new relaxation (P_{k+1}) given as above would be tighter. For (P_{k+1}) to qualify for a relaxation of (P), the feasible set $\widetilde{\mathcal{U}}_{k+1}$ must contain the feasible region $\mathcal{F}$ of (P). For SBOPs, i.e., $Z(x) \equiv Z$ is independent of x, the inequality (1.5) holds for all (x, y) satisfying $y \in S(x)$ and hence $\mathcal{F} \subseteq \widetilde{\mathcal{U}}_{k+1}$. However, for GBOPs, the condition $y \in S(x)$ may not necessarily imply $f(x, z^{(k)}) - f(x, y) \geq 0\ \forall (x, y) \in \mathcal{U}$ unless $z^{(k)} \in Z(x)$. Hence, the above $\widetilde{\mathcal{U}}_{k+1}$ may not contain the feasible set $\mathcal{F}$. To fix this issue, we propose to find a polynomial extension of the vector $z^{(k)}$, which is a polynomial function $q^{(k)}(x, y)$ satisfying $q^{(k)}(x^{(k)}, y^{(k)}) = z^{(k)}$ and $q^{(k)}(x, y) \in Z(x)$ for all $(x, y) \in \mathcal{U}$. Such a polynomial extension $q^{(k)}(x, y)$ satisfies

$$y \in S(x) \Longrightarrow f(x, q^{(k)}(x, y)) - f(x, y) \geq 0. \tag{1.7}$$

Therefore, we replace the feasible set in (1.6) by

$$\mathcal{U}_{k+1} := \{(x, y) \in \mathcal{U}_k | f(x, q^{(k)}(x, y)) - f(x, y) \geq 0\}$$

and the next polynomial optimization relaxation is

$$(P_{k+1}) \qquad \begin{cases} \min & F(x, y) \\ \text{s.t.} & (x, y) \in \mathcal{U}_k, f(x, q^{(k)}(x, y)) - f(x, y) \geq 0. \end{cases}$$

Continuing in this way, we either get an optimal solution of (P) for some k, or obtain an infinite sequence $\{(x^{(k)}, y^{(k)})\}_{k=1}^{\infty}$ such that each accumulation point is a global minimizer of (P).

The paper is organized as follows. In section 2, we review some basic facts in polynomial optimization as well as constraint qualifications for nonlinear optimization. In section 3, we propose a general approach for solving bilevel polynomial optimization. In section 4, we discuss how to get Lagrange multiplier expressions and the polynomial function $q^{(k)}(x, y)$. The numerical experiments are reported in section 5. Some conclusions and discussions are given in section 6.

2. Preliminaries.

Notation. The symbol $\mathbb{N}$ (resp., $\mathbb{R}, \mathbb{C}$) denotes the set of nonnegative integers (resp., real numbers, complex numbers). The $\mathbb{R}^n_+$ denotes the nonnegative orthant of $\mathbb{R}^n$. For a set S and a positive integer n, S^n denotes the n Cartesian products of S. For an integer $n > 0$, $[n] := \{1, \ldots, n\}$. Let $f(x, z)$ denote a continuously differentiable function. We use ∇f to denote its whole gradient and $\nabla_z f$ to denote its partial gradient with respect to z. For a vector $v := (v_1, \ldots, v_n)$ in $\mathbb{R}^n$, $\|v\|$ denotes the standard Euclidean norm and $diag[v]$ denotes an n-by-n diagonal matrix with the ith diagonal entry v_i for all $i \in [n]$. For $x := (x_1, \ldots, x_n) \in \mathbb{R}^n$ and $\alpha := (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$, denote the monomial

$$x^\alpha := x_1^{\alpha_1} \cdots x_n^{\alpha_n}.$$

For a positive integer k, $[x]_k$ denotes the vector of all monomials of the highest degree k ordered in the graded lexicographic ordering, i.e.,

$$[x]_k := (1, x_1, \dots, x_n, x_1^2, x_1x_2, \dots, x_n^k).$$

The symbol $\mathbb{R}[x] := \mathbb{R}[x_1, \dots, x_n]$ denotes the ring of polynomials in $\mathbb{R}[x]$ with real coefficients. For a polynomial $p \in \mathbb{R}[x]$, we use $\deg(p)$ to denote its degree, while for a tuple of polynomial $p = (p_1, \dots, p_r)$, $p_i \in \mathbb{R}[x]$, $i \in [r]$, we use $\deg(p)$ to denote the highest degree of p_i, i.e., $\deg(p) = \max\{\deg(p_1), \dots, \deg(p_r)\}$. For $k \in \mathbb{N}$, $\mathbb{R}[x]_k$ denotes the collection of all real polynomials in x with degrees at most k. For a given $p \in \mathbb{R}[x]$, define the set product $p \cdot \mathbb{R}[x] := \{pq | q \in \mathbb{R}[x]\}$. The symbol $\mathbf{1}_n$ is used to denote an all-ones vector in $\mathbb{R}^n$ and $\mathbf{1}$ denotes an all-ones vector with the dimension omitted. Denote by I_n the n-by-n identity matrix. For an optimization problem, `argmin` denotes the set of its minimizers.

2.1. Moment-SOS relaxations. For a tuple $p = (p_1, \dots, p_r)$ in $\mathbb{R}[x]$, $\mathrm{Ideal}(p)$ denotes the smallest ideal containing all p_i, i.e., $\mathrm{Ideal}(p) = p_1 \cdot \mathbb{R}[x] + \cdots + p_r \cdot \mathbb{R}[x]$. The kth *truncation* of the ideal $\mathrm{Ideal}(p)$, denoted as $\mathrm{Ideal}(p)_k$, is the set

$$p_1 \cdot \mathbb{R}[x]_{k-\deg(p_1)} + \cdots + p_r \cdot \mathbb{R}[x]_{k-\deg(p_r)}.$$

The real zero set of p is denoted as $\mathcal{V}(p) := \{x \in \mathbb{R}^n | \, p(x) = 0\}$.

A polynomial $\sigma \in \mathbb{R}[x]$ is said to be a sum of squares polynomial if $\sigma = \sigma_1^2 + \cdots + \sigma_k^2$ for some $\sigma_1, \dots, \sigma_k \in \mathbb{R}[x]$. We use the symbol $\Sigma[x]$ to denote the collection of all SOS polynomials in x. Its mth truncation is given by $\Sigma[x]_m := \Sigma[x] \cap \mathbb{R}[x]_m$. We define the *quadratic module* with respect to $q = (q_1, \dots, q_t) \in (\mathbb{R}[x])^t$ by

$$\mathrm{Qmod}(q) := \Sigma[x] + q_1 \cdot \Sigma[x] + \cdots + q_t \cdot \Sigma[x].$$

For $k \in \mathbb{N}$ and $2k \geq deg(q)$, the kth truncation of $\mathrm{Qmod}(q)$ is

$$\mathrm{Qmod}(q)_{2k} := \Sigma[x]_{2k} + q_1 \cdot \Sigma[x]_{2k-deg(q_1)} + \cdots + q_t \cdot \Sigma[x]_{2k-deg(q_t)}.$$

For a tuple of polynomials $q = (q_1, \dots, q_t)$ in $\mathbb{R}[x]$, denote the basic semialgebraic set $\mathcal{W}(q) := \{x \in \mathbb{R}^n | \; q(x) \geq 0\}$.

Given polynomial tuples p and q, if $f \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$, then it is easy to see that $f(x) \geq 0$ for all $x \in \mathcal{V}(p) \cap \mathcal{W}(q)$. To ensure $f \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$, we typically need more than $f(x) \geq 0$ for all $x \in \mathcal{V}(p) \cap \mathcal{W}(q)$. The sum $\mathrm{Ideal}(p) + \mathrm{Qmod}(q)$ is said to be *Archimedean* if there exists $b \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$ such that $\mathcal{W}(b) = \{x \in \mathbb{R}^n : b(x) \geq 0\}$ is a compact set. It is shown that $f \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$ if $f > 0$ on $\mathcal{V}(p) \cap \mathcal{W}(q)$ and $\mathrm{Ideal}(p) + \mathrm{Qmod}(q)$ is Archimedean [46]. This conclusion is often referenced as *Putinar's positivstellensatz*. When f is only nonnegative (but not strictly positive) on $\mathcal{V}(p) \cap \mathcal{W}(q)$, we still have $f \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$ under some generic conditions (cf. [40]).

We consider the polynomial optimization problem

$$f_{\min} := \min_{x \in \mathbb{R}^n} \; f(x) \quad \text{s.t.} \quad p(x) = 0, \, q(x) \geq 0, \tag{2.1}$$

where $f \in \mathbb{R}[x]$ and p, q are tuples of polynomials. The feasible set of problem (2.1) is $\mathcal{V}(p) \cap \mathcal{W}(q)$. It is obvious that a scalar $\gamma \leq f_{\min}$ if and only if $f - \gamma \geq 0$ on $\mathcal{V}(p) \cap \mathcal{W}(q)$, which can be ensured by the membership $f - \gamma \in \mathrm{Ideal}(p) + \mathrm{Qmod}(q)$.

The Moment-SOS hierarchy of semidefinite relaxations for solving problem (2.1) is to solve the relaxations

$$f_k := \max\ \gamma \quad \text{s.t.} \quad f - \gamma \in \mathrm{Ideal}(p)_{2k} + \mathrm{Qmod}(q)_{2k} \tag{2.2}$$

for $k = 1, 2, \ldots$. The asymptotic convergence $f_k \to f_{\min}$ as $k \to \infty$ was shown in [27]. Under the Archimedeanness and some classical optimality conditions, (i.e., LICQ, strict complementarity, and second order sufficiency conditions), it holds that $f_k = f_{\min}$ for all k big enough, as shown in [40]. The optimization problem (2.2) can be solved as a semidefinite program and hence can be solved by software packages such as `SeDuMi` [49] and `GloptiPoly 3` [21]. Moreover, after obtaining solutions for problem (2.2), we can extract an optimizer for (2.1) by using the so-called flat truncation condition [39].

2.2. Constraint qualifications. Consider the optimization problem

$$\begin{cases} \min & b(x) \\ \text{s.t.} & c_i(x) = 0\,(i \in \mathcal{E}), \\ & c_j(x) \geq 0\,(j \in \mathcal{I}), \end{cases} \tag{2.3}$$

where $b, c_i, c_j : \mathbb{R}^n \to \mathbb{R}$ are continuously differentiable. Let $\mathcal{I}(\bar{x}) := \{j \in \mathcal{I} | c_j(\bar{x}) = 0\}$ be the active index set of inequalities at a feasible point $\bar{x}$. The KKT condition is said to hold at $\bar{x}$ if there exist Lagrange multipliers λ_j such that

$$\sum_{j \in \mathcal{E} \cup \mathcal{I}} \lambda_j \nabla c_j(\bar{x}) = \nabla b(\bar{x}), \quad \lambda_j \geq 0,\ \lambda_j c_j(\bar{x}) = 0\,(j \in \mathcal{I}(\bar{x})).$$

A feasible point $\bar{x}$ is called a KKT point if it satisfies the KKT condition. A local minimizer must be a KKT point if all functions are linear. For nonlinear optimization, certain constraint qualifications are required for KKT points. The LICQ is said to hold at $\bar{x}$ if the gradient set $\{\nabla c_j(\bar{x})\}_{j \in \mathcal{E} \cup \mathcal{I}(\bar{x})}$ is linearly independent. The MFCQ is said to hold at $\bar{x}$ if the gradients $\nabla c_j(\bar{x})\,(j \in \mathcal{E})$ are linearly independent and there exists a vector $d \in \mathbb{R}^n$ satisfying

$$\nabla c_i(\bar{x})^T d = 0\,(i \in \mathcal{E}), \quad \nabla c_i(\bar{x})^T d > 0\,(i \in \mathcal{I}(\bar{x})).$$

The MFCQ is equivalent to the following statement:

$$\sum_{j \in \mathcal{E} \cup \mathcal{I}(\bar{x})} \lambda_j \nabla c_j(\bar{x}) = 0,\ \lambda_j \geq 0\,(j \in \mathcal{I}(\bar{x})) \quad \Longrightarrow \quad \lambda = 0.$$

When the functions $c_i(x)(i \in \mathcal{E})$ are linear and $c_j(x)(j \in \mathcal{I}(\bar{x}))$ are concave, Slater's condition is said to hold if there exists x_0 such that $c_i(x_0) = 0(i \in \mathcal{E}), c_i(x_0) > 0(i \in \mathcal{I})$. Slater's condition is equivalent to the MFCQ under the convexity assumption. If the MFCQ holds at a local minimizer $\bar{x}$, then $\bar{x}$ is a KKT point and the set of Lagrange multipliers is compact. If LICQ holds at $\bar{x}$, then the set of Lagrange multipliers is a singleton. We refer to [6] for constraint qualifications in nonlinear programming. For SBOPs, it is a generic assumption that each minimizer of the lower level optimization (P_x) is a KKT point (see [40]). When x is one-dimensional, this assumption is also generic (see [24]). When the dimension of x is bigger than one, we do not know whether or not this assumption is generic. However, in our computational experience, this assumption is often satisfied.

