Efficient Gradient Approximation Method for Constrained Bilevel Optimization

Siyuan Xu, Minghui Zhu*

School of Electrical Engineering and Computer Science
The Pennsylvania State University, University Park, USA
{spx5032, muz16}@psu.edu

Abstract

Bilevel optimization has been developed for many machine learning tasks with large-scale and high-dimensional data. This paper considers a constrained bilevel optimization problem, where the lower-level optimization problem is convex with equality and inequality constraints and the upper-level optimization problem is non-convex. The overall objective function is non-convex and non-differentiable. To solve the problem, we develop a gradient-based approach, called gradient approximation method, which determines the descent direction by computing several representative gradients of the objective function inside a neighborhood of the current estimate. We show that the algorithm asymptotically converges to the set of Clarke stationary points, and demonstrate the efficacy of the algorithm by the experiments on hyperparameter optimization and meta-learning.

1 Introduction

A general constrained bilevel optimization problem is formulated as follows:

$$\begin{aligned} &\min_{x\in\mathbb{R}^{d_x}} \ \Phi(x) = f\left(x, y^*(x)\right) \\ &\text{s.t. } \ r\left(x, y^*(x)\right) \leq 0; \ s\left(x, y^*(x)\right) = 0; \\ &y^*(x) = \arg\min_{y\in\mathbb{R}^{d_y}} \{g(x,y) : p\left(x,y\right) \leq 0; q\left(x,y\right) = 0\}. \end{aligned} \tag{1}$$

The bilevel optimization minimizes the overall objective function $\Phi(x)$ with respect to (w.r.t.) x, where $y^*(x)$ is the optimal solution of the lower-level optimization problem and parametric in the upper-level decision variable x. In this paper, we assume that $y^*(x)$ is unique for any $x \in \mathbb{R}^{d_x}$.

Existing methods to solve problem (1) can be categorized into two classes: single-level reduction methods (Bard and Falk 1982; Bard and Moore 1990; Shi, Lu, and Zhang 2005) and descent methods (Savard and Gauvin 1994; Dempe 1998). Single-level reduction methods use the KKT conditions to replace the lower-level optimization problem when it is convex. Then, they reformulate the bilevel optimization problem (1) as a single-level constrained optimization problem. Descent methods aim to find descent directions in which the new point is feasible and meanwhile reduces the objective function. Paper (Savard and Gauvin 1994) computes a descent direction of the objective function by solving a quadratic program. Paper (Dempe 1998) applies the gradient of the objective function computed in (Kolstad and Lasdon 1990; Fiacco and McCormick 1970) to compute a generalized Clarke Jacobian, and uses a bundle method (Schramm and Zowe 1992) for the optimization. When applied to machine learning, bilevel optimization faces additional challenges as the dimensions of decision variables in the upper-level and lower-level problems are high (Liu et al. 2021a).

Gradient-based methods have been shown to be effective in handling large-scale and high-dimensional data in a variety of machine learning tasks (Bottou and Bousquet 2008). They have been extended to solve the bilevel optimization problem where there is no constraint in the lower-level optimization. The methods can be categorized into the approximate implicit differentiation (AID) based approaches (Pedregosa 2016; Gould et al. 2016; Ghadimi and Wang 2018; Grazzi et al. 2020) and the iterative differentiation (ITD) approaches (Grazzi et al. 2020; Franceschi et al. 2017, 2018; Shaban et al. 2019; Ji, Yang, and Liang 2021). The AID based approaches evaluate the gradients of $y^*(x)$ and $\Phi(x)$ based on implicit differentiation (Bengio 2000). The ITD based approaches treat the iterative optimization steps in the lower-level optimization as a dynamical system, impose $y^*(x)$ as its stationary point, and compute $\nabla y^*(x)$ at each iterative step. The gradient-based algorithms have been applied to solve several machine learning tasks, including meta-learning (Franceschi et al. 2018; Rajeswaran et al. 2019; Ji et al. 2020), hyperparameter optimization (Pedregosa 2016; Franceschi et al. 2017, 2018), reinforcement learning (Hong et al. 2020; Konda and Tsitsiklis 2000), and network architecture search (Liu, Simonyan, and Yang 2018). The above methods are limited to unconstrained bilevel optimization and require the objective function to be differentiable. They cannot be directly applied when constraints are present in the lower-level optimization, as the objective function is non-differentiable.

