

Article

Exact-Penalty Prox-Linear Methods for Bilevel Optimization with ℓ_1 Lower-Level Gradient Penalty

Yutong Zheng ¹, Jiani Li ¹ and Qingna Li ^{1,2,*} 

¹ School of Mathematics and Statistics, Beijing Institute of Technology, Beijing 100081, China; 3120205698@bit.edu.cn (Y.Z.); lijiani@bit.edu.cn (J.L.)

² Beijing Key Laboratory on MCAACI, Beijing 100081, China

* Correspondence: qnl@bit.edu.cn

Abstract

Bilevel optimization is a fundamental framework for hierarchical decision-making, but its solution is challenging due to the implicit and typically set-valued nature of the lower-level optimality condition. In this paper, we study bilevel optimization problems through an exact-penalty reformulation based on the ℓ_1 -norm of the lower-level gradient. Under suitable regularity assumptions, we show that this penalty defines a distance-bound function and yields an exact penalty property for sufficiently large penalty parameters. To solve each fixed-penalty problem, we apply a prox-linear procedure that keeps the nonsmooth ℓ_1 penalty in its original form and linearizes the smooth mappings. We prove a stationarity-oriented convergence guarantee for the fixed-penalty prox-linear loop. For the unconstrained simple bilevel setting, the prox-linear subproblem admits explicit dual reformulation as a box-constrained quadratic program. This dual structure enables the use of a nonmonotone spectral projected gradient method together with a closed-form primal recovery formula. Numerical experiments on the Minimum Norm Solution Problem show that the proposed method consistently achieves lower-level feasibility and upper-level accuracy, and attains a higher success rate than several existing methods on the tested instances. A Lipschitz least-squares variant is further included to provide a numerical illustration of the global exactness theorem.

Keywords: bilevel optimization; exact penalty; prox-linear method; duality; spectral projected gradient

MSC: 90C30; 90C46; 65K05



Academic Editors: Cong Sun and
Gradimir V. Milovanović

Received: 31 May 2026

Revised: 3 July 2026

Accepted: 7 July 2026

Published: 8 July 2026

Copyright: © 2026 by the authors.
Licensee MDPI, Basel, Switzerland.
This article is an open access article
distributed under the terms and
conditions of the [Creative Commons
Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Bilevel optimization concerns hierarchical decision problems in which the feasible set of an upper-level problem is determined by the solution set of a lower-level optimization problem. Owing to this nested structure, bilevel models arise in a broad range of areas, including hyperparameter optimization, meta-learning, signal and image processing [1–3].

Bilevel optimization is a classical topic in mathematical programming. In machine learning, bilevel formulations now play a central role in hyperparameter optimization, meta-learning, neural architecture search, and other hierarchical learning pipelines. Representative examples include the bilevel framework for hyperparameter selection and meta-learning [3], truncated back-propagation through lower-level dynamics [4], implicit-gradient-based meta-learning [5], the generic first-order framework beyond the lower-level singleton assumption [6], and fully first-order stochastic bilevel methods such as [7].

For nonsmooth lower-level learning models, Ochs et al. developed early gradient-based techniques that have had considerable influence on later bilevel optimization methods [8]. Alongside these general-purpose machine-learning-oriented bilevel methods, another line of work studies problem-specific reformulations, especially for support vector classification and related learning models. For hyperparameter selection in ℓ_1 -loss support vector classification, Li, Li, and Zemkoho [9] formulated the cross-validation problem as a bilevel program, converted it into a mathematical program with equilibrium constraints (MPEC), and proposed a global-relaxation-based solution method. This MPEC perspective was further developed in [10], where several classical constraint qualifications were analyzed in the context of bilevel hyperparameter optimization for machine learning. For multiple hyperparameter selection with feature selection in support vector classification, Qian, Li, and Zemkoho [11] proposed a global relaxation-based linear programming–Newton method. For support vector classification with logistic loss, Wang and Li [12] transformed the bilevel model into a Karush–Kuhn–Tucker (KKT)-based nonlinear program and developed a fast smoothing Newton method with superlinear local convergence. For large-scale MPECs arising from hyperparameter selection in ℓ_1 -support vector classification, Wang, Li, and Zhang [13] proposed a smoothing damped Newton method that directly exploits problem structure. Beyond hyperparameter selection, Zheng and Li [14] studied bilevel models for adversarial learning and analyzed a case study in convex clustering. These works further illustrate the breadth of bilevel optimization in machine learning, while also showing that many successful approaches are strongly tailored to specific application structures and KKT/MPEC reformulations.

Penalty methods provide another important route for handling bilevel structure without explicitly differentiating through the lower-level solution map. A classical reference is the work of Ye, Zhu, and Zhu [15], which established exact penalization and necessary optimality conditions for generalized bilevel programming. In the modern first-order literature, Lu and Mei [16] studied penalty methods for a broad class of unconstrained and constrained bilevel problems by solving structured minimax subproblems. Shen, Xiao, and Chen [17] developed a penalty-based bilevel gradient descent method for constrained bilevel problems without lower-level strong convexity. In the simple bilevel setting, Chen et al. [18] further connected approximate solutions of the original problem and its penalized reformulation under Hölderian error bounds.

For simple bilevel optimization, a substantial body of work has focused on first-order algorithms tailored to convex and structured settings. Classical approaches include explicit descent methods [19], minimal-norm selection methods [20], and ϵ -subgradient schemes [21]. Subsequent developments introduced projected and incremental regularization strategies [22], methods for variational-inequality-constrained formulations [23], and inertial or fixed-point type algorithms [24,25]. More recently, the algorithmic scope has been broadened to accommodate nonsmooth outer objectives and more general first-order frameworks. For example, Doron and Shtern [26] proposed the Iterative Approximation and Level-set EXpansion (ITALEX) methodology, and Shen, Ho-Nguyen, and Kılınç-Karzan [27] developed an online-convex-optimization-based framework. In the convex simple bilevel setting with a nonsmooth outer objective, Merchav and Sabach [28] proposed the Bi-Sub-Gradient (Bi-SG) method, which combines a proximal-gradient step for the inner convex composite problem with a subgradient or proximal-gradient step for the outer objective.

A particularly active recent line of research studies non-asymptotic guarantees for convex simple bilevel optimization under weaker structural assumptions. Jiang et al. [29] proposed a conditional-gradient method for simple bilevel problems with convex lower-level structure, while Cao et al. [30] developed an accelerated method for the convex smooth case. Wang,

Shi, and Jiang [31] developed a bisection-type framework and established near-optimal complexity guarantees for convex simple bilevel optimization. Zhang et al. [32] studied a functionally constrained reformulation and derived near-optimal guarantees under standard convex smoothness assumptions. Giang-Tran, Ho-Nguyen, and Lee [33] proposed a projection-free conditional-gradient method with simultaneous inner and outer convergence guarantees.

Recent developments in computational intelligence and nonsmooth optimization also provide useful context for the present work. Structured Jacobian, Hessian, and matrix-inversion computations appear in many optimization-related algorithms, such as the fixed-time zeroing neural network in [34]. Nonsmoothness can also arise directly from modeling, for example, in nonsmooth cancer-treatment models [35]. Related concerns on large-scale efficiency and learning-based optimization have also been studied in sparse multiobjective optimization [36] and compiler tuning [37]. Although these studies address different models and algorithmic settings, they highlight related issues of nonsmoothness and computational efficiency, which are also relevant to the development and implementation of our method.

In this paper, we consider a bilevel optimization problem in which the upper-level decision variable is constrained by the solution set of a lower-level optimization problem. Rather than enforcing the lower-level optimality condition explicitly, we measure its violation by the ℓ_1 -norm of the lower-level gradient and incorporate this quantity into the upper-level objective as an exact-penalty term. This leads to a nonsmooth single-level penalized formulation.

Our approach is related to the penalty-based bilevel gradient descent method of Shen, Xiao, and Chen [17], since both approaches use lower-level gradient information to construct a penalized single-level problem and both exploit a prox-linear type model for the resulting nonsmooth objective. However, the focus of the present paper is different. The work [17] develops a general penalty-based first-order framework for bilevel optimization, whereas our main algorithmic contribution is to exploit the special ℓ_1 lower-level gradient penalty in the unconstrained simple bilevel setting. In this case, after linearizing the lower-level gradient, the prox-linear subproblem has a particular structure. We show that this structure admits an explicit dual reformulation as a box-constrained quadratic program, together with the closed-form primal recovery formula. This dual structure comes from the specific ℓ_1 norm-gradient penalty, rather than from the generic prox-linear framework alone. It allows the dual subproblem to be solved efficiently by a nonmonotone spectral projected gradient (SPG) method. Hence, the main novelty of our method is the exploitation of this ℓ_1 -induced dual quadratic structure and the associated primal recovery formula in the unconstrained simple bilevel setting.

The main contributions of this paper are summarized as follows.

- We study a norm-gradient exact-penalty reformulation for bilevel optimization based on the ℓ_1 -norm of the lower-level gradient. Under suitable regularity assumptions, we establish global exactness for sufficiently large penalty parameters. We also state a local exactness result under corresponding local assumptions.
- For the resulting nonsmooth penalized problem, we apply a fixed-penalty prox-linear procedure motivated by [17]. The method linearizes the smooth upper-level objective and the lower-level gradient mapping, while keeping the ℓ_1 penalty in its nonsmooth form. For a fixed penalty parameter, we establish a stationarity-oriented convergence guarantee for the prox-linear loop with inexact subproblem solutions.
- We specialize the approach to the unconstrained simple bilevel setting. In this case, the prox-linear subproblem admits an explicit Fenchel dual reformulation as a box-

constrained quadratic program. We also derive the projected fixed-point optimality condition of the dual problem and the closed-form primal recovery formula.

- Based on the dual box structure, we employ a nonmonotone spectral projected gradient method as the inner solver. Numerical experiments on YearPredictionMSD subsamples show that the method is effective and competitive on the tested bilevel least-squares instances.

The rest of the paper is organized as follows. Section 2 introduces the ℓ_1 norm-gradient penalty and establishes its exactness under suitable assumptions. Section 3 presents the fixed-penalty prox-linear procedure for the nonsmooth penalized problem and analyzes its stationarity-oriented convergence for a fixed penalty parameter. Section 4 specializes the method to the simple bilevel problem. Section 5 derives the dual reformulation of the unconstrained simple-bilevel prox-linear subproblem and presents the nonmonotone SPG solver. Section 6 reports numerical results on the YearPredictionMSD minimum-norm least-squares problem and on a Lipschitz least-squares variant. Finally, Section 7 concludes the paper.

Notations. We use $\|\cdot\|$ to denote the ℓ_2 norm. Given a nonempty closed set $\mathcal{S} \subseteq \mathbb{R}^d$, define the distance of $y \in \mathbb{R}^d$ to the set $\mathcal{S}$ by $d_{\mathcal{S}}(y) := \min_{y' \in \mathcal{S}} \|y - y'\|$. The notation $\langle a, b \rangle$ denotes the Euclidean inner product.

2. Exact Penalty Reformulation of Bilevel Problems

This section studies the relation between the solutions of the bilevel problem (1) and those of the norm-gradient penalized problem (3) by imposing certain generic conditions.