2.3. Lagrange multiplier representations. Consider the optimization problem (2.3) where b, c_i, c_j are real polynomials in $x \in \mathbb{R}^n$. For convenience, write that

$$\mathcal{E} \cup \mathcal{I} = \{1, \ldots, m\}, \quad c = (c_1, \ldots, c_m).$$

The KKT condition for (2.3) implies that

$$\underbrace{\begin{bmatrix} \nabla c_1(x) & \nabla c_2(x) & \cdots & \nabla c_m(x) \\ c_1(x) & 0 & \cdots & 0 \\ 0 & c_2(x) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_m(x) \end{bmatrix}}_{C(x)} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix}}_{\lambda} = \underbrace{\begin{bmatrix} \nabla b(x) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\hat{b}(x)}. \tag{2.4}$$

If there exists a polynomial matrix $L(x)$ such that $L(x)C(x) = I_m$, then

$$\lambda(x) = L(x)\hat{b}(x).$$

This gives an explicit expression for Lagrange multipliers as a function of x. When does such a polynomial matrix $L(x)$ exist? As showed in [41], it exists if and only if the constraining tuple c is nonsingular (i.e., the matrix $C(x)$ has full column rank for all complex vectors x). The nonsingularity is a generic condition in the Zariski topology [41, Proposition 5.7].

3. General bilevel polynomial optimization.

In this section, we propose a framework for solving the bilevel polynomial optimization (1.1). It is based on solving a sequence of polynomial optimization relaxations with the usage of KKT conditions and Lagrange multiplier representations.

3.1. Lagrange multiplier expressions and polynomial extensions.

For convenience, assume the constraining polynomial tuple in the lower level optimization (P_x) is $g := (g_1(x, z), \ldots, g_{m_2}(x, z))$, with $[m_2] := \mathcal{E}_2 \cup \mathcal{I}_2$. Then the KKT condition for (P_x) implies that

$$\underbrace{\begin{bmatrix} \nabla_z g_1(x, y) & \nabla_z g_2(x, y) & \cdots & \nabla_z g_{m_2}(x, y) \\ g_1(x, y) & 0 & \cdots & 0 \\ 0 & g_2(x, y) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_{m_2}(x, y) \end{bmatrix}}_{G(x,y)} \underbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{m_2} \end{bmatrix}}_{\lambda} = \underbrace{\begin{bmatrix} \nabla_z f(x, y) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\hat{f}(x,y)} \tag{3.1}$$

with $\lambda_j \geq 0, j \in \mathcal{I}_2$. Because of the dependence on x, the above matrix $G(x, y)$ is typically not full column rank for all complex pairs (x, y). Hence, there may not exist $L(x, y)$ such that $L(x, y)G(x, y) = I_{m_2}$. However, rational polynomial expressions always exist for Lagrange multipliers. Therefore, we make the following assumption.

Assumption 3.1. Suppose the KKT condition (3.1) holds for every minimizer of (1.1), there exist polynomials $d_1(x, y), \ldots, d_{m_2}(x, y) \geq 0$ on $\mathcal{U}$, and there are non-identically zero polynomials $\phi_1(x, y), \ldots, \phi_{m_2}(x, y)$ such that

$$\lambda_j d_j(x, y) = \phi_j(x, y), \quad j = 1, \ldots, m_2 \tag{3.2}$$

for all KKT points (x, y) as in (3.1).

Suppose there is a polynomial matrix $W(x,y)$ such that

$$W(x,y)G(x,y) = \operatorname{diag}[d(x,y)], \quad d(x,y) := (d_1(x,y),\dots,d_{m_2}(x,y)).$$

Then we can get Lagrange multiplier expressions as in (3.2), since

$$\operatorname{diag}[d(x,y)]\lambda = W(x,y)G(x,y)\lambda = W(x,y)\hat{f}(x,y), \tag{3.3}$$

which is the same as

$$d_j(x,y)\lambda_j = \big(W(x,y)\hat{f}(x,y)\big)_j. \tag{3.4}$$

(The subscript j denotes the jth entry.) The polynomial $\phi_j(x,y)$ in (3.2) is then $\big(W(x,y)\hat{f}(x,y)\big)_j$. Let $D(x,y)$ be the least common multiple of $d_1(x,y),\dots,d_{m_2}(x,y)$ and $D_j(x,y)$ be the quotient polynomial $D(x,y)/d_j(x,y)$. Under Assumption 3.1, the set of KKT points in (3.1) is contained in

$$\mathcal{K} := \left\{(x,y) \;\middle|\; \begin{array}{c} D(x,y)\nabla_z f(x,y) - \sum_{j=1}^{m_2} D_j(x,y)\phi_j(x,y)\nabla_z g_j(x,y) = 0, \\ \phi_j(x,y) \geq 0,\ \phi_j(x,y)g_j(x,y) = 0\,(j\in\mathcal{I}_2) \end{array}\right\}. \tag{3.5}$$

Indeed, the equivalence holds when $d(x,y)$ is positive on $\mathcal{U}$. If $d_j(\hat{x},\hat{y}) = 0$ for some j and $(\hat{x},\hat{y}) \in \mathcal{U}$, then $D(\hat{x},\hat{y}) = 0$ and hence the equations in (3.5) are automatically satisfied.

Assumption 3.2. For every pair $(\hat{x},\hat{y}) \in \mathcal{U}\cap\mathcal{K}$ and for every $\hat{z} \in S(\hat{x})$, there exists a polynomial tuple $q(x,y) := (q_1(x,y),\dots,q_p(x,y))$ such that

$$q(\hat{x},\hat{y}) = \hat{z}, \quad q(x,y) \in Z(x) \quad \forall\,(x,y) \in \mathcal{U}. \tag{3.6}$$

We call the function $q(x,y)$ in the above a *polynomial extension* of the point $\hat{z}$ at $(\hat{x},\hat{y})$. More details about Lagrange multiplier expressions and polynomial extensions, required in Assumptions 3.1 and 3.2, will be given in section 4.

3.2. An algorithm for bilevel polynomial optimization. Under Assumptions 3.1 and 3.2, we propose the following algorithm to solve the bilevel polynomial optimization (1.1). Recall that $Z(x)$ and $\mathcal{U}$ are the sets as in (1.2) and (1.4), respectively. We refer to section 2.1 for the Moment-SOS hierarchy.

ALGORITHM 3.3. *For the given polynomials* $F(x,y), h_i(x,y), f(x,z), g_j(x,z)$ *in* (1.1), *do the following:*

Step 0 *Find rational expressions for Lagrange multipliers as in* (3.2) *for Assumption* 3.1. *Let* $\mathcal{U}_0 := \mathcal{U}\cap\mathcal{K}$, *where* $\mathcal{K}$ *is the set in* (3.5). *Let* $k := 0$.

Step 1 *Apply the Moment-SOS hierarchy to solve the polynomial optimization*

$$(P_k)\quad \left\{\begin{array}{rl} F_k^* := \min\limits_{x\in\mathbb{R}^n, y\in\mathbb{R}^p} & F(x,y) \\ s.t. & (x,y) \in \mathcal{U}_k. \end{array}\right. \tag{3.7}$$

If (P_k) *is infeasible, then either* (1.1) *has no optimizers, or none of its optimizers satisfy the KKT condition* (3.1) *for the lower level optimization. If it is feasible and has a minimizer, solve it for a minimizer* $(x^{(k)},y^{(k)})$.

Step 2 *Apply the Moment-SOS hierarchy to solve the lower level optimization*

$$(Q_k)\quad \left\{\begin{array}{rl} v_k^* := \min\limits_{z\in\mathbb{R}^p} & f(x^{(k)},z) - f(x^{(k)},y^{(k)}), \\ s.t. & z \in Z(x^{(k)}),\ (x^{(k)},z) \in \mathcal{K}, \end{array}\right. \tag{3.8}$$

for an optimizer $z^{(k)}$. *If the optimal value* $v_k^* = 0$, *then* $(x^{(k)},y^{(k)})$ *is an optimizer for* (1.1) *and stop. Otherwise, go to the next step.*

Step 3 *Construct* $q^{(k)}(x,y)$*, a polynomial extension of the vector* $z^{(k)}$*, such that*

$$q^{(k)}(x^{(k)}, y^{(k)}) = z^{(k)}, \quad q^{(k)}(x,y) \in Z(x) \quad \forall (x,y) \in \mathcal{U}.$$

Update the set $\mathcal{U}_{k+1}$ *as*

$$\mathcal{U}_{k+1} := \{(x,y) \in \mathcal{U}_k | f(x, q^{(k)}(x,y)) - f(x,y) \geq 0\}.$$

Let $k := k+1$ *and go to Step* 1.

In Algorithm 3.3, the polynomial optimization problems $(P_k), (Q_k)$ need to be solved correctly. This can be done by using the Lasserre type Moment-SOS hierarchy of semidefinite relaxations. We refer to section 2.1 for the details. To solve (P_k), the Moment-SOS hierarchy produces a sequence of convergent lower bounds for F_k^*, say, $\{\rho_l\}_{l=1}^{\infty}$, such that

$$\rho_1 \leq \cdots \leq \rho_l \leq \cdots \leq F_k^*, \quad \lim_{l\to\infty} \rho_l = F_k^*,$$

where the subscript l is the relaxation order. For generic polynomial optimization problems, it has finite convergence, i.e., $\rho_l = F_k^*$ for some l. To check the convergence, we need to extract a feasible point $(\hat{x}, \hat{y})$ such that $\rho_l = F(\hat{x}, \hat{y}) = F_k^*$. It was shown in [39] that the flat truncation condition is a sufficient (and almost necessary) criterion for detecting the convergence. When the flat truncation condition is met, the Moment-SOS relaxation is tight and one (or more) minimizer $(x^{(k)}, y^{(k)})$ can be extracted for (P_k). The lower level polynomial optimization (Q_k) can be solved in the same way by the Moment-SOS hiearchy.

It was shown in [40] that the hierarchy of Moment-SOS relaxations has finite convergence, under the Archimedeanness and some classical optimality conditions (i.e., the LICQ, strict complementarity, and second order sufficiency condition). As a special case, this conclusion can also be applied to the suboptimization problem (P_k) in Algorithm 3.3, in particular when the lower level optimization has no inequality constraints (i.e., $\mathcal{I}_2 = \emptyset$), to ensure the finite convergence. However, when $\mathcal{I}_2 \neq \emptyset$, there is a complementarity constraint, so the problem (P_k) is a mathematical program with complementarity constraints. It is known that the usual constraint qualification such as MFCQ and LICQ will never hold for such problems (see [57, Proposition 1.1]) and hence the current theory is not applicable to guarantee the finite convergence. Therefore, we are not sure whether or not the Moment-SOS hierarchy has finite convergence for solving (P_k), when (1.1) is given by generic polynomials. We remark that it is possible that the Moment-SOS hierarchy fails to have finite convergence for some special cases of (P_k). For instance, this is the case if F is the Motzkin polynomial and (1.1) has a ball constraint and all f, g_i, g_j are zero polynomials (see [40]). However, in our computational experience, the suboptimization problem (P_k) is almost always solved successfully by the Moment-SOS hierarchy. In contrast, the suboptimization problem (Q_k) is easier to solve by the Moment-SOS hierarchy. This is because Lagrange multiplier expressions for (P_x) are used to formulate (Q_k). The Moment-SOS hierarchy has finite convergence for almost all cases. This is implied by results in [41].

In addition to the Lasserre type Moment-SOS relaxations, there exist other types of relaxations for solving polynomial optimization. For instance, the second order cone programming (SOCP) relaxations based on scaled diagonal (SDSOS) polynomials [1], the bounded degree SOS relaxations (BDSOS) [29], or a mixture of them [10]. In principle, these relaxation methods can also be used in Algorithm 3.3. However, we would like to remark that the performance of these relaxations is much worse than the classical Lasserre type Moment-SOS relaxations. Such a comparison is made in Example 6.2. A major reason is that these other types of relaxations cannot solve the suboptimization problems (P_k) or (Q_k) accurately enough. Note that Algorithm 3.3 requires that global optimizers of (P_k) and (Q_k) are computed successfully.

