Computing Binary Integer Programming via A New Exact Penalty Function*

Shuai Li† Shenglong Zhou†‡

Abstract: Unconstrained binary integer programming (UBIP) poses significant computational challenges due to its discrete nature. We introduce a novel reformulation approach using a piecewise cubic function that transforms binary constraints into continuous equality constraints. Instead of solving the resulting constrained problem directly, we develop an exact penalty framework with a key theoretical advantage: the penalty parameter threshold ensuring exact equivalence is independent of the unknown solution set, unlike classical exact penalty theory. To facilitate the analysis of the penalty model, we introduce the concept of P-stationary points and systematically characterize their optimality properties and relationships with local and global minimizers. The P-stationary point enables the development of an efficient algorithm called APPA, which is guaranteed to converge to a P-stationary point within a finite number of iterations under a single mild assumption, namely, strong smoothness of the objective function over the unit box. Comprehensive numerical experiments demonstrate that APPA outperforms established solvers in both accuracy and efficiency across diverse problem instances.

Keywords: UBIP, piecewise cubic function, exact penalty theory, P-stationary points, global convergence, termination within finite iterations

1 Introduction

This paper focus on the following unconstrained binary integer programming (UBIP),

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ f(\mathbf{x}), \ \ \text{s.t.} \ \ \mathbf{x}\in\{0,1\}^n, \tag{UBIP}$$

where function $f:\mathbb{R}^n\to\mathbb{R}\cup\{\infty\}$ is continuously differentiable. The UBIP problem finds extensive applications spanning traditional combinatorial optimization and contemporary machine learning. In combinatorial optimization, it models classical problems like the knapsack problem [2, 21] and the max-cut problem [5, 37]. In machine learning, UBIP has attracted considerable attention for adversarial attacks [9, 32], transductive inference [14], and binary neural networks [18, 26]. For additional applications, we refer readers to [15, 19, 25]. Despite its broad applicability, (UBIP) is known to be NP-hard due to the combinatorial nature of binary variables [10, 16]. Instead of directly solving (UBIP), we aim to address the following constrained optimization,

$$\min_{x\in\mathbb{R}^n} \ f(\mathbf{x}), \ \ \text{s.t.} \ \ g(x_i)=0, \ x_i\in B, \ i\in[n], \tag{1.1}$$

where $[n]:=\{1,2,\cdots,n\}, B:=\{x\in\mathbb{R}:0\le x\le 1\}$, and $g:\mathbb{R}\to\mathbb{R}\cup\{\infty\}$ is a piecewise cubic function defined as follows and its graph is shown in Figure 1,

$$g(x):=\begin{cases} x^3-3x^2+3x, & x\le 1/2,\\ 1-x^3, & x>1/2. \end{cases} \tag{1.2}$$

*This work is supported by the Fundamental Research Funds for the Central Universities (2024YJS091).

†School of Mathematics and Statistics, Beijing Jiaotong University, China.

‡Email: 24110488@bjtu.edu.cn, shlzhou@bjtu.edu.cn

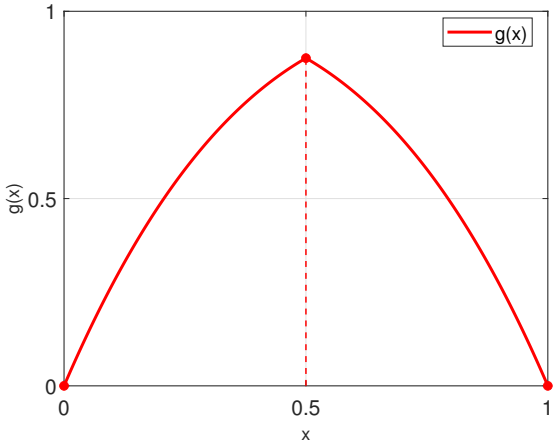


Figure 1: The graph of $g(x)$

1.1 Related work

Various approaches have been developed to process binary constraints in optimization problems, which can be broadly classified into two categories: relaxation methods and equivalent reformulations. Table 1 provides a summary of representative techniques from each category, which we briefly review below.

Relaxation approaches address binary constraints by transforming them into continuous optimization models. A widely adopted approach is linear programming (LP) relaxation [12, 17], which replaces the discrete binary constraint with a continuous box constraint. This reformulation converts the NP-hard binary problem into a tractable convex optimization problem that can be efficiently solved by modern first-order methods. Spectral relaxation [7, 30] offers an alternative strategy by substituting the binary constraint with a spherical constraint, allowing the problem to be solved via eigenvalue decomposition. Semi-definite programming (SDP) relaxation provides another powerful framework by introducing a matrix lifting technique and relaxing the binary constraint through a positive semidefinite cone [35, 1]. The resulting SDP formulations can be solved using various specialized algorithms, including interior point methods [35], quasi-Newton and smoothing Newton methods [31], spectral bundle methods [11], and branch-and-bound algorithms [4, 3]. To achieve tighter relaxations, doubly positive methods [13, 33] impose non-negativity constraints on both the eigenvalues and entries of the SDP solution matrix. Empirical studies have shown that these refined relaxation techniques can substantially improve solution quality compared to SDP approaches.

Equivalent reformulations provide another approach to addressing binary constraints in optimization problems. Unlike relaxation methods, these approaches transform the problem while preserving exact equivalence. A piecewise separable reformulation [40] converts UBIP into an equality-constrained optimization problem solvable by penalty methods. Continuous ℓ_2-box non-separable reformulation [27] has been employed to reformulate the UBIP as a continuous optimization problem, enabling the use of second-order interior-point methods [8, 23]. More generally, [36] proposed a continuous ℓ_p-norm ($p > 0$) box reformulation and applied ADMM for its solution. A different perspective was taken in [39], where the authors reformulated UBIP as an MPEC problem by embedding the ℓ_2 norm constraint within an equilibrium framework. The resulting formulation

Table 1: Main techniques to tackle the binary constraints.

Methods	References	Reformulations
Relaxation methods		
LP relaxation	[12, 17]	$\{0,1\}^n \approx \{\mathbf{x} \mid 0 \le \mathbf{x} \le 1\}$
Spectral relaxation	[7, 30]	$\{0,1\}^n \approx \{\mathbf{x} \mid \|2\mathbf{x}-1\|_2^2 = n\}$
SDP relaxation	[3, 31]	$\{0,1\}^n \approx \{\mathbf{x} \mid \mathbf{X} \succeq \mathbf{x}\mathbf{x}^\top, \mathrm{diag}(\mathbf{X}) = \mathbf{x}\}$
Doubly positive relaxation	[13, 33]	$\{0,1\}^n \approx \{\mathbf{x} \mid \mathbf{X} \succeq (2\mathbf{x}-1)(2\mathbf{x}-1)^\top, \mathrm{diag}(\mathbf{X}) = 1\}$
Equivalent reformulations		
Piecewise	[40]	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid \mathbf{x} \odot (1-\mathbf{x}) = 0\}$
ℓ_2 box	[23, 27]	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid 0 \le \mathbf{x} \le 1, \|2\mathbf{x}-1\|_2^2 = n\}$
ℓ_2 box MPEC	[39]	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid 0 \le \mathbf{x} \le 1, \|\mathbf{v}\|^2 \le n, \langle 2\mathbf{x}-1, \mathbf{v}\rangle = n\}$
ℓ_p box ($p > 0$)	[36]	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid 0 \le \mathbf{x} \le 1, \|2\mathbf{x}-1\|_p^p = n, p > 0\}$
ℓ_0 norm	[38]	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid \|\mathbf{x}\|_0 + \|\mathbf{x}-1\|_0 \le n\}$
Piecewise cubic	This paper	$\{0,1\}^n \Leftrightarrow \{\mathbf{x} \mid 0 \le \mathbf{x} \le 1, g(x_i) = 0, i \in [n]\}$

is an augmented biconvex optimization problem with a bilinear equality constraint, amenable to solution via exact penalty methods. Additionally, binary optimization can be reformulated as an ℓ_0 norm semi-continuous optimization problem [38]. Our proposed approach based on piecewise cubic function also belongs to this class of equivalent reformulations, as shown in Table 1.

1.2 Contribution and organization

In this paper, we introduce a piecewise cubic function to reformulate UBIP as the continuous model (1.1). This reformulation offers two key advantages: strong theoretical guarantees and the ability to develop efficient numerical algorithms. In other words, it provides a practical and effective continuous optimization framework for solving the discrete programming problem (UBIP). Our main contributions are summarized as follows.

1) New exact penalty theorems with solution-independent penalty parameters. To tackle (1.1) (equivalent to (UBIP)), we focus on its penalty model. By introducing the notion of P-stationary points, which serve as a key tool for both theoretical analysis and algorithm design, we establish novel exact penalty results showing that global minimizers of (1.1) and the penalty model coincide when the penalty parameter exceeds a threshold that is independent of the solution set of (1.1). This property distinguishes our theory from classical exact penalty frameworks, where the penalty parameter typically depends on the solution set of the original problem. Moreover, we systematically analyze the relationships between P-stationary points and local/global minimizers of (1.1) (see Figure 3). These relationships reveal that computing P-stationary points provides a practical and theoretically justified strategy for solving (UBIP), inspiring our algorithmic approach.

2) A simple numerical algorithm with strong convergence properties. To solve the penalty problem (2.1), we develop a novel algorithm called APPA (Adaptive Proximal Point Algorithm). Despite the adaptive updates of both the P-stationary parameter and the penalty parameter, we establish that

the whole sequence converges to a P-stationary point when f is strongly smooth on $\mathbb{B}$, as shown in Theorem 3.1. Remarkably, the algorithm exhibits finite termination, as outlined in Theorem 3.2, a property that is stronger than quadratic convergence.

3) Superior numerical performance. To evaluate the performance of APPA, we conduct comprehensive numerical experiments across diverse problem classes, benchmarking it against several state-of-the-art solvers, including the commercial solver GUROBI. The numerical experiments demonstrate that APPA achieves competitive solution quality while maintaining significantly faster computational times, especially in high-dimensional settings.

The remainder of this paper is organized as follows. The next subsection introduces the notation and preliminary concepts used throughout. Section 2 presents the penalty reformulation of (1.1) and establishes the exact penalty theory. Section 3 develops the APPA algorithm and analyzes its convergence properties. Extensive numerical experiments and concluding remarks are given in the last two sections.

1.3 Preliminaries

We end this section by introducing some notation to be used throughout the paper. The n-dimensional unit box is denoted by

$$\mathbb{B} := \{\mathbf{x} \in \mathbb{R}^n : x_i \in B, i \in [n]\},$$

where $B := \{x \in \mathbb{R} : 0 \leq x \leq 1\}$, $[n] = 1, 2, ..., n$, and ':=' means 'define'. We denote by $\|\mathbf{x}\|$ and $\|\mathbf{x}\|_\infty$ the ℓ_2- and ℓ_∞-norms of a vector $\mathbf{x} \in \mathbb{R}^n$, respectively. For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\langle \mathbf{x}, \mathbf{y} \rangle$ denotes their inner product, i.e., $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y} = \sum x_i y_i$. Let $\lceil t \rceil$ be the ceiling of t, i.e., the smallest inter no less than t. For a given set Ω, the indicator function $\delta_\Omega(\cdot)$ is defined as

$$\delta_\Omega(\mathbf{x}) := \begin{cases} 0, & \text{if } \mathbf{x} \in \Omega, \\ \infty, & \text{if } \mathbf{x} \notin \Omega. \end{cases}$$

We denote by $N_{\mathbb{B}}(\mathbf{x})$ the normal cone [28] of the set $\mathbb{B}$ at the point $\mathbf{x}$. Then any $\mathbf{v} \in N_{\mathbb{B}}(\mathbf{x})$ satisfies

$$v_i \in \begin{cases} (-\infty, 0], & \text{if } x_i = 0, \\ \{0\}, & \text{if } x_i \in (0, 1), \\ [0, +\infty), & \text{if } x_i = 1. \end{cases} \tag{1.3}$$

For a lower semi-continuous function $f : \mathbb{R}^n \to \mathbb{R}$, the definition of the (limiting) subdifferential denoted by ∂f can be found in [28, Definition 8.3], which allows us to calculate

$$\partial g(x) = \begin{cases} \left\{3x^2 - 6x + 3\right\}, & x < 1/2, \\ \{-3/4, 3/4\}, & x = 1/2, \\ \left\{-3x^2\right\}, & x > 1/2. \end{cases} \tag{1.4}$$

