A polynomial-time solvable class of sparse box-constrained polynomial optimization problems *

Aida Khajavirad †

April 27, 2026

Abstract

We study the problem of minimizing a multivariate polynomial function over the unit hypercube. By representing the polynomial through a hypergraph and exploiting its sparsity structure, we establish a new sufficient condition under which the problem can be solved in time polynomial in the encoding length of the input. Our approach identifies a subset of variables that attain binary values at optimality and shows how the remaining continuous variables can be eliminated locally when they appear in small, weakly coupled blocks, yielding a reduction to a structured binary optimization problem that can be solved efficiently. Our result extends the classical tractability result for binary polynomial optimization, namely, that problems with bounded treewidth are solvable in polynomial time, to box-constrained polynomial optimization.

Keywords: box-constrained polynomial optimization, binary polynomial optimization, sparsity, hypergraph, treewidth, polynomial-time algorithm.

1 Introduction

We consider the problem of minimizing a multivariate polynomial function of degree $d \geq 2$ over the unit hypercube. Let $x \in \mathbb{R}^n$ and $\alpha \in \mathbb{Z}^n_+$. Define $x^\alpha := x_1^{\alpha_1} \dots x_n^{\alpha_n}$ and $|\alpha| := \sum_{i=1}^n \alpha_i$. We define a *box-constrained polynomial optimization problem* of degree d as follows:

$$\begin{aligned} \min \ & \sum_{|\alpha| \leq d} c_\alpha x^\alpha \\ \text{s.t.} \ & x \in [0,1]^n, \end{aligned} \tag{1}$$

where $c_\alpha \in \mathbb{Q}$ for all $\alpha \in \mathbb{Z}^n_+$ with $|\alpha| \leq d$, and we assume that $c_\alpha \neq 0$ for at least one α with $|\alpha| = d$. Throughout the paper, we assume that the input polynomial is given explicitly in the standard monomial basis, that is, as a list of pairs (α, c_α) with $c_\alpha \neq 0$. The input length $\mathcal{L}$ denotes the total number of bits required to encode all coefficients and exponents. In particular, the degree d is not assumed to be fixed and is part of the input. Problem (1) is $\mathcal{NP}$-hard even for $d = 2$, as it contains the maximum cut problem as a special case. If $d = 2$ and the problem is convex, i.e., if the Hessian of the quadratic function is positive semidefinite, then thanks to the ellipsoid method, Problem (1) can be solved in time polynomial in $\mathcal{L}$ [17]. If $d \geq 3$ and the polynomial function is convex over the unit hypercube, then an ϵ-optimal solution can be computed in time polynomial in $\mathcal{L}$ and $\log(1/\epsilon)$ using the ellipsoid method [12] or interior point methods [14]. However, if Problem (1) is nonconvex, then its tractability has only been established in fixed dimension. Namely, if n is fixed, then Problem (1) is solvable in time polynomial in $\mathcal{L}$ via quantifier-elimination techniques [15].

*The author was supported in part by AFOSR grant FA9550-23-1-0123 and ONR grant N00014-25-1-2491.

†Department of Industrial and Systems Engineering, Lehigh University. E-mail: aida@lehigh.edu.

Binary polynomial optimization. In contrast to the continuous setting, the complexity of binary polynomial optimization, i.e., the problem of minimizing a multivariate polynomial function over the set of binary points, has been extensively studied in the literature [6, 3, 10, 7, 9, 4]. To formally define this problem, we use the hypergraph representation scheme of [8]. A hypergraph G is a pair (V, E), where V is a finite set of nodes and E is a set of subsets of V of cardinality at least two, called the edges of G. With any hypergraph $G = (V, E)$ and the cost vector $c \in \mathbb{R}^{V \cup E}$, we associate the following *binary polynomial optimization problem*:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \prod_{i \in e} z_i + \sum_{i \in V} c_i z_i \\ \text{s.t.} \quad & z \in \{0,1\}^V, \end{aligned} \tag{2}$$

where, without loss of generality, we assume that each node is contained in at least one edge, and $c_e \neq 0$ for all $e \in E$. Problem (2) is $\mathcal{NP}$-hard even for $d = 2$. We next review the existing sufficient conditions under which Problem (2) is solvable in polynomial-time. If the objective function of Problem (2) is submodular, i.e., if $c_e \leq 0$ for all $e \in E$, then this problem can be solved in strongly polynomial time [16]. By strongly polynomial time, we mean time that is polynomial in $|V|, |E|$. Given a hypergraph $G = (V, E)$, the *intersection graph of* G is the graph with node set V, and where two nodes $i, j \in V$ are adjacent if $i, j \in e$ for some $e \in E$. Henceforth, we refer to the treewidth of the intersection graph of a hypergraph G as the *primal treewidth* of G and denote it by $\text{ptw}(G)$. In [6], the authors prove that if $\text{ptw}(G)$ is bounded, then Problem (2) can be solved in strongly polynomial time. Given a hypergraph $G = (V, E)$, the *incidence graph of* G is a bipartite graph whose vertex set is $V \cup E$ and the edge set is $\{\{i, e\} : i \in V,\ e \in E,\ i \in e\}$. Henceforth, we refer to the treewidth of the incidence graph of a hypergraph G as the *incidence treewidth* of G and denote it by $\text{itw}(G)$. For any hypergraph G, we have $\text{itw}(G) \leq \text{ptw}(G)$. In [4], the authors prove that if $\text{itw}(G)$ is bounded, then Problem (2) can be solved in strongly polynomial time. We should remark that while the problem of computing the treewidth of a graph is $\mathcal{NP}$-hard in general, it is fixed-parameter tractable when parameterized by the treewidth [2]. Finally, in [7], the authors prove that if the hypergraph G is β-acyclic, then Problem (2) can be solved in strongly polynomial time. Note that the incidence treewidth of a β-acyclic hypergraph may be unbounded, in general.

Let us consider the important special case of Problem (2) with $d = 2$. With any graph $G = (V, E)$ and cost vector $c \in \mathbb{R}^{V \cup E}$, we associate the *binary quadratic optimization problem*:

$$\begin{aligned} \min \quad & \sum_{\{i,j\} \in E} c_{ij} z_i z_j + \sum_{i \in V} c_i z_i \\ \text{s.t.} \quad & z \in \{0,1\}^V. \end{aligned} \tag{3}$$

First, note that for a graph G we have $\text{ptw}(G) = \text{itw}(G)$. Second, for a graph, the notion of β-acyclicity in hypergraphs coincides with the usual notion of graph acyclicity. Therefore, for Problem (3) the most general sufficient condition for polynomial-time solvability is as follows: if the graph G has bounded treewidth, then Problem (3) can be solved in strongly polynomial time. In fact, the results of [5, 9] imply that bounded treewidth is essentially a necessary and sufficient condition for polynomial-time solvability of Problem (3).

Our contributions. In this paper, by building on existing algorithms for binary polynomial optimization and box-constrained polynomial optimization, we obtain new sufficient conditions under which Problem (1) can be solved in time polynomial in the length of the input $\mathcal{L}$.