3.3. Convergence analysis. We study the convergence of Algorithm 3.3. First, we show that if the problem (P_x) is convex for each x, then Algorithm 3.3 will find a global optimizer of the bilevel optimization (1.1) in the initial loop.

PROPOSITION 3.4. *Suppose that Assumptions* 3.1 *and* 3.2 *hold and all* $d_j(x,y) > 0$ *on* $\mathcal{U}$. *For every given* x, *assume that* $f(x,z)$ *is convex with respect to* z, $g_i(x,z)$ *is linear in* z *for* $i \in \mathcal{E}_2$, *and* $g_j(x,z)$ *is concave in* z *for* $j \in \mathcal{I}_2$. *Assume that Slater's condition holds for* $Z(x)$ *for all feasible* x. *Then, the bilevel optimization* (1.1) *is equivalent to* (P_0) *and Algorithm* 3.3 *terminates at the loop* $k = 0$.

Proof. Under the given assumptions, $y \in S(x)$ if and only if y is a KKT point for problem (P_x), which is then equivalent to $(x,y) \in \mathcal{K}$, since all $d_j(x,y) > 0$ on $\mathcal{U}$. Then, the feasible set of (1.1) is equivalent to $\mathcal{U} \cap \mathcal{K}$. This implies that (1.1) is equivalent to (P_0) and Algorithm 3.3 terminates at the initial loop $k = 0$. ◻

Second, if Algorithm 3.3 terminates at some loop k, we can show that it produces a global optimizer for the bilevel optimization (1.1).

PROPOSITION 3.5. *Suppose that Assumptions* 3.1 *and* 3.2 *hold. If Algorithm* 3.3 *terminates at the loop* k, *then the point* $(x^{(k)}, y^{(k)})$ *is a global optimizer of* (1.1).

Proof. By Assumption 3.1, the KKT condition (3.1) holds at each $(x,y) \in \mathcal{U} \cap \{(x,y) : y \in S(x)\} = \mathcal{F}$ and hence $\mathcal{F} \subseteq \mathcal{U}_0 := \mathcal{U} \cap \mathcal{K}$. By the construction of $q^{(k)}(x,y)$ as required for Assumptions 3.2, we have shown $\mathcal{F} \subseteq \mathcal{U}_k$ for each k, by virtue of (1.7). Hence we have $F_k^* \le F^*$ for all k, where F^* denotes the optimal value of (1.1). According to the stopping rule, if Algorithm 3.3 terminates at the kth loop, then $y^{(k)} \in S(x^{(k)})$. This means $(x^{(k)}, y^{(k)}) \in \mathcal{F}$. Consequently $F_k^* = F(x^{(k)}, y^{(k)}) \ge F^*$. Hence $(x^{(k)}, y^{(k)})$ is a global optimizer of (1.1). ◻

Last, we study the asymptotic convergence of Algorithm 3.3. To prove the convergence, we need to assume that the value function $v(x)$ is continuous at an accumulation point x^*. under the so-called *restricted inf-compactness* (RIC) condition (see e.g., [20, Definition 3.13]) and either $Z(x)$ is independent of x or the MFCQ holds at some $\bar{z} \in Z(x^*)$; see [19, Lemma 3.2] for the upper semicontinuity and [11, page 246] for the lower semicontinuity. The RIC holds at x^* for $v(x)$ if the value $v(x^*)$ is finite and there exist a compact set Ω and a positive number ϵ_0, such that for all $\|x - x^*\| < \epsilon_0$ with $v(x) < v(x^*) + \epsilon_0$, there exists $z \in S(x) \cap \Omega$. For instance, $v(x)$ satisfies the RIC at x^* (see [11, section 6.5.1]) under one of the following conditions.

- The set $Z(x)$ is uniformly compact around x^* (i.e., there is a neighborhood $N(x^*)$ of x^* such that the closure of $\cup_{x \in N(x^*)} Z(x)$ is compact).
- The lower level objective $f(x,z)$ satisfies the growth condition, i.e., there exists a positive constant $\delta > 0$ such that the set

$$\left\{ z \,\middle|\, \begin{array}{l} g_i(x^*,z) = \alpha_i (i \in \mathcal{E}_2),\, g_j(x^*,z) = \alpha_j (j \in \mathcal{I}_2), \\ f(x^*,z) \le \vartheta,\, \sum_{i \in \mathcal{E}_2 \cup \mathcal{I}_2} \alpha_i^2 \le \delta \end{array} \right\}$$

  is bounded for all real values ϑ.

- The objective $f(x,z)$ is weakly coercive in z with respect to $Z(x)$ for all x sufficiently close to x^*, i.e., there is a neighborhood $N(x^*)$ of x^* such that

$$\lim_{z\in Z(x),\|z\|\to\infty} f(x,z)=\infty \qquad \forall x\in N(x^*).$$

The following is the asymptotic convergence result for Algorithm 3.3.

THEOREM 3.6. *For Algorithm* 3.3, *we assume the following:*
(a) *All optimization problems* (P_k) *and* (Q_k) *have global minimizers.*
(b) *The Algorithm* 3.3 *does not terminate at any loop, so it produces the infinite sequence* $\{(x^{(k)},y^{(k)},z^{(k)})\}_{k=0}^{\infty}$.
(c) *Suppose* (x^*,y^*,z^*) *is an accumulation point of* $\{(x^{(k)},y^{(k)},z^{(k)})\}_{k=0}^{\infty}$ *and the value function* $v(x)$ *is continuous at* x^*.
(d) *The polynomial functions* $q^{(k)}(x,y)$ *converge to* $q^{(k)}(x^*,y^*)$ *uniformly for* $k\in\mathbb{N}$ *as* $(x,y)\to(x^*,y^*)$.

Then, (x^*,y^*) *is a global minimizer for the bilevel optimization* (1.1).

Proof. Since (x^*,y^*) is an accumulation point of the sequence $\{(x^{(k)},y^{(k)})\}_{k=0}^{\infty}$, there is a subsequence $\{k_\ell\}$ such that $k_\ell\to\infty$ and

$$(x^{k_\ell},y^{k_\ell},z^{k_\ell})\to(x^*,y^*,z^*).$$

Since each $z^{(k_\ell)}\in Z(x^{(k_\ell)})$, we can see that $z^*\in Z(x^*)$. The feasible set of (P_{k_ℓ}) contains that of (1.1), so

$$F(x^*,y^*)=\lim_{\ell\to\infty}F(x^{(k_\ell)},y^{(k_\ell)})\le F^*,$$

where F^* is the optimal value of the bilevel optimization (1.1). (The polynomial $F(x,y)$ is a continuous function.) To prove $F(x^*,y^*)\ge F^*$, we show that (x^*,y^*) is feasible for problem (1.1). Define the functions

$$H(x,y,z):=f(x,z)-f(x,y),\quad \phi(x,y):=\inf_{z\in Z(x)}H(x,y,z). \tag{3.9}$$

Observe that $\phi(x,y)=v(x)-f(x,y)\le 0$ for all $(x,y)\in\mathcal{U}$ and $\phi(x^*,y^*)=0$ if and only if (x^*,y^*) is feasible for (1.1). Since $v(x)$ is continuous at x^*, we have $\phi(x^*,y^*)\le 0$. Next, we show that $\phi(x^*,y^*)\ge 0$. For an arbitrary $k'\in\mathbb{N}$, and for all $k_\ell\ge k'$, the point $(x^{(k_\ell)},y^{(k_\ell)})$ is feasible for $(P_{k'})$, so

$$H(x^{(k_\ell)},y^{(k_\ell)},z)\ge 0\quad \forall z\in\mathcal{V}_{k_\ell}^{(k')},$$

where $\mathcal{V}_{k_\ell}^{(k')}$ is the set defined as

$$\mathcal{V}_{k_\ell}^{(k')}:=\left\{q^{(0)}(x^{(k_\ell)},y^{(k_\ell)}),\,q^{(1)}(x^{(k_\ell)},y^{(k_\ell)}),\dots,\,q^{(k'-1)}(x^{(k_\ell)},y^{(k_\ell)})\right\}.$$

As $\ell\to\infty$, we can get

$$H(x^*,y^*,z)\ge 0\quad \forall z\in\mathcal{V}_*^{(k')}, \tag{3.10}$$

where the set $\mathcal{V}_*^{(k')}$ is

$$\mathcal{V}_*^{(k')}:=\left\{q^{(0)}(x^*,y^*),\,q^{(1)}(x^*,y^*),\dots,q^{(k'-1)}(x^*,y^*)\right\}.$$

The inequality (3.10) holds for all k', so

$$H(x^*, y^*, z) \geq 0 \quad \forall z \in T := \{q^{(k)}(x^*, y^*)\}_{k \in \mathbb{N}}. \tag{3.11}$$

It follows that

$$H(x^*, y^*, q^{(k_\ell)}(x^*, y^*)) \geq 0.$$

In Algorithm 3.3, each point $z^{(k_\ell)} \in Z(x^{(k_\ell)})$ satisfies

$$\phi(x^{(k_\ell)}, y^{(k_\ell)}) = H(x^{(k_\ell)}, y^{(k_\ell)}, z^{(k_\ell)}).$$

Therefore, we have

$$\begin{aligned} \phi(x^*, y^*) &= \phi(x^{(k_\ell)}, y^{(k_\ell)}) + \phi(x^*, y^*) - \phi(x^{(k_\ell)}, y^{(k_\ell)}) \\ &\geq \Big(H(x^{(k_\ell)}, y^{(k_\ell)}, z^{(k_\ell)}) - H(x^*, y^*, q^{(k_\ell)}(x^*, y^*))\Big) \\ &\quad + \Big(\phi(x^*, y^*) - \phi(x^{(k_\ell)}, y^{(k_\ell)})\Big). \end{aligned} \tag{3.12}$$

Since $z^{(k_\ell)} = q^{(k_\ell)}(x^{(k_\ell)}, y^{(k_\ell)})$, by the condition (d), we know that

$$\lim_{\ell \to \infty} z^{(k_\ell)} = \lim_{\ell \to \infty} q^{(k_\ell)}(x^{(k_\ell)}, y^{(k_\ell)}) = \lim_{\ell \to \infty} q^{(k_\ell)}(x^*, y^*),$$
$$H(x^{(k_\ell)}, y^{(k_\ell)}, z^{(k_\ell)}) - H(x^*, y^*, q^{(k_\ell)}(x^*, y^*)) \to 0 \quad \text{as} \quad \ell \to \infty,$$

by the continuity of the polynomial function $H(x, y, z)$ at (x^*, y^*, z^*). By the assumption, $v(x)$ is continuous at x^*, so $\phi(x, y) = v(x) - f(x, y)$ is also continuous at (x^*, y^*). Letting $\ell \to \infty$ in (3.12), we get $\phi(x^*, y^*) \geq 0$. Thus, (x^*, y^*) is feasible for (1.1) and so $F(x^*, y^*) \geq F^*$. Earlier, we already proved $F(x^*, y^*) \leq F^*$, so (x^*, y^*) is a global optimizer of (1.1), i.e., (x^*, y^*) is a global minimizer of the bilevel optimization (1.1). ◻

Remark 3.7. To ensure that the sequence $\{(x^{(k)}, y^{(k)}, z^{(k)})\}$ has an accumulation point, one may assume it is bounded. A sufficient condition for this is that the set $\mathcal{U}$ is bounded or the the upper level objective $F(x, y)$ satisfies the growth condition, i.e., the set

$$\Big\{(x, y) \in \mathcal{U} \cap \mathcal{K} : \ F(x, y) \leq \vartheta\Big\}$$

is bounded for all values ϑ. The condition (d) in Theorem 3.6 can be either checked directly on $q^{(k)}(x, y)$ or implied by that the degrees and coefficients of polynomials $q^{(k)}(x, y)$ are uniformly bounded. For instance, if the polynomial sequence of $q^{(k)}(x, y)$ has bounded degrees and bounded coefficients, then $q^{(k)}(x, y)$ must converge to $q^{(k)}(x^*, y^*)$ uniformly as $(x, y) \to (x^*, y^*)$. As shown in subsection 4.2, when the lower level optimization (P_x) has the box, simplex, or annular type constraint, the $q^{(k)}(x, y)$ can be constructed explicitly, and the resulting polynomial sequence of $q^{(k)}(x, y)$ has bounded degrees and bounded coefficients. Therefore, the convergence of Algorithm 3.3 is guaranteed for these cases of (P_x), when the conditions (a), (b), (c) hold.