One can easily verify that

$$|\nu| \geq 3, \ \forall \nu \in \partial g(x), \ \forall x \in B. \tag{1.5}$$

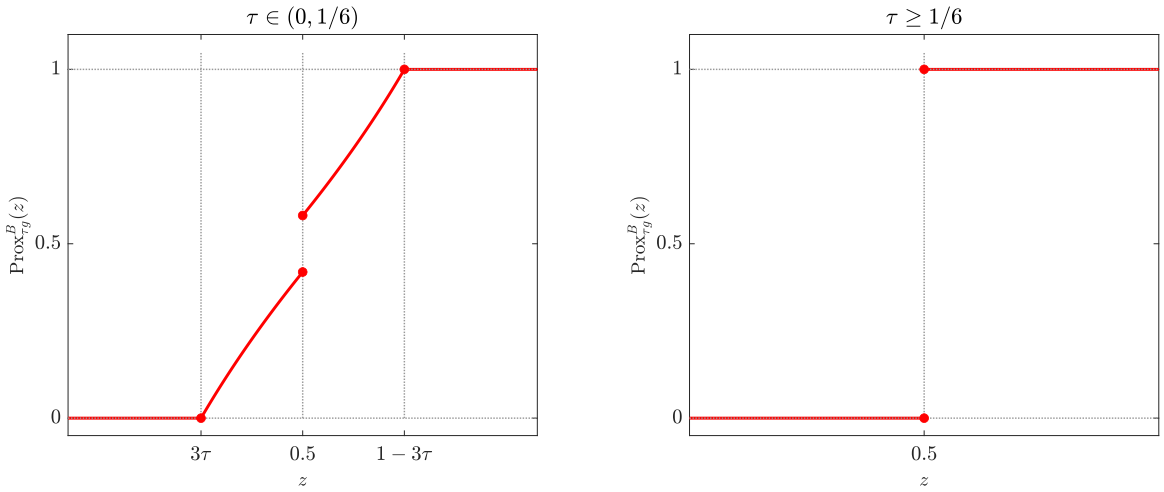


Figure 2: The graph of $\mathrm{Prox}_{\tau g}^{B}(z)$

We say function f is L-strong smoothness on box $\mathbb{B}$ if

$$f(\mathbf{x}) \leq f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{x} - \mathbf{w} \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{w}\|^2, \quad \forall\, \mathbf{x}, \mathbf{w} \in \mathbb{B},$$

where $L > 0$. This is a weaker condition than the strong smoothness of f on whole space $\mathbb{R}^n$. A sufficient condition to L-strong smoothness on box $\mathbb{B}$ is the twice continuously differentiability of f. Moreover, we say function f is ℓ-strong convex on box $\mathbb{B}$ if

$$f(\mathbf{x}) \geq f(\mathbf{w}) + \langle \nabla f(\mathbf{w}), \mathbf{x} - \mathbf{w} \rangle + \frac{\ell}{2}\|\mathbf{x} - \mathbf{w}\|^2, \quad \forall\, \mathbf{x}, \mathbf{w} \in \mathbb{B},$$

where $\ell > 0$. The proximal operator of a function $p : \mathbb{R}^n \to \mathbb{R}$, associated with a parameter $\tau > 0$, is defined by

$$\mathrm{Prox}_{\tau p}(\mathbf{z}) := \underset{\mathbf{x} \in \mathbb{R}^n}{\mathrm{argmin}}\ p(\mathbf{x}) + \frac{1}{2\tau}\|\mathbf{x} - \mathbf{z}\|^2. \tag{1.6}$$

Specifically, we define the proximal operator of the function $p + \delta_\Omega$ as follows:

$$\mathrm{Prox}_{\tau p}^{\Omega}(\mathbf{z}) := \mathrm{Prox}_{\tau(p+\delta_\Omega)}(\mathbf{z}) = \underset{\mathbf{x} \in B}{\mathrm{argmin}}\ p(\mathbf{x}) + \frac{1}{2\tau}\|\mathbf{x} - \mathbf{z}\|^2, \tag{1.7}$$

where Ω is a given set. Finally, we present the explicit form of $\mathrm{Prox}_{\tau g}^{B}(z)$ in the following lemma, with its graph shown in Figure 2.

Lemma 1.1. *For any $z \in \mathbb{R}^n$ and $\tau > 0$, proximal operator $\mathrm{Prox}_{\tau g}^{B}(z)$ takes the following forms.*

- *If $\tau \geq 1/6$, then*

$$\mathrm{Prox}_{\tau g}^{B}(z) = \begin{cases} \{0\}, & z < 1/2, \\ \{0, 1\}, & z = 1/2, \\ \{1\}, & z > 1/2. \end{cases} \tag{1.8}$$

- *If $\tau \in (0, 1/6)$, then*

$$\mathrm{Prox}_{\tau g}^{B}(z) = \begin{cases} \{0\}, & z \leq 3\tau, \\ \{z_1^*\}, & z \in (3\tau, 1/2), \\ \{z_1^*, z_2^*\}, & z = 1/2, \\ \{z_2^*\}, & z \in (1/2, 1-3\tau), \\ \{1\}, & z \geq 1 - 3\tau, \end{cases} \tag{1.9}$$

where $z_1^* := 1 + \dfrac{\sqrt{1+12\tau(z-1)}-1}{6\tau}$ *and* $z_2^* := \dfrac{1-\sqrt{1-12\tau z}}{6\tau}$.

Proof. For a given $z \in \mathbb{R}$ and $\eta > 0$, denote $g^* := \min\limits_{x\in B} g(x) + \dfrac{(x-z)^2}{2\tau}$ and

$$\begin{aligned} g_1(x) :=& x^3 - 3x^2 + 3x + \frac{(x-z)^2}{2\tau}, \\ g_2(x) :=& 1 - x^3 + \frac{(x-z)^2}{2\tau}. \end{aligned}$$

To derive g^*, we need to compare the following two optimal objective function values, namely,

$$g_1^* := \min_{x\in[0,\frac{1}{2}]} g_1(x), \qquad g_2^* := \min_{x\in(\frac{1}{2},1]} g_2(x).$$

Direct calculation enables us to obtain their values under different τ, which are summarized in Table 2, where $\tau_1 := 1/2 - 3\tau/4$ and $\tau_2 := 1/2 + 3\tau/4$. Finally, a comparison of g_1^* and g_2^* in different cases yields (1.8) and (1.9). □

Table 2: The optimal function values of g_1 and g_2.

τ	g_1^*		g_2^*	
$0 < \tau < 1/6$	$g_1(0)$,	$z < 3\tau$	$g_2(1/2)$,	$z < \tau_1$
	$g_1(z_1^*)$,	$z \in [3\tau, \tau_2)$	$g_2(z_2^*)$,	$z \in [\tau_1, 1-3\tau)$
	$g_1(1/2)$,	$z \geq \tau_2$	$g_2(1)$,	$z \geq 1-3\tau$
$1/6 \leq \tau < 2/9$	$\min\{g_1(0), g_1(z_1^*)\}$,	$z < 3\tau$	$g_2(1/2)$,	$z < \tau_1$
	$g_1(z_1^*)$,	$z \in [3\tau, \tau_2)$	$g_2(z_2^*)$,	$z \in [\tau_1, 1-3\tau)$
	$g_1(1/2)$,	$z \geq \tau_2$	$\min\{g_2(1), g_1(z_2^*)\}$,	$z \geq 1-3\tau$
$2/9 \leq \tau < 1/3$	$\min\{g_1(0), g_1(z_1^*)\}$,	$z < \tau_2$	$g_2(1/2)$,	$z < 1-3\tau$
	$\min\{g_1(0), g_1(1/2)\}$,	$z \in [\tau_2, 3\tau)$	$\min\{g_2(1/2), g_2(1)\}$,	$z \in [1-3\tau, \tau_1)$
	$g_1(1/2)$,	$z \geq 3\tau$	$\min\{g_2(z_2^*), g_2(1)\}$,	$z \geq \tau_2$
$\tau \geq 1/3$	$g_1(0)$,	$z < \tau_2$	$g_2(1/2)$,	$z < 1-3\tau$
	$\min\{g_1(0), g_1(1/2)\}$,	$z \in [\tau_2, 3\tau)$	$\min\{g_2(1/2), g_2(1)\}$,	$z \in [1-3\tau, \tau_1)$
	$g_1(1/2)$,	$z \geq 3\tau$	$g_2(1)$,	$z \geq \tau_2$

2 Exact Penalty Theory

Instead of solving problem (1.1), we focus on solving its penalty formulation,

$$\min_{\mathbf{x}\in\mathbb{B}} \; F(\mathbf{x};\lambda) := f(\mathbf{x}) + \lambda p(\mathbf{x}), \qquad \text{with} \;\; \lambda \geq \overline{\lambda} \;\; \text{and} \;\; p(\mathbf{x}) := \sum_{i=1}^{n} g(x_i), \tag{2.1}$$

where $\overline{\lambda}$ is defined by

$$\overline{\lambda} := \max_{\mathbf{x}\in\mathbb{B}} \frac{\|\nabla f(\mathbf{x})\|_\infty}{3}. \tag{2.2}$$

Since f is continuously differentiable and $\mathbb{B}$ is bounded, ∇f is continuous on $\mathbb{B}$, which implies that $\overline{\lambda}$ is well-defined and finite. In the sequel, we show that (2.1) is an exact penalty model of (1.1).

To derive the exact penalty theory, we define a P-stationary point associated with the proximal operator of p, which also plays a crucial role in our algorithm design.

Definition 2.1. *Point $\overline{\mathbf{x}}$ is called a P-stationary point of (2.1) if there is a $\tau > 0$ such that*

$$\begin{aligned} \overline{\mathbf{x}} &\in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}\Big(\overline{\mathbf{x}} - \tau\nabla f(\overline{\mathbf{x}})\Big), \\ &= \mathrm{argmin}_{\mathbf{z}\in\mathbb{B}}\ \frac{1}{2}\|\mathbf{z} - (\overline{\mathbf{x}} - \tau\nabla f(\overline{\mathbf{x}}))\|^2 + \tau\lambda p(\mathbf{z}). \end{aligned} \tag{2.3}$$

By Lemma 1.1, we can directly obtain the closed-form of $\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}$, namely,

$$\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}(\mathbf{z}) = \{\mathbf{v}\in\mathbb{R}^n : v_i \in \mathrm{Prox}^{B}_{\tau\lambda g}(z_i), i\in[n]\},$$

which allows us to design a fast numerical algorithm given in the next section. The following result establishes the relationships among P-stationary points and local minimizers of problem (2.1).

Theorem 2.1. *The following relationships hold for problem (2.1).*

1) Any P-stationary point is binary.

2) A local minimizer is a P-stationary point if f is L-strongly smooth on $\mathbb{B}$.

3) A P-stationary point with $\tau \geq 1/\ell$ is a global minimizer if f is ℓ-strongly convex on $\mathbb{B}$.

Proof. 1) Suppose $\tilde{\mathbf{x}} \notin \{0,1\}^n$ is a P-stationary point of (2.1). Then, there exists a $\tau > 0$ such that

$$\tilde{\mathbf{x}} \in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}\Big(\tilde{\mathbf{x}} - \tau\nabla f(\tilde{\mathbf{x}})\Big), \tag{2.4}$$

According to the generalized Fermat's rule [28, Theorem 10.1(P422)], it follows that

$$\begin{aligned} 0 &\in \partial\left(\frac{1}{2}\|\tilde{\mathbf{x}} - (\tilde{\mathbf{x}} - \tau\nabla f(\tilde{\mathbf{x}}))\|^2 + \tau\lambda p(\tilde{\mathbf{x}}) + \delta_{\mathbb{B}}(\overline{\mathbf{x}})\right) \\ &= \tau\nabla f(\tilde{\mathbf{x}}) + \partial\left(\tau\lambda p(\tilde{\mathbf{x}}) + \delta_{\mathbb{B}}(\tilde{\mathbf{x}})\right) \\ &= \tau\nabla f(\tilde{\mathbf{x}}) + \tau\lambda\partial p(\tilde{\mathbf{x}}) + N_{\mathbb{B}}(\tilde{\mathbf{x}}), \end{aligned} \tag{2.5}$$

where the first equation are from three facts: 1) f is continuously differentiable, 2) $p(\widetilde{\mathbf{x}})$ is lower semi-continuous and bounded, and 3) [28, 10.10 Exercise], the second equation holds because the unique nondifferentiable point of g is $1/2$, which lies in the interior of $[0,1]$.

Since $\tilde{\mathbf{x}} \notin \{0,1\}^n$, there is an $\widetilde{x}_i \in (0,1)$. Then (2.5) implies that there is $\nu_i \in \partial g(\tilde{x}_i)$ such that $\nabla_i f(\widetilde{\mathbf{x}}) = -\lambda\nu_i$. Moreover, from (1.5), we have $|\nu_i| > 3$ for any $\nu_i \in \partial g(\tilde{x}_i)$, which yields

$$\overline{\lambda} = \max_{\mathbf{x}\in\mathbb{B}} \frac{\|\nabla f(\mathbf{x})\|_\infty}{3} \geq \frac{|\nabla_i f(\widetilde{\mathbf{x}})|}{3} = \frac{\lambda|\nu_i|}{3} > \lambda \geq \overline{\lambda}, \tag{2.6}$$

leading to a contradiction. Therefore, any P-stationary point must be binary.