Define $f : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \rightarrow \mathbb{R}$ and $g : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \rightarrow \mathbb{R}$. We consider the following bilevel problem:

$$\min_{x,y} f(x,y) \quad \text{s.t.} \quad x \in C, \quad y \in S(x), \quad (1)$$

where

$$S(x) := \arg \min_{u \in \mathbb{R}^{d_y}} g(x,u).$$

Here $C \subseteq \mathbb{R}^{d_x}$ is a nonempty closed convex set, and f, g are continuously differentiable. For every $x \in C$, the lower-level solution set $S(x)$ is assumed to be nonempty and closed.

Assumption 1. For every $x \in C$, the function $g(x, \cdot)$ is continuously differentiable on $\mathbb{R}^{d_y}$ and satisfies the Polyak–Łojasiewicz (PL) inequality with constant $1/\sqrt{\mu}$, namely,

$$\|\nabla_y g(x,y)\|_2^2 \geq \frac{1}{\sqrt{\mu}} (g(x,y) - v(x)), \quad \forall y \in \mathbb{R}^{d_y},$$

where $v(x) := \min_{u \in \mathbb{R}^{d_y}} g(x,u)$.

Under Assumption 1, the implicit lower-level optimality condition $y \in S(x)$ can be equivalently replaced by the lower-level gradient equation. Indeed, by Fermat's rule ([38], Theorem 10.1), every global minimizer of the unconstrained differentiable lower-level problem satisfies the stationarity condition $\nabla_y g(x,y) = 0$. Conversely, the PL condition implies that every stationary point is a global minimizer [39]. Hence, for every $x \in C$,

$$y \in S(x) \iff \nabla_y g(x,y) = 0.$$

Therefore, (1) is equivalent to the following gradient-based reformulation:

$$\min_{x,y} f(x,y) \quad \text{s.t.} \quad x \in C, \quad \nabla_y g(x,y) = 0. \quad (2)$$

A central difficulty in (1) is that the lower-level optimality condition $y \in S(x)$ is implicit and typically set-valued. Under Assumption 1, this condition can be equivalently represented by the gradient equation in (2). Nevertheless, (2) is still a nonlinear equality-constrained problem, and directly solving it would require handling the constraint $\nabla_y g(x, y) = 0$ and its associated Jacobian, which involves second-order information of the lower-level objective. To avoid enforcing this equality constraint explicitly, we adopt an exact-penalty reformulation. Motivated by the classical theory of nonsmooth exact penalties and recent penalty-based first-order methods for bilevel optimization [15–17], we penalize the gradient equation as follows

$$p(x, y) := \|\nabla_y g(x, y)\|_1.$$

For a penalty parameter $\gamma > 0$, we consider the norm-gradient penalized problem

$$\min_{x, y} F_\gamma(x, y) := f(x, y) + \gamma p(x, y) \quad \text{s.t.} \quad x \in C, \quad y \in \mathbb{R}^{d_y}. \quad (3)$$

To analyze the relation of (2) and (3), we need the following assumptions and definitions.

Definition 1. A function $p : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \rightarrow \mathbb{R}$ is a ρ -distance-bound if there exists $\rho > 0$ such that for any $x \in C$ and $y \in \mathbb{R}^{d_y}$, it holds

$$p(x, y) \geq 0, \quad d_{S(x)}(y) \leq \rho p(x, y), \quad (4)$$

$$p(x, y) = 0 \iff d_{S(x)}(y) = 0. \quad (5)$$

Assumption 2. There exists a constant $L > 0$ such that for any $x \in C$, the mapping $y \mapsto f(x, y)$ is L -Lipschitz continuous on $\mathbb{R}^{d_y}$, i.e.,

$$|f(x, y) - f(x, y')| \leq L\|y - y'\|, \quad \forall y, y' \in \mathbb{R}^{d_y}.$$

Lemma 1. Suppose Assumption 1 holds, and suppose that $g(x, \cdot)$ is Lipschitz-smooth for any $x \in C$. Then

$$p(x, y) := \|\nabla_y g(x, y)\|_1$$

is a ρ -distance-bound function with $\rho := 2\sqrt{\mu}$.

Proof. See Appendix A.1. $\square$

Remark 1. The penalty $p(x, y) := \|\nabla_y g(x, y)\|_1$ has both modeling and computational motivations. It measures the violation of the gradient reformulation (2), and under Assumption 1, $p(x, y) = 0$ is equivalent to $y \in S(x)$. In addition, the dual representation of the ℓ_1 -norm leads, after linearization, to a box-constrained quadratic dual subproblem in the unconstrained simple bilevel case. This structure is exploited in the dual SPG solver developed later.

Now we can prove the exactness of the penalized problem (3). The following result clarifies the relation between global minimizers of (1) and (3).

Theorem 1. Suppose Assumptions 1 and 2 hold, $g(x, \cdot)$ is Lipschitz-smooth for any $x \in C$, and $p(x, y) = \|\nabla_y g(x, y)\|_1$ satisfies Lemma 1 with $\rho = 2\sqrt{\mu}$. Then, for any $\gamma > L\rho = 2L\sqrt{\mu}$, the global solutions of (3) are the global solutions of (1).

Proof. By Lemma 1, $p(x, y)$ is a ρ -distance-bound function. Hence, for the general exact-penalty statement in [17], Theorem 8 applies with $\gamma^* = L\rho$. Since we consider exact

global minimizers of (3), we have $\epsilon_2 = 0$ in [17], Theorem 8. Therefore, the corresponding approximate accuracy satisfies

$$0 \leq \epsilon_\gamma \leq \frac{\epsilon_2}{\gamma - \gamma^*} = 0,$$

and hence $\epsilon_\gamma = 0$. Thus, the approximate problem reduces to the original bilevel problem (1). The result follows. $\square$

The following result gives the corresponding local exactness statement. It is stated separately because the local relation requires additional local conditions.

Theorem 2. Suppose Assumption 1 holds, $g(x, \cdot)$ is Lipschitz-smooth for any $x \in C$, and $p(x, y) = \|\nabla_y g(x, y)\|_1$ satisfies Lemma 1 with $\rho = 2\sqrt{\mu}$. Let (x_γ, y_γ) be a local solution of (3) on a neighborhood $N((x_\gamma, y_\gamma), r)$. Suppose that $f(x_\gamma, \cdot)$ is L -Lipschitz continuous on $N(y_\gamma, r)$. If $p(x_\gamma, \cdot)$ is continuous and convex on $\mathbb{R}^{d_y}$, then, for any $\gamma > L\rho = 2L\sqrt{\mu}$, (x_γ, y_γ) is a local solution of (1).

Proof. By Lemma 1, $p(x, y)$ is a ρ -distance-bound function. Hence, for the local exact-penalty statement in [17], Theorem 9 applies with $\gamma^* = L\rho$, since $f(x_\gamma, \cdot)$ is assumed to be L -Lipschitz continuous near y_γ , the present unconstrained lower-level setting gives $U(x_\gamma) = \mathbb{R}^{d_y}$, and $p(x_\gamma, \cdot)$ is assumed to be continuous and convex. The result follows. $\square$

We terminate the ℓ_1 penalty method by combining a feasibility residual for the gradient equation in (2) and a stationarity residual. Specifically, after obtaining (x^{k+1}, y^{k+1}) , we compute

$$r_f^{k+1} := \|\nabla_y g(x^{k+1}, y^{k+1})\|_1, \quad r_s^{k+1} := \frac{1}{\lambda_k} \|(x^{k+1}, y^{k+1}) - (x^k, y^k)\|_2, \quad (6)$$

where $\lambda_k > 0$ is the proximal stepsize used in the fixed-penalty subproblem at the k -th penalty stage. The first residual measures the violation of the equality constraint $\nabla_y g(x, y) = 0$ in (2), while the second residual measures the stationarity of the step. The algorithm stops once

$$r_f^{k+1} \leq \epsilon_f, \quad r_s^{k+1} \leq \epsilon_s. \quad (7)$$

If (7) is not satisfied, the penalty parameter is increased.

The details of the Classical ℓ_1 Penalty Method applied to the norm-gradient penalized problem (3) are given in Algorithm 1.

Algorithm 1 Classical ℓ_1 Penalty Method for (3).

1: **Input:** (x^0, y^0) and $\gamma_0 > 0, \tau > 1, \epsilon_f > 0, \epsilon_s > 0$.

2: Set $k := 0, r_f^0 := +\infty$, and $r_s^0 := +\infty$.

3: **while** $r_f^k > \epsilon_f$ or $r_s^k > \epsilon_s$, **do**

4: Solve

$$\min_{x \in C, y \in \mathbb{R}^{d_y}} F_{\gamma_k}(x, y) := f(x, y) + \gamma_k p(x, y). \quad (8)$$

to obtain (x^{k+1}, y^{k+1}) .

5: Set $\gamma_{k+1} := \tau\gamma_k$ and $k := k + 1$

6: **end while**

7: **Output:** $(\bar{x}, \bar{y}) = (x^k, y^k)$.

3. Solving the Penalized Problem

In this section, we develop an algorithmic framework for solving the penalized problem (3). Our goal is to show how the special structure of the exact-penalty objective can be exploited algorithmically. In particular, we adapt the prox-linear framework of [40] to our bilevel penalty formulation and then derive a problem-specific subproblem that will be further exploited in the later sections.

At the k -th penalty stage, the task is to solve the single-level nonsmooth problem (8). In principle, problem (8) can be handled by existing nonsmooth optimization methods, smoothing techniques, or nonlinear programming solvers after introducing auxiliary variables for the ℓ_1 term. However, such generic approaches either smooth the exact nonsmooth penalty or introduce additional nonlinear constraints involving $\nabla_y g(x, y)$. More importantly, they do not explicitly exploit the composite structure $f(x, y) + \gamma_k \|\nabla_y g(x, y)\|_1$. For this reason, we apply a prox-linear approach. The key idea is to linearize the smooth mappings f and $\nabla_y g$ at the current iterate while keeping the ℓ_1 norm in its original nonsmooth form. This yields a structured strongly convex subproblem and avoids introducing a smoothing parameter.

We recall the prox-linear idea inspired by [40]. The prox-linear method is designed for composite nonsmooth optimization problems of the form

$$\min_{z \in \mathcal{Z}} c_1(z) + c_3(c_2(z)), \quad (9)$$

where $\mathcal{Z}$ is a closed convex set, $c_1 : \mathbb{R}^m \rightarrow \mathbb{R}$ and $c_2 : \mathbb{R}^m \rightarrow \mathbb{R}^n$ are Lipschitz smooth, and $c_3 : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex but possibly nonsmooth. The idea of the prox-linear method is to linearize both $c_1(z)$ and $c_2(z)$ at a given iteration to solve the resulting subproblem.

To be specific, we specialize the framework of [40] to the norm-gradient penalized problem (3). For this problem, we denote $z = (x, y)$ and set

$$c_1(z) = f(x, y), \quad c_2(z) = \nabla_y g(x, y), \quad c_3(w) = \gamma \|w\|_1.$$

This specialization is nontrivial for two reasons. Firstly, the nonsmooth term $c_3(\cdot)$ is not an arbitrary composite penalty, but the ℓ_1 -norm of the lower-level gradient, whose exact-penalty role was justified in Section 2. Secondly, this particular structure will later allow us, in the simple bilevel setting, to derive an explicit dual reformulation of the subproblem as a box-constrained quadratic program (See Section 5).