4. Constructions of polynomials. In Algorithm 3.3, we need Lagrange multiplier expressions as in (3.2), required for Assumption 3.1, and the polynomial function $q(x, y)$, required in Assumption 3.2. This section discusses how they can be obtained.

4.1. Lagrange multiplier expressions. Lagrange multiplier expressions (LMEs) are discussed in [41]. For the classical single level polynomial optimization (2.3), the existence of a polynomial matrix $L(x)$ satisfying $L(x)C(x) = I_m$ is equivalent to that the constraining polynomial tuple c is nonsingular. If the feasible set $Z(x)$ of the lower level optimization (P_x) does not depend on x, i.e., (1.1) is an SBOP, the matrix $G(x, y)$ does not depend on x, and then there exists a polynomial matrix $W(y)$ satisfying $W(y)G(y) = I_{m_2}$ for generic g [41]. If $Z(x)$ depends on x, there typically does not exist $W(x, y)$ such that $W(x, y)G(x, y) = I_{m_2}$. This is because the matrix $G(x, y)$ in (3.1) is typically not full column rank for all complex $x \in \mathbb{C}^n$, $y \in \mathbb{C}^p$. We generally do not expect polynomial expressions for Lagrange multipliers of (P_x) for GBOPs.

However, we can always find a matrix polynomial $W(x, y)$ such that

$$W(x, y)G(x, y) = \mathrm{diag}[d(x, y)] \tag{4.1}$$

for a denominator polynomial vector

$$d(x, y) := \big(d_1(x, y), \ldots, d_{m_2}(x, y)\big)$$

which is nonnegative on $\mathcal{U}$. This ensures the Assumption 3.1. The $W(x, y), d(x, y)$ satisfying (4.1) are not unique. In computation, we prefer that $W(x, y), d(x, y)$ have low degrees and $d(x, y) > 0$ on $\mathcal{U}$ (or $d(x, y)$ has as few as possible zeros on $\mathcal{U}$). We would like to remark that there always exist such $W(x, y), d(x, y)$ satisfying (4.1). Note that $H(x, y) := G(x, y)^T G(x, y)$ is a positive semidefinite matrix polynomial. If the determinant $\det H(x, y)$ is not identically zero (this is the general case), then the adjoint matrix $\mathrm{adj}\big(H(x, y)\big)$ satisfies

$$\mathrm{adj}\big(H(x, y)\big)H(x, y) = \det H(x, y) I_{m_2}.$$

Then the equation (4.1) is satisfied for

$$W(x, y) := \mathrm{adj}\big(H(x, y)\big)G(x, y)^T, \quad d(x, y) = \det H(x, y)\mathbf{1}_{m_2}.$$

The above choice for $W(x, y), d(x, y)$ may not be very practical in computation, because they typically have high degrees. In applications, there often exist more suitable choices for $W(x, y), d(x, y)$ with much lower degrees.

Example 4.1. Consider the lower level optimization problem

$$\begin{cases} \min\limits_{y \in \mathbb{R}^2} & x_1 y_1 + x_2 y_2 \\ \text{s.t.} & (2y_1 - y_2, x_1 - y_1, y_2, x_2 - y_2) \geq 0. \end{cases}$$

The matrix $G(x, y)$ and $\hat{f}(x, y)$ in (3.1) are

$$G(x, y) = \begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & 0 & 1 & -1 \\ 2y_1 - y_2 & 0 & 0 & 0 \\ 0 & x_1 - y_1 & 0 & 0 \\ 0 & 0 & y_2 & 0 \\ 0 & 0 & 0 & x_2 - y_2 \end{pmatrix}, \quad \hat{f}(x, y) = \begin{pmatrix} x_1 \\ x_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Equation (4.1) holds for the denominator vector

$$d(x, y) = \big(2x_1 - y_2,\, 2x_1 - y_2,\, x_2(2x_1 - y_2),\, x_2(2x_1 - y_2)\big)$$

and the matrix $W(x,y)$ which is given as follows:

$$\begin{pmatrix} x_1-y_1 & 0 & 1 & 1 & 0 & 0 \\ y_2-2y_1 & 0 & 2 & 2 & 0 & 0 \\ (x_2-y_2)(x_1-y_1) & (x_2-y_2)(2x_1-y_2) & x_2-y_2 & x_2-y_2 & 2x_1-y_2 & 2x_1-y_2 \\ y_2(y_1-x_1) & y_2(y_2-2x_1) & -y_2 & -y_2 & 2x_1-y_2 & 2x_1-y_2 \end{pmatrix}.$$

Note that $d(x,y) \geq 0$ for all feasible (x,y).

In numerical computation, we often choose $W(x,y), d(x,y)$ in (4.1) to have low degrees and $d(x,y) > 0$ on $\mathcal{U}$ (or $d(x,y)$ has as few as possible zeros on $\mathcal{U}$). Although we prefer explicit expressions for $W(x,y)$ and $d(x,y)$, it may be too complicated to do that for some problems. In the following, we give a numerical method for finding $W(x,y)$ and $d(x,y)$. Select a point $(\hat{x},\hat{y}) \in \mathcal{U}$. For a priori low degree ℓ, we consider the following convex optimization in $W(x,y)$, $d(x,y)$:

$$(4.2) \qquad \begin{cases} \max & \gamma_1 + \cdots + \gamma_{m_2} \\ \text{s.t.} & W(x,y)G(x,y) = \operatorname{diag}[d(x,y)], \\ & d(\hat{x},\hat{y}) = \mathbf{1}_{m_2},\ \gamma_1 \geq 0, \ldots, \gamma_{m_2} \geq 0, \\ & W(x,y) \in \left(\mathbb{R}[x,y]_{2\ell-\deg(G)}\right)^{m_2\times(p+m_2)}, \\ & d_j(x,y) - \gamma_j \in \operatorname{Ideal}(\Phi)_{2\ell} + \operatorname{Qmod}(\Psi)_{2\ell}\ (j \in [m_2]). \end{cases}$$

In the above, the polynomial tuples Φ, Ψ are

$$(4.3) \qquad \Phi := \{h_i\}_{i\in\mathcal{E}_1} \cup \{g_i\}_{i\in\mathcal{E}_2}, \quad \Psi := \{h_j\}_{j\in\mathcal{I}_1} \cup \{g_j\}_{j\in\mathcal{I}_2}.$$

The first equality constraint in (4.2) is (4.1), which gives a set of linear constraints about coefficients of $W(x,y), d(x,y)$. The last constraint implies that each $d_j(x,y) \geq \gamma_j \geq 0\, \forall (x,y) \in \mathcal{U}$. The equality $d(\hat{x},\hat{y}) = \mathbf{1}_{m_2}$ ensures that $d(x,y)$ is not identically zero. As commented on earlier in this subsection, we have shown that (4.2) must have a solution if the degree ℓ is large enough, when $G(x,y)^T G(x,y)$ is not identically singular. In practice, we always start with a low degree ℓ. If (4.2) is infeasible, we then increase the value of ℓ until it becomes feasible.

4.2. The construction of polynomial extensions. We can construct a polynomial extension, required in Assumption 3.2, for many bilevel optimization problems. If (P_x) has linear equality constraints, we can get rid of them by eliminating variables. If (P_x) has nonlinear equality constraints, generally there is no polynomial $q(x,y)$ satisfying Assumption 3.2, unless the corresponding algebraic set is rational. So, we consider cases in which (P_x) has no equality constraints, i.e., the label set $\mathcal{E}_2 = \emptyset$. Moreover, we assume the polynomials $g_j(x,z)$ are linear in z for each $j \in \mathcal{E}_2$. Recall the polynomial tuples Φ, Ψ given in (4.3). For a priori degree ℓ and for given $\hat{x}, \hat{y}, \hat{z}$, we consider the following polynomial system about q:

$$(4.4) \qquad \begin{cases} q(\hat{x},\hat{y}) = \hat{z}, \\ g_j(x,q) \in \operatorname{Ideal}(\Phi)_{2\ell} + \operatorname{Qmod}(\Psi)_{2\ell}\, (j \in \mathcal{I}_2), \\ q = (q_1,\ldots,q_p) \in \left(\mathbb{R}[x,y]\right)^p. \end{cases}$$

The second constraint in (4.4) implies that $g_j(x, q(x,y)) \geq 0\ \forall (x,y) \in \mathcal{U}, j \in \mathcal{I}_2$. Hence q obtained as above must satisfy Assumption 3.2. The above program can be solved using the software `Yalmip` [33].

Example 4.2. Consider Example 4.1 with

$$\hat{x} = (1,0),\ \hat{y} = (1,0),\ \hat{z} = (0,0),$$
$$h(x,y) = (3x_1 - x_2, x_2, x_2 - x_1 + 1),$$
$$g(x,y) = (2y_1 - y_2, x_1 - y_1, y_2, x_2 - y_2).$$

For $\ell = 2$, a satisfactory $q := (q_1, q_2)$ for (4.4) is

$$q_1(x,y) = x_2/3, \quad q_2(x,y) = 2x_2/3,$$

because $g(x,q) = \frac{1}{3}(0, h_1(x,y), 2h_2(x,y), h_2(x,y))$ and

$$h_1(x,y),\ h_2(x,y) \in \mathrm{Ideal}(\Phi)_{2\ell} + \mathrm{Qmod}(\Psi)_{2\ell}.$$

For computational convenience, we prefer explicit expressions for $q(x,y)$. In the following, we give explicit expressions for various cases of bilevel optimization problems.

4.2.1. Simple bilevel optimization. If the feasible set $Z(x)$ of the lower level optimization (P_x) is independent of x, i.e., $Z(x) \equiv Z$, then we can just simply choose

$$q(x,y) := z$$

in Assumption 3.2 for all $z \in Z$ and all $(x,y) \in \mathcal{U}$. It is a constant polynomial function. Therefore, Assumption 3.2 is always satisfied for all simple bilevel optimization problems.

4.2.2. Box constraints. A typical case is that the lower level problem (P_x) has box constraints. Suppose the feasible set $Z(x)$ of (P_x) is given as

$$l(x) \le z \le u(x),$$

where $l(x) := (l_1(x), \dots, l_p(x))$, $u(x) := (u_1(x), \dots, u_p(x))$. For every $(\hat{x}, \hat{y}) \in \mathcal{U} \cap \mathcal{K}$ and $\hat{z} \in S(\hat{x})$, we can choose $q := (q_1, \dots, q_p)$ as

$$q_j(x,y) := \mu_j l_j(x) + (1 - \mu_j) u_j(x), \quad j = 1, \dots, p,$$

where each scalar $\mu_j := (u_j(\hat{x}) - \hat{z}_j)/(u_j(\hat{x}) - l_j(\hat{x})) \in [0,1]$. (For the special case that $u_j(\hat{x}) = l_j(\hat{x})$, we have $\hat{z}_j = u_j(\hat{x}) = l_j(\hat{x})$ and simply choose $\mu_j = 0$.) Then, for each j,

$$q_j(\hat{x}, \hat{y}) = \mu_j l_j(\hat{x}) + (1 - \mu_j) u_j(\hat{x}) = \hat{z}_j.$$

Clearly, $q(x,y) \in Z(x)$ for all $(x,y) \in \mathcal{U}$. The following is a more general case.

Example 4.3. Suppose the feasible set $Z(x)$ of (P_x) is given as

$$l(x) \le Az \le u(x),$$

where $A := [a_1, \dots, a_{m_2}]^T \in \mathbb{R}^{m_2 \times p}$ is a full row rank matrix and $m_2 \le p$. Let $a_{m_2+1}, \dots, a_p$ be vectors such that the matrix

$$B := [a_1, \dots, a_{m_2}, a_{m_2+1}, \dots, a_p]^T \in \mathbb{R}^{p \times p}$$

is invertible. Then the linear coordinate transformation $z = B^{-1}w$ makes the constraints become the box constraints $l_j(x) \le w_j \le u(x)_j$, $j \in [m_2]$. Hence we can choose $q = B^{-1}q'$, where $q' := (q'_1, \dots, q'_p)$ as

$$q'_j(x,y) := \begin{cases} \mu_j l_j(x) + (1 - \mu_j) u_j(x), & j = 1, \dots, m_2, \\ (By)_j, & j = m_2 + 1, \dots, p, \end{cases}$$

where each scalar

$$\mu_j := (u_j(\hat{x}) - (B\hat{z})_j)/(u_j(\hat{x}) - l_j(\hat{x})) \in [0,1].$$

For the special case that $u_j(\hat{x}) - l_j(\hat{x}) = 0$, we just set $\mu_j = 0$. One can similarly verify that $q(x,y) \in Z(x)$ for all $(x,y) \in \mathcal{U}$.