2) Let $\mathbf{x}$ be a local minimizer of (2.1) and $\mathbf{z}$ satisfy $\mathbf{z} \in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}\Big(\mathbf{x} - \tau\nabla f(\mathbf{x})\Big)$. By Fermat's rule [28, Theorem 10.1(P422)], it holds that

$$\begin{aligned} 0 &\in \partial(f(\widetilde{\mathbf{x}}) + \lambda p(\widetilde{\mathbf{x}}) + \delta_{\mathbb{B}}(\widetilde{\mathbf{x}})) \\ &= \nabla f(\widetilde{\mathbf{x}}) + \lambda\partial p(\widetilde{\mathbf{x}}) + N_{\mathbb{B}}(\widetilde{\mathbf{x}}), \end{aligned} \tag{2.7}$$

where the equality holds for the same reason as in the previous part. Similar to the argument in (2.6), it follows from (2.7) that $\mathbf{x}$ is binary, and hence $p(\mathbf{x}) = 0$. The definition of $\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}$ yields

$$\|\mathbf{z} - (\mathbf{x} - \tau\nabla f(\mathbf{x}))\|^2 + 2\tau\lambda p(\mathbf{z}) \le \|\mathbf{x} - (\mathbf{x} - \tau\nabla f(\mathbf{x}))\|^2 + 2\tau\lambda p(\mathbf{x}),$$

which results in

$$\begin{aligned}\|\mathbf{z} - \mathbf{x}\|^2 &\le 2\tau\langle \mathbf{x} - \mathbf{z}, \nabla f(\mathbf{x})\rangle + 2\tau\lambda(p(\mathbf{x}) - p(\mathbf{z}))\\ &\le 2\tau\langle \mathbf{x} - \mathbf{z}, \nabla f(\mathbf{x})\rangle\\ &\le 2\tau\|\mathbf{z} - \mathbf{x}\|\|\nabla f(\mathbf{x})\|\\ &\le 2\tau\sqrt{n}\|\mathbf{z} - \mathbf{x}\|\|\nabla f(\mathbf{x})\|_\infty\\ &\le 6\tau\overline{\lambda}\sqrt{n}\|\mathbf{z} - \mathbf{x}\|,\end{aligned} \tag{2.8}$$

where the last inequality is from (2.2). This implies $\|\mathbf{z} - \mathbf{x}\| \le 2\tau c\overline{\lambda}\sqrt{n}$, meaning $\mathbf{z}$ is around $\mathbf{x}$ when τ is sufficiently small. Using the L-strong smoothness of f on $\mathbb{B}$ and the first inequality in (2.8), we obtain

$$\begin{aligned}&2F(\mathbf{z};\lambda) - 2F(\mathbf{x};\lambda)\\ \le\ & 2\langle \mathbf{z} - \mathbf{x}, \nabla f(\mathbf{x})\rangle + 2\lambda(p(\mathbf{z}) - p(\mathbf{x})) + L\|\mathbf{z} - \mathbf{x}\|^2\\ \le\ & (L - 1/\tau)\,\|\mathbf{z} - \mathbf{x}\|^2 \le 0,\end{aligned}$$

where the last inequality is because τ is sufficiently small. The above condition leads to $\mathbf{z} = \mathbf{x}$ due to the local optimality of $\mathbf{x}$, namely, $\mathbf{x}$ is a P-stationary point.

3) Let $\overline{\mathbf{x}}$ be a P-stationary point. Then it satisfies

$$\overline{\mathbf{x}} \in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}\Big(\overline{\mathbf{x}} - \tau\nabla f(\overline{\mathbf{x}})\Big).$$

By the definition of $\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}$, we obtain

$$\|\overline{\mathbf{x}} - (\overline{\mathbf{x}} - \tau\nabla f(\overline{\mathbf{x}}))\|^2 + 2\tau\lambda p(\overline{\mathbf{x}}) \le \|\mathbf{w} - (\overline{\mathbf{x}} - \tau\nabla f(\overline{\mathbf{x}}))\|^2 + 2\tau\lambda p(\mathbf{w}),$$

for any $\mathbf{w} \in \mathbb{B}$, which results in

$$2\langle \overline{\mathbf{x}} - \mathbf{w}, \nabla f(\overline{\mathbf{x}})\rangle + 2\lambda(p(\overline{\mathbf{x}}) - p(\mathbf{w})) \le (1/\tau)\|\mathbf{w} - \overline{\mathbf{x}}\|^2. \tag{2.9}$$

Using the above condition and the ℓ-strong convexity of f, we have

$$\begin{aligned}&2F(\overline{\mathbf{x}};\lambda) - 2F(\mathbf{w};\lambda)\\ \le\ & 2\langle \overline{\mathbf{x}} - \mathbf{w}, \nabla f(\overline{\mathbf{x}})\rangle + 2\lambda(p(\mathbf{x}) - p(\mathbf{w})) - \ell\|\mathbf{w} - \overline{\mathbf{x}}\|^2\\ \le\ & (1/\tau - \ell)\,\|\mathbf{z} - \overline{\mathbf{x}}\|^2 \le 0,\end{aligned}$$

due to $\tau \ge 1/\ell$. Therefore, $\overline{\mathbf{x}}$ is global minimizer of (2.1). □

To highlight the advantages of our proposed penalty function $g(x)$ over existing alternatives, we provide a comparative example. Specifically, we contrast our piecewise cubic penalty with the ℓ_p-norm based penalty $h(x) = n - \|2\mathbf{x} - \mathbf{1}\|_p^p$ from [36].

Example 2.1. *Consider the following two single-variable penalty problems:*

$$\min_{x\in[0,1]}\ F_1(x;\lambda) = \frac{1}{2}\left(x - \frac{1}{2}\right)^2 + \lambda(1 - |2x - 1|^3), \tag{2.10}$$

$$\min_{x\in[0,1]}\ F_2(x;\lambda) = \frac{1}{2}\left(x - \frac{1}{2}\right)^2 + \lambda g(x). \tag{2.11}$$

- ***Problem (2.10) with ℓ_p-norm penalty:*** *For any $\lambda > 0$, the non-binary point $x = 1/2$ is always a local minimizer. To verify this, note that $\partial F_1(1/2;\lambda) = \{0\}$ for all λ. Furthermore, for sufficiently small $|\varepsilon|$:*

$$\begin{aligned} F_1\left(\frac{1}{2}+\varepsilon;\lambda\right) - F_1\left(\frac{1}{2};\lambda\right) &= \frac{1}{2}\varepsilon^2 + \lambda\left(1 - 8|\varepsilon|^3\right) - \lambda \\ &= \left(\frac{1}{2|\varepsilon|} - 8\lambda\right)|\varepsilon|^3 \geq 0, \end{aligned}$$

confirming that $x = 1/2$ is indeed a local minimizer regardless of the penalty parameter value.

- ***Problem (2.11) with our piecewise cubic penalty:*** *When $\lambda > \overline{\lambda} = 1/6$, all local minimizers are binary. To see this, observe that*

$$0 \notin \partial F_2(x;\lambda) = x - \frac{1}{2} + \lambda \begin{cases} \{3x^2 - 6x + 3\}, & x \in (0, 1/2), \\ [-3/4, 3/4], & x = 1/2, \\ \{-3x^2\}, & x \in (1/2, 1). \end{cases}$$

Hence, Problem (2.11) has no local minimizers in $(0,1)$. The binary points $x = 0$ and $x = 1$ are the only local (and global) minimizers.

This example reveals a key distinction: penalties like $h(x)$ may admit non-binary local minimizers regardless of the penalty parameter, whereas $g(x)$ ensures all local minimizers are binary when λ exceeds a computable threshold.

Based on Theorem 2.1, we establish the exact penalty theorem for problems (1.1) and (2.1).

Theorem 2.2 (Exact Penalty Theorem)**.** *A point is a global minimizer of problem (1.1) if and only if it is a global minimizer of problem (2.1).*

Proof. Let $\mathbf{x}^*$ and $\widetilde{\mathbf{x}}$ be global minimizer of problems (1.1) and (2.1). Similar to the reasoning in the proof of part 3) of Theorem 2.1, any global minimizer of (2.1) satisfies the optimality condition (2.7) and thus $\widetilde{\mathbf{x}} \in \{0,1\}^n$. Note that the feasible region of (1.1) is $\{0,1\}^n$. Since $\widetilde{\mathbf{x}} \in \{0,1\}^n$, we have $f(\mathbf{x}^*) \leq f(\widetilde{\mathbf{x}})$ by the global optimality of $\mathbf{x}^*$ for problem (1.1). This yields

$$F(\widetilde{\mathbf{x}};\lambda) \leq F(\mathbf{x}^*;\lambda) = f(\mathbf{x}^*) \leq f(\widetilde{\mathbf{x}}) = F(\widetilde{\mathbf{x}};\lambda),$$

where the first inequality holds because $\widetilde{\mathbf{x}}$ is a global minimizer of problem (2.1) and $\mathbf{x}^* \in \{0,1\}^n \subseteq \mathbb{B}$, the two equations hold due to $p(\mathbf{x}^*) = p(\widetilde{\mathbf{x}}) = 0$. The above condition implies the four values are the same, concluding the conclusion. □

Remark 2.1. *We make a few comments on Theorem 2.1 and 2.2.*

Based on Theorems 2.1–2.2, we summarize the relationships among different solutions to problems (UBIP), (1.1), and (2.1), as shown in Figure 3. Specifically, the global minimizers of these three problems are equivalent; a global minimizer $\mathbf{x}$ of problem (2.1) is a P-stationary point if f is strongly smooth on $\mathbb{B}$; a P-stationary point of problem (2.1) is also a global minimizer if f is strongly convex on $\mathbb{B}$. These relationships indicate that finding a P-stationary point of problem (2.1) provides an effective approach for solving problem (1.1) or (UBIP).

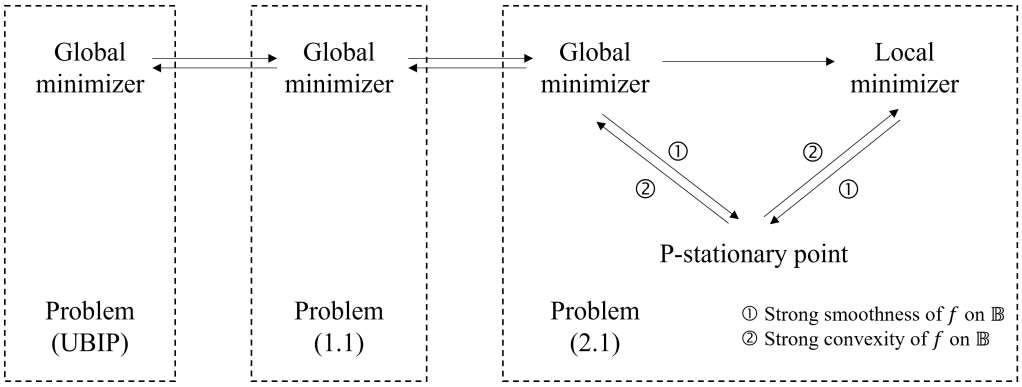


Figure 3: Relationships among different points for problems (UBIP) and (2.1).

Theorem 2.2 shows that when the penalty parameter $\lambda > \overline{\lambda}$, the global minimizers of the penalty model (2.1) and the original problem (1.1) coincide. Recall that $\overline{\lambda}$ defined in (2.2) is a constant independent of the solution set of (1.1), which distinguishes Theorem 2.2 from traditional exact penalty theory (e.g., [24, Theorem 17.3]), where the penalty parameter threshold typically depends on the solution to the original problem. Additionally, a result similar to Theorem 2.2 was established in [39], but under the assumption that f is Lipschitz continuous and convex on $\mathbb{B}$. However, deriving Theorem 2.2 requires no additional conditions on f.

To end this section, we present a result to guarantee a binary solution to $\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}$.