First, we focus on the important special case of Problem (1) with $d = 2$; namely, the *box-constrained quadratic optimization problem*:

$$\begin{aligned} \min \quad & x^\top Q x + c^\top x \\ \text{s.t.} \quad & x \in [0,1]^n, \end{aligned} \tag{4}$$

where $c \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix that is not positive semidefinite. We partition the variables according to the sign of the diagonal entries of Q: variables x_i with $q_{ii} \leq 0$ can be restricted to binary values in an optimal solution. Henceforth, we will refer to these variables as *hidden binary variables*, while the remaining variables form the continuous part. We prove that if the "interaction graph" of Problem 4 has treewidth bounded by $O(\log n)$, and if each connected component of continuous variables together with hidden binary variables that directly interact with it has size bounded by $O(\log n)$, then Problem (4) can be solved in polynomial time. The key idea is to eliminate each continuous block independently: since each block involves only logarithmically many variables, even nonconvex quadratic optimization over that block can be solved efficiently. This produces a reduced binary polynomial optimization problem. A crucial step is to show that, after the elimination of continuous blocks, the resulting binary polynomial optimization problem still has treewidth bounded by $O(\log n)$, which allows it to be solved efficiently by dynamic programming (see Theorem 1).

Next, we extend this approach to higher-degree box-constrained polynomial optimization problems. We again separate variables into a hidden binary part and a continuous part and decompose the continuous variables into connected components of the "interaction" hypergraph. When each continuous component has constant size and directly interacts with only a logarithmic number of binary variables, and when the incidence treewidth of the interaction hypergraph is logarithmically bounded, we can eliminate each continuous component locally using polynomial-time algorithms for fixed-dimension polynomial optimization using tools from real algebraic geometry. As in the quadratic case, this yields a reduced binary polynomial optimization problem. We then prove that the structural assumptions ensure that the incidence treewidth of this reduced problem remains logarithmically bounded, and hence it can be solved in polynomial time (see Theorem 2).

Organization. The remainder of the paper is structured as follows. In Section 2 we obtain a sufficient condition under which box-constrained quadratic optimization problems can be solved in polynomial time. In Section 3 we obtain a sufficient condition under which higher-degree box-constrained polynomial optimization problems can be solved in polynomial time.

2 Box-constrained quadratic optimization

In this section, we consider the box-constrained quadratic optimization problem defined in (4). To exploit the sparsity of the objective function, we define an *interaction graph* $G = (V, E)$ for this problem as follows. For each variable x_i, $i \in [n] := \{1, \cdots, n\}$, we define a node i. Two distinct nodes i and j are adjacent if the coefficient q_{ij} is nonzero. Henceforth, given a graph G, we denote by $\mathrm{tw}(G)$ the treewidth of G. We are now ready to state our sufficient condition for polynomial-time solvability of Problem (4):

Theorem 1. *Let $G = (V, E)$ be the interaction graph of Problem (4). Define*

$$V^+ := \{i \in V : q_{ii} > 0\}. \tag{5}$$

Denote by G_{V^+} the subgraph of G induced by V^+, and denote by $\mathcal{C}$ the set of connected components of G_{V^+}. For each connected component $C \in \mathcal{C}$, define its neighborhood as:

$$N(C) := \{i \in V \setminus C : \{i, j\} \in E, \text{ for some } j \in C\}. \tag{6}$$

Suppose that the following conditions are satisfied:

(i) $\mathrm{tw}(G) \in O(\log |V|)$
(ii) $|C \cup N(C)| \in O(\log |V|)$ *for all* $C \in \mathcal{C}$

Then Problem (4) *can be solved in time polynomial in the input length $\mathcal{L}$.*

We should remark that the assumptions of Theorem 1 can be verified in time polynomial in the input length $\mathcal{L}$. The interaction graph $G = (V, E)$ can be constructed in polynomial time by inspecting the nonzero entries of the matrix Q. The set V^+ can be identified by checking the signs of the diagonal entries q_{ii}, which is immediate. The connected components of the induced subgraph G_{V^+} and the corresponding neighborhoods $N(C)$ can be computed using standard graph traversal algorithms in time $O(|V| + |E|)$. Once these sets are obtained, verifying that $\mathrm{tw}(G) \in O(\log |V|)$ and $|C \cup N(C)| \in O(\log |V|)$ for all $C \in \mathcal{C}$ can be done in polynomial time; in particular, although computing treewidth exactly is $\mathcal{NP}$-hard, it is fixed-parameter tractable, and one can check whether $\mathrm{tw}(G) \leq k$ for $k = O(\log |V|)$ in polynomial time for fixed k [2].

To prove Theorem 1, we first present a number of intermediate results that will be used in the proof. The first proposition is a celebrated result in discrete optimization that has been independently discovered in different communities. Recall that given a hypergraph $G = (V, E)$, the primal treewidth of G, denoted by $\mathrm{ptw}(G)$ is the treewidth of its intersection graph. In the following, by $\mathrm{poly}(|V|, |E|)$, we denote a polynomial function in $|V|, |E|$.

Proposition 1 ([6])**.** *Let $G = (V, E)$ be a hypergraph with the primal treewidth $\mathrm{ptw}(G) = \kappa$. Then Problem* (2) *can be solved by dynamic programming in $O(2^\kappa |V|)$ operations. In particular, if $\kappa \in O(\log \mathrm{poly}(|V|, |E|))$, then Problem* (2) *is solvable in strongly polynomial time.*

The next result is concerned with Problem (4) and essentially states that variables x_i with $q_{ii} \leq 0$ can be treated as binary variables. Henceforth, we refer to such variables as *hidden binary variables*.

Lemma 1. *Let $G = (V, E)$ be the interaction graph of Problem* (4)*. Then there exists an optimal solution x^* of Problem* (4) *such that $x_i^* \in \{0, 1\}$ for all $i \in V \setminus V^+$, where V^+ is defined by* (5)*.*

Proof. Consider some $i \in V \setminus V^+$ and fix all other variables at some arbitrary values in the unit box. The objective function of Problem (4) then becomes

$$\phi(t) = q_{ii} t^2 + \alpha t + \beta, \quad t \in [0, 1],$$

with $q_{ii} \leq 0$. Thus, ϕ is a concave function and attains its minimum at $t \in \{0, 1\}$. Repeating the same argument for all $i \in V \setminus V^+$ yields the result. □

The next lemma implies that nonconvex box-constrained quadratic optimization in fixed dimension is solvable in polynomial time.

Lemma 2. *Problem* (4) *can be solved in time $O(3^n \cdot \mathrm{poly}(\mathcal{L}))$.*

Proof. Denote by f the objective function of Problem (4). Let I_0, I_1 denote disjoint subsets of $[n]$. Define a face of the unit box:

$$F(I_0, I_1) := \big\{x \in [0,1]^n : x_i = 0 \ \forall i \in I_0,\ x_i = 1 \ \forall i \in I_1\big\},$$

and set $J := [n] \setminus (I_0 \cup I_1)$. Every point of $[0,1]^n$ lies in the relative interior of a unique such face, and the total number of faces of $[0,1]^n$ is 3^n. Fix I_0, I_1 and write $y = (x_j)_{j\in J}$. Substituting the fixed coordinates x_i, $i \in I_0 \cup I_1$ into the objective function f yields

$$f_{I_0,I_1}(y) = y^\top Q' y + r^\top y + s,$$

where $y \in [0,1]^J$. Note that Q', r, s are computable in time poly$(\mathcal{L})$. Let y^* be a minimizer of f_{I_0,I_1} over $[0,1]^J$. If $y^* \in (0,1)^J$, then

$$\nabla f_{I_0,I_1}(y^*) = 2Q'y^* + r = 0.$$

Thus interior minimizers are precisely the solutions of a linear system, which can be found (or shown not to exist) in time poly$(\mathcal{L})$. If such a solution exists in $(0,1)^J$, all such solutions give the same value of f_{I_0,I_1}, since their differences lie in the kernel of Q' and f_{I_0,I_1} is constant along these directions. If no solution lies in $(0,1)^J$, then the minimum of f_{I_0,I_1} is attained on the boundary of $[0,1]^J$, i.e., , on lower-dimensional faces, which are included in the same family.