4.2.3. Simplex constraints. We consider the case that the lower level optimization (P_x) has the simplex type constraints

$$l(x) \le z, \quad \mathbf{1}^T z \le u(x),$$

where $l(x)$ is a p-dimensional polynomial function, $\mathbf{1}$ is the vector of all ones, and $u(x)$ is a scalar polynomial function in x. For every $(\hat{x},\hat{y}) \in \mathcal{U}$ and $\hat{z} \in S(\hat{x})$, we can choose $q := (q_1, \dots, q_p)$ as

$$q_j(x,y) := c_j \cdot \big(u(x) - \mathbf{1}^T l(x)\big) + l_j(x), \quad j = 1, \dots, p,$$

where each scalar $c_j := (\hat{z}_j - l_j(\hat{x}))/(u(\hat{x}) - \mathbf{1}^T l(\hat{x})) \ge 0$. (For the special case that $u(\hat{x}) - \mathbf{1}^T l(\hat{x}) = 0$, we just simply set all $c_j = 0$.) Note that

$$q_j(\hat{x},\hat{y}) = c_j\big(u(\hat{x}) - \mathbf{1}^T l(\hat{x})\big) + l_j(\hat{x}) = \hat{z}_j.$$

For all $(x,y) \in \mathcal{U}$, it is clear that $q(x,y) \ge l(x)$. Moreover, we also have

$$\mathbf{1}^T q(x,y) = \mathbf{1}^T l(x) \left(1 - \sum_{j=1}^{p} c_j\right) + \left(\sum_{j=1}^{p} c_j\right) u(x) \le u(x),$$

since $\mathbf{1}^T l(x) \le u(x)$ and $c_1 + \cdots + c_p \le 1$. Therefore, $q(x,y) \in Z(x)$ for all $(x,y) \in \mathcal{U}$. In the above, $\mathbf{1}$ can be replaced by a nonnegative vector. The following is the more general case.

Example 4.4. Suppose that the feasible set $Z(x)$ of (P_x) is given as

$$a^T z \le u(x), \quad z_j \ge l_j(x)\, (j = 1, \dots, p),$$

where $a := (a_1, \dots, a_p) \in \mathbb{R}^p_+$, $u(x)$ and all $l_j(x)$ are polynomials in x. We can choose $q := (q_1, \dots, q_p)$ as

$$q_j(x,y) := c_j \cdot \big(u(x) - a^T l(x)\big) + l_j(x),$$

where each $c_j := (\hat{z}_j - l_j(\hat{x}))/(u(\hat{x}) - a^T l(\hat{x})) \ge 0$. In particular, we set all $c_j = 0$ if $u(\hat{x}) - a^T l(\hat{x}) = 0$. Note that

$$q_j(\hat{x},\hat{y}) = l_j(\hat{x}) + c_j \cdot (u(\hat{x}) - a^T l(\hat{x})) = \hat{z}_j.$$

For all $(x,y) \in \mathcal{U}$, it is clear that $q(x,y) \ge l(x)$. In addition, we have

$$a^T q(x,y) = a^T l(x) \left(1 - \sum_{j=1}^{p} a_j c_j\right) + \left(\sum_{j=1}^{p} a_j c_j\right) u(x) \le u(x)$$

since $a^T l(x) \le u(x)$ and $a_1 c_1 + \cdots + a_p c_p \le 1$. Therefore, $q(x,y) \in Z(x)$ for all $(x,y) \in \mathcal{U}$.

4.2.4. Annular constraints. Suppose the lower level feasible set is

$$Z(x) = \left\{ y \in \mathbb{R}^p \middle| \; r(x) \le \|y - a(x)\|_d \le R(x) \; \right\},$$

where $\|z\|_d := \sqrt[d]{\sum_{i=1}^p |z_i|^d}$ and $a(x) := [a_1(x), \ldots, a_p(x)]$ is a polynomial vector, and $r(x), R(x)$ are polynomials such that $0 \le r(x) \le R(x)$ on $\mathcal{U}$. We can choose

$$q(x, y) := a(x) + q'(x)s,$$

where $q'(x) := \mu_1 r(x) + \mu_2 R(x)$, μ_1, μ_2 are scalars such that

$$\|\hat{z} - a(\hat{x})\|_d = \mu_1 r(\hat{x}) + \mu_2 R(\hat{x}), \quad \mu_1, \mu_2 \ge 0, \quad \mu_1 + \mu_2 = 1,$$

and $s := (s_1, \ldots, s_p)$ is the vector such that

$$s_i := \frac{\hat{z}_i - a_i(\hat{x})}{\|\hat{z} - a(\hat{x})\|_d}, \quad i = 1, \ldots, p.$$

(For the special case that $\hat{z} = a(\hat{x})$, we just set all $s_i = p^{-1/d}$.) Then,

$$\begin{aligned} \hat{z} - q(\hat{x}, \hat{y}) &= (\hat{z} - a(\hat{x})) - (q(\hat{x}, \hat{y}) - a(\hat{x})) \\ &= (\hat{z} - a(\hat{x})) - q'(\hat{x})s = 0, \end{aligned}$$

since $q'(\hat{x}) = \|\hat{z} - a(\hat{x})\|_d$. Moreover,

$$\|q(x, y) - a(x)\|_d = \|q'(x)s\|_d = |q'(x)| \cdot \|s\|_d = |q'(x)|.$$

Because $0 \le r(x) \le R(x)$ on $\mathcal{U}$, we must have

$$r(x) \le \|q(x, y) - a(x)\|_d \le R(x).$$

This means that $q(x, y)$ satisfies Assumption 3.2.

5. Numerical experiments. In this section, we report numerical results of applying Algorithm 3.3 to solve bilevel polynomial optimization problems. The computation is implemented in MATLAB R2018a, on a Laptop with CPU 8th Generation Intel R CoreTM i5-8250U and RAM 16 GB. The software `GloptiPoly 3` [21] and `SeDuMi` [49] are used to solve the polynomial optimization problems in Algorithm 3.3. In this section, we use the following notation.

- The matrix $G(x, y)$ and vector $\hat{f}(x, y)$ are given as in (3.1). The polynomials $\phi_j(x, y), d_j(x, y)$ for LMEs in Assumption 3.1 are given by (3.3), i.e., $\phi_j(x, y)$ is the jth entry of $W(x, y)\hat{f}(x, y)$, for a matrix polynomial $W(x, y)$ satisfying (4.1). In our examples, such $W(x, y)$ is determined by symbolic Gaussian elimination on (4.1).
- The notation (P) denotes the bilevel optimization (1.1). Its optimal value and optimizers are denoted by F^* and (x^*, y^*), respectively.
- The (P_k) denotes the relaxed polynomial optimization in the kth loop of Algorithm 3.3. Its optimal value and minimizers are denoted as F_k^* and $(x^{(k)}, y^{(k)})$, respectively.

Table 1
Computational results for some SBOPs.

$\min_{x,y\in\mathbb{R}} \; x+y$ s.t. $(x+1, 1-x) \geq 0,$ $y \in \arg\min_{z\in\mathbb{R}} \frac{1}{2}xz^2 - \frac{1}{3}z^3$ s.t. $(z+1, 1-z) \geq 0$	$F^* = -1.2380 \cdot 10^{-8}$, $v^* = -3.9587 \cdot 10^{-8}$, $x^* = -1.0000$, $y^* = 1.0000$.
$\min_{x,y\in\mathbb{R}^2} \; x_1^2 - 2x_1 + x_2^2 - 2x_2 + y_1^2 + y_2^2$ s.t. $(x_1, x_2, y_1, y_2, 2-x_1) \geq 0,$ $y \in \arg\min_{z\in\mathbb{R}^2} z_1^2 - 2x_1z_1 + z_2^2 - 2x_2z_2$ s.t. $0.25 - (z_1-1)^2 \geq 0,$ $0.25 - (z_2-1)^2 \geq 0.$	$F^* = -1.0000$, $v^* = -1.3113 \cdot 10^{-9}$, $x^* = (0.5000, 0.5000)$, $y^* = (0.5000, 0.5000)$.
$\min_{x,y\in\mathbb{R}^2} \; 2x_1 + x_2 - 2y_1 + y_2$ s.t. $(1+x_1, 1-x_1, 1+x_2, -0.75-x_2) \geq 0,$ $y \in \arg\min_{z\in\mathbb{R}^2} x^Tz$ s.t. $(2z_1 - z_2, 2 - z_1) \geq 0,$ $(z_2, 2-z_2) \geq 0.$	$F^* = -5.0000$, $v^* = -1.4163 \cdot 10^{-8}$, $x^* = (-1.0000, -1.0000)$, $y^* = (2.0000, 2.0000)$.
$\min_{x\in\mathbb{R}^2, y\in\mathbb{R}^3} \; x_1y_1 + x_2y_2 + x_1x_2y_1y_2y_3$ s.t. $(1-x_1^2, 1-x_2^2, x_1^2 - y_1y_2) \geq 0,$ $y \in \arg\min_{z\in\mathbb{R}^3} x_1z_1^2 + x_2^2z_2z_3 - z_1z_3^2$ s.t. $(z^Tz - 1, 2 - z^Tz) \geq 0.$	$F^* = -1.7095$, $v^* = -1.3995 \cdot 10^{-9}$, $x^* = (-1.0000, -1.0000)$, $y^* = (1.1097, 0.3143, -0.8184)$.
$\min_{x,y\in\mathbb{R}^2} \; (x_1-30)^2 + (x_2-20)^2 - 20y_1 + 20y_2$ s.t. $(x_1 + 2x_2 - 30, 25 - x_1 - x_2, 15 - x_2) \geq 0,$ $y \in \arg\min_{z\in\mathbb{R}^2} (x_1-z_1)^2 + (x_2-z_2)^2$ s.t. $(10-z_1, 10-z_2, z_1, z_2) \geq 0.$	$F^* = 225.0000$, $v^* = -1.6835 \cdot 10^{-9}$, $x^* = (20.0000, 5.0000)$, $y^* = (10.0000, 5.0000)$.

- The (Q_k) denotes the lower level optimization problem (3.8) in the kth loop of Algorithm 3.3. Its optimal value and minimizers are denoted as v_k and $z^{(k)}$,
- We always have $v_k \leq 0$. Note that $y^{(k)}$ is a minimizer of (3.8) if and only if $v_k = 0$. Due to numerical roundoff errors, we cannot have $v_k = 0$ exactly. We view $y^{(k)}$ as a minimizer of (3.8) if $v_k \geq -\epsilon$ for a tiny scalar ϵ (e.g., 10^{-6}).

Example 5.1. First, we apply Algorithm 3.3 to solve SBOPs. The displayed problems are, respectively, from [31, Example 5.2], [2, Example 3], [15, Example 3.8], [42, Example 5.2], and [47, Example 2]. All but the first problem are solved successfully in the initial loop $k = 0$. The computational results are shown in Table 1, where arg min denotes the set of minimizer(s). In Table 1, we use v^* to denote the value of v_k in the last loop. Algorithm 3.3 computed global optimizers for all of them. In Table 2, we compare Algorithm 3.3 with some prior methods for solving SBOPs in existing references, for the quality of computed solutions and the consumed CPU time (in seconds). For the SBOP in [2, Example 3] and [15, Example 3.8], no CPU time was given in the work, so we implement their methods with the `MATLAB` function `fmincon`. For [2, Example 3], the method requires one to choose starting points. The performance depends on the choice. We chose 100 random starting points. For some of them, the method converges; for the others, it does not. We report the minimum CPU time for cases that it converges in Table 2. In [47, Example 2], it was mentioned that 225 is the true optimal value but the method there cannot compute it accurately. There is no publicly available code for implementing that method, so its CPU time is not reported.

TABLE 2
Comparison with prior methods for some SBOPs.