Lemma 2.1. *Let $\mathbf{x} \in \mathbb{B}$ and $\mathbf{z} \in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}(\mathbf{x} - \tau\nabla f(\mathbf{x}))$. Then $\mathbf{z} \in \{0,1\}^n$ when $\lambda \geq \overline{\lambda} + 1/(3\tau)$.*

Proof. By the definition of $\mathrm{Prox}^{\mathbb{B}}_{\tau\lambda p}$, we have

$$\mathbf{z} \in \mathrm{argmin}_{\mathbf{w}\in\mathbb{B}} \ \frac{1}{2}\|\mathbf{w} - (\mathbf{x} - \tau\nabla f(\mathbf{x}))\|^2 + \tau\lambda p(\mathbf{w}).$$

Then the similar reasoning to derive (2.5) enables the following optimality condition,

$$0 \in \mathbf{z} - \mathbf{x} + \tau\nabla f(\mathbf{x}) + \tau\lambda\partial\varphi(\mathbf{z}) + N_{\mathbb{B}}(\mathbf{z}).$$

If there is i such that $z_i \in (0,1)$, then $z_i - x_i + \tau\nabla_i f(\mathbf{x}) + \tau\lambda v_i = 0$, where $\mathbf{v} \in \partial\varphi(\mathbf{z})$. Hence,

$$\begin{aligned} 3\tau\lambda < |\tau\lambda v_i| &= |z_i - x_i + \tau\nabla_i f(\mathbf{x})| \\ &\leq |z_i - x_i| + \tau|\nabla_i f(\mathbf{x})| \\ &\leq 1 + 3\tau\overline{\lambda} \\ &< 3\tau\lambda, \end{aligned}$$

where the third inequality is from $z_i \in (0,1), x_i \in [0,1]$, and (2.2). This contradiction implies that $\mathbf{z}$ is binary, as desired. □

3 Algorithm and convergence

In this section, we develop the algorithm to solve problem (2.1). Specifically, for current point $\mathbf{x}^k \in \mathbb{R}^n$, the next point is updated by,

$$\mathbf{x}^{k+1} \in \mathrm{Prox}^{\mathbb{B}}_{\tau_k\lambda_k p}\Big(\mathbf{x}^k - \tau_k\nabla f(\mathbf{x}^k)\Big), \tag{3.1}$$

where τ_k and λ_k are updated adaptively. Stopping criteria can be set as

$$\mathbf{x}^k \in \{0,1\}^n, \qquad \left\|\mathbf{x}^k - \mathbf{x}^{k+1}\right\| = \left\|\mathbf{x}^k - \mathrm{Prox}^{\mathbb{B}}_{\tau_k\lambda_k p}\left(\mathbf{x}^k - \tau_k \nabla f(\mathbf{x}^k)\right)\right\| < \varepsilon, \tag{3.2}$$

where $\varepsilon \in (0,1)$ is a given tolerance. One can observe that if $\mathbf{x}^k$ satisfies the above conditions for any given ε, then it is a binary P-stationary point of problem (2.1) with a particular λ, as outlined in Theorem 3.1. The main update, (3.1), is a typical step of the proximal point algorithm. To facilitate binary solutions, we increase penalty parameter λ adaptively. Overall, we present the algorithmic framework in Algorithm 1 and call it an adaptive proximal point algorithm (APPA).

Algorithm 1: Adaptive proximal point algorithm (APPA).

Input $\mathbf{x}^0 \in \mathbb{B}$, $(\lambda_0, \eta, \sigma, \theta) > 0, \alpha \in (0,1)$, $\pi > 1$, and an integer $k_0 > 0$.

for $k = 0, 1, 2, \ldots$ **do**

Find the smallest integer $s_k \in \{0, 1, 2, \ldots\}$ such that

$$\tau_k = \eta\alpha^{s_k}, \tag{3.3}$$

$$\mathbf{x}^{k+1} \in \mathrm{Prox}^{\mathbb{B}}_{\tau_k\lambda_k p}\left(\mathbf{x}^k - \tau_k \nabla f(\mathbf{x}^k)\right), \tag{3.4}$$

$$F(\mathbf{x}^{k+1}; \lambda_k) \le F(\mathbf{x}^k; \lambda_k) - (\sigma/2)\|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2. \tag{3.5}$$

If (3.2) holds, then stop.

If $\mathrm{mod}(k+1, k_0) = 0$ and $\lambda_k < \theta$, then $\lambda_{k+1} = \pi\lambda_k$, else $\lambda_{k+1} = \lambda_k$.

end

In Algorithm 1, we increase the penalty parameter only when $\mathrm{mod}(k+1, k_0) = 0$ and $\lambda_k < \theta$, where $\mathrm{mod}(a, b)$ returns the remainder after division of a by b. This is because condition $\mathrm{mod}(k+1, k_0) = 0$ prevents λ_k from increasing too quickly, while threshold θ ensures that λ_k does not tend to ∞. According to Lemma 2.1, there exists a finite threshold beyond which $\mathbf{x}^{k+1}$ is ensured to be binary. Therefore, there is no reason to continue increasing λ_k once it exceeds the threshold.

3.1 Convergence analysis

To establish the convergence, we always set

$$\theta \ge \overline{\lambda} + \frac{\sigma + L}{3\alpha}. \tag{3.6}$$

Based on this θ, we define some constants:

$$\begin{aligned} &\tau_\infty := \eta\alpha^S \qquad \text{with} \qquad S := -\left\lceil \log_\alpha(\eta(\sigma + L)) \right\rceil, \\ &\lambda_\infty := \lambda_0\pi^K \qquad \text{with} \qquad K := \left\lceil k_0 \max\left\{0, \log_\pi\left(\frac{\theta}{\lambda_0}\right)\right\}\right\rceil. \end{aligned} \tag{3.7}$$

Based on the above constants, our first result shows that τ_k is well defined, the point generated by APPA is always binary, and sequence $\{f(\mathbf{x}^k)\}$ is non-increasing after finitely many iterations.

Lemma 3.1. *The following claims hold for APPA if f is L-strong smoothness on $\mathbb{B}$.*

1) For any $k \ge 1$, conditions (3.3)-(3.5) are ensured with $\tau_k = \tau_\infty$ and

$$\frac{1}{\sigma + L} \ge \tau_\infty \ge \frac{\alpha}{\sigma + L}. \tag{3.8}$$

2) For any $k > K$, it holds $\lambda_k \equiv \lambda_\infty$, $\mathbf{x}^k \in \{0,1\}^n$, and

$$f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k) \leq \frac{\sigma}{2}\|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2. \tag{3.9}$$

Proof. 1) For any $\tau > 0$, the following problem

$$\mathbf{w}^{k+1} \in \mathrm{Prox}^{\mathbb{B}}_{\tau\lambda_k p}\left(\mathbf{x}^k - \tau\nabla f(\mathbf{x}^k)\right),$$

enables us to derive that

$$\|\mathbf{w}^{k+1} - \mathbf{x}^k + \tau\nabla f(\mathbf{x}^k)\|^2 + 2\tau\lambda_k p(\mathbf{w}^{k+1}) \leq \|\mathbf{x}^k - \mathbf{x}^k + \tau\nabla f(\mathbf{x}^k)\|^2 + 2\tau\lambda_k p(\mathbf{x}^k),$$

which leads to

$$2\lambda_k p(\mathbf{w}^{k+1}) - 2\lambda_k p(\mathbf{x}^k) + 2\langle\nabla f(\mathbf{x}^k), \mathbf{w}^{k+1} - \mathbf{x}^k\rangle \leq -(1/\tau)\|\mathbf{w}^{k+1} - \mathbf{x}^k\|^2.$$

Using the above condition and the L-strongly smoothness of f yield that

$$\begin{aligned} & 2F(\mathbf{w}^{k+1};\lambda_k) - 2F(\mathbf{x}^k;\lambda_k) \\ \leq\ & 2\lambda_k p(\mathbf{w}^{k+1}) - 2\lambda_k p(\mathbf{x}^k) + 2\langle\nabla f(\mathbf{x}^k), \mathbf{w}^{k+1} - \mathbf{x}^k\rangle + L\|\mathbf{w}^{k+1} - \mathbf{x}^k\|^2 \\ \leq\ & (L - 1/\tau)\,\|\mathbf{w}^{k+1} - \mathbf{x}^k\|^2. \end{aligned}$$

Therefore, if $\tau \in (0, 1/(\sigma + L)]$, then we have

$$F(\mathbf{w}^{k+1};\lambda_k) - F(\mathbf{x}^k;\lambda_k) \leq -\frac{\sigma}{2}\|\mathbf{w}^{k+1} - \mathbf{x}^k\|^2.$$

This means conditions (3.3)-(3.5) hold with $\mathbf{w}^{k+1} = \mathbf{x}^{k+1}$ when

$$\tau_k = \eta\alpha^{s_k} = \eta\alpha^S \in \left[\frac{\alpha}{\sigma + L}, \frac{1}{\sigma + L}\right].$$

2) If $\lambda_0 \geq \theta$, then we always have $\lambda_k \equiv \lambda_0$ as $\{\lambda_k\}$ is non-decreasing. Therefore,

$$\lambda_k\tau_k - \overline{\lambda}\tau_k \geq \frac{(\lambda_0 - \overline{\lambda})\alpha}{\sigma + L} \geq \frac{(\theta - \overline{\lambda})\alpha}{\sigma + L} \geq \frac{1}{3}, \tag{3.10}$$

where the first and last inequalities are from (3.8) and (3.6). Thus $\mathbf{x}^{k+1} \in \{0,1\}^n$ from Lemma 2.1. We note that if $\lambda_0 \geq \theta$ then $K = 0$, thereby leading to the conclusion.

Now we consider $\lambda_0 < \theta$. It follows from (3.7) that

$$\lambda_K = \lambda_0\pi^{\lceil K/k_0\rceil} \geq \theta.$$

Therefore, for any $k \geq K$, we have $\lambda_k \equiv \lambda_K \equiv \lambda_\infty$. Then similar reasoning to show (3.10) can show $\lambda_k\tau_k \geq \overline{\lambda}\tau_k + 1/3$, which by Lemma 2.1 yields $\mathbf{x}^k \in \{0,1\}^n$ and thus $p(\mathbf{x}^k) = 0$. Overall

$$\lambda_k \begin{cases} \in [\lambda_0, \lambda_\infty), & k \in [0, K], \\ \equiv \lambda_\infty, & k > K, \end{cases}$$

namely, $\{\lambda_k\}$ is bounded. Based on these facts, for any $k > K$, we have

$$\begin{aligned} f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k) &= F(\mathbf{x}^{k+1};\lambda_{k+1}) - F(\mathbf{x}^k;\lambda_k) \\ &= F(\mathbf{x}^{k+1};\lambda_\infty) - F(\mathbf{x}^k;\lambda_\infty) \\ &\leq (\sigma/2)\|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2. \end{aligned}$$

where the last inequality is from (3.5). □

Theorem 3.1. *The following results hold for APPA if f is L-strong smoothness on $\mathbb{B}$.*

1) Whole sequence $\{f(\mathbf{x}^k)\}$ converges and $\lim_{k\to\infty}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|=0$.

2) Whole sequence $\{\mathbf{x}^k\}$ converges to a P-stationary point with $\tau \geq 1/(3\lambda_\infty - 3\overline{\lambda})$ of

$$\min_{\mathbf{x}\in\mathbb{B}} f(\mathbf{x}) + \lambda_\infty p(\mathbf{x}). \tag{3.11}$$

Proof. 1) Since $\{\mathbf{x}^k\}\subseteq\mathbb{B}$, we have $\{f(\mathbf{x}^k)\}$ is bounded because f is continuous. Therefore, $\{f(\mathbf{x}^k)\}$ converges by (3.9). Taking the limit of the both sides of (3.9) derives $\lim_{k\to\infty}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|=0$.

2) By (3.10), for any $k > K$ we have

$$\lambda_\infty\tau_\infty = \lambda_{k+1}\tau_{k+1} \geq \overline{\lambda}\tau_\infty + 1/3. \tag{3.12}$$

Let $\{\mathbf{x}^k : k\in T\}$ be a convergent subsequence of $\{\mathbf{x}^k\}\subseteq\mathbb{B}$ and $\lim_{k(\in T)\to\infty}\mathbf{x}^k = \mathbf{x}^\infty$, where $T\subseteq\{0,1,2,3\ldots\}$. Moreover, it follow from Lemma 3.1 that

$$\begin{aligned}\mathbf{x}^{k+1} &\in \mathrm{Prox}^{\mathbb{B}}_{\tau_\infty\lambda_\infty p}\Big(\mathbf{x}^k - \tau_\infty\nabla f(\mathbf{x}^k)\Big)\\ &= \mathrm{Prox}_{\tau_\infty\lambda_\infty(p+\delta_{\mathbb{B}})}\Big(\mathbf{x}^k - \tau_\infty\nabla f(\mathbf{x}^k)\Big),\end{aligned}$$

for any $k > K$. Using these facts, [28, Theorem 1.25], and taking the limit of both sides of the above condition along with $k(\in T)\to\infty$ enable us to derive that

$$\begin{aligned}\mathbf{x}^{\infty} &\in \mathrm{Prox}_{\tau_\infty\lambda_\infty(p+\delta_{\mathbb{B}})}\Big(\mathbf{x}^\infty - \tau_\infty\nabla f(\mathbf{x}^\infty)\Big)\\ &= \mathrm{Prox}^{\mathbb{B}}_{\tau_\infty\lambda_\infty p}\Big(\mathbf{x}^\infty - \tau_\infty\nabla f(\mathbf{x}^\infty)\Big).\end{aligned}$$

Therefore $\mathbf{x}^\infty$ is a P-stationary point of (3.11) with $\tau_\infty \geq 1/(3\lambda_\infty - 3\overline{\lambda})$. Since any $\mathbf{x}^\infty\in\{0,1\}^n$ which means it is isolated around its neighbour region. This together with $\lim_{k\to\infty}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|=0$ and [22, Lemma 4.10] delivers the whole sequence convergence. □

Theorem 3.2. *APPA terminates within finitely many steps if f is L-strong smoothness on $\mathbb{B}$.*

Proof. For any $k > K$, if $\mathbf{x}^{k+1}\neq\mathbf{x}^k$, we have $\|\mathbf{x}^{k+1}-\mathbf{x}^k\|>1$ due to $\mathbf{x}^k\in\{0,1\}^n$, contradicting to $\lim_{k\to\infty}\|\mathbf{x}^{k+1}-\mathbf{x}^k\|=0$. Therefore, we can conclude that there is $K' > K$ such that $\mathbf{x}^{k+1}\equiv\mathbf{x}^k$ for any $k\geq K'$, which indicates conditions in (3.2) are satisfied. □

4 Numerical experiment

In this section, we evaluate the performance of APPA on three problems: the recovery problems, the Multiple-Input-Multiple-Output (MIMO) detection, and the quadratic unconstrained binary optimization (QUBO). All codes are implemented in MATLAB (R2021a) and executed on a desktop computer with an Intel(R) Xeon W-3465X CPU (2.50 GHz) and 128 GB of RAM.