Therefore, the minimum of f over $[0,1]^n$ is attained either at an interior stationary point of some face or on a lower-dimensional face. Evaluating f at all such candidate points over all 3^n faces yields the minimum value. Each face requires solving a linear system and evaluating f, both in time poly$(\mathcal{L})$. Hence, the total running time is at most $3^n \cdot$ poly$(\mathcal{L})$. □

To prove Theorem 1, we need to show that the treewidth of a graph obtained from the interaction graph G by adding certain cliques of small size to G is remains bounded. The next two lemmas establish this fact. In the following, given a graph $G = (V, E)$ and a node $v \in V$, we say that the graph $G - v := (V', E')$ is obtained from G by *removing* v if $V' = V \setminus \{v\}$ and $E' = \{\{i, j\} \in E : i \neq v, j \neq v\}$.

Lemma 3 ([13])**.** *Let $G = (V, E)$ be a graph. Let $C \subseteq V$ be such that the subgraph of G induced by C is a connected graph. Denote by G' a graph obtained from G by removing all nodes in C and then making $N(C)$ a clique, i.e., adding all missing edges between pairs of nodes in $N(C)$, where $N(C)$ is defined by* (6)*. Then the treewidth of G' is upper bounded by*

$$\mathrm{tw}(G') \leq \max(\mathrm{tw}(G),\ |N(C)| - 1). \tag{7}$$

The next result is obtained by a repeated application of the above lemma.

Lemma 4 ([13])**.** *Consider a graph $G = (V, E)$ and let $S \subseteq V$ be nonempty. Denote by G_S the subgraph of G induced by S. Denote by $C_1, \cdots, C_p$ the connected components of G_S, for some $p \geq 1$. Define $G_0 := G$. For each $i \in [p]$, let G_i be the graph obtained from G_{i-1} by removing the nodes in C_i and adding edges between the pairs of nodes in $G_{i-1} - C_i$ that are both adjacent to some node in C_i in G_{i-1}. Define*

$$N(C_i) := \{\, u \in V \setminus C_i : \exists v \in C_i \text{ with } \{u, v\} \in E\}, \quad \forall i \in [p],$$

and

$$d_{\max} := \max_{i\in[p]} |N(C_i)|.$$

We then have

$$\mathrm{tw}(G_p) \leq \max\big(\mathrm{tw}(G),\ d_{\max} - 1\big),$$

where G_p is the graph obtained from G after p component removal and clique addition operations.

We are now ready to prove Theorem 1.

Proof of Theorem 1. Define $V^- := V \setminus V^+$, where V^+ is defined by (5). Let x_{V^-} (resp. x_{V^+}) contain the components x_i of x with $i \in V^-$ (resp. $i \in V^+$). Denote by $f(x)$ the objective function of Problem (4). Then by Lemma 1, Problem (4) can be equivalently written as:

$$\begin{array}{ll} \min & f(x_{V^-}, x_{V^+}) \\ \text{s.t.} & x_{V^-} \in \{0,1\}^{V^-},\ x_{V^+} \in [0,1]^{V^+}. \end{array}$$

Let G_{V^+} be the subgraph of G induced by V^+, and let $\mathcal{C}$ denote the set of connected components of G_{V^+}. Define

$$f_{V^-}(x_{V^-}) := \sum_{i \in V^-} q_{ii} x_i^2 + \sum_{\substack{\{i,j\} \in E: \\ i,j \in V^-}} q_{ij} x_i x_j + \sum_{i \in V^-} c_i x_i,$$

and, for each $C \in \mathcal{C}$, define

$$f_C(x_C, x_{N(C)}) := \sum_{i \in C} q_{ii} x_i^2 + \sum_{\substack{\{i,j\} \in E: \\ i,j \in C}} q_{ij} x_i x_j + \sum_{\substack{\{i,j\} \in E \\ i \in C,\ j \in N(C)}} q_{ij} x_i x_j + \sum_{i \in C} c_i x_i,$$

where $N(C)$ is defined by (6). Since the sets $C \in \mathcal{C}$ are the connected components of G_{V^+} and $N(C) \subseteq V^-$ for all $C \in \mathcal{C}$, we deduce that:

$$f(x_{V^-}, x_{V^+}) = f_{V^-}(x_{V^-}) + \sum_{C \in \mathcal{C}} f_C(x_C, x_{N(C)}).$$

Since $C \cap C' = \emptyset$ for any $C \neq C' \in \mathcal{C}$, for any fixed x_{V^-} we have:

$$\min_{x_{V^+} \in [0,1]^{V^+}} f(x_{V^-}, x_{V^+}) = f_{V^-}(x_{V^-}) + \sum_{C \in \mathcal{C}} \min_{x_C \in [0,1]^C} f_C(x_C, x_{N(C)}). \tag{8}$$

Now for each $C \in \mathcal{C}$, and for each $z \in \{0,1\}^{N(C)}$, define

$$\psi_C(z) := \min_{x_C \in [0,1]^C} f_C(x_C, z).$$

From (8) we deduce that:

$$\min_{x_{V^-}, x_{V^+}} f(x_{V^-}, x_{V^+}) = \min_{x_{V^-} \in \{0,1\}^{V^-}} \left(f_{V^-}(x_{V^-}) + \sum_{C \in \mathcal{C}} \psi_C(x_{N(C)}) \right).$$

Thus, once we compute $\psi_C(z)$ for all $z \in \{0,1\}^{N(C)}$ and for all $C \in \mathcal{C}$, the original continuous problem is reduced to a binary optimization problem. Let us now consider the complexity of computing the function values $\psi_C(z)$, $z \in \{0,1\}^{N(C)}$ for all $C \in \mathcal{C}$. For each fixed $z \in \{0,1\}^{N(C)}$, computing $\psi_C(z)$ amounts to minimizing a quadratic function over $[0,1]^C$. From assumption (ii) in the statement of the theorem it follows that $|C| \in O(\log |V|)$. Therefore, by Lemma 2, for a fixed z, the function $\psi_C(z)$ can be computed in $O(|V|^{O(1)}\text{poly}(\mathcal{L}))$ time. Moreover, again by assumption (ii), we have $|N(C)| \in O(\log |V|)$. Therefore, $\psi_C(z)$ for all $z \in \{0,1\}^{N(C)}$ can be computed in $O(|V|^{O(1)}\text{poly}(\mathcal{L}))$ time. Finally, we have $|\mathcal{C}| \leq |V|$. Therefore, all such functional computations can be done in time polynomial in the input length $\mathcal{L}$.