Let $\mathcal{Z} := C \times \mathbb{R}^{d_y}$. To solve (8), we apply the prox-linear method with the penalty parameter γ_k fixed. At the k -th penalty stage, we initialize $z^{k,0} := z^k$ and generate a sequence $\{z^{k,j}\}_{j \geq 0}$ by minimizing prox-linear models. At the j -th prox-linear iteration, we linearize both f and $\nabla_y g$ at $z^{k,j} = (x^{k,j}, y^{k,j})$. The prox-linear model is defined by

$$\mathcal{L}_{\gamma_k, \lambda}(z; z^{k,j}) := f(z^{k,j}) + \nabla f(z^{k,j})^\top (z - z^{k,j}) + \gamma_k \left\| \nabla_y g(z^{k,j}) + \nabla(\nabla_y g)(z^{k,j})^\top (z - z^{k,j}) \right\|_1 + \frac{1}{2\lambda} \|z - z^{k,j}\|_2^2, \quad (10)$$

where $\lambda > 0$ is the proximal regularization parameter. The same parameter is used in the prox-gradient mapping defined later in (15). Here,

$$\nabla(\nabla_y g)(z^{k,j}) := \begin{bmatrix} \nabla_{xy} g(x^{k,j}, y^{k,j}) \\ \nabla_{yy} g(x^{k,j}, y^{k,j}) \end{bmatrix}.$$

Let $\bar{z}^{k,j+1}$ denote the exact minimizer of the model:

$$\bar{z}^{k,j+1} := \arg \min_{z \in \mathcal{Z}} \mathcal{L}_{\gamma_k, \lambda}(z; z^{k,j}). \quad (11)$$

In practice, the subproblem is usually solved only approximately. Therefore, for a prescribed accuracy $\delta_{k,j} \geq 0$, we call $z^{k,j+1} \in \mathcal{Z}$ a $\delta_{k,j}$ -approximate solution if

$$\mathcal{L}_{\gamma_k, \lambda}(z^{k,j+1}, z^{k,j}) \leq \mathcal{L}_{\gamma_k, \lambda}(\bar{z}^{k,j+1}, z^{k,j}) + \delta_{k,j}. \quad (12)$$

The fixed- γ_k prox-linear loop is terminated when

$$r_s^{k,j+1} := \frac{1}{\lambda} \|z^{k,j+1} - z^{k,j}\|_2 \leq \epsilon_s^{\text{in}}. \quad (13)$$

Let J_k be the final prox-linear index at the k -th penalty stage. We then set

$$z^{k+1} := z^{k, J_k}. \quad (14)$$

Thus, the overall idea is as follows. We apply the prox-linear framework to the norm-gradient penalized problem (3). The special structure of this subproblem, especially in the unconstrained simple bilevel case, is what distinguishes our method from a prox-linear scheme and leads to the dual reformulation and SPG solver developed later.

We emphasize that our contribution is to apply the prox-linear method to the norm-gradient penalized problem (3) and to exploit the specific ℓ_1 norm-gradient structure in the subsequent algorithm design.

The details of the fixed-penalty prox-linear procedure are given in Algorithm 2. When the fixed-penalty prox-linear procedure is embedded into the penalty scheme in Algorithm 1, we obtain the exact-penalty prox-linear method, abbreviated as EPPL.

Algorithm 2 Fixed-Penalty Prox-Linear Procedure for (8).

```

1: Input:  $\gamma_k, z^{k,0} := z^k, \lambda > 0, \epsilon_s^{\text{in}} > 0, \{\delta_{k,j}\}_{j \geq 0}$ .
2: Set  $j := 0$  and  $r_s^{k,0} := +\infty$ .
3: while  $r_s^{k,j} > \epsilon_s^{\text{in}}$ , do
4:   Compute  $\nabla f(z^{k,j}), \nabla_y g(z^{k,j}), \nabla(\nabla_y g)(z^{k,j})$ .
5:   Compute a  $\delta_{k,j}$ -approximate solution  $z^{k,j+1}$  satisfying (12).
6:   Compute the step residual  $r_s^{k,j+1} := \frac{1}{\lambda} \|z^{k,j+1} - z^{k,j}\|$ .
7:   Set  $j := j + 1$ .
8: end while
9: Set  $J_k := j$  and  $z^{k+1} := z^{k, J_k}$ .
10: Output:  $z^{k+1}$ .

```

Convergence Analysis for a Fixed Penalty Parameter

In this subsection, we analyze the fixed-penalty prox-linear procedure in Algorithm 2. More precisely, we fix an arbitrary penalty stage $\bar{k}$ and set $\gamma := \gamma_{\bar{k}}$. This shorthand is used only in this subsection. The penalty update rule is not analyzed here. Hence, the following result should be interpreted as a convergence guarantee for the prox-linear loop with a fixed penalty parameter, not as a convergence theorem for the whole continuation scheme with varying γ_k .

To simplify notation, we write $z^j := z^{\bar{k}, j}, z^{j+1} := z^{\bar{k}, j+1}, \bar{z}^{j+1} := \bar{z}^{\bar{k}, j+1}, J := J_{\bar{k}}$ and $\delta_j := \delta_{\bar{k}, j}$. The fixed penalized objective is denoted by $F_{\gamma}(z) := f(z) + \gamma p(z)$, $p(z) := \|\nabla_y g(z)\|_1$. We introduce the following smoothness assumption.

Assumption 3. *There exist constants $L_f > 0$ and $L_{g,2} > 0$ such that the $f(x, y)$ and $\nabla_y g(x, y)$ are, respectively, L_f -Lipschitz-smooth and $L_{g,2}$ -Lipschitz-smooth in (x, y) , i.e.,*

- (i) $\|\nabla f(z) - \nabla f(z')\| \leq L_f \|z - z'\|, \quad \forall z, z' \in C \times \mathbb{R}^{d_y}.$
- (ii) $\|\nabla(\nabla_y g)(z) - \nabla(\nabla_y g)(z')\| \leq L_{g,2} \|z - z'\|, \quad \forall z, z' \in C \times \mathbb{R}^{d_y}.$

For each j , the exact model minimizer is $\bar{z}^{j+1} := \arg \min_{z \in \mathcal{Z}} \mathcal{L}_{\gamma, \lambda}(z; z^j)$. The associated exact prox-gradient mapping [40] is defined by

$$\mathcal{G}_{\gamma, \lambda}(z^j) := \lambda^{-1}(z^j - \bar{z}^{j+1}). \quad (15)$$

Moreover, by (12), the computed point z^{j+1} satisfies

$$\mathcal{L}_{\gamma, \lambda}(z^{j+1}; z^j) \leq \mathcal{L}_{\gamma, \lambda}(\bar{z}^{j+1}; z^j) + \delta_j. \quad (16)$$

The following lemma quantifies the approximation error between the true penalized objective and its prox-linear model.

Lemma 2. Let $\mathcal{Z} := C \times \mathbb{R}^{d_y}$ and suppose that Assumption 3 holds. Define

$$L_\gamma := L_f + \gamma \sqrt{d_y} L_{g, 2}. \quad (17)$$

Then, for any $z, w \in \mathcal{Z}$, it holds that

$$-\frac{L_\gamma + \lambda^{-1}}{2} \|z - w\|^2 \leq F_\gamma(z) - \mathcal{L}_{\gamma, \lambda}(z; w) \leq \frac{L_\gamma - \lambda^{-1}}{2} \|z - w\|^2. \quad (18)$$

Proof. See Appendix B.1. $\square$

The next lemma gives the descent estimate for inexact prox-linear steps.

Lemma 3. Suppose Assumption 3 holds and $\lambda \leq \frac{1}{L_\gamma}$. Assume that z^{j+1} satisfies the δ_j -approximate condition (16). Then

$$F_\gamma(z^j) \geq F_\gamma(z^{j+1}) - \delta_j + \frac{\lambda}{2} \|\mathcal{G}_{\gamma, \lambda}(z^j)\|^2. \quad (19)$$

Proof. See Appendix B.2. $\square$

Now we can derive the main convergence guarantee for Algorithm 2.

Theorem 3. Suppose Assumption 3 holds and $\lambda \leq \frac{1}{L_\gamma}$. Let $F_\gamma^{\inf} := \inf_{z \in \mathcal{Z}} F_\gamma(z) > -\infty$. Assume that, for each $j \geq 0$, the computed iteration z^{j+1} satisfies the δ_j -approximate condition (16). Then, for any integer $J \geq 1$,

$$\min_{j=0, \dots, J-1} \|\mathcal{G}_{\gamma, \lambda}(z^j)\|^2 \leq \frac{2(F_\gamma(z^0) - F_\gamma^{\inf} + \sum_{j=0}^{J-1} \delta_j)}{\lambda J}. \quad (20)$$

In particular, if $\sum_{j=0}^{\infty} \delta_j < \infty$, then $\min_{j=0, \dots, J-1} \|\mathcal{G}_{\gamma, \lambda}(z^j)\| = \mathcal{O}(J^{-1/2})$.

Proof. Summing (19) over $j = 0, \dots, J-1$ yields

$$\sum_{j=0}^{J-1} (F_\gamma(z^j) - F_\gamma(z^{j+1})) \geq \sum_{j=0}^{J-1} \left(-\delta_j + \frac{\lambda}{2} \|\mathcal{G}_{\gamma, \lambda}(z^j)\|^2 \right).$$

Since the left-hand side telescopes, we obtain

$$F_\gamma(z^0) - F_\gamma(z^J) \geq -\sum_{j=0}^{J-1} \delta_j + \frac{\lambda}{2} \sum_{j=0}^{J-1} \|\mathcal{G}_{\gamma, \lambda}(z^j)\|^2.$$

Rearranging terms gives

$$\frac{\lambda}{2} \sum_{j=0}^{J-1} \|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2 \leq F_{\gamma}(z^0) - F_{\gamma}(z^J) + \sum_{j=0}^{J-1} \delta_j.$$

Using the lower bound $F_{\gamma}(z^J) \geq F_{\gamma}^{\inf}$, we further have

$$\frac{\lambda}{2} \sum_{j=0}^{J-1} \|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2 \leq F_{\gamma}(z^0) - F_{\gamma}^{\inf} + \sum_{j=0}^{J-1} \delta_j.$$

Dividing both sides by $\lambda J/2$ and using

$$\min_{j=0,\dots,J-1} \|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2 \leq \frac{1}{J} \sum_{j=0}^{J-1} \|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2$$

yields (20).

Finally, if $\sum_{j=0}^{\infty} \delta_j < \infty$, then the partial sums $\sum_{j=0}^{J-1} \delta_j$ are uniformly bounded in J , and the estimate (20) implies

$$\min_{j=0,\dots,J-1} \|\mathcal{G}_{\gamma,\lambda}(z^j)\| = \mathcal{O}(J^{-1/2}).$$

□

Remark 2. The exact prox-gradient mapping is defined by the exact minimizer $\bar{z}^{j+1}$ of the prox-linear model, whereas Algorithm 2 uses the computed point z^{j+1} and the step residual (13). If z^{j+1} satisfies the inexactness condition (16), then the strong convexity of $\mathcal{L}_{\gamma,\lambda}(\cdot; z^j)$ implies $\|z^{j+1} - \bar{z}^{j+1}\| \leq \sqrt{2\lambda\delta_j}$. Therefore, by the triangle inequality,

$$\begin{aligned} \|\mathcal{G}_{\gamma,\lambda}(z^j)\| &= \lambda^{-1} \|z^j - \bar{z}^{j+1}\| \leq \lambda^{-1} \|z^{j+1} - z^j\| + \lambda^{-1} \|z^{j+1} - \bar{z}^{j+1}\| \\ &\leq \lambda^{-1} \|z^{j+1} - z^j\| + \sqrt{\frac{2\delta_j}{\lambda}}. \end{aligned} \quad (21)$$

Thus, the step residual used in Algorithm 2 provides a valid stationarity certificate, provided that the prox-linear subproblem is solved to a sufficiently small objective gap. In the unconstrained simple bilevel case, where the subproblem is solved through a box-constrained quadratic dual problem, this quantity can be certified by the primal-dual gap of the model subproblem. The projected-gradient residual (40) used by SPG is a practical stopping criterion, while the condition (16) is the theoretical inexactness measure used in the convergence analysis.