4.1 Recovery problem

We first demonstrate the performance of APPA for solving a recovery problem. The problem aims to recover a binary signal $\mathbf{x}^*$ from the linear system $\mathbf{b} = \mathbf{A}\mathbf{x}^*$ with $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$. To find the original signal $\mathbf{x}^*$, we consider the following optimization problem,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_q^q, \quad \text{s.t.} \quad \mathbf{x} \in \{0,1\}^n,$$

where $\|\mathbf{x}\|_q^q = \sum_i |x_i|^q$ and $q > 1$. Let $\mathbf{b} \in \mathbb{R}^m$ be generated by the linear regression model

$$\mathbf{b} = \mathbf{A}\mathbf{x}^* + \mathrm{nf} \cdot \boldsymbol{\varepsilon},$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is the measurement matrix whose entries are independently and identically distributed (i.i.d.) from the standard normal distribution $\mathcal{N}(0,1)$, followed by normalization $\mathbf{A} \leftarrow \mathbf{A}/c$ with $c = \sqrt{m}$ when $n \le 10000$ and $c = 1$ otherwise. The ground truth signal $\mathbf{x}^*$ is taken from $\{0,1\}^n$ with randomly picked s indices on which entries are 1. The additive noise $\boldsymbol{\varepsilon} \in \mathbb{R}^m$ has i.i.d. components $\varepsilon_i \sim \mathcal{N}(0,1)$ for $i = 1,\ldots,m$, and the parameter $\mathrm{nf} \ge 0$ controls the noise level.

4.1.1 Benchmark methods

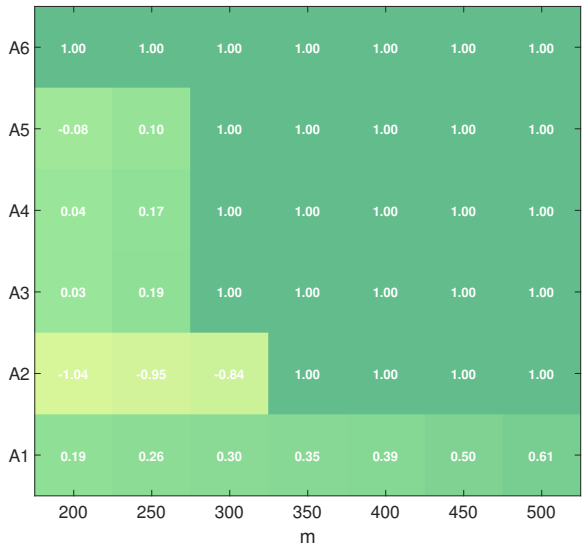
We compare APPA against NPGM, MEPM [39], L2ADMM [36], EMADM [19], and GUROBI, where NPGM applies Nesterov's proximal gradient method to the LP relaxation of (UBIP) and EMADM solves its SDP relaxation. Note that GUROBI and EMADM only handle binary quadratic programs ($q = 2$). For APPA, we set $(\eta, \sigma, \lambda, \pi) = (1, 10^{-8}, 0.25, 1.5)$, $\theta = \|\mathbf{A}\|_\infty + \|\mathbf{b}\|_\infty$, update frequency $k_0 = 100$ (if $n < 10000$) or $k_0 = 50$ (otherwise), and initial penalty $\lambda_0 = r\|\mathbf{b}^\top\mathbf{A}\|_\infty$ with $r = \min\{0.01, 0.1^{4\sqrt{s}/\log_2(mn)}\}$ if $2m \le n < 10s$ and $r = 0.05$ otherwise. For MEPM and L2ADMM, we set $(\rho, \sigma, T) = (0.01, \sqrt{10}, 10)$ and $(\alpha, \sigma, T) = (0.01, \sqrt{10}, 10)$ respectively, both with 200 maximum iterations, tolerance 10^{-2}, and Lipschitz constant $L = \|\mathbf{A}\|_F^2/n$. For EMADM, we set $(\beta, \lambda_0, \delta) = (0.1, 0.001, 10^{-4})$ with 10^4 maximum iterations and tolerance 10^{-4}. Finally, we initialize all algorithms by $\mathbf{x}_0 = 0$ and set the maximum running time to 600 seconds. To evaluate the algorithmic performance, we let $\mathbf{x}$ be the solution obtained by one algorithm and report the recovery accuracy ($\mathrm{Acc} := 1 - \|\mathbf{x} - \mathbf{x}^*\|/\|\mathbf{x}^*\|$) and the computational time in seconds.

4.1.2 Numerical comparison

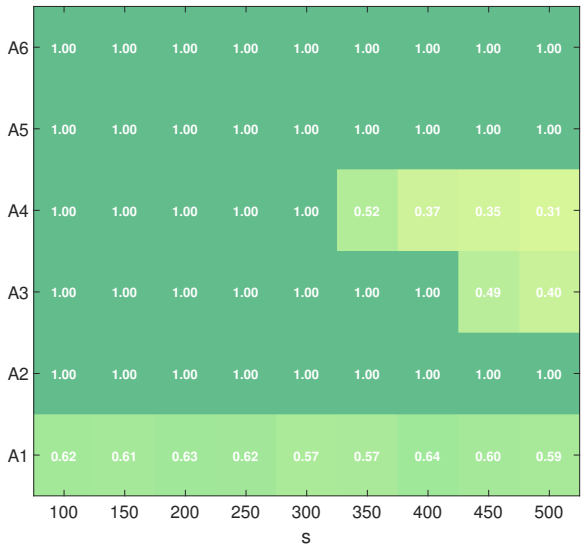
To evaluate the efficiency of selected algorithms for solving the recovery problem in different scenarios. We alter one factor of $(m, n, s, q, \mathrm{nf})$ to see its effect by fixing the other four factors.

a) Effect of m. We set $(n, s, q, \mathrm{nf}) = (1000, 100, 2, 0)$ and vary the number of measurements $m \in \{250, 300, \ldots, 500\}$. For each parameter setting $(m, n, s, q, \mathrm{nf})$, we perform 20 independent trials and report the median result. As illustrated in Figure 4a, the recovery accuracy improves monotonically as m increases. Notably, when $m \ge 350$, all algorithms except NPGM successfully recover the true signal with 100% accuracy. Meanwhile, APPA attains exact recovery in all cases.

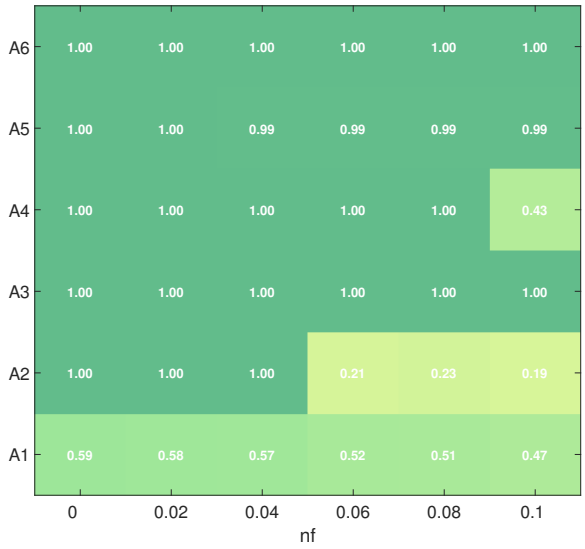
b) Effect of s. We fix $(m, n, q, \mathrm{nf}) = (500, 1000, 2, 0)$ and vary $s \in \{250, 300, \ldots, 500\}$. The median values over 20 trials are presented in Figure 4b. It is evident that the larger the value of s, the more difficult it becomes to detect the ground-truth signal. In all cases, APPA, L2ADMM, and GUROBI



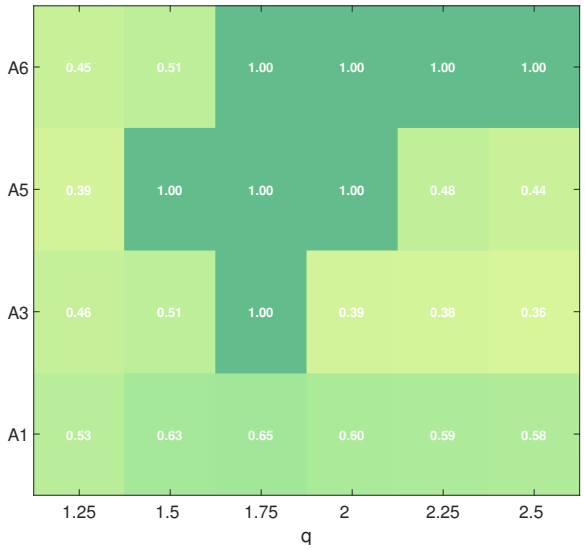
(a) Acc v.s. m



(b) Acc v.s. s



(c) Acc v.s. nf



(d) Acc v.s. q

Figure 4: Results on recovery problems, where A1-A6 stand for NPGM, GUROBI, MEPM, EMEDM, L2ADMM and APPA, respectively.

successfully achieve exact recovery, whereas EMADM and MEPM fail to do so when $s \geq 300$ and $s \geq 450$, respectively.

c) Effect of nf**.** We fix $(m, n, s, q) = (500, 1000, 300, 2)$ and vary the noise level nf $\in \{0, 0.02, 0.04, \ldots, 0.1\}$. Figure 4c presents the median recovery accuracy over 20 trials. The results demonstrate that APPA, MEPM, and L2ADMM exhibit robustness to noise. In contrast, GUROBI (within 600 seconds) fails to recover the signal when nf ≥ 0.06, while EMADM fails at nf $= 0.1$.

d) Effect of q**.** We fix $(m, n, s, \text{nf}) = (500, 1000, 500, 0)$ but increase q over range $\{1.25, 1.5, \ldots, 2.5\}$. The median results over 20 trials are reported in Figure 4d. As noted previously, GUROBI and EMADM are only applicable to the quadratic case ($q = 2$), and thus their results are omitted. Again, APPA maintains the highest accuracy, with the exception of scenarios where $q \leq 1.5$.

e) Effect of higher dimensions. To evaluate performance on large-scale problems, we consider dimensions $n \in \{10^4, 10^5, 10^6\}$ with $(m, s) = (0.5n, 0.01n)$ and noise levels nf $\in \{0, 0.5\}$. GUROBI

Table 3: Effect of higher dimensions, where A1-A4 stand for APPA, MEPM, L2ADMM, and NPGM, respectively.

n	nf	Acc				Time			
		A1	A2	A3	A4	A1	A2	A3	A4
					$q = 1.5$				
10^4	0.0	1.000	1.000	0.989	0.627	0.192	24.32	39.14	3.085
10^4	0.5	1.000	1.000	0.988	0.692	0.218	23.54	38.87	3.241
10^5	0.0	1.000	1.000	1.000	-0.108	71.89	923.7	1214.6	37.96
10^5	0.5	1.000	1.000	0.981	-0.387	102.8	941.5	1298.4	39.12
10^6	0.0	1.000	-3.397	-3.142	-1.805	4.427	3312.8	3600.0	117.8
10^6	0.5	1.000	-3.425	-3.203	-1.912	67.54	3198.6	3600.0	118.3
					$q = 2$				
10^4	0.0	1.000	1.000	1.000	0.671	0.169	18.52	27.18	2.963
10^4	0.5	1.000	1.000	0.968	0.639	0.176	23.69	34.21	3.105
10^5	0.0	1.000	-5.594	-5.467	-4.586	2.521	1261.7	1239.8	36.87
10^5	0.5	1.000	-5.597	-5.488	-4.551	4.037	1272.4	1283.5	37.45
10^6	0.0	1.000	-5.578	-4.106	-4.547	5.628	3225.9	3438.2	106.9
10^6	0.5	1.000	-5.541	-4.132	-4.573	6.891	3287.3	3421.6	109.7
					$q = 2.5$				
10^4	0.0	1.000	-5.851	-5.822	-5.718	0.201	39.24	38.79	3.146
10^4	0.5	1.000	-5.843	-5.869	-5.711	0.203	38.76	39.28	3.197
10^5	0.0	1.000	-5.859	-5.831	-5.794	4.938	1314.6	1295.7	8.751
10^5	0.5	1.000	-5.817	-5.826	-5.798	4.652	1293.2	1283.4	37.84
10^6	0.0	1.000	-5.803	-5.809	-5.693	9.582	3528.3	3600.0	123.4
10^6	0.5	1.000	-5.776	-5.836	-5.721	11.43	3445.7	3600.0	117.6

and EMADM are excluded as their computational requirements become prohibitive when $n \geq 10^4$. For $n \geq 10^5$, we employ sparse measurement matrices $\mathbf{A}$ with 10^8 nonzero entries. The median results over 10 independent trials are presented in Table 3. Remarkably, APPA consistently outperforms all competing methods across all problem sizes, achieving superior recovery accuracy while maintaining the fastest computation time.