Finally, let us consider the cost of solving the binary optimization problem:

$$\begin{aligned} \min \quad & f_{V^-}(x_{V^-}) + \sum_{C \in \mathcal{C}} \psi_C(x_{N(C)}) \\ \text{s.t.} \quad & x_{V^-} \in \{0,1\}^{V^-}. \end{aligned} \tag{9}$$

Since each term $\psi_C(x_{N(C)})$ depends only on binary variables x_i, $i \in N(C)$, it can be written as a polynomial function in binary variables x_i, $i \in N(C)$. Moreover, since by assumption (ii), $|N(C)| \in O(\log |V|)$, such a formulation can be constructed in polynomial time. Therefore, without loss of generality, we can assume that Problem (9) is a binary polynomial optimization problem. Denote by G_{V^-} the subgraph of G induced by V^-. Denote by $\bar{G}$ the graph obtained from G_{V^-} by making each $N(C)$ a clique for all $C \in \mathcal{C}$. It then follows that $\bar{G}$ is the intersection graph of the hypergraph of the objective function of Problem (9). By Lemma 4, and assumptions (i)-(ii), we deduce that

$$\mathrm{tw}(\bar{G}) \leq \max\big(\mathrm{tw}(G),\ \max_{C \in \mathcal{C}} |N(C)| - 1\big) \in O(\log |V|).$$

Therefore, by Proposition 1, Problem (9) can be solved in time polynomial in $|V| 2^{O(\log |V|)} \mathrm{poly}(\mathcal{L}) = |V|^{O(1)} \cdot \mathrm{poly}(\mathcal{L})$, and this completes the proof. □

Remark 1. Assumptions (i)-(ii) of Theorem 1 do not imply any nontrivial lower bound on the number of hidden binary variables. In fact, it is possible to have $|V^+| = \Theta(|V|)$. To see this, for each integer $m \geq 1$, define a graph $G = (V, E)$ as follows. Let $V := \{p_1, \ldots, p_m\} \cup \{z_1, \ldots, z_{m-1}\}$, and $E := \big\{\{p_i, z_i\}, \{z_i, p_{i+1}\} : i = 1, \ldots, m-1\big\}$. Let $V^+ = \{p_1, \ldots, p_m\}$. Then G is a path, hence $\mathrm{tw}(G) = 1$ and assumption (i) is satisfied. Since there are no edges between distinct nodes p_i, the connected components of G_{V^+} are the singletons $C_i := \{p_i\}$ for $i = 1, \ldots, m$. We have $N(C_1) = \{z_1\}$, $N(C_m) = \{z_{m-1}\}$, and $N(C_i) = \{z_{i-1}, z_i\}$ for all $2 \leq i \leq m-1$. Hence, assumption (ii) is satisfied. Finally, we have $|V| = 2m - 1$ and $|V^+| = m$, implying that $|V^+| = \Theta(|V|)$.

Remark 2. In [11, 13], the authors propose new convex relaxations for Problem (4). To this end, they associate a graph $G = (V, E, L)$ with this problem, where V and E are node set and edges set, respectively, of G as defined for our interaction graph, and L denotes the loop set, where $\{i, i\} \in L$, if $q_{ii} \neq 0$. They further define $L = L^- \cup L^+$, where L^- (resp. L^+) contain all loops with $q_{ii} < 0$ (resp. $q_{ii} > 0$). Subsequently, they study the facial structure of the set:

$$\begin{aligned} \mathrm{QP}(G) := \mathrm{conv}\Big\{ z \in \mathbb{R}^{V \cup E \cup L} : z_{ii} \geq z_i^2,\ \forall \{i,i\} \in L^+,\ z_{ii} \leq z_i^2,\ \forall \{i,i\} \in L^-,\ z_{ij} = z_i z_j, \\ \forall \{i,j\} \in E,\ z_i \in [0,1], \forall i \in V \Big\}, \end{aligned}$$

where $\mathrm{conv}(\cdot)$ denotes the convex hull. They obtain sufficient conditions under which $\mathrm{QP}(G)$ admits a polynomial-size second-order cone (SOC) or semidefinite programming (SDP) representable formulation that can be constructed in polynomial time. We next summarize these sufficient conditions. Suppose that $\mathrm{tw}(G) \in O(\log |V|)$. We have the following cases:

1. if $|C| = 1$ and $|N(C)| \in O(\log |V|)$ for all $C \in \mathcal{C}$, then $\mathrm{QP}(G)$ admits a polynomial-size SOC-representable formulation that can be constructed in polynomial time (see theorem 3 in [11] and corollary 2 in [13]).
2. if $|C| = 2$ and $|N(C)| \in O(\log |V|)$ for all $C \in \mathcal{C}$, then $\mathrm{QP}(G)$ admits a polynomial-size SDP-representable formulation that can be constructed in polynomial time (see theorem 6 in [13]).

If the above conditions are satisfied, then the optimal value of Problem (4) is equal to that of a polynomial-size convex optimization problem. First notice that conditions (1) and (2) are special

cases of Theorem 1 because assumption (ii) can be equivalently stated as $|C| \in O(\log |V|)$ and $|N(C)| \in O(\log |V|)$ for all $C \in \mathcal{C}$. Second, the tightness of a polynomial-size SOCP or SDP relaxation of Problem (4) does not imply its polynomial-time solvability because when the optimal solution of Problem (4) is not unique, the solution returned by the SOCP or SDP solver may not be feasible for the nonconvex problem (4). It is also important to note that Theorem 1 does not imply the results of [11, 13] either because polynomial-time solvability of Problem (4) does not imply that QP(G) admits a polynomial-size extended formulation that can be constructed in polynomial time.

3 Higher-degree polynomial optimization

In this section, we consider a box-constrained polynomial optimization problem of degree $d \geq 3$. We obtain a sufficient condition under which this problem can be solved in polynomial time. As in the previous section, we first identify a subset of variables that can be treated as binary variables because there is an optimal solution of Problem (1) at which these variables take binary values. As before, we refer to such variables as hidden binary variables. For a vector $x \in \mathbb{R}^n$ and an index $i \in [n]$, we denote by $x_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \in \mathbb{R}^{n-1}$ the vector obtained from x by removing its i-th coordinate. Accordingly, for $t \in \mathbb{R}$ we write $(x_{-i}, t) := (x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n) \in \mathbb{R}^n$. The *standard monomial basis* of the space of polynomials in n variables of degree at most d is the set $\{x^\alpha : \alpha \in \mathbb{Z}^n_+, |\alpha| \leq d\}$; a polynomial is said to be written in this basis if it is expressed as a linear combination of these monomials. The following lemma provides easily verifiable sufficient conditions for identifying hidden binary variables:

Lemma 5. *Let $f(x)$, $x \in \mathbb{R}^n$ be a polynomial and fix some $i \in [n]$. Then we have the following:*

1. *Write $f(x)$ as a polynomial in x_i:*

$$f(x) = \sum_{m=0}^{d} a_m(x_{-i})\, x_i^m,$$

where each $a_m(x_{-i})$ is a polynomial in the remaining variables x_{-i}. Assume that for every $m \geq 2$, the polynomial $a_m(x_{-i})$ has only nonpositive coefficients when written in the standard monomial basis. Then there exists a minimizer x^* of $f(x)$ over $[0,1]^n$ such that $x_i^* \in \{0,1\}$.