4. EPPL for Simple Bilevel Problem

We now specialize our EPPL to the simple bilevel problem. This setting serves as a natural starting point for two reasons. Firstly, by removing explicit feasible-set constraints, it isolates the essential difficulty caused by the exact-penalty term $\|\nabla g(x)\|_1$ and allows us to focus on the core mechanism of the proposed method. Secondly, in this setting, the prox-linear subproblem admits an explicit dual reformulation as a box-constrained quadratic program, which in turn enables an efficient dual first-order solver with closed-form primal recovery.

Define $F, G : \mathbb{R}^{d_x} \rightarrow \mathbb{R}$. We consider the simple bilevel problem:

$$\begin{aligned} \min_{x \in U} \quad & F(x) \\ \text{s.t.} \quad & x \in \arg \min_{z \in \mathbb{R}^{d_x}} G(z). \end{aligned} \quad (22)$$

Here, $U \subseteq \mathbb{R}^{d_x}$ is a nonempty closed convex set and $S := \arg \min_{z \in \mathbb{R}^{d_x}} G(z)$ is assumed to be nonempty with $S \subseteq U$. As a direct specialization of the assumptions introduced for the general bilevel problem, we assume: (i) G satisfies the Polyak–Łojasiewicz inequality on U ; (ii) F is Lipschitz continuous on U ; (iii) F , G , and ∇G are Lipschitz smooth on U . Under assumption (i), problem (22) is equivalent to the following gradient-based reformulation:

$$\min_{x \in U} F(x) \quad \text{s.t.} \quad \nabla G(x) = 0. \quad (23)$$

Under assumptions (i)–(iii), the exact-penalty analysis in Theorem 1 applies to the present simple bilevel setting. Hence, for sufficiently large $\gamma > 0$, the following penalized problem provides a globally exact reformulation of (23):

$$\min_{x \in U} \Phi_\gamma(x) := F(x) + \gamma P(x), \quad (24)$$

where $P(x) := \|\nabla G(x)\|_1$. Moreover, if P is continuous and convex, then the local exactness statement in Theorem 2 also applies.

We terminate the algorithm by combining a feasibility residual for the gradient equation in (23) and a stationarity residual.

$$R_f^{k+1} := \|\nabla G(x^{k+1})\|_1, \quad R_s^{k+1} := \frac{1}{\lambda_k} \|x^{k+1} - x^k\|_2. \quad (25)$$

where $\lambda_k > 0$ is the prox-linear stepsize used at the k -th penalty stage. The lower-level feasibility residual measures the violation of the equality constraint $\nabla G(x) = 0$ in (23), while the stationarity residual measures the stationarity of the step. The algorithm stops once

$$R_f^{k+1} \leq \varepsilon_f, \quad R_s^{k+1} \leq \varepsilon_s. \quad (26)$$

If (26) is not satisfied, the penalty parameter is increased.

The details of the ℓ_1 Penalty Method are given in Algorithm 3.

Algorithm 3 Classical ℓ_1 Penalty Method for (24).

1: **Input:** x^0 and $\gamma_0 > 0$, $\tau > 1$, $\varepsilon_f > 0$, $\varepsilon_s > 0$.

2: Set $k := 0$, $R_f^0 := +\infty$, and $R_s^0 := +\infty$.

3: **while** $R_f^k > \varepsilon_f$, or $R_s^k > \varepsilon_s$, **do**

4: Solve

$$\min_{x \in U} \Phi_{\gamma_k} := F(x) + \gamma_k P(x) \quad (27)$$

to obtain x^{k+1} .

5: Set $\gamma_{k+1} := \tau \gamma_k$ and $k := k + 1$

6: **end while**

7: **Output:** $x_{\text{opt}} := x^k$.

To solve (27), we apply the prox-linear method with the penalty parameter γ_k fixed. More precisely, at the k -th penalty stage, we initialize $x^{k,0} := x^k$. Then, for the fixed value γ_k , we perform several prox-linear iterations indexed by $j = 0, 1, 2, \dots$. At the j -th prox-linear iteration, the model of Φ_{γ_k} at $x^{k,j}$ is defined by

$$\ell_{\gamma_k, \lambda}(x; x^{k,j}) := F(x^{k,j}) + \nabla F(x^{k,j})^\top (x - x^{k,j}) + \gamma_k \left\| \nabla G(x^{k,j}) + \nabla^2 G(x^{k,j})(x - x^{k,j}) \right\|_1 + \frac{1}{2\lambda} \|x - x^{k,j}\|_2^2. \quad (28)$$

The next prox-linear iterate $x^{k,j+1}$ is obtained by solving

$$\min_{x \in U} \ell_{\gamma_k, \lambda}(x; x^{k,j}). \quad (29)$$

The fixed- γ_k prox-linear loop is terminated when

$$R_s^{k,j+1} := \frac{1}{\lambda} \|x^{k,j+1} - x^{k,j}\|_2 \leq \epsilon_{\text{in}}. \quad (30)$$

Let J_k be the final prox-linear iteration index at the k -th penalty stage. We then set

$$x^{k+1} := x^{k,J_k}. \quad (31)$$

For each fixed penalty parameter γ_k , the penalized subproblem (27) is solved by the fixed-penalty prox-linear procedure described in Algorithm 4. Together with the penalty-continuation scheme in Algorithm 3, this gives the exact-penalty prox-linear method for simple bilevel problems, abbreviated as EPPL-SBP.

Algorithm 4 Fixed-Penalty Prox-Linear Procedure for (27).

```

1: Input:  $\gamma_k, x^{k,0} := x^k$ , and  $\lambda > 0, \epsilon_{\text{in}} > 0$ .
2: Set  $j := 0, R_s^{k,0} := +\infty$ .
3: while  $R_s^{k,j} > \epsilon_{\text{in}}$ , do
4:   Compute  $\nabla F(x^{k,j}), \nabla G(x^{k,j}), \nabla^2 G(x^{k,j})$ .
5:   Compute a  $\delta_{k,j}$ -approximate solution  $x^{k,j+1}$  of (29).
6:   Set  $j := j + 1$ .
7: end while
8: Set  $J_k := j$  and  $x^{k+1} := x^{k,J_k}$ .
9: Output:  $x^{k+1}$ .

```

5. Solving Dual Problems via Spectral Projected Gradient

In this section, we derive the dual reformulation of (29). For the dual reformulation developed below, we specialize to the unconstrained case $U = \mathbb{R}^{d_x}$. In this case, the prox-linear subproblem admits a box-constrained quadratic dual reformulation.

For notational simplicity, throughout this section, we fix one penalty stage k and suppress this outer index. Hence, the penalty parameter γ_k is denoted simply by γ , and the prox-linear iterates $x^{k,j}$ and $x^{k,j+1}$ are denoted by x^j and x^{j+1} , respectively. Thus, all quantities in this section are associated with a fixed penalty parameter γ and a local prox-linear iteration index j .

Ignoring the constant term $F(x^j)$, consider

$$\min_{x \in \mathbb{R}^{d_x}} \frac{1}{2\lambda} \|x - x^j\|_2^2 + \nabla F(x^j)^\top (x - x^j) + \gamma \left\| \nabla G(x^j) + \nabla^2 G(x^j)(x - x^j) \right\|_1. \quad (32)$$

Define

$$v^j := x^j - \lambda \nabla F(x^j), \quad B^j := \nabla^2 G(x^j), \quad a^j := \nabla G(x^j) - B^j x^j.$$

where $v^j \in \mathbb{R}^{d_x}$, $B^j \in \mathbb{R}^{d_x \times d_x}$, $a^j \in \mathbb{R}^{d_x}$. Dropping constants independent of x and completing the square, (32) is equivalent to

$$\min_{x \in \mathbb{R}^{d_x}} \frac{1}{2\lambda} \|x - v^j\|_2^2 + \gamma \|a^j + B^j x\|_1. \quad (33)$$

Remark 3. In the remainder of this section, we focus on the unconstrained upper-level case $U = \mathbb{R}^{d_x}$. This setting covers the simple bilevel model used in our numerical experiments and leads

to a box-constrained quadratic dual subproblem. For a general closed set U , the subproblem can be written as

$$\min_{x \in \mathbb{R}^{d_x}} \frac{1}{2\lambda} \|x - v^j\|_2^2 + \gamma \|a^j + B^j x\|_1 + \delta_U(x),$$

where δ_U is the indicator function of U , namely,

$$\delta_U(x) = \begin{cases} 0, & x \in U, \\ +\infty, & x \notin U. \end{cases}$$

However, in this case the dual objective is generally no longer a simple quadratic function.

5.1. Dual Subproblem and Optimality Condition

In this subsection, we reformulate the primal subproblem (33) as an equivalent dual problem by using the Fenchel conjugate of the ℓ_1 term. The following proposition summarizes the dual reformulation and its explicit quadratic form.

Proposition 1. *The dual problem of (33) is*

$$\min_{y \in \Omega_\gamma} d(y) := \frac{\lambda}{2} \|(B^j)^\top y\|_2^2 - \langle y, c^j \rangle. \quad (34)$$

where Ω_γ is defined by $\Omega_\gamma := \{y \in \mathbb{R}^{d_x} : \|y\|_\infty \leq \gamma\}$ and $c^j := a^j + B^j v^j$. Moreover, if y^* is a dual solution, the corresponding primal solution is recovered by

$$x^* = v^j - \lambda (B^j)^\top y^*. \quad (35)$$

Proof. See Appendix C.1. $\square$

Now we study the first-order optimality condition of the dual problem derived in Proposition 1. In particular, we show that the optimality system can be equivalently written as a projected fixed-point equation, which naturally motivates projected first-order methods for solving the dual problem.

Since $d(\cdot)$ is continuously differentiable and Ω_γ is nonempty, closed, and convex, a point y^* solves (34) if and only if

$$0 \in \nabla d(y^*) + N_{\Omega_\gamma}(y^*), \quad (36)$$

where $N_{\Omega_\gamma}(\cdot)$ denotes the normal cone to Ω_γ . By the standard projection characterization, (36) is equivalent to

$$y^* = \Pi_{\Omega_\gamma}(y^* - t \nabla d(y^*)), \quad \forall t > 0. \quad (37)$$

This projected fixed-point form motivates the use of projected first-order methods for the dual problem, where the only nonsmoothness is handled by the projection onto the box-constrained set Ω_γ .

5.2. Dual Spectral Projected Gradient

In this subsection, we employ the classical spectral projected gradient (SPG) method of [41,42] to solve the dual problem (34)

For a fixed outer iterate x^j , consider the dual minimization problem (34), its gradient is

$$\nabla d_j(y) = \lambda B^j (B^j)^\top y - c^j. \quad (38)$$

Moreover, the projection onto Ω_γ is given component-wise by

$$(\Pi_{\Omega_\gamma}(u))_i = \min\{\gamma, \max\{-\gamma, u_i\}\}, \quad i = 1, \dots, d_x. \quad (39)$$

The inner SPG solver is terminated according to the projected-gradient residual

$$r_{\text{spg}}(y) := \left\| \Pi_{\Omega_\gamma}(y - \eta \nabla d_j(y)) - y \right\|_2 \leq \varepsilon_{\text{spg}}, \quad (40)$$

where $\eta > 0$ is the current spectral step-length. The detailed SPG method, including the nonmonotone Armijo line search, is given in Algorithm A1 in Appendix C.2.