4.2 MIMO detection

Two variants of the MIMO detection problem are considered in this section: classical MIMO detection, characterized by a quadratic objective function, and one-bit MIMO detection, which features a non-quadratic objective function.

4.2.1 Classical MIMO detection

The classical MIMO detection problem seeks to reconstruct an unknown signal $\mathbf{w}^*$ from noisy linear observations $\mathbf{y} = \mathbf{H}\mathbf{w}^* + \boldsymbol{\varepsilon}$. Here, $\mathbf{H} \in \mathbb{C}^{m\times n}$ denotes the channel matrix, $\mathbf{y} \in \mathbb{C}^m$ represents the

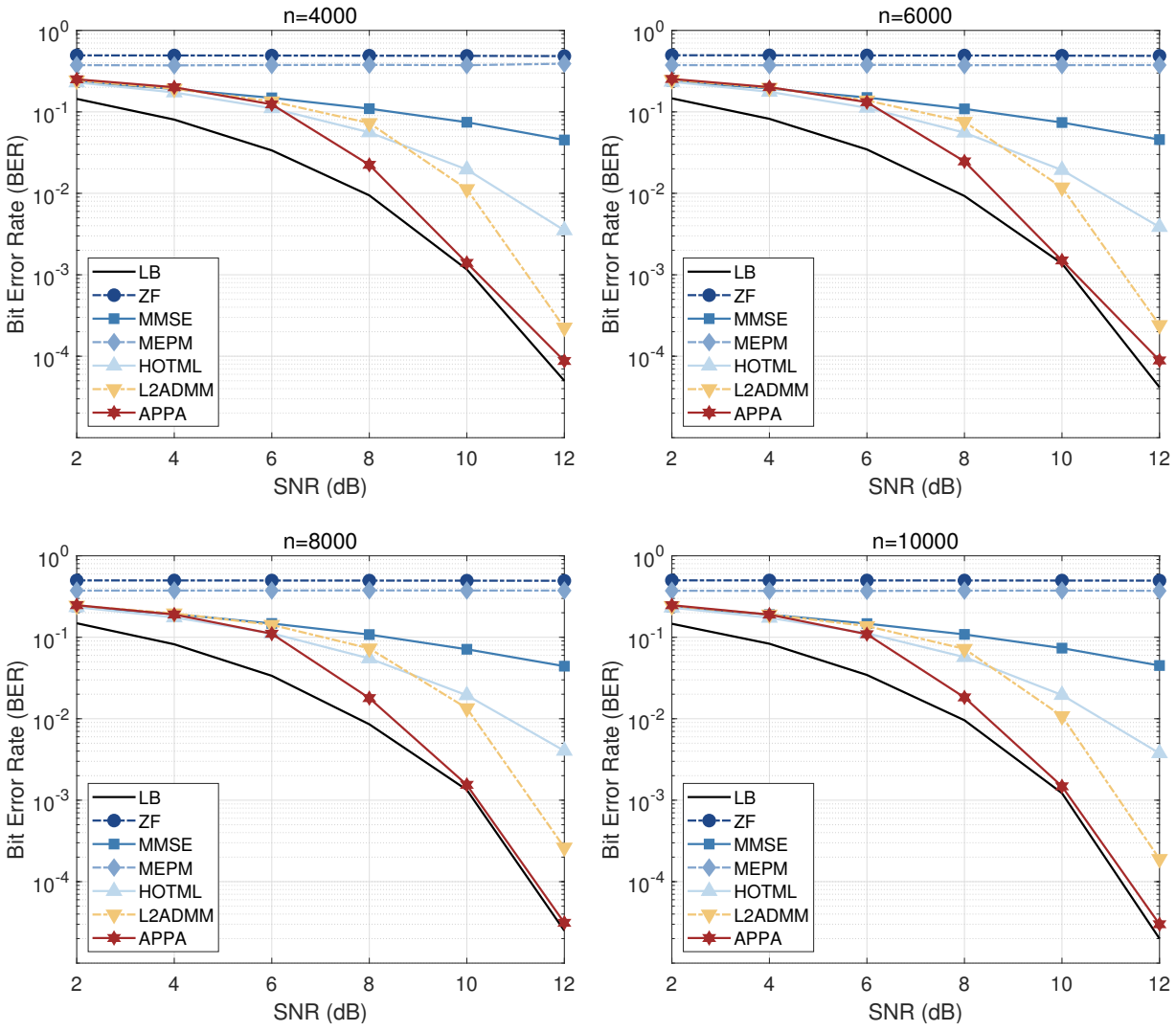


Figure 5: Results on classical MIMO detection problems for i.i.d channels.

observed signal, and $\boldsymbol{\varepsilon} \in \mathbb{C}^m$ is additive Gaussian noise with i.i.d. entries drawn from $\mathcal{N}(0, \varrho^2)$. The true signal $\mathbf{w}^*$ belongs to the constraint set $\Omega := \{\mathbf{w} \in \mathbb{C}^n : \mathrm{Re}(\mathbf{w}) \in \{0,1\}^n, \mathrm{Im}(\mathbf{w}) \in \{0,1\}^n\}$, where $\mathrm{Re}(\mathbf{w})$ and $\mathrm{Im}(\mathbf{w})$ denote the real and imaginary components of $\mathbf{w}$, respectively. This detection problem can be formulated as a maximum likelihood estimation:

$$\min_{\mathbf{x}\in\mathbb{R}^{2n}} \quad \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2, \quad \text{s.t.} \quad \mathbf{x} \in \{0,1\}^{2n},$$

where

$$\mathbf{A} := \begin{bmatrix} \mathrm{Re}(\mathbf{H}) & -\mathrm{Im}(\mathbf{H}) \\ \mathrm{Im}(\mathbf{H}) & \mathrm{Re}(\mathbf{H}) \end{bmatrix}, \qquad \mathbf{b} := \begin{bmatrix} \mathrm{Re}(\mathbf{y}) \\ \mathrm{Im}(\mathbf{y}) \end{bmatrix}, \qquad \mathbf{x} := \begin{bmatrix} \mathrm{Re}(\mathbf{w}) \\ \mathrm{Im}(\mathbf{w}) \end{bmatrix}.$$

We consider two types of channel matrices $\mathbf{H}$:

- **i.i.d. Gaussian channel:** The real and imaginary parts $\mathrm{Re}(\mathbf{H})$ and $\mathrm{Im}(\mathbf{H})$ are generated independently with entries sampled from the standard normal distribution, following the same procedure used for matrix $\mathbf{A}$ in Section 4.1.

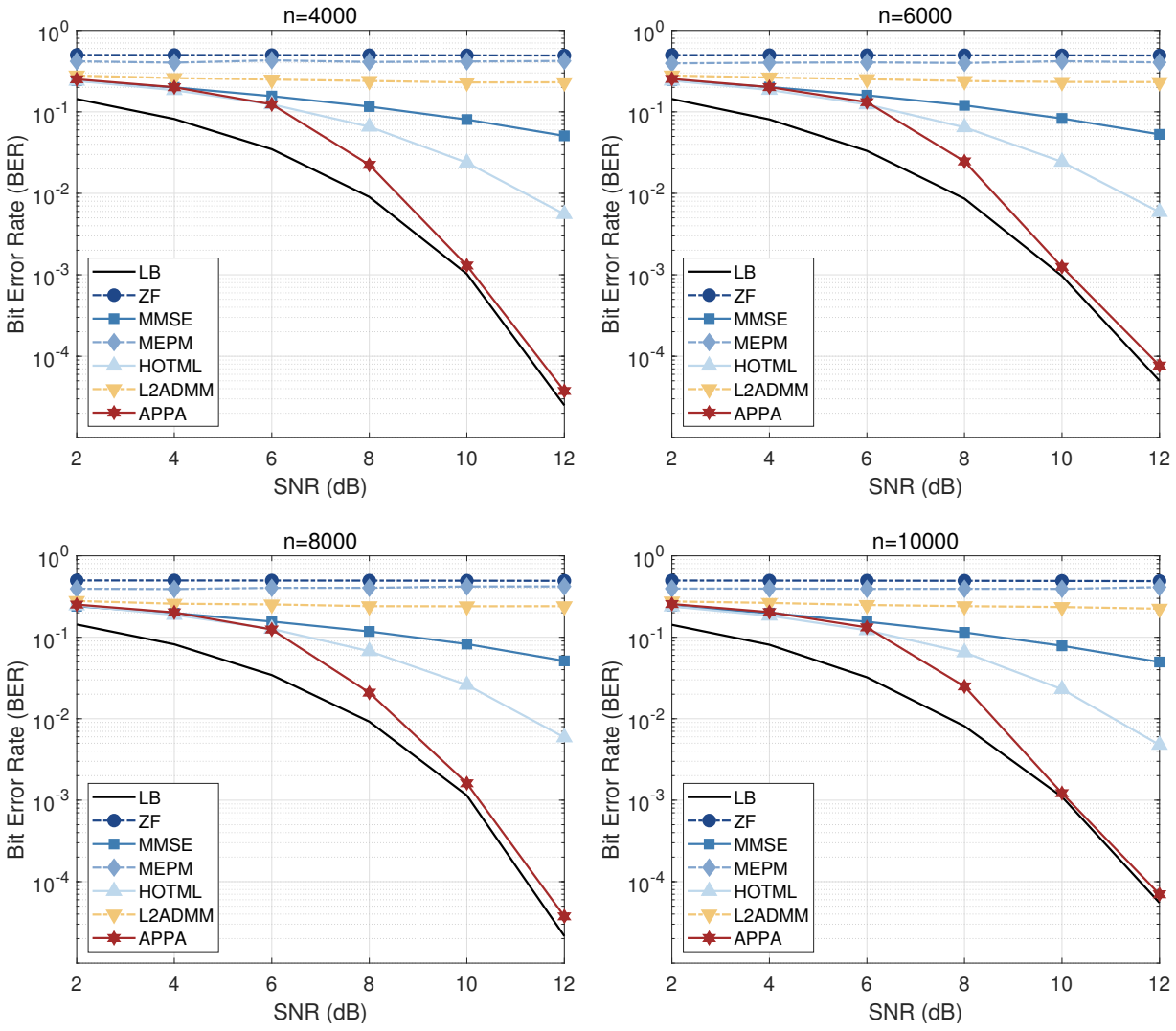


Figure 6: Results on classical MIMO detection problems for correlated channels.

- **Correlated MIMO channel:** Following [20, 29], we construct a spatially correlated channel as follows. Let $\widetilde{\mathbf{H}}$ be an element-wise i.i.d. circularly symmetric complex Gaussian matrix with zero mean and unit variance. Define spatial correlation matrices $\mathbf{R} \in \mathbb{C}^{m\times m}$ and $\mathbf{T} \in \mathbb{C}^{n\times n}$ at the receiver and transmitter, respectively, with entries

$$R_{ij} = \begin{cases} r^{i-j}, & \text{if } i \le j, \\ R_{ji}^*, & \text{if } i > j, \end{cases}$$

  where $|r| \le 1$. Matrix $\mathbf{T}$ is constructed analogously. Performing Cholesky-like decompositions $\mathbf{R} = \mathbf{P}\mathbf{P}^*$ and $\mathbf{T} = \mathbf{Q}\mathbf{Q}^*$, the correlated channel matrix is given by $\mathbf{H} = \mathbf{P}\widetilde{\mathbf{H}}\mathbf{Q}$. We set $r = 0.2$ and $\mathbf{R} = \mathbf{T}$ in our experiments.