2. *Define*

$$\Delta_i(x) := f(x) - (1 - x_i) f(x_{-i}, 0) - x_i f(x_{-i}, 1),$$

and write $\Delta_i(x) = x_i(1 - x_i) H_i(x)$. Assume that H_i has only nonnegative coefficients when written in the standard monomial basis. Then there exists a global minimizer x^* of $f(x)$ over $[0,1]^n$ such that $x_i^* \in \{0,1\}$.

Proof. First consider part (1). Fix $x_{-i} \in [0,1]^{n-1}$ and consider

$$\phi(t) = f(x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n) = \sum_{m=0}^{d} a_m(x_{-i})\, t^m.$$

Since each monomial in the variables x_{-i} is nonnegative on $[0,1]^{n-1}$, and since each coefficient of $a_m(x_{-i})$ is nonpositive for every $m \geq 2$, it follows that $\phi(t)$ is concave on $[0,1]$. Therefore, there exists a minimizer of $f(x)$, $x \in [0,1]^n$ at $x_i^* \in \{0,1\}$

Next, consider part (2). Since every monomial is nonnegative on $[0,1]^n$ and H_i has only nonnegative coefficients in the standard monomial basis, we have $H_i(x) \geq 0$ for all $x \in [0,1]^n$. Therefore $\Delta_i(x) \geq 0$ for all $x \in [0,1]^n$. Thus, the slice of $f(x)$ in x_i lies above its chord for every fixing of the other variables, implying that there exists a minimizer x^* of $f(x)$ with $x_i^* \in \{0,1\}$. □

Remark 3. In Lemma 5, while condition 1 is simpler to verify, condition 2 is strictly more general. To see this, consider $f(x_1, x_2) = x_1(1 - x_1)(x_1 + x_2)$. Expanding in powers of x_1, we obtain $f(x_1, x_2) = -x_1^3 + (1 - x_2)x_1^2 + x_1x_2$. The coefficient of x_1^2 is the polynomial $1 - x_2$, which has a positive coefficient in the standard monomial basis. Therefore, condition 1 of Lemma 5 does not apply. On the other hand, $f(0, x_2) = f(1, x_2) = 0$, implying that $\Delta_1(x_1, x_2) = f(x_1, x_2) = x_1(1 - x_1)H_1(x_1, x_2)$, with $H_1(x_1, x_2) = x_1 + x_2$. Since H_1 has only nonnegative coefficients in the standard monomial basis, condition 2 of Lemma 5 applies. Hence there exists a minimizer of f with $x_1 \in \{0, 1\}$.

Henceforth, we refer to as variables that satisfy the conditions of Lemma 5 as hidden binary variables. We should remark that to identify a larger set of hidden binary variables, one can obtain more general sufficient conditions that those of Lemma 5. However, to the best of our knowledge, such conditions cannot be verified in polynomial time.

Consider the polynomial $f(x) = \sum_{|\alpha| \leq d} c_\alpha x^\alpha$, $x \in \mathbb{R}^n$. We associate with f the *interaction hypergraph* $G = (V, E)$ with $V = [n]$ and for each monomial $c_\alpha x^\alpha$ with $c_\alpha \neq 0$ and $\text{card}(\alpha) \geq 2$, we include the edge $\text{supp}(\alpha) := \{i \in V : \alpha_i \neq 0\}$ in E. Let $G = (V, E)$ be a hypergraph and let $C \subseteq V$. We define the *subhypergrah* of G induced by C as $G_C = (C, E_C)$ where $E_C := \{e \cap C : e \in E,\ |e \cap C| \geq 2\}$. A hypergraph $G = (V, E)$ is *connected* if for every $u, v \in V$ there exists a sequence of edges $e_1, \ldots, e_k \in E$ such that $u \in e_1$, $v \in e_k$, $e_j \cap e_{j+1} \neq \emptyset$ for all $j \in [k - 1]$. The *connected components* of a hypergraph are the maximal subsets of nodes that induce connected subhypergraphs. Recall that the incidence graph $I(G)$ of a hypergraph G is the bipartite graph with node set $V \cup E$, in which $v \in V$ is adjacent to $e \in E$ if and only if $v \in e$. We then define the *incidence treewidth* of G, denoted by $\text{itw}(G)$ as the treewidth of $I(G)$.

We are now ready to state the main result of this section.

Theorem 2. *Consider Problem* (1) *with the interaction hypergraph denoted by* $G = (V, E)$. *Let* V^- *be the set of hidden binary variables as defined by Lemma 5, and let* $V^+ = V \setminus V^-$. *Denote by* G_{V^+} *the subhypergraph of* G *induced by* V^+ *and denote by* $\mathcal{C}$ *the set of connected components of* G_{V^+}. *For each component* $C \in \mathcal{C}$, *define its neighborhood:*

$$N(C) := \{u \in V \setminus C : \exists\, e \in E \text{ such that } e \cap C \neq \emptyset,\ u \in e\}.$$

Suppose that the following conditions are satisfied:

(i) $\text{itw}(G) \in O(\log |V|)$;
(ii) $|C| \in O(1)$ *for all* $C \in \mathcal{C}$;
(iii) $|N(C)| \in O(\log |V|)$ *for all* $C \in \mathcal{C}$.

Then Problem (1) *can be solved in time polynomial in the length of the input* $\mathcal{L}$.

We note that the assumptions of Theorem 2 can be verified in time polynomial in the input length $\mathcal{L}$. Indeed, the interaction hypergraph $G = (V, E)$ can be constructed in polynomial time from the monomial representation of the polynomial. The set of hidden binary variables V^- can be identified by checking the sufficient conditions of Lemma 5 for each variable, which requires examining the coefficients of the polynomial in the standard monomial basis and can be done in time polynomial in $\mathcal{L}$. The connected components of the induced subhypergraph G_{V^+}, as well as the neighborhoods $N(C)$ for each component C, can be computed by performing a graph traversal on the incidence graph of G in time linear in the size of the incidence representation, namely $O\big(|V| + |E| + \sum_{e \in E} |e|\big)$. Once these sets are obtained, verifying that $|C| \in O(1)$ and $|N(C)| \in O(\log |V|)$ is immediate. Finally, although computing the incidence treewidth $\text{itw}(G)$ exactly is $\mathcal{NP}$-hard in general, it is fixed-parameter tractable; in particular, one can either verify that $\text{itw}(G) \leq k$ or produce a tree decomposition of width k in time polynomial in $|V|$ for any fixed $k = O(\log |V|)$ [2].

To prove Theorem 2 we first present a number of intermediate results that will be used in the proof. The first result serves as the strongest sufficient condition for polynomial-time solvability of binary polynomial optimization.

Proposition 2 ([4]). *There is a strongly polynomial time algorithm to solve Problem* (2) *if the hypergraph G is β-acyclic or the incidence treewidth of G is bounded by* $\log \mathrm{poly}(|V|, |E|)$.