The use of SPG in Algorithm A1 is motivated by three features of the dual subproblem (34). Firstly, the objective $d_j(\cdot)$ is a smooth convex quadratic function. Secondly, the feasible set Ω_γ is a simple box, so the projection (39) is explicit. Thirdly, the dual iterate can be converted back to a primal iterate by the closed-form recovery formula (35). These properties make SPG a natural and efficient inner solver within our EPPL-SBP.

Remark 4. For each fixed outer iterate x^j , the dual subproblem (34) is a smooth convex quadratic optimization problem over the closed and bounded box Ω_γ . Therefore, Algorithm A1 is precisely the classical nonmonotone SPG method of [41] applied to (34). By the standard convergence theory of SPG, every accumulation point $\bar{y}$ of the generated dual sequence $\{y^\ell\}$ satisfies the first-order optimality condition $0 \in \nabla d_j(\bar{y}) + N_{\Omega_\gamma}(\bar{y})$. Equivalently, for any $t > 0$, $\bar{y} = \Pi_{\Omega_\gamma}(\bar{y} - t \nabla d_j(\bar{y}))$. Since d_j is convex and Ω_γ is convex, every such accumulation point is a global minimizer of the dual problem (34).

5.3. Computational Cost and Limitations

We briefly discuss the main computational cost of EPPL-SBP, which consists of the penalty-continuation scheme in Algorithm 3, the fixed-penalty prox-linear procedure in Algorithm 4, and the inner SPG solver in Algorithm A1. Since the simple bilevel problem considered in this section has a single decision variable $x \in \mathbb{R}^{d_x}$, we set $d := d_x$.

We first consider one iteration of Algorithm 4. At each prox-linear iteration, the algorithm evaluates $\nabla F(x^{k,j})$, $\nabla G(x^{k,j})$, $\nabla^2 G(x^{k,j})$. When these derivatives are explicitly formed and $\nabla^2 G(x^{k,j})$ is treated as a dense $d \times d$ matrix, the derivative-evaluation cost is dominated by forming the Hessian and is, therefore, $O(d^2)$.

After the derivatives are computed, forming the dual data $v^j = x^{k,j} - \lambda \nabla F(x^{k,j})$, $a^j = \nabla G(x^{k,j}) - B^j x^{k,j}$, $c^j = a^j + B^j v^j$, where $B^j = \nabla^2 G(x^{k,j})$, requires matrix-vector products with B^j and, therefore, costs $O(d^2)$. We next consider the inner SPG solver in Algorithm A1. The gradient of the dual objective is $\nabla d_j(y) = \lambda B^j (B^j)^\top y - c^j$. For dense B^j , each evaluation of $\nabla d_j(y)$ costs $O(d^2)$. The evaluation of the dual objective $d_j(y)$ also costs $O(d^2)$, while the projection onto the box Ω_γ costs only $O(d)$. Thus, if the number of nonmonotone line-search trials is uniformly bounded, the dominant cost of one SPG iteration is $O(d^2)$. Let $T_{k,j}$ be the number of SPG iterations used by Algorithm A1 at the j -th prox-linear iteration of the k -th penalty stage. Combining the derivative-evaluation cost, the cost of forming the dual data, and the cost of the inner SPG iterations, one iteration of Algorithm 4 costs $O((1 + T_{k,j})d^2)$. If J_k prox-linear iterations are performed in Algorithm 4 at the k -th penalty stage, then the cost of this penalty stage is

$$O\left(\sum_{j=0}^{J_k-1} (1 + T_{k,j})d^2\right).$$

Finally, Algorithm 3 repeats the fixed-penalty prox-linear procedure while increasing the penalty parameter. If $J_k \leq J$ and $T_{k,j} \leq T$ uniformly, and if Algorithm 3 terminates

after K penalty stages, then the overall computational cost of EPPL-SBP is $O(KJ(1+T)d^2)$, or simply $O(KJTd^2)$ when the inner SPG cost is dominant. The memory cost is $O(d^2)$, dominated by storing the dense Hessian matrix B^j .

The proposed EPPL-SBP method has some limitations. Theoretically, its exact-penalty guarantee requires a lower-level error-bound or PL-type condition, together with the Lipschitz assumptions. These conditions may not hold for general nonconvex lower-level problems. The box-constrained dual reformulation is also specific to the unconstrained case $U = \mathbb{R}^{d_x}$. For a general closed convex set U , the dual subproblem, may lose this simple structure. Computationally, EPPL-SBP relies on first- and second-order information of G , and the inner SPG solver may become costly in high dimensions unless sparsity or fast Hessian-vector products are available. Large penalty parameters may also cause ill-conditioning, so the continuation strategy for γ needs to be chosen with care.

6. Numerical Experiments

In this section, we evaluate EPPL-SBP from two perspectives. Firstly, we apply EPPL-SBP to Minimum Norm Solution Problem (MNP), and compare it with several existing methods [20,23,24,31,43]. Secondly, we include a Lipschitz least-squares variant to provide a numerical illustration of the global exactness result under its stated assumptions. The numerical tests are conducted in MATLAB R2023a on a MacBook Air (13-inch, M3, 2024) (Apple Inc., Cupertino, CA, USA), running macOS Sonoma 14.6 with an Apple M3 chip and 16 GB of memory.

6.1. EPPL-SBP on the MNP

We consider the simple bilevel problem

$$\min_{x \in \mathbb{R}^n} F(x) := \frac{1}{2} \|x\|_2^2 \quad \text{s.t.} \quad x \in \arg \min_{z \in \mathbb{R}^n} G(z) := \frac{1}{2} \|Az - b\|_2^2. \quad (41)$$

This model seeks the minimum-norm least-squares solution and follows the MNP experiment used in [31]. We use it mainly as a test problem for comparison with existing methods. We note that (41) is not directly covered by the global exactness theorem, because $F(x)$ is not Lipschitz continuous on $\mathbb{R}^n$. However, it is covered by the local exactness result on bounded neighborhoods. Indeed, F is Lipschitz continuous on every bounded region. Therefore, the numerical results for (41) are justified by the local exactness theorem.

We use the YearPredictionMSD dataset (<https://archive.ics.uci.edu/dataset/203/yearpredictionmsd> (accessed on 27 January 2026)), which consists of 515,345 songs released between 1992 and 2011. Each sample contains the release year and 90 additional attributes. To evaluate the robustness of the algorithms with respect to random sampling, we perform experiments over ten prescribed random seeds. For each seed, we independently and uniformly sample 1000 songs from the dataset [31]. The feature matrix is normalized columnwise to $[-1, 1]$, an intercept column is appended, and the response vector is normalized to $[0, 1]$. Thus, for every run, the resulting data matrix satisfies $A \in \mathbb{R}^{1000 \times 91}$, and $b \in \mathbb{R}^{1000}$ denotes the corresponding normalized response vector.

For each sampled instance, motivated by [31], we compute a reference solution $x^\dagger$ by MATLAB `lsqminnorm(A, b)`. This is justified because, for the simple bilevel problem (41) considered here, the bilevel solution is exactly the minimum-norm least-squares solution. Therefore, `lsqminnorm(A, b)` is used only to obtain the reference solution for this special benchmark problem, not as a general-purpose solver for bilevel optimization. We then report the corresponding benchmark quantities

$$F^* := F(x^\dagger), \quad G^* := G(x^\dagger).$$

Table 1 reports the basic information of the ten randomly sampled instances used in the YearPredictionMSD experiment. The last two rows report the mean and standard deviation (Std.) over the ten random subsamples.

Table 1. Data information for the YearPredictionMSD MNP experiment.

Seed	Rank (A)	$\ A^T A\ _2$	F^*	G^*
123456	91	1.2671×10^4	0.7545	7.3916
123457	91	1.2089×10^4	0.8537	6.3737
123458	91	1.1773×10^4	0.7030	5.9077
123459	91	1.2538×10^4	0.5739	6.7935
123460	91	1.1609×10^4	0.9301	8.6109
223456	91	1.3695×10^4	0.7559	6.2904
223457	91	1.2646×10^4	0.6605	7.2526
223458	91	1.0375×10^4	0.6733	12.7804
223459	91	1.2861×10^4	0.7676	7.1585
223460	91	1.4655×10^4	0.7940	9.3361
Mean	91	1.2491×10^4	0.7466	7.7895
Std.	0	1.1671×10^3	0.1014	2.0441

The algorithmic parameters of EPPL-SBP used in the YearPredictionMSD experiment are summarized in Table 2. The same parameter setting is used for all ten random subsamples.

Table 2. Parameter settings of EPPL-SBP for the YearPredictionMSD experiment.

Category	Parameter	Value
Initialization	initial point	$x^0 = 0$
Penalty continuation	initial penalty parameter	$\gamma_0 = 100$
	penalty growth factor	$\tau = 1.2$
	lower-level residual tolerance	$\varepsilon_f = 10^{-5}$
	stationarity residual tolerance	$\varepsilon_s = 10^{-5}$
Prox-linear step	stepsize parameter	$\lambda = 10^{-2}$
	prox-linear residual tolerance	$\varepsilon_{in} = 10^{-5}$
Inner SPG solver	inner tolerance	adaptive: $10^{-3}, 10^{-4}, 10^{-6}$
	maximum SPG iterations	adaptive: 200, 400, 1000
	nonmonotone memory length	$M = 10$
	spectral stepsize bounds	$\eta_{\min} = 10^{-12}, \eta_{\max} = 10^{12}$
	Armijo forcing constant	$\sigma = 10^{-4}$
	backtracking factor	$\beta = 0.5$

The stopping criterion of EPPL-SBP is

$$\|\nabla G(x^k)\|_1 \leq 10^{-5}, \quad \frac{\|x^k - x^{k-1}\|_2}{\lambda} \leq 10^{-5}. \quad (42)$$

Table 3 reports the performance of EPPL-SBP when the algorithm is terminated by its stopping criterion (42). The quantities are summarized over the ten random subsamples by their mean, standard deviation, minimum, and maximum values.

Table 3 shows that EPPL-SBP is stable under its own stopping criterion. The method terminates successfully on all ten runs. The final $\|\nabla G(x)\|_1$ is on average 5.6472×10^{-6} , and the final $\|x^k - x^{k-1}\|_2/\lambda$ is on average 6.7782×10^{-6} , both below the prescribed tolerance 10^{-5} . Moreover, the final lower-level objective gap $G(x) - G^*$ is 4.5040×10^{-13} on average, and the mean upper-level gap $|F(x) - F^*|$ is 4.0048×10^{-7} . These values indicate that the computed solution is very close to the reference minimum-norm least-

squares solution on all ten subsamples. The main computational effort is spent in the inner SPG solver: EPPL-SBP uses 5.9718×10^4 SPG iterations on average. Nevertheless, because each SPG iteration only requires matrix-vector products and box projections, the average CPU time remains below one second.

Table 3. Performance of EPPL-SBP under stopping criterion (42) over ten random subsamples.