The transmitted signal $\mathbf{w}^*$ is generated element-wise from a uniform distribution over the quaternary phase-shift keying (QPSK) constellation (see [29] for details). The noise variance ϱ^2 is determined by the signal-to-noise ratio (SNR), defined as

$$\text{SNR} = \mathbb{E}\|\mathbf{H}\mathbf{w}^*\|^2 / \mathbb{E}\|\boldsymbol{\varepsilon}\|^2.$$

a) Benchmark methods. We evaluate APPA against MEPM, L2ADMM, HOTML [29], and two classical detectors: zero-forcing (ZF) and minimum-mean-square-error decision-feedback (MMSE-DF). A theoretical lower bound (LB) assuming no inter-signal interference is computed following [29]. The hyperparameters for all algorithms are summarized as follows. For APPA, we set $(\eta, \lambda, \pi, k_0) = (2, 0.25, 1.25, 10)$ with initial penalty $\lambda_0 = 0.01\|\mathbf{b}^\top \mathbf{A}\|_\infty$; other parameters follow Section 4.1. MEPM and L2ADMM use identical settings to Section 4.1. For HOTML, we choose $\sigma_0 = 0.5$ and $\lambda_0 = 0.001$, with termination criteria: either $k > 200$ iterations or $\|\lambda_k - \lambda_{k-1}\| \leq 10^{-4}$. To ensure fair comparison, all algorithms start from $\mathbf{x}^0 = \mathbf{0}$. We assess detection accuracy through the bit error rate (BER), defined as

$$\text{BER} = \frac{|\{i \in [2n] : x_i \neq x_i^*\}|}{2n},$$

where $|\Omega|$ is the cardinality of Ω. A smaller BER corresponds to more accurate signal recovery.

b) Numerical comparison. We evaluate performance on critically determined systems ($m = n$) with $n \in \{4000, 6000, 8000, 10000\}$. The BER versus SNR curves for i.i.d. and correlated channels are shown in Figures 5 and 6, respectively, where each data point represents the average of 20 trials. In the i.i.d. channel scenario, APPA delivers the best BER performance, with L2ADMM and HOTML ranking second and third, while ZF exhibits the poorest accuracy. Under correlated channels, APPA consistently achieves the minimum BER at most SNR levels, with HOTML as the second-best performer. Notably, L2ADMM fails to produce reliable results in the correlated setting. The corresponding computational time is reported in Table 4 for problem sizes $n = 10^3$ and $n = 10^4$. APPA exhibits superior speed across all test cases. As an illustrative example, for the correlated channel with $n = 10^4$ and SNR $= 12$ dB, APPA requires only 13.32 seconds, while the other three methods each require over 200 seconds.

4.2.2 One-bit MIMO detection

The one-bit MIMO detection [29] aims to recover the transmitted binary signal $\mathbf{z} \in \{-1, 1\}^n$ from one-bit quantized observations $\mathbf{y} \in \{-1, 1\}^m$ according to the measurement model

$$\mathbf{y} = \text{sgn}(\mathbf{H}\mathbf{z} + \mathbf{v}), \tag{4.1}$$

where $\text{sgn}(\cdot)$ denotes the element-wise sign function that maps each component to $\{-1, 1\}$, $\mathbf{H} \in \mathbb{R}^{m\times n}$ represents the known channel matrix, and $\mathbf{v} \in \mathbb{R}^m$ is additive Gaussian noise with i.i.d. entries drawn from $\mathcal{N}(0, \varrho^2)$. The noise variance ϱ^2 characterizes the quality of the communication channel. Under this observation model, the maximum-likelihood detector takes the form

$$\min_{\mathbf{z}\in\{-1,1\}^n} \quad -\sum_{i=1}^{m} \log \Phi\left(\frac{y_i \langle \mathbf{h}_i, \mathbf{z}\rangle}{\varrho}\right), \tag{4.2}$$

where $\Phi(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-\tau^2/2} d\tau$ is the cumulative distribution function (CDF) of the standard normal distribution, and $\mathbf{h}_i$ denotes the i-th row of $\mathbf{H}$. To reformulate problem (4.2) into the standard form (UBIP), we perform the variable transformation $\mathbf{x} = (\mathbf{z} + \mathbf{1})/2$, which maps the bipolar constraint set $\{-1, 1\}^n$ to the binary set $\{0, 1\}^n$.

a) Benchmark methods. The benchmark algorithms include the zero-forcing (ZF) detector, MEPM, L2ADMM, and HOTML. For APPA, we set $(\eta, \lambda, \pi, k_0) = (0.1, 0.5, 1.2, 10)$, $\theta = \|\mathbf{H}\|_\infty +$

Table 4: Average CPU time (in seconds) for classical MIMO detection problems, where A1-A4 stand for APPA, MEPM, L2ADMM, and HOTML, respectively.

	I.I.D channels				Correlated channels			
SNR	A1	A2	A3	A4	A1	A2	A3	A4
				$n = 4000$				
2	2.496	28.09	38.18	26.08	2.478	27.70	36.72	25.85
4	2.498	28.17	38.22	26.12	2.625	28.42	34.68	25.86
6	2.437	28.04	38.38	26.18	2.472	27.29	34.46	25.87
8	2.289	27.94	38.47	26.40	2.354	27.99	34.39	25.85
10	1.716	27.98	38.38	26.12	1.988	27.93	34.44	25.84
12	1.763	27.48	33.89	26.42	1.911	27.72	35.24	25.92
				$n = 6000$				
2	5.760	68.48	92.39	103.4	5.926	70.58	91.65	101.5
4	5.852	68.74	92.60	103.3	5.856	70.24	86.82	101.5
6	5.728	68.24	92.21	103.4	5.617	69.64	84.01	101.5
8	5.532	68.21	92.66	103.8	5.599	70.35	84.27	101.5
10	4.027	68.33	92.45	103.7	4.794	68.53	85.37	101.6
12	3.823	68.08	83.09	103.8	3.976	69.68	86.70	101.5
				$n = 8000$				
2	12.25	134.1	180.7	251.6	12.36	131.9	177.5	261.9
4	12.81	134.2	180.9	251.3	12.16	132.4	157.5	261.9
6	12.34	134.1	181.8	251.3	12.09	129.9	157.3	261.8
8	10.48	134.5	182.5	251.7	10.65	130.7	156.9	262.0
10	8.994	133.0	179.9	253.1	8.716	127.6	159.7	261.8
12	8.657	133.7	174.4	253.1	7.189	128.4	164.6	262.0
				$n = 10000$				
2	19.76	204.4	274.7	496.4	17.19	205.4	290.1	509.4
4	19.31	204.2	275.3	496.7	17.84	204.6	263.0	509.6
6	19.87	205.0	275.3	496.2	18.37	202.5	246.4	498.9
8	16.59	204.4	277.8	496.5	15.65	206.2	244.0	509.1
10	13.63	204.8	276.6	497.0	13.00	206.2	258.9	509.1
12	13.32	200.9	265.0	497.1	12.29	200.5	251.4	509.0

$\|\mathbf{y}\|_\infty$, and $\lambda_0 = 0.005\|\mathbf{y}^\top \mathbf{H}\|_\infty$. For MEPM and L2ADMM, we set $L = \|\mathbf{H}\|_F^2/n$. HOTML maintains the parameter settings used in the classical MIMO case. Its maximum number of iterations is set to 300.

b) Numerical comparison. We first investigate how the measurement-to-dimension ratio m/n affects detection performance. Fixing SNR = 15 dB, we vary $m/n \in \{1, 1.5, \ldots, 3\}$ and test different problem dimensions. Figure 7 presents the BER averaged over 20 independent trials. The results show that APPA consistently delivers the lowest BER across all tested configurations.

Next, we examine performance under varying noise levels by fixing $m/n = 2$ and testing SNR $\in$

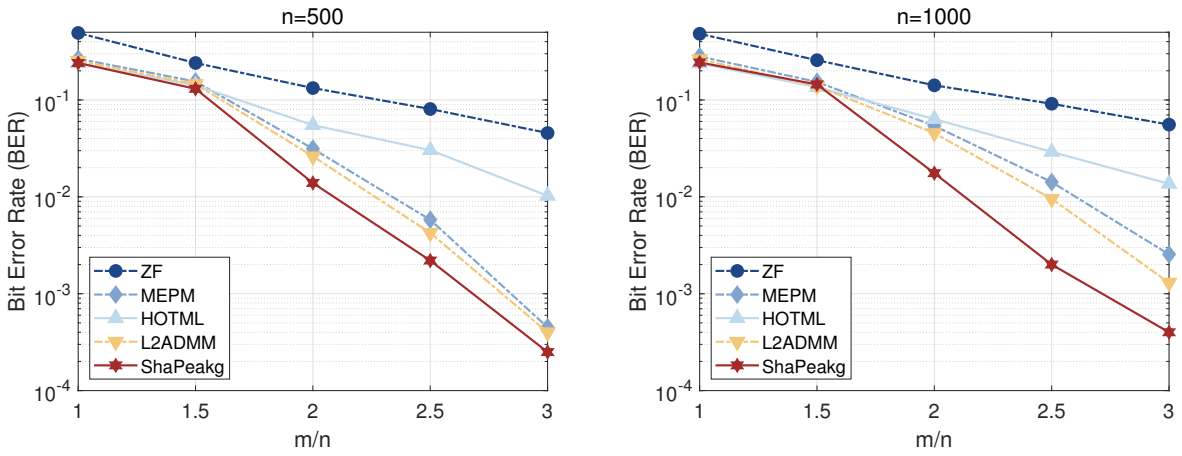


Figure 7: Effect of m/n for one-bit MIMO detection problems.

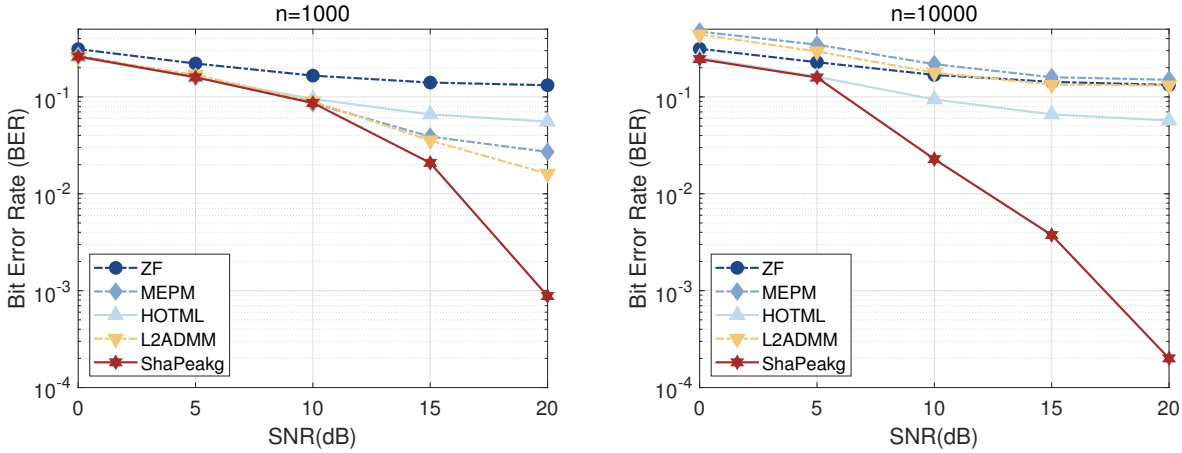


Figure 8: Effect of SNR for one-bit MIMO detection problems.

Table 5: Average CPU time (in seconds) for one-bit MIMO detection problems, where A1-A4 stand for APPA, MEPM, L2ADMM, and HOTML, respectively.

	$m = 1000, n = 500$				$m = 10000, n = 5000$			
SNR	A1	A2	A3	A4	A1	A2	A3	A4
0	0.024	0.609	0.856	0.024	1.356	142.9	188.4	4.021
5	0.030	0.649	0.912	0.033	1.547	143.4	196.8	6.502
10	0.081	0.673	0.934	0.069	4.399	139.8	196.3	13.88
15	0.105	0.678	0.994	0.088	7.287	144.0	195.7	14.28
20	0.120	0.693	1.017	0.087	8.919	143.0	197.2	14.45

$\{0, 5, \ldots, 20\}$. The average BER over 20 runs is displayed in Figure 8. At low SNR levels (SNR $\leq$ 10 dB), all algorithms exhibit poor performance, failing to reliably recover the transmitted signal. However, as SNR increases beyond 10 dB, APPA begins to outperform competing methods, with its superiority becoming increasingly evident at higher SNR values. Table 5 reports the runtime comparison. APPA demonstrates substantial computational advantages across all scenarios. For example, in the case of $n = 5000$ and SNR $= 20$ dB, APPA completes in approximately 9 seconds, whereas L2ADMM requires nearly 200 seconds.

4.3 QUBO

The QUBO problem takes the following form,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ \frac{1}{2}\langle \mathbf{x}, \mathbf{Q}\mathbf{x}\rangle, \quad \text{s.t.} \ \ \mathbf{x}\in\{0,1\}^n,$$

where $\mathbf{Q}\in\mathbb{R}^n$ is a symmetric matrix. We consider two types of test instances for evaluating algorithmic performance on QUBO problems. The first corresponds standard benchmark problems from the Beasley instances [34], which are widely used for evaluating QUBO solvers. For this instance, we test three problem sizes: dimensions $n\in\{100, 250, 500\}$. Each size category contains 10 instances with densities ranging from 0.2 to 0.8. The second corresponds to synthetically generated instances following [6], where $\mathbf{Q}$ is constructed with density 0.8 and nonzero entries uniformly distributed over $[10, 100]$ when $n < 10^4$, and with density 0.005 when $n \geq 10^4$.