	Prior methods		Algorithm 3.3	
	F^*	time	F^*	time
[31, Example 5.2]	$4.7260 \cdot 10^{-8}$	47.48	$-1.2380 \cdot 10^{-8}$	0.89
[2, Example 3]	-1.0000	0.05	-1.0000	0.34
[15, Example 3.8]	-5.0000	0.06	-5.0000	0.27
[42, Example 5.2]	-1.7095	13.45	-1.7095	6.43
[47, Example 2]	228.7000	not available	225.0000	0.27

TABLE 3
Computational results for Example 5.2.

(P_0)	$F_0^* = -0.7688$,
	$x^{(0)} = (0.6819,\ 1.7059)$, $y^{(0)} = (0.3997,\ 0.6819)$,
(Q_0)	$\upsilon_0 = -3.3569 \cdot 10^{-7} \to$ stop.
Time	0.31 seconds,
Output	$F^* = F_0^*,\ x^* = x^{(0)},\ y^* = y^{(0)}$.

Example 5.2. Consider the GBOP

$$\begin{cases} \min\limits_{x,y\in\mathbb{R}^2} & x_1y_1^3 + x_2y_2^3 - x_1^2x_2^2 \\ \text{s.t.} & (x_1x_2 - 1,\ x_1,\ x_2,\ 4 - x_1^2 - x_2^2 - y_1^2 - y_2^2) \geq 0, \\ & y \in S(x), \end{cases}$$

where $S(x)$ is the optimizer set of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^2} & z_1^2 + z_2^2 - 2x_2z_1 - x_1x_2z_2 \\ \text{s.t.} & (z_1,\ z_2 - x_2z_1,\ 2x_1 - x_2z_1 - z_2) \geq 0. \end{cases}$$

The polynomial matrix $W(x, y)$ satisfying (4.1) is

$$\begin{pmatrix} 2x_1 - 2x_2y_1 & 2x_1x_2 - 2x_2y_2 & 2x_2 & 2x_2 & 2x_2 \\ -y_1 & 2x_1 - y_2 & 1 & 1 & 1 \\ -y_1 & -y_2 & 1 & 1 & 1 \end{pmatrix}$$

for the denominators

$$d_1(x,y) = d_2(x,y) = d_3(x,y) = 2x_1 > 0\ \forall (x,y) \in \mathcal{U}.$$

The lower level optimization is convex for given x. According to Proposition 3.4, we get the optimizer for this bilevel optimization in the initial loop $k = 0$ by Algorithm 3.3. The computational results are shown in Table 3.

Example 5.3 (see [38, Example 2]). Consider the general bilevel optimization

$$\begin{cases} \min\limits_{x\in\mathbb{R}^2, y\in\mathbb{R}^3} & y_1^2 + y_3^2 - y_1y_3 - 4y_2 - 7x_1 + 4x_2 \\ \text{s.t.} & (x_1, x_2, 1 - x_1 - x_2) \geq 0,\ y \in S(x), \end{cases}$$

where $S(x)$ is the optimizer set of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^3} & z_1^2 + 0.5z_2^2 + 0.5z_3^2 + z_1z_2 + (1 - 3x_1)z_1 + (1 + x_2)z_2 \\ \text{s.t.} & (-2z_1 - z_2 + z_3 - x_1 + 2x_2 - 2, z_1, z_2, z_3) \geq 0. \end{cases}$$

TABLE 4
Computational results for Example 5.3.

(P_0)	$F_0^* = 0.6389$,
	$x^{(0)} = (0.6111, 0.3889)$, $y^{(0)} = (0.0000, 0.0000, 1.8332)$,
(Q_0)	$v_0 = -6.7295 \cdot 10^{-9} \to$ stop.
Time	1.09 seconds,
Output	$F^* = F_0^*,\ x^* = x^{(0)},\ y^* = y^{(0)}$.

The polynomial matrix $W(x, y)$ satisfying (4.1) is

$$\begin{pmatrix} y_1 & y_2 & y_3 & -1 & -1 & -1 & -1 \\ 2 + x_1 + 2y_1 - 2x_2 & 2y_2 & 2y_3 & -2 & -2 & -2 & -2 \\ y_1 & 2 + x_1 + y_2 - 2x_2 & y_3 & -1 & -1 & -1 & -1 \\ -y_1 & -y_2 & 2 + x_1 - 2x_2 - y_3 & 1 & 1 & 1 & 1 \end{pmatrix}$$

for the denominators ($i = 1, 2, 3, 4$)

$$d_i(x, y) = 2 + x_1 - 2x_2 = 3h_1(x, y) + 2h_3(x, y) \geq 0 \ \forall (x, y) \in \mathcal{U}.$$

By Algorithm 3.3, we get the optimizer for this bilevel optimization in the initial loop $k = 0$. The computational results are shown in Table 4.

Example 5.4 (see [42, Example 5.8]). Consider the general bilevel optimization

$$\begin{cases} \min\limits_{x,y \in \mathbb{R}^4} & (x_1 + x_2 + x_3 + x_4)(y_1 + y_2 + y_3 + y_4) \\ \text{s.t.} & (1 - x^T x, x_4 - y_3^2, x_1 - y_2 y_4) \geq 0,\ y \in S(x), \end{cases}$$

where $S(x)$ is the set of optimizer(s) of

$$\begin{cases} \min\limits_{z \in \mathbb{R}^4} & x_1 z_1 + x_2 z_2 + 0.1 z_3 + 0.5 z_4 - z_3 z_4 \\ \text{s.t.} & (x_1^2 + x_3^2 + x_2 + x_4 - z_1^2 - 2z_2^2 - 3z_3^2 - 4z_4^2, z_2 z_3 - z_1 z_4) \geq 0. \end{cases}$$

The matrix polynomial $W(x, y)$ satisfying (4.1) is

$$y_4 \cdot \begin{pmatrix} -y_1 y_4 & -y_2 y_4 & -y_3 y_4 & -y_4^2 & 2y_4 & 2y_4 \\ 2y_1^2 - 2(x_1^2 + x_3^2 + x_2 + x_4) & 2y_1 y_2 & 2y_1 y_3 & 2y_1 y_4 & -4y_1 & -4y_1 \end{pmatrix}$$

for the denominators

$$\begin{aligned} d_1(x, y) = d_2(x, y) &= 2y_4^2(x_1^2 + x_3^2 + x_2 + x_4) \\ &\geq 2y_4^2(y_1^2 + 2y_2^2 + 3y_3^2 + 4y_4^2) \geq 0 \ \forall (x, y) \in \mathcal{U}. \end{aligned}$$

By Algorithm 3.3, we get the optimizer for this bilevel optimization in the initial loop $k = 0$. The computational results are shown in Table 5.

Example 5.5. Consider the GBOP

$$\begin{cases} \min\limits_{x,y \in \mathbb{R}^4} & x_1^2 y_3^2 - 2x_3 x_4 + 1.2 x_1 x_3 - x_4^2(y_3 + 2y_4) \\ \text{s.t.} & (\mathbf{1}^T x, 8 - \mathbf{1}^T x, 4x_1 x_2 - y_1^2 - y_2^2) \geq 0, \\ & (x_1 - y_1, x_2 - y_2, 4 - x_1 - x_2, 4 - x_3^2 - x_4^2) \geq 0, \\ & y \in S(x), \end{cases}$$

TABLE 5
Computational results for Example 5.4.

(P_0)	$F_0^* = -3.5050$,
	$x^{(0)} = (0.5442, 0.4682, 0.4904, 0.4942)$,
	$y^{(0)} = (-0.7791, -0.5034, -0.2871, -0.1855)$,
(Q_0)	$\upsilon_0 = -1.6143 \cdot 10^{-9} \to$ stop.
Time	49.08 seconds,
Output	$F^* = F_0^*,\ x^* = x^{(0)},\ y^* = y^{(0)}$.

TABLE 6
Computational results for Example 5.5.

(P_0)	$F_0^* = -24.6491$,
	$x^{(0)} = (0.0000,\ 0.0000,\ 0.3204,\ 1.9742)$,
	$y^{(0)} = (0.0000,\ -0.0000,\ 0.0000,\ 3.0000)$,
(Q_0)	$\upsilon_0 = -2.5204 \cdot 10^{-8} \to$ stop;
Time	2.90 seconds
Output	$F^* = F_0^*$, $x^* = x^{(0)}$, $y^* = y^{(0)}$.

where $S(x)$ is the set of optimizer(s) of

$$\begin{cases} \min\limits_{z \in \mathbb{R}^4} & x_1 z_1^2 + x_2 z_2^2 + x_3 z_3 - x_4 z_4 \\ \text{s.t.} & (z_1 - z_2 - x_2, x_1 - z_1 + z_2, z_1 + z_2 + x_1 + x_2) \geq 0, \\ & (4x_1 - 2x_2 - z_1 - z_2, z_3, z_4, 3 - z_3 - z_4) \geq 0. \end{cases}$$

The matrix polynomial $W(x, y)$ satisfying (4.1) is

$$\begin{pmatrix} x_1 - y_1 + y_2 & y_1 - y_2 - x_1 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\ x_2 - y_1 + y_2 & y_1 - y_2 - x_2 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\ 4x_1 - 2x_2 - y_1 - y_2 & 4x_1 - 2x_2 - y_1 - y_2 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ y_1 + y_2 + x_1 + x_2 & y_1 + y_2 + x_1 + x_2 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & y_4 & -y_4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -y_3 & y_3 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -y_3 & -y_4 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

for the denominator vector

$$\begin{aligned} d(x, y) &= \big(2x_1 - 2x_2,\ 2x_1 - 2x_2,\ 10x_1 - 2x_2,\ 10x_1 - 2x_2,\ y_4,\ y_3,\ 3\big) \\ &= \big(2(g_1(x, y) + g_2(x, y)) \cdot \mathbf{1}_2,\ 2(g_3(x, y) + g_4(x, y)) \cdot \mathbf{1}_2,\ g_6(x, y),\ g_5(x, y),\ 3\big). \end{aligned}$$

The denominators are all nonnegative on $\mathcal{U}$. By Algorithm 3.3, we get the optimizer of this bilevel optimization in the initial loop $k = 0$. The computational results are shown in Table 6.

Example 5.6. Consider the general bilevel optimization

$$\begin{cases} \min\limits_{x, y \in \mathbb{R}^4} & y_1 x_1^2 + y_2 x_2^2 - y_3 x_3 - y_4 x_4 \\ \text{s.t.} & (x_1 - 1, x_2 - 1, 4 - x_1 - x_2) \geq 0, \\ & (x_3 - 1, 2 - x_4, x_3^2 - 2x_4, 8 - x^T x) \geq 0, \\ & y \in S(x), \end{cases}$$

where $S(x)$ is the set of optimizer(s) of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^4} & -z_1z_2+z_3+z_4 \\ \text{s.t.} & (z_1,z_2,z_3-x_4,z_4-x_3)\geq 0, \\ & (4x_1x_2-x_1z_1-x_2z_2,3-z_3-z_4)\geq 0. \end{cases}$$

The polynomial matrix $W(x,y)$ satisfying (4.1) is

$$\begin{pmatrix} x_1(4x_2+y_2)-x_1y_1-x_2y_2 & -x_1y_2 & 0 & 0 & x_1 & x_1 & 0 & 0 & x_1 & 0 \\ -x_1y_1 & 4x_1x_2-x_2y_2 & 0 & 0 & x_1 & x_1 & 0 & 0 & x_1 & 0 \\ 0 & 0 & 3-x_3-y_3 & x_3-y_4 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & x_4-y_3 & 3-x_4-y_4 & 0 & 0 & 1 & 1 & 0 & 1 \\ -y_1 & -y_2 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_4-y_3 & x_3-y_4 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

for the denominator vector $d(x,y)$ as follows:

$$d(x,y)=(4x_1x_2+x_1y_2-x_2y_2,4x_1x_2+x_1y_2-x_2y_2,\\ 3-x_3-x_4,3-x_3-x_4,4x_1x_2+x_1y_2-x_2y_2,3-x_3-x_4).$$

It is clear that $d(x,y)\geq 0$ for all feasible (x,y). As in the subsection 4.2.3, the polynomial function $q:=(q_1,q_2,q_3,q_4)$ in Assumption 3.2 can be given as

$$q_1=\mu_1x_2,\ q_2=\mu_2x_1,\ q_3=x_4+\mu_3(3+x_3+x_4),\ q_4=x_3+\mu_4(3+x_3+x_4), \tag{5.1}$$

where (see subsections 4.2.3 for the notation $\hat{x},\hat{y},\hat{z}$)

$$\mu_1=\frac{\hat{z}_1}{\hat{x}_2},\ \mu_2=\frac{\hat{z}_2}{\hat{x}_1},\ \mu_3=\frac{\hat{z}_3-\hat{x}_4}{3+\hat{x}_3+\hat{x}_4},\ \mu_4=\frac{\hat{z}_4-\hat{x}_3}{3+\hat{x}_3+\hat{x}_4}$$

for given $(\hat{x},\hat{y})\in\mathcal{U}$. Since $x_1,x_2,x_3\geq 1$ and $x_4\geq -2\sqrt{2}$, the above μ_1,μ_2,μ_3,μ_4 are well defined. Applying Algorithm 3.3, we get the optimizer for this bilevel optimization in the loop $k=1$. The computational results are shown in Table 7.