Quantity	Mean	Std.	Min	Max
Number of penalty stages	3.40	0.84	2	5
Final penalty parameter γ	1.5658×10^2	2.4714×10^1	1.2000×10^2	2.0736×10^2
Total prox-linear iterations	109.0	29.31	71	164
Total SPG iterations	5.9718×10^4	2.1685×10^4	3.4362×10^4	1.0609×10^5
Final $\ \nabla G(x)\ _1$	5.6472×10^{-6}	8.8155×10^{-7}	3.6765×10^{-6}	6.7883×10^{-6}
Final $\ x^k - x^{k-1}\ _2 / \lambda$	6.7782×10^{-6}	3.0212×10^{-6}	2.5412×10^{-6}	9.9891×10^{-6}
Final $ F(x) - F^* $	4.0048×10^{-7}	2.7467×10^{-7}	9.6566×10^{-8}	8.6599×10^{-7}
Final $G(x) - G^*$	4.5040×10^{-13}	1.5663×10^{-13}	2.2382×10^{-13}	6.7502×10^{-13}
Final $\ x - x^\dagger\ _2$	9.4011×10^{-7}	2.9744×10^{-7}	5.3531×10^{-7}	1.3634×10^{-6}
CPU time (s)	6.5352×10^{-1}	2.4178×10^{-1}	3.5411×10^{-1}	1.1263

6.2. Comparison with Existing Methods for Solving (41)

We compare the performance of EPPL-SBP with several existing methods (The implementations of the benchmark methods are taken from the public repository <https://github.com/XuShi22/BisecBiO> (accessed on 23 February 2026)) for solving (41), including the minimal norm gradient method (MNG), the bilevel gradient SAM method (BiG-SAM) [24], the averaging iteratively regularized gradient method (a-IRG) [23], the dynamic barrier gradient descent method (DBGD) [43], and the bisection-based method (Bisec-BiO) [31]. We use the same ten YearPredictionMSD subsamples as in Section 6.1.

For comparison with existing methods, we use a common time-to-accuracy criterion. A run is counted as successful if the method reaches

$$G(x^k) - G^* \leq 10^{-6}, \quad |F(x^k) - F^*| \leq 10^{-6}.$$

For each successful run, we record the first iteration at which both conditions are satisfied and the corresponding CPU time. For methods with success rate below 100%, the reported first-hit time is averaged only over the successful runs. A dash in Table 4 means that the method did not reach the prescribed accuracy in any of the ten runs.

Table 4. Time-to-accuracy comparison over ten random subsamples.

Method	Success Rate	First-Hit Time (s)	$G(x) - G^*$ at Hit	$ F(x) - F^* $ at Hit
Bisec-BiO	4/10	$9.6161 \times 10^{-2} \pm 6.4289 \times 10^{-2}$	1.2136×10^{-10}	3.8938×10^{-7}
a-IRG	0/10	—	—	—
BiG-SAM	0/10	—	—	—
MNG	0/10	—	—	—
DBGD	0/10	—	—	—
EPPL-SBP	10/10	$4.7397 \times 10^{-1} \pm 1.5857 \times 10^{-1}$	1.7642×10^{-11}	4.0715×10^{-7}

Bold values indicate the proposed method and the best success rate among all compared methods.

Table 4 shows that EPPL-SBP reaches the prescribed accuracy in all ten runs. Bisec-BiO is faster on the four runs where it succeeds, but it only reaches the target accuracy in 4 out of 10 runs. The other compared methods do not reach the prescribed pair of accuracy conditions. Therefore, on these YearPredictionMSD subsamples, EPPL-SBP achieves the highest success rate among the tested methods for simultaneously attaining lower-level feasibility and upper-level minimum-norm accuracy.

6.3. A Numerical Illustration of Global Exactness

The preceding MNP experiment serves as the main test problem for comparison with existing methods. As noted above, its upper-level objective $F(x) = \frac{1}{2}\|x\|^2$ is not globally Lipschitz continuous on $\mathbb{R}^n$, and hence the global exactness theorem does not apply directly. To illustrate the global exactness result under its stated assumptions, we additionally consider the following Lipschitz variant:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & F_{\text{ge}}(x) := \sqrt{1 + \|x\|^2} \\ \text{s.t.} \quad & x \in \arg \min_{z \in \mathbb{R}^n} G(z) := \frac{1}{2}\|Az - b\|^2. \end{aligned} \quad (43)$$

Here the function F_{ge} is globally Lipschitz continuous on $\mathbb{R}^n$.

We use the same ten YearPredictionMSD subsamples as in Section 6.1. Table 5 shows that EPPL-SBP performs consistently on the Lipschitz least-squares variant. The final residuals $\|\nabla G(x)\|_1$ and $\|x^k - x^{k-1}\|_2/\lambda$ are on average 5.6727×10^{-6} and 7.7033×10^{-6} , respectively, both below the prescribed tolerance 10^{-5} . The final gaps are also small, with average values $G(x) - G^* = 4.4773 \times 10^{-13}$, $|F_{\text{ge}}(x) - F_{\text{ge}}^*| = 2.7119 \times 10^{-7}$. Numerically, EPPL-SBP approximates this reference solution with high accuracy, with an average CPU time of about 1.45 seconds.

Table 5. Performance of EPPL-SBP under stopping criterion (42) on the problem (43) over ten random subsamples.

Quantity	Mean	Std.	Min	Max
Number of penalty stages	4.60	1.43	3	7
Final penalty parameter γ	2.2484	1.1640	9.9190×10^{-1}	4.2328
Total prox-linear iterations	165.2	55.99	93	279
Total SPG iterations	1.1350×10^5	4.5617×10^4	5.0340×10^4	1.9955×10^5
Final $\ \nabla G(x)\ _1$	5.6727×10^{-6}	7.8019×10^{-7}	4.7454×10^{-6}	7.5396×10^{-6}
Final $\ x^k - x^{k-1}\ _2/\lambda$	7.7033×10^{-6}	2.6756×10^{-6}	1.0001×10^{-6}	9.9255×10^{-6}
Final $ F_{\text{ge}}(x) - F_{\text{ge}}^* $	2.7119×10^{-7}	1.8940×10^{-7}	6.1620×10^{-8}	5.9907×10^{-7}
Final $G(x) - G^*$	4.4773×10^{-13}	1.6661×10^{-13}	2.7711×10^{-13}	7.6827×10^{-13}
Final $\ x - x^\dagger\ _2$	9.1199×10^{-7}	2.7567×10^{-7}	6.1646×10^{-7}	1.3328×10^{-6}
CPU time (s)	1.4544	7.7589×10^{-1}	5.2665×10^{-1}	2.8272

7. Conclusions

In this paper, we studied an exact-penalty approach for bilevel optimization based on the ℓ_1 -norm of the lower-level gradient. Under suitable assumptions, the resulting norm-gradient penalty yields an exact reformulation for sufficiently large penalty parameters. To solve the nonsmooth penalized problem, we applied a fixed-penalty prox-linear procedure and established a stationarity-oriented convergence guarantee for the fixed-penalty loop with inexact subproblem solutions. We then specialized the method to the unconstrained simple bilevel problem. In this setting, the prox-linear subproblem admits an explicit dual reformulation as a box-constrained quadratic program. This dual structure leads to a nonmonotone SPG solver for the dual subproblem, together with a closed-form formula for recovering the primal solution. Numerical experiments on the Minimum Norm Solution Problem show that EPPL-SBP is effective in attaining both lower-level feasibility and upper-level accuracy, and achieves the highest success rate among the tested methods on the considered YearPredictionMSD instances. A Lipschitz least-squares variant further provides a numerical illustration of the global exactness theorem under its stated assumptions.

Several directions remain for future work. One is to extend the present framework to constrained simple bilevel problems and, more generally, to bilevel models with explicit lower-level constraints. It would also be of interest to investigate adaptive penalty update rules and more scalable inner solvers for large-scale machine learning applications.

Author Contributions: Conceptualization, Y.Z. and Q.L.; methodology, Y.Z. and Q.L.; software, Y.Z.; formal analysis, Y.Z.; investigation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Q.L. and J.L.; supervision, Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: The corresponding author’s research is supported by National Natural Science Foundation of China (NSFC) No. 12271526.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Omitted Proof in Section 2

Appendix A.1. Proof of Lemma 1

We first recall the definition of error bound and its relation to the PL condition, which will be used to establish the distance-bound property of the penalty function.

Definition A1. For a fixed $x \in C$, the function $g(x, \cdot)$ is said to satisfy the error bound with constant $\kappa > 0$ if

$$\kappa d_{S(x)}(y) \leq \|\nabla_y g(x, y)\|_2, \quad \forall y \in \mathbb{R}^{d_y}.$$

Theorem A1 ([39], Theorem 2). Suppose that, for every $x \in C$, the function $g(x, \cdot)$ is Lipschitz-smooth. If $g(x, \cdot)$ satisfies the Polyak–Łojasiewicz inequality, then $g(x, \cdot)$ also satisfies the error bound.

Remark A1. According to the proof of [39], Theorem 2, under the PL inequality in Assumption 1, the above error bound holds with $\kappa = \frac{1}{2\sqrt{\mu}}$.

We next give the proof of Lemma 1.

Proof. The nonnegativity of p is immediate. Since the lower-level problem is unconstrained in y , Fermat’s rule gives

$$y \in S(x) \implies \nabla_y g(x, y) = 0,$$

and hence $p(x, y) = 0$. Conversely, if $p(x, y) = 0$, then

$$\nabla_y g(x, y) = 0.$$

By Assumption 1, we have

$$0 = \|\nabla_y g(x, y)\|_2^2 \geq \frac{1}{\sqrt{\mu}} (g(x, y) - v(x)).$$

Since $g(x, y) \geq v(x)$, it follows that $g(x, y) = v(x)$, and therefore, $y \in S(x)$. Since $S(x)$ is closed, this is equivalent to $d_{S(x)}(y) = 0$. Thus,

$$p(x, y) = 0 \iff y \in S(x) \iff d_{S(x)}(y) = 0.$$

It remains to prove the distance-bound inequality. By Theorem A1 and the subsequent remark, the PL condition in Assumption 1 implies the error bound with $\kappa = 1/(2\sqrt{\mu})$. Therefore,

$$\frac{1}{2\sqrt{\mu}}d_{S(x)}(y) \leq \|\nabla_y g(x, y)\|_2, \quad \forall x \in C, \quad \forall y \in \mathbb{R}^{d_y}.$$

Equivalently,

$$d_{S(x)}(y) \leq 2\sqrt{\mu} \|\nabla_y g(x, y)\|_2.$$

Finally, since $\|a\|_2 \leq \|a\|_1$ for any vector a , we have

$$d_{S(x)}(y) \leq 2\sqrt{\mu} \|\nabla_y g(x, y)\|_1 = 2\sqrt{\mu} p(x, y).$$

Therefore, p is a ρ -distance-bound function with $\rho = 2\sqrt{\mu}$. $\square$

Appendix B. Omitted Proof in Section 3

Appendix B.1. Proof of Lemma 2

Proof. We estimate separately the approximation errors of the smooth term f and the composite penalty term. Let $s := z - w$. Since f is L_f -smooth, we have, by the integral form of Taylor's formula,

$$f(z) - f(w) - \nabla f(w)^\top s = \int_0^1 \langle \nabla f(w + ts) - \nabla f(w), s \rangle dt.$$