The algorithms benchmarked in this experiment include MEPM, L2ADMM, and GUROBI. The hyperparameters are given as follows. For APPA, we set $(\eta, \sigma, \lambda, \pi, k_0) = (1, 10^{-8}, 0.25, 1.5, 100)$, $\theta = \|\mathbf{Q}\|_\infty$ and $\lambda_0 = 0.001\|\mathbf{Q}\|_F$. For MEPM and L2ADMM, we set $L = 1/n\|\mathbf{Q}\|_F^2$, while other parameters remain consistent with those used in previous experiments. Default settings are adopted for GUROBI. To evaluate performance, we compute the relative optimality gap by

$$\text{Gap} = \frac{|\text{obj} - \text{lowest}|}{|\text{lowest}|} \times 100\%,$$

where 'obj' denotes the objective value obtained by a specific algorithm, and 'lowest' denotes the the latest best-known values (for Beasley benchmarks) or lowest objective value achieved among all compared methods (for synthetic instances).

a) Comparisons on Beasley instances. Table 6 presents the results on Beasley benchmark instances. In terms of solution quality, APPA achieves gaps second only to GUROBI across most instances, and even attains the optimal solution in several cases (e.g., bqp100-1, bqp100-10, bqp250-7). Regarding computational efficiency, APPA maintains the shortest computational time in the majority of instances, while GUROBI's runtime reaches 600 seconds for all problems with $n \geq 250$.

b) Comparisons on synthetic instances. We first evaluate performance on small-scale problems, with results over 20 independent runs presented in Table 7. Here, 'best' and 'mean' denote the minimum and average optimality gaps, respectively, among the 20 trials. Across all test cases, GUROBI achieves the smallest gap (i.e., the best objective values), with APPA as the second-best performer. However, APPA demonstrates a significant computational advantage, requiring substantially less time than competing methods.

For large-scale experiments with $n\in\{80000, 100000\}$, we exclude GUROBI due to its excessive runtime requirements. Here, cases 1-5 represent five different proportions of nonnegative entries in $\mathbf{Q}$: $\{0.4995, 0.4999, 0.49995, 0.49999, 0.5\}$, respectively. Table 8 shows that APPA consistently obtains lower objective values and dramatically faster computation times compared to MEPM and L2ADMM. For example, in case 5 with $n = 10^5$, APPA completes in approximately 80 seconds, while MEPM and L2ADMM require 857 and 1083 seconds.

Table 6: Results on Beasley benchmark instances

	APPA		MEPM		L2ADMM		GUROBI	
Instance	gap	time	gap	time	gap	time	gap	time
bqp100-1	0.00	0.015	2.31	0.053	2.08	0.082	0.00	0.420
bqp100-2	0.43	0.004	0.45	0.037	0.45	0.047	0.00	0.337
bqp100-3	0.28	0.005	0.28	0.034	0.28	0.051	0.28	0.474
bqp100-4	0.46	0.004	0.60	0.035	0.60	0.034	0.46	0.422
bqp100-5	2.37	0.004	0.74	0.036	0.74	0.050	0.00	0.398
bqp100-6	1.68	0.071	1.97	0.046	1.97	0.054	0.90	0.499
bqp100-7	1.15	0.004	1.47	0.048	1.31	0.070	0.00	0.405
bqp100-8	1.03	0.054	1.63	0.055	1.63	0.066	0.00	0.326
bqp100-9	0.10	0.003	0.10	0.034	0.10	0.029	0.00	0.330
bqp100-10	0.00	0.003	0.14	0.044	0.14	0.057	0.00	0.327
bqp250-1	0.62	0.046	0.90	0.079	0.88	0.119	0.00	600.0
bqp250-2	0.85	0.003	1.58	0.054	1.58	0.052	0.00	600.0
bqp250-3	0.24	0.040	0.25	0.075	0.25	0.092	0.00	600.0
bqp250-4	0.38	0.002	0.39	0.066	0.39	0.073	0.00	600.0
bqp250-5	0.38	0.006	0.47	0.062	0.47	0.091	0.20	600.0
bqp250-6	0.28	0.003	0.62	0.075	0.62	0.084	0.10	600.0
bqp250-7	0.00	0.002	0.11	0.079	0.11	0.112	0.00	600.0
bqp250-8	4.11	0.044	3.09	0.078	2.76	0.116	0.00	600.0
bqp250-9	0.56	0.021	1.43	0.083	1.29	0.128	0.03	600.0
bqp250-10	0.21	0.003	0.40	0.075	0.40	0.081	0.00	600.0
bqp500-1	1.44	0.005	2.25	0.185	1.78	0.252	0.06	600.0
bqp500-2	0.25	0.009	0.51	0.192	0.45	0.270	0.00	600.0
bqp500-3	0.22	0.006	0.59	0.156	0.45	0.218	0.00	600.0
bqp500-4	0.23	0.006	0.39	0.150	0.39	0.215	0.00	600.0
bqp500-5	0.86	0.005	1.27	0.150	1.23	0.228	0.12	600.0
bqp500-6	0.54	0.005	2.38	0.165	2.27	0.233	0.00	600.0
bqp500-7	0.81	0.018	1.85	0.171	1.70	0.244	0.08	600.0
bqp500-8	0.52	0.009	1.26	0.166	1.26	0.265	0.00	600.0
bqp500-9	0.51	0.007	1.14	0.161	1.06	0.225	0.00	600.0
bqp500-10	1.06	0.024	0.69	0.155	0.69	0.210	0.00	600.0

5 Conclusion

The proposed piecewise cubic function provides an exact continuous reformulation of binary constraints, transforming the combinatorially hard problem into a tractable continuous optimization framework. The key contribution is threefold: exact penalty theory with a solution-independent threshold, P-stationary points for optimality characterization, and the APPA algorithm with finite-iteration convergence guarantee. Extensive numerical experiments demonstrate that APPA achieves superior performance compared to other solvers across diverse problems. Future work includes the extension to constrained binary optimization and exploring applications to other applications.

Table 7: Gap and time (in seconds) for low-dimensional instances

	APPA			MEPM			L2ADMM			GUROBI		
n	best	mean	time	best	mean	time	best	mean	time	best	mean	time
1000	0.39	0.80	0.057	2.17	2.82	0.352	1.92	2.39	0.462	0.00	0.00	600.0
3000	0.49	0.59	0.228	2.97	3.74	3.089	2.68	3.49	3.742	0.00	0.00	600.0
5000	0.39	0.53	1.041	3.86	4.56	22.05	3.42	4.15	25.73	0.00	0.00	600.0
7000	0.30	0.39	2.691	4.56	4.97	47.86	4.04	4.53	57.33	0.00	0.00	600.0
10000	0.28	0.37	6.082	5.43	5.76	112.3	4.88	5.32	129.8	0.00	0.00	600.0

Table 8: Gap and time (in seconds) for higher dimensional instances

		APPA			MEPM			L2ADMM		
n	case	best	mean	time	best	mean	time	best	mean	time
	1	0.00	0.00	47.50	1.53	1.63	613.7	1.29	1.36	768.5
	2	0.00	0.00	49.38	1.56	1.65	613.5	1.29	1.37	780.8
8e4	3	0.00	0.00	44.67	1.58	1.64	605.8	1.30	1.36	770.4
	4	0.00	0.00	52.11	1.66	1.69	613.8	1.36	1.40	760.3
	5	0.00	0.00	43.05	1.63	1.67	607.7	1.36	1.39	765.4
	1	0.00	0.00	87.10	1.79	1.80	901.5	1.49	1.51	1133.8
	2	0.00	0.00	67.87	1.69	1.80	899.8	1.69	1.50	1138.7
1e5	3	0.00	0.00	75.53	1.78	1.82	898.8	1.46	1.51	1125.5
	4	0.00	0.00	77.02	1.76	1.80	873.3	1.47	1.50	1109.1
	5	0.00	0.00	76.69	1.78	1.80	857.6	1.50	1.51	1083.6

References

[1] M. F. Anjos and J. B. Lasserre. *Handbook on semidefinite, conic and polynomial optimization*, volume 166. Springer Science & Business Media, 2011.

[2] E. Baş. Binary aquila optimizer for 0–1 knapsack problems. *Engineering Applications of Artificial Intelligence*, 118:105592, 2023.

[3] C. Buchheim, M. Montenegro, and A. Wiegele. SDP-based branch-and-bound for non-convex quadratic integer optimization. *Journal of Global Optimization*, 73:485–514, 2019.

[4] C. Buchheim and A. Wiegele. Semidefinite relaxations for non-convex quadratic mixed-integer programming. *Mathematical Programming*, 141:435–452, 2013.

[5] S. Burer, R. D. Monteiro, and Y. Zhang. Rank-two relaxation heuristics for max-cut and other binary quadratic programs. *SIAM Journal on Optimization*, 12(2):503–521, 2002.

[6] C. Chen, R. Chen, T. Li, R. Ao, and Z. Wen. Monte carlo policy gradient method for binary optimization, 2023.

[7] T. Cour and J. Shi. Solving markov random fields with spectral relaxation. In *Artificial Intelligence and Statistics*, pages 75–82. PMLR, 2007.

[8] M. De Santis and F. Rinaldi. Continuous reformulations for zero–one programming problems. *Journal of Optimization Theory and Applications*, 153:75–84, 2012.

[9] M. Esmaeilpour, P. Cardinal, and A. L. Koerich. A robust approach for securing audio classification against adversarial attacks. *IEEE Transactions on Information Forensics and Security*, 15:2147–2159, 2019.

[10] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 29. wh freeman New York, 2002.

[11] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 2000.

[12] C.-J. Hsieh, N. Natarajan, and I. Dhillon. PU learning for matrix completion. In *International Conference on Machine Learning*, pages 2445–2453. PMLR, 2015.

[13] Q. Huang, Y. Chen, and L. Guibas. Scalable semidefinite relaxation for maximum a posterior estimation. In *International Conference on Machine Learning*, pages 64–72. PMLR, 2014.

[14] T. Joachims et al. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, volume 99, pages 200–209, 1999.

[15] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang. The unconstrained binary quadratic programming problem: a survey. *Journal of combinatorial optimization*, 28(1):58–81, 2014.

[16] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang. The unconstrained binary quadratic programming problem: A survey. *Journal of Combinatorial Optimization*, 28:58–81, 2014.

[17] N. Komodakis and G. Tziritas. Approximate labeling via graph cuts based on linear programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1436–1453, 2007.

[18] J. B. Lasserre. A max-cut formulation of 0/1 programs. *Operations Research Letters*, 44(2):158–164, 2016.

[19] H. Liu, K. Deng, H. Liu, and Z. Wen. An entropy-regularized ADMM for binary quadratic programming. *Journal of Global Optimization*, 87(2):447–479, 2023.

[20] S. L. Loyka. Channel capacity of MIMO architecture using the exponential correlation matrix. *IEEE Communications letters*, 5(9):369–371, 2001.

[21] R. Merkle and M. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions on Information Theory*, 24(5):525–530, 1978.

[22] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

[23] W. Murray and K.-M. Ng. An algorithm for nonlinear optimization problems with binary variables. *Computational Optimization and Applications*, 47(2):257–288, 2010.

[24] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer, 1999.

[25] J.-S. Pan, P. Hu, V. Snášel, and S.-C. Chu. A survey on binary metaheuristic algorithms and their engineering applications. *Artificial Intelligence Review*, 56(7):6101–6167, 2023.

[26] H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe. Binary neural networks: A survey. *Pattern Recognition*, 105:107281, 2020.

[27] M. Raghavachari. On connections between zero-one integer programming and concave programming under linear constraints. *Operations Research*, 17(4):680–684, 1969.

[28] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[29] M. Shao and W.-K. Ma. Binary MIMO detection via homotopy optimization and its deep adaptation. *IEEE Transactions on Signal Processing*, 69:781–796, 2020.

[30] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[31] P. Wang, C. Shen, A. van den Hengel, and P. H. Torr. Large-scale binary quadratic optimization using semidefinite relaxation and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(3):470–485, 2016.

[32] S. Weerasinghe, T. Alpcan, S. M. Erfani, and C. Leckie. Defending support vector machines against data poisoning attacks. *IEEE Transactions on Information Forensics and Security*, 16:2566–2578, 2021.

[33] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2(3):203–230, 2010.

[34] A. Wiegele. Biq mac library—a collection of max-cut and quadratic 0-1 programming instances of medium size. *Preprint*, 51:112–127, 2007.

[35] H. Wolkowicz, R. Saigal, and L. Vandenberghe. *Handbook of semidefinite programming: theory, algorithms, and applications*, volume 27. Springer Science & Business Media, 2012.

[36] B. Wu and B. Ghanem. ℓ_p-box ADMM: A versatile framework for integer programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1695–1708, 2018.

[37] X. Yang, Z. Song, I. King, and Z. Xu. A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(9):8934–8954, 2022.

[38] G. Yuan and B. Ghanem. Sparsity constrained minimization via mathematical programming with equilibrium constraints. *arXiv preprint arXiv:1608.04430*, 2016.

[39] G. Yuan and B. Ghanem. An exact penalty method for binary optimization based on MPEC formulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[40] Z. Zhang, T. Li, C. Ding, and X. Zhang. Binary matrix factorization with applications. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 391–400. IEEE, 2007.