Example 5.7. Consider the GBOP

$$\begin{cases} \min\limits_{x,y\in\mathbb{R}^4} & x_1^2y_4^2-x_2y_3^2+x_3y_1-x_4y_2 \\ \text{s.t.} & (4-x_1^2-x_2^2,\ -x_1-x_2^2,\ y_1-x_1,\ \mathbf{1}^Tx)\geq 0, \\ & (x_3+x_4-3,\ 1+x_3-x_4,\ 3-x_3,\ x_4)\geq 0, \\ & y\in S(x), \end{cases}$$

where $S(x)$ is the optimizer(s) set of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^4} & (x_1-z_1)^2+(x_2-z_2)^2+z_3-z_4 \\ \text{s.t.} & 4x_3^2-x_1^2-x_2^2+2x_1z_1+2x_2z_2-z^Tz\geq 0, \\ & (z_3,\ x_3-z_3,\ z_4,\ x_4-z_4)\geq 0. \end{cases}$$

TABLE 7
Computational results for Example 5.6.

(P_0)	$F_0^* = -4.4575$,
	$x^{(0)} = (1.1548,\ 1.1546,\ 1.6458,\ 1.3542)$,
	$y^{(0)} = (0.0000,\ 0.0000,\ 1.3542,\ 1.6458)$,
(Q_0)	$v_0 = -5.3362 \to$ next loop;
	$z^{(0)} = (2.3093,\ 2.3096,\ 1.3542,\ 1.6458)$,
	$q^{(0)} = (2x_2,\ 2x_1,\ x_4,\ x_3)$ as in (5.1).
(P_1)	$F_1^* = -0.4574$,
	$x^{(1)} = (1.0000,\ 1.0000,\ 1.6458,\ 1.3542)$,
	$y^{(1)} = (2.0000,\ 2.0000,\ 1.3542,\ 1.6458)$,
(Q_1)	$v_1 = -1.9402 \cdot 10^{-9} \to$ stop.
Time	102.21 seconds,
Output	$F^* = F_1^*,\ x^* = x^{(1)},\ y^* = y^{(1)}$.

The matrix polynomial $W(x,y)$ satisfying (4.1) is

$$\begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -(x_3-y_3)y_3 & 0 & (x_3-y_3)(y_1-x_1) & 0 & 0 & 0 & y_1-x_1 & 0 & 0 \\ y_3^2 & 0 & -y_3(y_1-x_1) & 0 & 0 & y_1-x_1 & 0 & 0 & 0 \\ -(x_4-y_4)y_4 & 0 & 0 & (x_4-y_4)(y_1-x_1) & 0 & 0 & 0 & 0 & y_1-x_1 \\ y_4^2 & 0 & 0 & -y_4(y_1-x_1) & 0 & 0 & 0 & y_1-x_1 & 0 \end{pmatrix}$$

for the denominator vector

$$d(x,y) = (y_1 - x_1) \cdot \big(2,\ x_3 - y_3,\ y_3,\ x_4 - y_4,\ y_4\big).$$

It is clear that $d(x,y) \geq 0$ for all feasible (x,y). The lower level feasible set $Z(x)$ is a mixture of separable and annular constraints:

$$Z(x) = \left\{ z \in \mathbb{R}^4 \;\middle|\; \begin{array}{c} (z_1 - x_1)^2 + (z_2 - x_2)^2 + z_3^2 + z_4^2 \leq 4x_3^2, \\ 0 \leq z_3 \leq x_3,\ 0 \leq z_4 \leq x_4 \end{array} \right\}.$$

As in subsections 4.2.2 and 4.2.4, the polynomial function $q := (q_1, q_2, q_3, q_4)$ in Assumption 3.2 can be given as

$$q_1 = x_1 + \mu_1 x_3,\ q_2 = x_2 + \mu_2 x_3,\ q_3 = \mu_3 x_3,\ q_4 = \mu_4 x_4, \tag{5.2}$$

where (for a given value $(\hat{x}, \hat{y}, \hat{z})$ of $(x^{(k)},\ y^{(k)},\ z^{(k)})$, q satisfies $q(\hat{x}, \hat{y}) = \hat{z}$)

$$\mu_1 = \frac{\hat{z}_1 - \hat{x}_1}{\hat{x}_3},\ \mu_2 = \frac{\hat{z}_2 - \hat{x}_2}{\hat{x}_3},\ \mu_3 = \frac{\hat{z}_3}{\hat{x}_3},\ \mu_4 = \frac{\hat{z}_4}{\hat{x}_4}.$$

Since $1 \leq \hat{x}_3 \leq 3$ and $0 \leq \hat{x}_4 \leq 1 + \hat{x}_3$, we have $\mu_4 = 0$ for the special case when $\hat{x}_4 = 0$, thus the above q is well defined. This bilevel optimization was solved by Algorithm 3.3 in the loop $k = 1$. The computational results are shown in Table 8.

TABLE 8
Computational results for Example 5.7.

(P_0)	$F_0^* = -41.7143$,
	$x^{(0)} = (-1.5616, 1.2496, 3.0000, 4.0000)$,
	$y^{(0)} = (-1.5616, 6.4458, 3.0000, 0.0008)$,
(Q_0)	$\upsilon_0 = -33.9991$,
	$z^{(0)} = (-1.5615, 1.2496, 0.0000, 4.0000)$,
	$q^{(0)} = (x_1, x_2, 0, x_4)$ as in (5.2).
(P_1)	$F_1^* = -6.0000$,
	$x^{(1)} = (-2.0000, 0.0001, 3.0000, 0.0001)$,
	$y^{(1)} = (-2.0000, 0.0001, -0.0000, 0.0001)$,
(Q_1)	$\upsilon_1 = -2.7612 \cdot 10^{-9} \to$ stop.
Time	3.42 seconds,
Output	$F^* = F_1^*,\ x^* = x^{(1)},\ y^* = y^{(1)}$.

6. Conclusions and discussions. We propose a new method for solving general bilevel polynomial optimization problems, which consists of solving a sequence of polynomial optimization relaxations. Each relaxation is obtained by using KKT conditions for the lower level optimization. For KKT conditions, the Lagrange multipliers are represented as a polynomial or a rational function. The Moment-SOS relaxations are used to solve each polynomial relaxation, which is then refined by the exchange technique from SIP. Under some suitable assumptions, we prove the convergence for both simple and general bilevel polynomial optimization problems. Numerical experiments are presented to show the efficiency of the method. In all of our numerical experiments, the algorithm converges to optimizers in a few loops. An interesting future work is to explore the complexity of the algorithm.

We would like to emphasis that when the lower level optimization (P_x) is replaced by its KKT system, the resulting new optimization may not be equivalent to the original bilevel optimization (1.1) in the sense that optimal solutions (1.1) may not be recovered by solving the the initial polynomial optimization relaxation (P_0). There exists such an example of exponential functions as in [36]. In the following, we provide a new example of polynomial functions.

Example 6.1. Consider the SBOP

$$\begin{cases} \min\limits_{x\in\mathbb{R}^1, y\in\mathbb{R}^1} & xy - y + \frac{1}{2}y^2 \\ \text{s.t.} & 1-x^2 \geq 0, 1-y^2 \geq 0, \\ & y \in S(x), \end{cases}$$

where $S(x)$ is the optimizer set of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^1} & -xz^2 + \frac{1}{2}xz^4 \\ \text{s.t.} & 1-z^2 \geq 0. \end{cases}$$

The KKT condition $\nabla_z f(x,z) - \lambda \nabla_z g(z) = 0$ for the lower level optimization is

$$-2xz + 2xz^3 + 2\lambda z = 0.$$

Therefore, the initial polynomial optimization relaxation (P_0) is equivalent to

$$\begin{cases} \min\limits_{x,y,\lambda\in\mathbb{R}} & xy - y + \frac{1}{2}y^2 \\ \text{s.t.} & 1-x^2 \geq 0,\ 1-y^2 \geq 0, \\ & -xy + xy^3 + \lambda y = 0,\ \lambda \geq 0,\ \lambda(1-y^2) = 0. \end{cases}$$

TABLE 9
Computational results for Example 6.1.

(P_0)	$F_0^* = -1.5000$,
	$(x^{(0)}, y^{(0)}) = (-1.0000,\ 1.0000)$,
(Q_0)	$v_0 = -0.5000 \to$ next loop;
	$z^{(0)} = 1.1385 \cdot 10^{-17}$, $q^{(0)} = z^{(0)}$ as in section 4.2.1.
(P_1)	$F_1^* = -0.5000$,
	$(x^{(1)}, y^{(1)}) = (2.4099 \cdot 10^{-9},\ 1.0000)$,
(Q_1)	$v_1 = -3.5197 \cdot 10^{-12} \to$ stop.
Time	0.75 second,
Output	$F^* = F_1^*,\ x^* = x^{(1)},\ y^* = y^{(1)}$.

By a brute force computation, the above optimization has the optimal value and minimizer, respectively,

$$F_c^* = -1.5, \quad (x_c^*, y_c^*) = (-1, 1).$$

However, (x_c^*, y_c^*) is not even feasible for the original bilevel optimization, since $y_c^* \notin S(x_c^*) = \{0\}$. We can apply Algorithm 3.3 to solve this SBOP, with the LME $\lambda(x, y) = x - xy^2$. It terminated in the loop $k = 1$. The computational results are shown in Table 9.

One may consider solving the suboptimization problems (P_k) and (Q_k) in Algorithm 3.3 by methods other than the classical Lasserre type Moment-SOS relaxations, e.g., the BDSOS relaxations [29], and the bounded degree SOCP (BDSOCP) relaxations [10] that is a mixture of both BDSOS and SDSOS polynomials [1]. As requested by referees, we give a computational comparison in the following example. We remark that Algorithm 3.3 fails when these two new relaxations are used to solve (P_k).

Example 6.2 (see [44, Example 3.1]). Consider the GBOP

$$\begin{cases} \min\limits_{x\in\mathbb{R}^1, y\in\mathbb{R}^2} & 0.5(y_1-3)^2 + 0.5(y_2-4)^2 \\ \text{s.t.} & (x, 10-x) \geq 0,\ y \in S(x), \end{cases}$$

where $S(x)$ is the optimizer set of

$$\begin{cases} \min\limits_{z\in\mathbb{R}^2} & 0.5(1+0.2x)z_1^2 + 0.5(1+0.1x)z_2^2 - (3+1.333x)z_1 - xz_2 \\ \text{s.t.} & (0.333z_1 - z_2 - 0.1x + 1,\ 9 + 0.1x - z_1^2 - z_2^2,\ z_1,\ z_2) \geq 0. \end{cases}$$

The polynomial matrix $W(x, y)$ satisfying (4.1) is

$$\begin{pmatrix} 6.006y_1y_2^2 & -6.006y_1^2y_2 & 0 & 0 & -6.006y_2^2 & 6.006y_1^2 \\ -3.003y_1y_2 & -y_1y_2 & 0 & 0 & 3.003y_2 & y_1 \\ 0 & 0 & 0 & 0 & 6.006y_1y_2 + 2y_2^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6.006y_1^2 + 2y_1y_2 \end{pmatrix}$$

with the denominators

$$d_1(x, y) = d_2(x, y) = 6.006y_1^2y_2 + 2y_1y_2^2 \geq 0\ \forall (x, y) \in \mathcal{U}.$$

When the classical Lasserre type Moment-SOS relaxations are used to solve (P_k) and (Q_k) in Algorithm 3.3, we get the correct solution successfully in the initial loop $k = 0$.

TABLE 10
Computational results for Example 6.2.