Hence, by the Cauchy–Schwarz inequality,

$$|f(z) - f(w) - \nabla f(w)^\top s| \leq \int_0^1 \|\nabla f(w + ts) - \nabla f(w)\| \|s\| dt \leq \int_0^1 L_f t \|s\|^2 dt = \frac{L_f}{2} \|s\|^2.$$

Therefore,

$$-\frac{L_f}{2} \|z - w\|^2 \leq f(z) - f(w) - \nabla f(w)^\top (z - w) \leq \frac{L_f}{2} \|z - w\|^2.$$

Let $h(z) := \nabla_y g(z)$, $J(w) := \nabla h(w) = \nabla(\nabla_y g(w))$. Using the integral form of Taylor's formula again, we obtain

$$h(z) - h(w) - J(w)^\top s = \int_0^1 [J(w + ts) - J(w)]^\top s dt.$$

By Assumption 3 (ii),

$$\|h(z) - h(w) - J(w)^\top s\|_2 \leq \int_0^1 \|J(w + ts) - J(w)\| \|s\| dt \leq \int_0^1 L_{g,2} t \|s\|^2 dt = \frac{L_{g,2}}{2} \|s\|^2.$$

That is,

$$\|h(z) - h(w) - J(w)^\top (z - w)\|_2 \leq \frac{L_{g,2}}{2} \|z - w\|^2.$$

Moreover, by the reverse triangle inequality for the ℓ_1 norm,

$$|\|h(z)\|_1 - \|h(w) + J(w)^\top (z - w)\|_1| \leq \|h(z) - h(w) - J(w)^\top (z - w)\|_1.$$

Therefore,

$$\|h(z) - h(w) - J(w)^\top (z - w)\|_1 \leq \sqrt{d_y} \|h(z) - h(w) - J(w)^\top (z - w)\|_2 \leq \frac{\sqrt{d_y} L_{g,2}}{2} \|z - w\|^2.$$

It follows that

$$-\frac{\gamma \sqrt{d_y} L_{g,2}}{2} \|z - w\|^2 \leq \gamma \|h(z)\|_1 - \gamma \|h(w) + J(w)^\top (z - w)\|_1 \leq \frac{\gamma \sqrt{d_y} L_{g,2}}{2} \|z - w\|^2.$$

Combining the estimates for f and the penalty term, and recalling that $\mathcal{L}_{\gamma,\lambda}(z;w)$ contains the proximal term $\frac{1}{2\lambda}\|z-w\|^2$, we obtain

$$-\frac{L_\gamma}{2}\|z-w\|^2 \leq F_\gamma(z) - \mathcal{L}_{\gamma,\lambda}(z;w) + \frac{1}{2\lambda}\|z-w\|^2 \leq \frac{L_\gamma}{2}\|z-w\|^2.$$

Rearranging gives (18). $\square$

Appendix B.2. Proof of Lemma 3

Proof. We first verify that $\mathcal{L}_{\gamma,\lambda}(\cdot; z^j)$ is $1/\lambda$ -strongly convex on $\mathcal{Z}$. For fixed z^j , the term

$$f(z^j) + \nabla f(z^j)^\top (z - z^j)$$

is affine in z . Moreover, the mapping

$$z \mapsto \nabla_y g(z^j) + \nabla(\nabla_y g)(z^j)^\top (z - z^j)$$

is affine, and hence

$$z \mapsto \left\| \nabla_y g(z^j) + \nabla(\nabla_y g)(z^j)^\top (z - z^j) \right\|_1$$

is convex, because the ℓ_1 -norm is convex and convexity is preserved under affine composition. Finally,

$$z \mapsto \frac{1}{2\lambda}\|z - z^j\|^2$$

is $1/\lambda$ -strongly convex. Therefore, the sum $\mathcal{L}_{\gamma,\lambda}(\cdot; z^j)$ is $1/\lambda$ -strongly convex on $\mathcal{Z}$. Consequently, since $\bar{z}^{j+1}$ is the exact minimizer of $\mathcal{L}_{\gamma,\lambda}(\cdot; z^j)$ over $\mathcal{Z}$, we have

$$\mathcal{L}_{\gamma,\lambda}(z; z^j) \geq \mathcal{L}_{\gamma,\lambda}(\bar{z}^{j+1}; z^j) + \frac{1}{2\lambda}\|z - \bar{z}^{j+1}\|^2, \quad \forall z \in \mathcal{Z}.$$

Taking $z = z^j$ gives

$$\mathcal{L}_{\gamma,\lambda}(z^j; z^j) \geq \mathcal{L}_{\gamma,\lambda}(\bar{z}^{j+1}; z^j) + \frac{1}{2\lambda}\|z^j - \bar{z}^{j+1}\|^2.$$

Recalling that $\mathcal{L}_{\gamma,\lambda}(z^j; z^j) = F_\gamma(z^j)$ and using (15), this becomes

$$F_\gamma(z^j) \geq \mathcal{L}_{\gamma,\lambda}(\bar{z}^{j+1}; z^j) + \frac{\lambda}{2}\|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2. \quad (\text{A1})$$

Next, by the definition of δ_j -approximate solution, $\mathcal{L}_{\gamma,\lambda}(\bar{z}^{j+1}; z^j) \geq \mathcal{L}_{\gamma,\lambda}(z^{j+1}; z^j) - \delta_j$. Substituting this into (A1) yields

$$F_\gamma(z^j) \geq \mathcal{L}_{\gamma,\lambda}(z^{j+1}; z^j) - \delta_j + \frac{\lambda}{2}\|\mathcal{G}_{\gamma,\lambda}(z^j)\|^2.$$

Finally, by Lemma 2 in the rearranged form, with $z = z^{j+1}$ and $w = z^j$, and using the stepsize condition $\lambda^{-1} \geq L_\gamma$, we obtain

$$F_\gamma(z^{j+1}) \leq \mathcal{L}_{\gamma,\lambda}(z^{j+1}; z^j).$$

Combining the last two inequalities proves (19). $\square$

Appendix C. Omitted Proof in Section 5

Appendix C.1. Proof of Proposition 1

Proof. Introduce an auxiliary variable $r = a^j + B^j x$. Then (33) can be equivalently written as

$$\begin{aligned} \min_{x \in \mathbb{R}^{d_x}, r \in \mathbb{R}^{d_x}} \quad & \frac{1}{2\lambda} \|x - v^j\|_2^2 + \gamma \|r\|_1 \\ \text{s.t.} \quad & r = a^j + B^j x. \end{aligned} \quad (\text{A2})$$

The Lagrangian associated with the equality constraint $r = a^j + B^j x$ is

$$\mathcal{L}(x, r; y) = \frac{1}{2\lambda} \|x - v^j\|_2^2 + \gamma \|r\|_1 + \langle y, a^j + B^j x - r \rangle,$$

where $y \in \mathbb{R}^{d_x}$ is the Lagrange multiplier.

The dual function is obtained by minimizing the Lagrangian over x and r :

$$q(y) := \inf_{x \in \mathbb{R}^{d_x}, r \in \mathbb{R}^{d_x}} \mathcal{L}(x, r; y).$$

First, consider the minimization with respect to r : $\inf_{r \in \mathbb{R}^{d_x}} \{\gamma \|r\|_1 - \langle y, r \rangle\}$. By the conjugacy relation of the ℓ_1 -norm, we have

$$\inf_{r \in \mathbb{R}^{d_x}} \{\gamma \|r\|_1 - \langle y, r \rangle\} = \begin{cases} 0, & \|y\|_\infty \leq \gamma, \\ -\infty, & \text{otherwise.} \end{cases}$$

Therefore, the dual feasible set is $\Omega_\gamma = \{y \in \mathbb{R}^{d_x} : \|y\|_\infty \leq \gamma\}$.

Next, for $y \in \Omega_\gamma$, we minimize the remaining part with respect to x :

$$\inf_{x \in \mathbb{R}^{d_x}} \left\{ \frac{1}{2\lambda} \|x - v^j\|_2^2 + \langle (B^j)^\top y, x \rangle + \langle y, a^j \rangle \right\}.$$

The first-order optimality condition with respect to x gives $\frac{1}{\lambda}(x - v^j) + (B^j)^\top y = 0$. Hence

$$x = v^j - \lambda (B^j)^\top y.$$

Substituting this minimizer into the Lagrangian yields

$$q(y) = \langle y, a^j \rangle + \langle (B^j)^\top y, v^j \rangle - \frac{\lambda}{2} \|(B^j)^\top y\|_2^2.$$

Since $\langle (B^j)^\top y, v^j \rangle = \langle y, B^j v^j \rangle$, we obtain

$$q(y) = \langle y, a^j + B^j v^j \rangle - \frac{\lambda}{2} \|(B^j)^\top y\|_2^2 = \langle y, c^j \rangle - \frac{\lambda}{2} \|(B^j)^\top y\|_2^2.$$

Thus the Lagrange dual problem is

$$\max_{y \in \Omega_\gamma} \left\{ \langle y, c^j \rangle - \frac{\lambda}{2} \|(B^j)^\top y\|_2^2 \right\}.$$

Equivalently, changing maximization into minimization, we obtain

$$\min_{y \in \Omega_\gamma} \left\{ \frac{\lambda}{2} \|(B^j)^\top y\|_2^2 - \langle y, c^j \rangle \right\},$$

which is exactly (34).

Finally, the primal recovery formula follows from the minimizer of the Lagrangian with respect to x , namely

$$x^* = v^j - \lambda(B^j)^\top y^*.$$

The proof is complete. $\square$

Appendix C.2. SPG Method for (34)

Algorithm A1 SPG Method for (34).

- 1: **Input:** B^j , c^j , $\gamma > 0$, $\lambda > 0$, $\varepsilon_{\text{spg}} > 0$, y^0 , $\eta_0 > 0$, $\eta_{\max} > \eta_{\min} > 0$, $M \in \mathbb{N}$, $\sigma \in (0, 1)$, $\beta \in (0, 1)$.
 - 2: Set $\ell := 0$ and $r_{\text{spg}}(y^0)$. Project the initial point onto Ω_γ : $y^0 := \Pi_{\Omega_\gamma}(y^0)$.
 - 3: **while** $r_{\text{spg}}(y^\ell) > \varepsilon_{\text{spg}}$ **do**
 - 4: Compute the projected search direction $p^\ell := \Pi_{\Omega_\gamma}(y^\ell - \eta_\ell \nabla d_j(y^\ell)) - y^\ell$.
 - 5: Set $D_\ell := \max_{0 \leq i \leq \min\{\ell, M-1\}} d_j(y^{\ell-i})$. Find the smallest integer $m_\ell \geq 0$ such that $d_j(y^\ell + \alpha_\ell p^\ell) \leq D_\ell + \sigma \alpha_\ell \langle \nabla d_j(y^\ell), p^\ell \rangle$, $\alpha_\ell := \beta^{m_\ell}$.
 - 6: Set $y^{\ell+1} := y^\ell + \alpha_\ell p^\ell$, $s^\ell := y^{\ell+1} - y^\ell$, $q^\ell := \nabla d_j(y^{\ell+1}) - \nabla d_j(y^\ell)$.
 - 7: Update $\eta_{\ell+1} := \min\{\eta_{\max}, \max\{\eta_{\min}, \frac{\langle s^\ell, s^\ell \rangle}{\langle s^\ell, q^\ell \rangle}\}\}$.
 - 8: Set $\ell := \ell + 1$.
 - 9: **end while**
 - 10: Set $y^* := y^\ell$. Recover the primal point by $x^{j+1} := v^j - \lambda(B^j)^\top y^*$.
 - 11: **Output:** the final dual iterate y^* and the recovered primal point x^{j+1} .
-