Now let us turn attention to a continuous polynomial optimization problem. The problem of minimizing a polynomial over a compact semialgebraic set can be reduced to deciding the feasibility of semialgebraic systems. In particular, deciding whether

$$\exists x \in [0,1]^k \text{ such that } f(x) \leq \lambda$$

is an instance of the existential theory of the reals. It is a classical result that such problems can be solved in time polynomial in the encoding length of the input when the number of variables is fixed; see Renegar [15] and Basu–Pollack–Roy [1]. More precisely, we have the following result:

Proposition 3 ([15]). *Given a polynomial $f(x)$, $x \in \mathbb{R}^n$ of degree at most d and with rational coefficients, the problem of minimizing f over $[0,1]^n$ can be solved in time $(nd)^{O(n)}\mathrm{poly}(\mathcal{L})$.*

Proposition 3 implies that Problem (1) in fixed-dimension can be solved in polynomial time. We should remark that even in fixed dimension, the optimal value of a polynomial optimization problem over $[0,1]^n$ may not be rational for $d \geq 3$. In general, optimal solutions and optimal values are algebraic numbers, i.e., roots of polynomials with rational coefficients. In this setting, an "exact" algorithm outputs such values in a symbolic form, typically by specifying a univariate polynomial $p(t) \in \mathbb{Q}[t]$ such that the optimal value λ^* is a root of p.

To prove Theorem 2 we need to show that the incidence treewidth of the interaction hypergraph remains bounded even if we add certain complete hypergraphs with small node sets to it. The next two lemmas establish this fact. To state these results, we first need to introduce some notation and terminology.

Given a graph $G = (V, E)$, a *tree decomposition* of G is a pair $(\mathcal{X}, T)$, where $\mathcal{X} = \{X_1, \cdots, X_m\}$ is a family of subsets of V, called *bags*, and T is a tree with m nodes, where each node of T corresponds to a bag such that:

1. $V = \bigcup_{i \in [m]} X_i$.
2. For every edge $\{v_j, v_k\} \in E$, there is a bag X_i for some $i \in [m]$ such that $X_i \ni v_j, v_k$.
3. For each node $v \in V$, the set of all bags containing v induces a connected subtree of T.

The *width* $\omega(\mathcal{X})$ of a tree decomposition $(\mathcal{X}, T)$ is the size of its largest bag X_i minus one. The *treewidth* $\mathrm{tw}(G)$ of a graph G is the minimum width among all possible tree decompositions of G. Let $G = (V, E)$ be a hypergraph and let $C \subseteq V$. We define the hypergraph obtained from G by *removing* C as $G - C := (V \setminus C, E')$, where $E' := \{e \setminus C : e \in E,\ |e \setminus C| \geq 2\}$.

Lemma 6. *Let $G = (V, E)$ be a hypergraph, and let $C \subseteq V$ be such that the subhypergraph of G induced by C is connected. Define the neighborhood of C as*

$$N(C) := \{u \in V \setminus C : \exists e \in E \text{ with } u \in e \text{ and } e \cap C \neq \emptyset\}. \tag{10}$$

Let G' be the hypergraph obtained from G by removing C and subsequently adding the complete hypergraph on $N(C)$, i.e., adding as edges all subsets $f \subseteq N(C)$ with $|f| \geq 2$. Then we have

$$\mathrm{itw}(G') \leq \max\big(\mathrm{itw}(G),\ |N(C)|\big).$$

Proof. Let $I(G) = (\bar{V}, \bar{E})$ denote the incidence graph of the hypergraph G. Define

$$S := C \cup \{e \in E : e \cap C \neq \emptyset\}.$$

Note that $S \subseteq \bar{V}$. We first show that the subgraph of $I(G)$ induced by S is connected. Denote by $G_C = (C, E_C)$ the subhypergraph of G induced by C. Take any two nodes $u, v \in C$. Since by assumption G_C is connected, there exist edges $f_1, \ldots, f_k \in E_C$ for some $k \geq 1$ such that $u \in f_1$, $v \in f_k$, and $f_j \cap f_{j+1} \neq \emptyset$ for all $j = 1, \ldots, k-1$. For each $j \in [k]$, choose an edge $e_j \in E$ such that $f_j = e_j \cap C$. Moreover, for each $j \in [k-1]$, choose a node $w_j \in f_j \cap f_{j+1} \subseteq C$. Define the sequence of nodes of $I(G)$: $u, e_1, w_1, e_2, w_2, \ldots, w_{k-1}, e_k, v$. This is a path in $I(G)$ because $u \in e_1$, $v \in e_k$ and for each $j \in [k-1]$, we have $w_j \in e_j \cap e_{j+1}$. Moreover, every node on this path belongs to S, so u and v are connected in the subgraph $I(G)$ induced by S. Now let $e \in E$ be any edge with $e \cap C \neq \emptyset$ implying that $e \in S$. Pick $u \in e \cap C$. Then e is adjacent to u in $I(G)$. Since every node in C lies in the same connected component of the subgraph $I(G)$ induced by S, it follows that every such e also lies in that same connected component. Hence, the subgraph $I(G)$ induced by S is connected.

Denote by $N_{I(G)}(S)$ the neighborhood of S in the graph $I(G)$. We claim that

$$N_{I(G)}(S) = N(C), \tag{11}$$

where $N(C)$ is defined by (10). To see this, let $v \in V \setminus C$. Then v is adjacent in $I(G)$ to a node of S if and only if there exists an edge $e \in E$ such that $v \in e$ and $e \cap C \neq \emptyset$. This is precisely the definition of $v \in N(C)$. Therefore, $N_{I(G)}(S) \cap V = N(C)$. On the other hand, consider some $e \in E \setminus S$. By definition of S, this means that $e \cap C = \emptyset$. If e were adjacent to some node of S in $I(G)$, then there would exist a node $v \in C$ such that $v \in e$, which contradicts $e \cap C = \emptyset$. Hence, no $e \in E \setminus S$ is adjacent to S. Therefore, equality (11) holds.

Let H be the graph obtained from $I(G)$ by removing S and making its neighborhood a clique. Since the subgraph of $I(G)$ induced by S is connected, by Lemma 3 and equality (11) we have:

$$\mathrm{tw}(H) \leq \max\big(\mathrm{itw}(G),\, |N(C)| - 1\big). \tag{12}$$

Let $(\mathcal{Y}, T')$ be a tree decomposition of H of width at most the right-hand side of (12). Since $N(C)$ is a clique in H, there exists a node $t^* \in V(T')$ whose bag Y_{t^*} contains $N(C)$; i.e.,

$$N(C) \subseteq Y_{t^*}. \tag{13}$$

We now construct a tree decomposition of $I(G')$ from $(\mathcal{Y}, T')$. From the definition of the hypergraph G' it follows that the incidence graph $I(G')$ is obtained from H by deleting the edges having both incident nodes in $N(C)$ and, for each subset $f \subseteq N(C)$ with $|f| \geq 2$, adding one new node w_f adjacent to all nodes in f. First, observe that deleting edges does not affect the validity of a tree decomposition, so $(\mathcal{Y}, T')$ is also a tree decomposition of the graph obtained from H after removing the edges with incident nodes in $N(C)$. For each subset $f \subseteq N(C)$ with $|f| \geq 2$, add a new leaf node t_f adjacent to t^*, and define its bag by

$$Y_{t_f} := f \cup \{w_f\}. \tag{14}$$

Keep all old bags unchanged. Let $(\mathcal{Y}', T'')$ be the resulting family of bags and tree. We next verify that $(\mathcal{Y}', T'')$ is a tree decomposition of $I(G')$.