(P_0)	$F_0^* = 3.2077$,
	$x^{(0)} = 4.0604, \quad y^{(0)} = (2.6822, 1.4871)$,
(Q_0)	$v_0 = -3.7906 \cdot 10^{-6} \to$ stop.
Time	1.031 seconds,
Output	$F^* = F_0^*,\ x^* = x_0^*,\ y^* = y_0^*$.

TABLE 11
Computational results for Example 6.2 *with BDSOCP and BDSOS on suboptimization problem* (P_0).

		BDSOCP		BDSOS	
M	d	$\hat{F}^*_{d,M}$	Time	$\tilde{F}^*_{d,M}$	Time
10	1	2.6011	0.40	2.6011	0.27
	2	3.5516	1.03	5.6719	0.92
	3	5.8476	51.17	8.4647	51.01
100	1	2.6011	0.16	2.6011	0.09
	2	3.2121	0.86	3.5696	0.63
	3	3.7170	50.45	5.8199	48.30
1000	1	2.6012	0.16	2.6011	0.09
	2	2.7375	0.72	3.7937	0.95
	3	3.7297	41.52	7.9497	39.33

The computational results are shown in Table 10, which are the same as in [44]. Now we apply BDSOCP [10] and BDSOS [29] to solve (P_0). We implement these two new relaxation methods in SPOT [35] and solve the resulting SOCP and SDP by `MOSEK`. Both BDSOCP and BDSOS require one to use a parameter M at the beginning. A scale factor $1/M$ will be multiplied to each inequality constraint, to ensure that the constraining function value is always between 0 and 1. We remark that estimating such an M exactly is quite difficult, which is equivalent to solving another polynomial optimization problem [10]. Here, we tune the parameter M. Let d be the relaxation order for both BDSOCP and BDSOS. Denote by $\hat{F}^*_{d,M}$ and $\tilde{F}^*_{d,M}$ the objective values that are computed by BDSOCP and BDSOS, respectively. The computational results are shown in Table 11. The time there is measured in seconds. None of these two methods solved the initial optimization (P_0) well, so Algorithm 3.3 fails to continue. This GBOP was not solved accurately by either BDSOCP or BDSOS relaxations.

In this paper, we assumed the KKT conditions are satisfied at global optimizers of the lower level optimization (P_x). When the KKT conditions fail to hold for (P_x), we do not know how to apply our proposed method. For such a case, we may consider to use Fritz John conditions and Jacobian representations as in the work [42]. The KKT approach has advantages, as well as potential drawbacks, for solving bilevel optimization. We refer to the work [9] for this issue. It is important future work to solve BOPs when the KKT conditions fail for (P_x).

REFERENCES

[1] A. A. AHMADI AND A. MAJUMDAR, *DSOS and SDSOS optimization: More tractable alternatives to sum of squares and semidefinite optimization*, SIAM J. Appl. Algebra Geom., 3 (2019), pp. 193–230.

[2] G. ALLENDE AND G. STILL, *Solving bilevel programs with the KKT-approach*, Math. Program., 138 (2013), pp. 309–332.

[3] K. Bai and J.J. Ye, *Directional necessary optimality conditions for bilevel programs*, Math. Oper. Res., to appear.
[4] J. Bard, *Practical Bilevel Optimization: Algorithms and Applications*, Kluwer Academic, Dordrecht, Netherlands, 1998.
[5] A. Ben-Tal and C. Blair, *Computational difficulties of bilevel linear programming*, Oper. Res., 38 (1990), pp. 556–560.
[6] D. Bertsekas, *Nonlinear Programming*, 2nd ed., Athena Scientific, Belmont, MA, 1995.
[7] M. Bjørndal and K. Jørnsten, *The deregulated electricity market viewed as a bilevel programming problem*, J. Global Optim., 33 (2005), pp. 465–475.
[8] G. Boglárka and K. Kovács, *Solving a Huff-like Stackelberg location problem on networks*, J. Global Optim., 64 (2016), pp. 233–247.
[9] G. Bouza Allende and G. Still, *Solving bilevel programs with the KKT-approach*, Math. Program., 138 (2013), pp. 309–332.
[10] T. Chuong, V. Jeyakumar, and G. Li, *A new bounded degree hierarchy with SOCP relaxations for global polynomial optimization and conic convex semi-algebraic programs*, J. Global Optim., 75 (2019), pp. 885–919.
[11] F. H. Clarke, *Optimization and Nonsmooth Analysis*,Classics Appl. Math. 5, SIAM, Philadelphia, 1990.
[12] B. Colson, P. Marcotte, and G. Savard, *An overview of bilevel optimization*, Ann. Oper. Res., 153 (2007), pp. 235–256.
[13] S. Dempe, *Bilevel optimization: Theory, algorithms and applications*, preprint, optimization online 2018-11, 2018.
[14] S. Dempe and J. Dutta, *Is bilevel programming a special case of a mathematical program with complementarity constraints?*, Math. Program., 131 (2012), pp. 37–48.
[15] S. Dempe and S. Franke, *Solution algorithm for an optimistic linear Stackelberg problem*, Comput. Oper. Res., 41 (2014), pp. 277–281.
[16] S. Dempe, V. Kalashnikov, G. Pérez-Valdés, and N. Kalashnykova, *Bilevel Programming Problems*, Energy Syst., Springer, Berlin, 2015.
[17] S. Dempe and A. Zemkoho, *Bilevel Optimization: Advances and Next Challenges*, Springer Optim. Appl. 161, Springer, Cham, Switzerland, 2020.
[18] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil, *Bilevel programming for hyperparameter optimization*, in International Conference on Machine Learning, Curran Associates, Red Hook, NY, (2018), pp. 1563–1572.
[19] J. Gauvin and F. Dubeau, *Differential properties of the marginal function in mathematical programming*, in Optimality and Stability in Mathematical Programming, Math. Program. Stud., 19, Springer, Berlin, 1982, pp. 101–119.
[20] L. Guo, G. Lin, J.J. Ye, and J. Zhang, *Sensitivity analysis of the value function for parametric mathematical programs with equilibrium constraints*, SIAM J. Optim., 24 (2014), pp. 1206–1237.
[21] D. Henrion, J. Lasserre, and J. Lofberg, *GloptiPoly* 3: *Moments, optimization and semidefinite programming*, Optim. Methods Softw., 24 (2009), pp. 761–779.
[22] R. Hettich and K. O. Kortanek, *Semi-infinite programming: Theory, methods, and applications*, SIAM Rev., 35 (1993), pp. 380–429.
[23] V. Jeyakumar, J. B. Lasserre, G. Li, and T. S. Pham, *Convergent semidefinite programming relaxations for global bilevel polynomial optimization problems*, SIAM J. Optim., 26 (2016), pp. 753–780.
[24] H. Th. Jongen, P. Jonker, and F. Twilt, *Critical sets in parametric optimization*, Math. Program., 34 (1986), pp. 333–353.
[25] G. Kunapuli, K. Bennett, J. Hu, and J-S. Pang, *Classification model selection via bilevel programming*, Optim. Methods Softw., 23 (2008), pp. 475–489.
[26] L. Lampariello and S. Sagratella, *A bridge between bilevel programs and Nash games*, J. Optim. Theory Appl., 174 (2017), pp. 613–635.
[27] J. B. Lasserre, *Global optimization with polynomials and the problem of moments*, SIAM J. Optim., 11 (2001), pp. 796–817.
[28] J. Lasserre, *Moments, Positive Polynomials and Their Applications*, Optim. Ser. 1, Imperial College Press, London, 2009.
[29] J. B. Lasserre, K. Toh, and S. Yang, *A bounded degree SOS hierarchy for polynomial optimization*, EURO J. Comput. Optim., 5 (2017), pp. 87–117.
[30] M. Laurent, *Optimization over polynomials: Selected topics*, Proceedings of the International Congress of Mathematicians, Kyung Moon, Seoul, 2014.
[31] G. Lin, M. Xu, and J. Ye, *On solving simple bilevel programs with a nonconvex lower level program*, Math. Program., 144 (2014), pp. 277–305.

[32] R. Liu, P. Mu, X. Yuan, S. Zeng, and J. Zhang, *A generic first-order algorithmic framework for bi-level programming beyond lower-level singleton*, International Conference on Machine Learning, 2020, Proceedings of the 37th International Conference on Machine Learning, PMLR, 119 (2020), pp. 6305–6315.

[33] J. Lofberg, *YALMIP: A toolbox for modeling and optimization in MATLAB*, in 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No. 04CH37508), IEEE, Piscataway, NJ, 2004.

[34] Z. Luo, J. Pang, and D. Ralph, *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge, 1996.

[35] A. Megretski, *SPOT (Systems polynomial optimization tools) Manual*, http://web.mit.edu/ameg/www/images/spot_manual.pdf (2010).

[36] J. Mirrlees, *The theory of moral hazard and unobservable behaviour: Part* I, Rev. Eco. Stud., 66 (1999), pp. 3–22.

[37] A. Mitsos, P. Lemonidis, and P.I. Barton, *Global solution of bilevel programs with a nonconvex inner program*, J. Global Optim., 42 (2008), pp. 475–513.

[38] L. Muu and N. Quy, *A global optimization method for solving convex quadratic bilevel programming problems*, J. Global Optim., 26 (2003), pp. 199–219.

[39] J. Nie, *Certifying convergence of Lasserre's hierarchy via flat truncation*, Math. Program., 1422 (2013), pp. 485–510.

[40] J. Nie, *Optimality conditions and finite convergence of Lasserre's hierarchy*, Math. Program., 146 (2014), pp. 97–121.

[41] J. Nie, *Tight relaxations for polynomial optimization and Lagrange multiplier expressions*, Math. Program., 178 (2019), pp. 1–37.

[42] J. Nie, L. Wang, and J.J. Ye, *Bilevel polynomial programs and semidefinite relaxation methods*, SIAM J. Optim., 27 (2017), pp. 1728–1757.

[43] J. Outrata, *On the numerical solution of a class of Stackelberg problems*, Z. Oper. Resear., 34 (1990), pp. 255–277.

[44] J. V. Outrata, *On optimization problems with variational inequality constraints*, SIAM J. Optim., 4 (1994), pp. 340–357.

[45] J. Outrata, M. Kocvara, and J. Zowe, *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical Results*, Kluwer Academic, Boston, 1998.

[46] M. Putinar, *Positive polynomials on compact semi-algebraic sets*, Indiana Univ. Math. J., 42 (1993), pp. 969–984.

[47] K. Shimizu and E. Aiyoshi, *A new computational method for Stackelberg and min-max problems by use of a penalty method*, IEEE Trans. Automat. Control, 26 (1981), pp. 460–466.

[48] K. Shimizu, Y. Ishizuka, and J. Bard, *Nondifferentiable and Two-level Mathematical Programming*, Kluwer Academic, Boston, 1997.

[49] J. Sturm, *Using SeDuMi* 1.02*, a MATLAB toolbox for optimization over symmetric cones*, Optim. Methods Softw., 11 (1999), pp. 625–653.

[50] M. Xu and J.J. Ye, *A smoothing augmented Lagrangian method for solving simple bilevel programs*, Comput. Optim. Appl., 59 (2014), pp. 353–377.

[51] M. Xu and J.J. Ye, *Relaxed constant positive linear dependence constraint qualification and its application to bilevel programs*, J. Global Optim., 78 (2020), pp. 181–205.

[52] M. Xu, J.J. Ye, and L. Zhang, *Smoothing augmented Lagrangian method for solving nonsmooth and nonconvex constrained optimization problems*, J. Global Optim., 62 (2014), pp. 675–694.

[53] M. Xu, J.J. Ye, and L. Zhang, *Smoothing SQP methods for solving degenerate nonsmooth constrained optimization problems with applications to bilevel programs*, SIAM J. Optim., 25 (2015), pp. 1388–1410.

[54] J.J. Ye, *Constraint qualifications and optimality conditions in bilevel optimization*, in Bilevel Optimization: Advances and Next Challenges, Springer Optim. Appl. 161, Springer, Cham, Switzerland, 2020.

[55] J.J. Ye and D. Zhu, *Optimality conditions for bilevel programming problems*, Optimization, 33 (1995), pp. 9–27.

[56] J.J. Ye and D. Zhu, *New necessary optimality conditions for bilevel programs by combining the MPEC and value function approaches*, SIAM J. Optim., 20 (2010), pp. 1885–1905.

[57] J.J. Ye, D.L. Zhu, and Q.J. Zhu, *Exact penalization and necessary optimality conditions for generalized bilevel programming problems*, SIAM J. Optim., 7 (1997), pp. 481–507.