References

1. Colson, B.; Marcotte, P.; Savard, G. An Overview of Bilevel Optimization. *Ann. Oper. Res.* **2007**, *153*, 235–256. <https://doi.org/10.1007/s10479-007-0176-2>.
2. Sinha, A.; Malo, P.; Deb, K. A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. *IEEE Trans. Evol. Comput.* **2018**, *22*, 276–295. <https://doi.org/10.1109/TEVC.2017.2712906>.
3. Franceschi, L.; Frascioni, P.; Salzo, S.; Grazi, R.; Pontil, M. Bilevel Programming for Hyperparameter Optimization and Meta-Learning. In *Proceedings of the 35th International Conference on Machine Learning*; PMLR: Cambridge, MA, USA, 2018; Volume 80, pp. 1568–1577.
4. Shaban, A.; Cheng, C.A.; Hatch, N.; Boots, B. Truncated Back-propagation for Bilevel Optimization. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*; PMLR: Cambridge, MA, USA, 2019; Volume 89, pp. 1723–1732.
5. Rajeswaran, A.; Finn, C.; Kakade, S.M.; Levine, S. Meta-Learning with Implicit Gradients. *Proc. Adv. Neural Inf. Process. Syst.* **2019**, *32*, 113–124.
6. Liu, R.; Mu, P.; Yuan, X.; Zeng, S.; Zhang, J. A Generic First-Order Algorithmic Framework for Bi-Level Programming Beyond Lower-Level Singleton. In *Proceedings of the 37th International Conference on Machine Learning*; PMLR: Cambridge, MA, USA, 2020; Volume 119, pp. 6305–6315.
7. Kwon, J.; Kwon, D.; Wright, S.; Nowak, R.D. A Fully First-Order Method for Stochastic Bilevel Optimization. In *Proceedings of the 40th International Conference on Machine Learning*; PMLR: Cambridge, MA, USA, 2023; Volume 202, pp. 18083–18113.
8. Ochs, P.; Ranftl, R.; Brox, T.; Pock, T. Techniques for Gradient-Based Bilevel Optimization with Non-smooth Lower Level Problems. *J. Math. Imaging Vis.* **2016**, *56*, 175–194. <https://doi.org/10.1007/s10851-016-0663-7>.
9. Li, Q.; Li, Z.; Zemkoho, A.B. Bilevel Hyperparameter Optimization for Support Vector Classification: Theoretical Analysis and a Solution Method. *Math. Methods Oper. Res.* **2022**, *96*, 315–350. <https://doi.org/10.1007/s00186-022-00798-6>.
10. Li, J.; Li, Q.; Zemkoho, A.B. On Constraint Qualifications for MPECs with Applications to Bilevel Hyperparameter Optimization for Machine Learning. *arXiv* **2025**, arXiv:2508.12850.
11. Qian, Y.; Li, Q.; Zemkoho, A.B. Global Relaxation-based LP–Newton Method for Multiple Hyperparameter Selection in Support Vector Classification with Feature Selection. *arXiv* **2023**, arXiv:2312.10848.
12. Wang, Y.; Li, Q. A Fast Smoothing Newton Method for Bilevel Hyperparameter Optimization for SVC with Logistic Loss. *Optimization* **2025**, *74*, 2793–2822. <https://doi.org/10.1080/02331934.2024.2394612>.
13. Wang, Y.; Li, Q.; Zhang, L. An Efficient Smoothing Damped Newton Method for Large-Scale Mathematical Programs with Equilibrium Constraints. *arXiv* **2025**, arXiv:2506.22603.

14. Zheng, Y.; Li, Q. Bilevel Models for Adversarial Learning and a Case Study. *Mathematics* **2025**, *13*, 3910. <https://doi.org/10.3390/math13243910>.
15. Ye, J.J.; Zhu, D.L.; Zhu, Q.J. Exact Penalization and Necessary Optimality Conditions for Generalized Bilevel Programming Problems. *Siam J. Optim.* **1997**, *7*, 481–507. <https://doi.org/10.1137/S1052623493257344>.
16. Lu, Z.; Mei, S. First-Order Penalty Methods for Bilevel Optimization. *Siam J. Optim.* **2024**, *34*, 1937–1969. <https://doi.org/10.1137/23M1566753>.
17. Shen, H.; Xiao, Q.; Chen, T. On Penalty-based Bilevel Gradient Descent Method. *Math. Program.* **2025**, *214*, 539–589. <https://doi.org/10.1007/s10107-025-02194-4>.
18. Chen, P.; Shi, X.; Jiang, R.; Wang, J. Penalty-based Methods for Simple Bilevel Optimization under Hölderian Error Bounds. In *Proceedings of the Advances in Neural Information Processing Systems*; Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C., Eds.; Neural Information Processing Systems: Red Hook, NY, USA, 2024; Volume 37. <https://doi.org/10.52202/079017-4466>.
19. Solodov, M.V. An Explicit Descent Method for Bilevel Convex Optimization. *J. Convex Anal.* **2007**, *14*, 227–237.
20. Beck, A.; Sabach, S. A First Order Method for Finding Minimal Norm-Like Solutions of Convex Optimization Problems. *Math. Program.* **2014**, *147*, 25–46. <https://doi.org/10.1007/s10107-013-0708-2>.
21. Helou, E.S.; Simões, L.E.A. ϵ -subgradient Algorithms for Bilevel Convex Optimization. *Inverse Probl.* **2017**, *33*, 055020.
22. Amini, M.; Yousefian, F. An Iterative Regularized Incremental Projected Subgradient Method for a Class of Bilevel Optimization Problems. In *Proceedings of the 2019 American Control Conference (ACC)*; IEEE: New York, NY, USA, 2019; pp. 4069–4074. <https://doi.org/10.23919/ACC.2019.8814637>.
23. Kaushik, H.D.; Yousefian, F. A Method with Convergence Rates for Optimization Problems with Variational Inequality Constraints. *Siam J. Optim.* **2021**, *31*, 2171–2198. <https://doi.org/10.1137/20M1357378>.
24. Sabach, S.; Shtern, S. A First Order Method for Solving Convex Bilevel Optimization Problems. *Siam J. Optim.* **2017**, *27*, 640–660. <https://doi.org/10.1137/16M105592X>.
25. Shehu, Y.; Vuong, P.T.; Zemkoho, A. An Inertial Extrapolation Method for Convex Simple Bilevel Optimization. *Optim. Methods Softw.* **2021**, *36*, 1–19. <https://doi.org/10.1080/10556788.2019.1619729>.
26. Doron, L.; Shtern, S. Methodology and First-Order Algorithms for Solving Nonsmooth and Non-strongly Convex Bilevel Optimization Problems. *Math. Program.* **2023**, *201*, 521–558. <https://doi.org/10.1007/s10107-022-01914-4>.
27. Shen, L.; Ho-Nguyen, N.; Kılınç-Karzan, F. An Online Convex Optimization-based Framework for Convex Bilevel Optimization. *Math. Program.* **2023**, *198*, 1519–1582. <https://doi.org/10.1007/s10107-022-01894-5>.
28. Merchav, R.; Sabach, S. Convex Bi-Level Optimization Problems with Nonsmooth Outer Objective Function. *Siam J. Optim.* **2023**, *33*, 3114–3142. <https://doi.org/10.1137/22M1533608>.
29. Jiang, R.; Abolfazli, N.; Mokhtari, A.; Yazdandoost Hamedani, E. A Conditional Gradient-based Method for Simple Bilevel Optimization with Convex Lower-level Problem. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*; PMLR: Cambridge, MA, USA, 2023; Volume 206, pp. 10305–10323.
30. Cao, J.; Jiang, R.; Yazdandoost Hamedani, E.; Mokhtari, A. An Accelerated Gradient Method for Convex Smooth Simple Bilevel Optimization. In *Proceedings of the Advances in Neural Information Processing Systems*; Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C., Eds.; Neural Information Processing Systems: Red Hook, NY, USA, 2024; Volume 37. <https://doi.org/10.52202/079017-1434>.
31. Wang, J.; Shi, X.; Jiang, R. Near-Optimal Convex Simple Bilevel Optimization with a Bisection Method. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*; PMLR: Cambridge, MA, USA, 2024; Volume 238, pp. 2008–2016.
32. Zhang, H.; Chen, L.; Xu, J.; Zhang, J. Functionally Constrained Algorithm Solves Convex Simple Bilevel Problems. In *Proceedings of the Advances in Neural Information Processing Systems*; Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C., Eds.; Neural Information Processing Systems: Red Hook, NY, USA, 2024; Volume 37. <https://doi.org/10.52202/079017-1836>.
33. Giang-Tran, K.H.; Ho-Nguyen, N.; Lee, D. A Projection-Free Method for Solving Convex Bilevel Optimization Problems. *Math. Program.* **2025**, *213*, 907–940. <https://doi.org/10.1007/s10107-024-02157-1>.
34. Jin, J.; Zhu, J.; Zhao, L.; Chen, L. A fixed-time convergent and noise-tolerant zeroing neural network for online solution of time-varying matrix inversion. *Appl. Soft Comput.* **2022**, *130*, 109691. <https://doi.org/10.1016/j.asoc.2022.109691>.
35. Wu, X.; Qin, W.; Tang, S. Analyzing the impact of threshold strategies on cancer treatment using non-smooth models. *Math. Biosci.* **2025**, *390*, 109559. <https://doi.org/10.1016/j.mbs.2025.109559>.
36. Shao, S.; Tian, Y.; Zhang, Y.; Zhang, X. Knowledge Learning-Based Dimensionality Reduction for Solving Large-Scale Sparse Multiobjective Optimization Problems. *IEEE Trans. Cybern.* **2025**, *55*, 3471–3484. <https://doi.org/10.1109/TCYB.2025.3558354>.
37. Huang, D.; Shi, X.; Feng, Y.; Zhao, C.; Wen, J.; Shang, M. Automatic Compiler Tuning for Inlining with Machine Learning. *Int. J. Pattern Recognit. Artif. Intell.* **2025**, *39*, 2552016. <https://doi.org/10.1142/S0218001425520160>.
38. Rockafellar, R.T.; Wets, R.J.B. *Variational Analysis*; Grundlehren der Mathematischen Wissenschaften; Springer: Berlin/Heidelberg, Germany, 1998; Volume 317.

39. Karimi, H.; Nutini, J.; Schmidt, M. Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak–ojasiewicz Condition. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 795–811.
40. Drusvyatskiy, D.; Paquette, C. Efficiency of minimizing compositions of convex functions and smooth maps. *Math. Program.* **2019**, *178*, 503–558.
41. Birgin, E.G.; Martínez, J.M.; Raydan, M. Nonmonotone Spectral Projected Gradient Methods on Convex Sets. *Siam J. Optim.* **2000**, *10*, 1196–1211. <https://doi.org/10.1137/S1052623497330963>.
42. Birgin, E.G.; Martínez, J.M.; Raydan, M. Algorithm 813: SPG—Software for Convex-Constrained Optimization. *ACM Trans. Math. Softw.* **2001**, *27*, 340–349. <https://doi.org/10.1145/502800.502803>.
43. Gong, C.; Liu, X. Bi-objective trade-off with dynamic barrier gradient descent. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*; Neural Information Processing Systems: Red Hook, NY, USA, 2021.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.