1. Every old node of $I(G')$ already belongs to some bag of $(\mathcal{Y}, T')$. Each new node w_f belongs to the new bag Y_{t_f} defined by (14). Hence condition 1 of a tree decomposition holds.

2. Every old edge of $I(G')$ is an edge of the graph obtained from H by deleting the edges inside $N(C)$, hence it is contained in some old bag. Now let (u, w_f) be a new edge of $I(G')$. By construction, $u \in f$, hence both u, w_f belong to the bag Y_{t_f} defined by (14). Hence condition (2) of a tree decomposition holds.
3. Let u be a node of $I(G')$. Then the following cases arise:
   - If u is an old node not in $N(C)$, then no new bag contains u, implying that the set of bags containing u is unchanged and remains connected.
   - If u is an old node and $u \in N(C)$, then by (13) we have $u \in Y_{t^*}$. Moreover, u belongs to a new bag Y_{t_f} exactly when $u \in f$, and every such new bag is attached as a leaf to t^*. Therefore, the set of bags containing u is obtained from the old connected subtree containing u by attaching some leaves to the node t^*, and hence remains connected.
   - If u is a new node,i.e., $u = w_f$, then it belongs only to the single bag Y_{t_f} defined by (14), so the set of bags containing u is connected.

   Hence condition (3) of a tree decomposition holds.

Thus, $(\mathcal{Y}', T'')$ is a tree decomposition of $I(G')$. By (14), we can upper bound the size of the new bags as $|Y_{t_f}| = |f| + 1 \leq |N(C)| + 1$. Therefore, the width of the tree decomposition $(\mathcal{Y}', T'')$ is at most

$$\max\big(\mathrm{itw}(G),\ |N(C)|\big),$$

and this completes the proof. □

Lemma 7. *Let $G = (V, E)$ be a hypergraph, and let $S \subseteq V$ be nonempty. Denote by G_S the subhypergraph of G induced by S and let $C_1, \ldots, C_p$ denote the connected components of G_S. Define $G_0 := G$. For each $r \in [p]$, let G_r be the hypergraph obtained from G_{r-1} by first removing C_r and then adding the complete hypergraph on $N(C_r)$. Set*

$$d_{\max} := \max_{r \in [p]} |N(C_r)|.$$

Then

$$\mathrm{itw}(G_p) \leq \max\big(\mathrm{itw}(G),\, d_{\max}\big).$$

Proof. We prove the statement by induction on p. In the base case, we have $p = 1$ and the proof follows directly from Lemma 6. Now assume that $p \geq 2$ and that the statement holds for the first k components, where $1 \leq k < p$. That is, assume that

$$\mathrm{itw}(G_k) \leq \max\Big(\mathrm{itw}(G),\, \max_{r \in [k]} |N(C_r)|\Big). \tag{15}$$

We must prove that

$$\mathrm{itw}(G_{k+1}) \leq \max\Big(\mathrm{itw}(G),\, \max_{r \in [k+1]} |N(C_r)|\Big).$$

To do so, we will apply Lemma 6 to the hypergraph G_k and the set C_{k+1}. To this end, we need to verify two facts:

1. The subhypergraph of G_k induced by C_{k+1} is connected. Let $G_k = (V_k, E_k)$. Define $E[C_{k+1}] := \{e \cap C_{k+1} : e \in E, |e \cap C_{k+1}| \geq 2\}$ and $E_k[C_{k+1}] := \{e \cap C_{k+1} : e \in E_k, |e \cap C_{k+1}| \geq 2\}$. The prove the statement, it suffices to show that

$$E_k[C_{k+1}] = E[C_{k+1}].$$

   First we show that $E_k[C_{k+1}] \subseteq E[C_{k+1}]$. Let $f \in E_k[C_{k+1}]$. Then there exists $e \in E_k$ such that $f = e \cap C_{k+1}$ and $|e \cap C_{k+1}| \geq 2$. If $e \notin E$, then e was added during the elimination of some

C_r with $r \in [k]$, implying that $e \subseteq N(C_r)$. Pick $v \in e \cap C_{k+1}$. Then $v \in N(C_r)$, so there exists $e' \in E$ such that $v \in e'$ and $e' \cap C_r \neq \emptyset$. Hence $e' \cap S$ intersects both C_r and C_{k+1}, contradicting that they are distinct connected components of G_S. Therefore $e \in E$, and $f \in E[C_{k+1}]$.
Next we show that $E[C_{k+1}] \subseteq E_k[C_{k+1}]$. Let $f \in E[C_{k+1}]$. Then there exists $e \in E$ such that $f = e \cap C_{k+1}$ and $|e \cap C_{k+1}| \geq 2$. If $e \notin E_k$, then e was removed during the elimination of some C_r with $r \in [k]$, so $e \cap C_r \neq \emptyset$. Then $e \cap S$ intersects both C_r and C_{k+1}, again contradicting the definition of the components of G_S. Hence $e \in E_k$, and $f \in E_k[C_{k+1}]$.

2. The neighborhood of C_{k+1} in G_k is $N(C_{k+1})$; i.e.,

$$N_{G_k}(C_{k+1}) = N(C_{k+1}).$$

We first show that $N_{G_k}(C_{k+1}) \subseteq N(C_{k+1})$. Let $u \in N_{G_k}(C_{k+1})$. Then there exists $e \in E_k$ such that $u \in e$ and $e \cap C_{k+1} \neq \emptyset$. If $e \notin E$, then $e \subseteq N(C_r)$ for some $r \in [k]$. Pick $v \in e \cap C_{k+1}$. Then $v \in N(C_r)$, so there exists $e' \in E$ such that $v \in e'$ and $e' \cap C_r \neq \emptyset$, yielding a contradiction as above. Hence $e \in E$, and $u \in N(C_{k+1})$. Next we show that $N(C_{k+1}) \subseteq N_{G_k}(C_{k+1})$. Let $u \in N(C_{k+1})$. Then there exists $e \in E$ such that $u \in e$, $e \cap C_{k+1} \neq \emptyset$. If $e \notin E_k$, then $e \cap C_r \neq \emptyset$ for some $r \in [k]$, which again contradicts the definition of the connected components of G_S. Hence $e \in E_k$, and $u \in N_{G_k}(C_{k+1})$.

Therefore we can apply Lemma 6 to the hypergraph G_k and the set C_{k+1} to obtain

$$\operatorname{itw}(G_{k+1}) \leq \max\big(\operatorname{itw}(G_k),\, |N(C_{k+1})|\big).$$

Using the induction hypothesis (15), we obtain

$$\operatorname{itw}(G_{k+1}) \leq \max\Big(\operatorname{itw}(G),\, \max_{r\in[k]} |N(C_r)|,\, |N(C_{k+1})|\Big),$$

which by definition of $d_{\max}$ completes the proof. □

We are now ready to prove Theorem 2.

Proof of Theorem 2. For any $U \subseteq V$, denote by x_U the vector containing all components x_i of x with $i \in U$. Denote by $f_{V^-}(x_{V^-})$ the polynomial that collects all monomials in f whose support is contained in V^-. For each $C \in \mathcal{C}$, denote by $f_C(x_C, x_{N(C)})$ the polynomial that collects all monomials in f whose support intersects with C. We then claim that $f(x)$ an be written as:

$$f(x) = f_{V^-}(x_{V^-}) + \sum_{C\in\mathcal{C}} f_C(x_C, x_{N(C)}). \tag{16}$$

First consider a monomial x^α with $\operatorname{card}(\alpha) = 1$; suppose that $\alpha_k \geq 1$ for some $k \in [n]$. If $k \in V^-$, then this monomial is included in $f_{V^-}(x_{V^-})$. Otherwise, $k \in C$ for exactly one $C \in \mathcal{C}$ and hence is included in $f_C(x_C, x_{N(C)})$. Now suppose that $\operatorname{card}(\alpha) \geq 2$. Then by construction, this monomial corresponds to an edge $e \in E$. If $e \subseteq V^-$, then the monomial is included in f_{V^-}. Otherwise, $e \cap V^+ \neq \emptyset$. If $|e \cap V^+| \geq 2$, then $e \cap V^+ \in E_{V^+}$, and hence all the nodes in $e \cap V^+$ lie in a single connected component $C \in \mathcal{C}$. If $|e \cap V^+| = 1$, then e contains one node from V^+ and is assigned to the connected component containing that node. From the definition of $N(C)$ it follows that each such monomial depends only on variables in C and $N(C)$. Thus, every monomial is assigned to exactly one f_C implying that equation (16) is valid.

Since $C \cap C' = \emptyset$ for all $C \neq C' \in \mathcal{C}$, from (16) it follows that

$$\min_{x_{V^+}\in[0,1]^{V^+}} f(x_{V^-}, x_{V^+}) = f_{V^-}(x_{V^-}) + \sum_{C\in\mathcal{C}} \min_{x_C\in[0,1]^C} f_C(x_C, x_{N(C)}),$$

For each $C \in \mathcal{C}$ and each $z \in \{0,1\}^{N(C)}$, define

$$\psi_C(z) := \min_{x_C \in [0,1]^C} f_C(x_C, z).$$

We then deduce that

$$\min_{x_{V^-}, x_{V^+}} f(x_{V^-}, x_{V^+}) = \min_{x_{V^-} \in \{0,1\}^{V^-}} \Big(f_{V^-}(x_{V^-}) + \sum_{C \in \mathcal{C}} \psi_C(x_{N(C)}) \Big). \tag{17}$$

Therefore, once $\psi_C(z)$ is evaluated for all $z \in \{0,1\}^{N(C)}$ and $C \in \mathcal{C}$, Problem (1) reduces to a binary optimization problem. Let us now examine the cost of computing $\psi_C(z)$ at some $z \in \{0,1\}^{N(C)}$. By assumption (ii) this problem amounts to solving a polynomial optimization problem in fixed dimension, which can be solved in polynomial time in $\mathcal{L}$. Therefore, by assumption (iii) computing $\psi_C(z)$ for all $z \in \{0,1\}^{N(C)}$ can be done in $n^{O(1)}\text{poly}(\mathcal{L})$ time. Since $|\mathcal{C}| \leq |V|$, computing all required function values can be done in time polynomial in $\mathcal{L}$.

Each function $\psi_C(z)$ is defined on $\{0,1\}^{N(C)}$, hence can be written in polynomial time as polynomial function of binary variables z_i, $i \in N(C)$. Denote by $\bar{G}$ the hypergraph obtained from G by removing nodes in C and adding the complete hypergraph on $N(C)$ for all $C \in \mathcal{C}$. It then follows that $\bar{G}$ is the interaction hypergraph of the binary polynomial optimization problem (17). By Lemma 7, anf assumptions (i) and (iii) in the statement of the theorem, we have

$$\text{itw}(\bar{G}) \leq \max\Big(\text{itw}(G),\ \max_{C \in \mathcal{C}} |N(C)|\Big) \in O(\log |V|).$$

Therefore, it remains to solve a binary polynomial optimization problem over $\bar{G}$ with the incidence treewidth $\text{itw}(\bar{G})$. By Proposition 2, the latter can be solved in strongly polynomial time, and this completes the proof. □

Let us conclude by commenting on some of the assumptions and limitations of Theorem 2. First, unlike the sufficient condition for the quadratic case in Theorem 1, where logarithmic-size continuous components suffice, Theorem 2 requires that the continuous components have constant size. This difference arises from the complexity of the underlying subproblems: quadratic optimization over $[0,1]^k$ can be solved in time $3^k\text{poly}(\mathcal{L})$, whereas for general polynomial optimization the best known algorithms have complexity of the form $(kd)^{O(k)}\text{poly}(\mathcal{L})$ (see Proposition 3). As a result, only constant-dimensional continuous components can be handled while preserving overall polynomial-time solvability. Second, Proposition 2 shows that binary polynomial optimization is tractable not only under bounded incidence treewidth, but also under β-acyclicity of the corresponding hypergraph. However, the proof of Theorem 2 relies on eliminating continuous components by introducing complete hypergraphs on their neighborhoods. This operation creates β-cycles even when the original interaction hypergraph is β-acyclic. Consequently, our approach does not preserve β-acyclicity, and extending Theorem 2 to this setting would require different techniques.

References

[1] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*. Springer, 2006.

[2] H. L. Bodlaender and A. Koster. Combinatorial optimization on graphs of bounded treewidth. *The Computer Journal*, 51(3):255–269, 2008.

[3] E. Boros and P.L. Hammer. Pseudo-Boolean optimization. *Discrete applied mathematics*, 123(1):155–225, 2002.

[4] F. Capelli, A. Del Pia, and S. Di Gregorio. A knowledge compilation take on binary polynomial optimization. *arXiv preprint arXiv:2311.00149*, 2023.

[5] V. Chandrasekaran, N. Srebro, and P. Harsha. Complexity of inference in graphical models. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence UAI'08*, 2008.

[6] Y. Crama, P. Hansen, and B. Jaumard. The basic algorithm for pseudo-Boolean programming revisited. *Discrete Applied Mathematics*, 29(2–3):171–185, 1990.

[7] A. Del Pia and S. Di Gregorio. On the complexity of binary polynomial optimization over acyclic hypergraphs. *Algorithmica*, 85:2189–2213, 2023.

[8] A. Del Pia and A. Khajavirad. A polyhedral study of binary polynomial programs. *Mathematics of Operations Research*, 42(2):389–410, 2017.

[9] A. Del Pia and A. Khajavirad. Beyond hypergraph acyclicity: limits of tractability for pseudo-boolean optimization. *arXiv preprint arxiv:2410.23045*, 2024.

[10] A. Del Pia and A. Khajavirad. The pseudo-boolean polytope and polynomial-size extended formulations for binary polynomial optimization. *Mathematical Programming*, 212(1):717–761, 2025.

[11] S. S. Dey and A. Khajavirad. A second-order cone representable class of nonconvex quadratic programs. *Mathematical Programming*, 2026.

[12] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[13] A. Khajavirad. Tight semidefinite programming relaxations for sparse box-constrained quadratic programs. *arXiv preprint arXiv:2601.18545*, 2026.

[14] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

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

[16] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

[17] S. P. Tarasov and L. G. Khachiyan. Polynomial solvability of convex quadratic programming. In *Doklady akademii nauk*, volume 248, pages 1049–1051. Russian Academy of Sciences, 1979.