Contributions. In this paper, we consider a special case of problem (1) where the upper-level constraints r and s are not included. In general, the objective function Φ is nonconvex and non-differentiable, even if the upper-level and lower-level problems are convex and functions f, g, p, q are differentiable (Hansen, Jaumard, and Savard 1992; Liu et al. 2021a). Most

*Corresponding author.
Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

methods for this bilevel optimization problem are highly complicated and computationally expensive, especially when the dimension of the problem is large (Liu et al. 2021a; Dempe and Franke 2016). Addressing the challenge, we determine the descent direction by computing several gradients which can represent the gradients of the objective function of all points in a ball, and develop a computationally efficient algorithm with convergence guarantee for the constrained bilevel optimization problem. The overall contributions are summarized as follows. (i) Firstly, we derive the conditions under which the lower-level optimal solution $y^*(x)$ is continuously differentiable or directional differentiable. In addition, we provide analytical expressions for the gradient of $y^*(x)$ when it is continuously differentiable and the directional derivative of $y^*(x)$ when it is directional differentiable. (ii) Secondly, we propose the gradient approximation method, which applies the Clarke subdifferential approximation of the non-convex and non-differentiable objective function Φ to the line search method. In particular, a set of derivatives is used to approximate the gradients or directional derivatives on all points in a neighborhood of the current estimate. Then, the Clarke subdifferential is approximated by the derivatives, and the approximate Clarke subdifferential is employed as the descent direction for line search. (iii) It is shown that, the Clarke subdifferential approximation errors are small, the line search is always feasible, and the algorithm asymptotically converges to the set of Clarke stationary points. (iv) We empirically verify the efficacy of the proposed algorithm by conducting experiments on hyperparameter optimization and meta-learning.

Related Works. Differentiation of the optimal solution of a constrained optimization problem has been studied for a long time. Sensitivity analysis of constrained optimization (Fiacco 1983; Fiacco and Ishizuka 1990; Fiacco and McCormick 1970) shows the optimal solution $y^*(x)$ of a convex optimization problem is directional differentiable but may not differentiable at all points. It implies that the objective function $\Phi(x)$ in problem (1) may not be differentiable. Based on the implicit differentiation of the KKT conditions, the papers also compute $\nabla y^*(x)$ when y^* is differentiable at x. Optnet (Amos, Xu, and Kolter 2017; Amos and Kolter 2017; Agrawal et al. 2019) applies the gradient computation to the constrained bilevel optimization, where a deep neural network is included in the upper-level optimization problem. In particular, the optimal solution $y^*(x)$ serves as a layer in the deep neural network and $\nabla y^*(x)$ is used as the backpropagation gradients to optimize the neural network parameters. However, all the above methods do not explicitly consider the non-differentiability of $y^*(x)$ and $\Phi(x)$, and cannot guarantee convergence. Recently, papers (Liu et al. 2021b; Sow et al. 2022) consider that the lower-level optimization problem has simple constraints, such that projection onto the constraint set can be easily computed, and require that the constraint set is bounded. In this paper, we consider inequality and equality constraints, which are more general than those in (Liu et al. 2021b; Sow et al. 2022).

Notations. Denote $a > b$ for vectors $a, b \in \mathbb{R}^n$, when $a_i > b_i$ for all $1 \leq i \leq n$. Notations $a \geq b$, $a = b$, $a \leq b$, and $a < b$ are defined in an analogous way. Denote the l_2 norm of vectors by $\|\cdot\|$. The directional derivative of a function f at x on the direction d with $\|d\| = 1$ is defined as $\nabla_d f(x) \triangleq \lim_{h \to 0^+} \frac{f(x+hd)-f(x)}{h}$. A ball centered at x with radius ϵ is denoted as $\mathcal{B}(x, \epsilon)$. The complementary set of a set S is denoted as S^C. The distance between the point x and the set S is defined as $d(x, S) \triangleq \inf\{\|x - a\| \mid a \in S\}$. The convex hull of S is denoted by $\operatorname{conv} S$. For set S and function f, we define the image set $f(S) \triangleq \{f(x) \mid x \in S\}$. For a finite positive integer set I and a vector function p, we denote the subvector function $p_I \triangleq [p_{k_1}, \cdots, p_{k_j}, \cdots]^\top$ where $k_j \in I$.

2 Problem Statement

Consider the constrained bilevel optimization problem:

$$\min_{x \in \mathbb{R}^{d_x}} \Phi(x) = f\left(x, y^*(x)\right) \tag{2}$$
$$\text{s.t. } y^*(x) = \arg\min_{y \in \mathbb{R}^{d_y}} \{g(x, y) : p\left(x, y\right) \leq 0; q\left(x, y\right) = 0\},$$

where $f, g : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}$; $p : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}^m$; $q : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}^n$. Given $x \in \mathbb{R}^{d_x}$, we denote the lower-level optimization problem in (2) as $P(x)$. The feasible set of $P(x)$ is defined as $K\left(x\right) \triangleq \{y \in \mathbb{R}^{d_y} : p\left(x, y\right) \leq 0, q\left(x, y\right) = 0\}$. Suppose the following assumptions hold.

Assumption 1. *The functions f, g, p and q are twice continuously differentiable.*

Assumption 2. *For all $x \in \mathbb{R}^{d_x}$, the function $g(x, y)$ is μ-strongly-convex w.r.t. y; $p_j(x, y)$ is convex w.r.t. y for each j; $q_i(x, y)$ is affine w.r.t. y for each i.*

Note that the upper-level objective function $f(x, y)$ and the overall objective function $\Phi(x)$ are non-convex. The lower-level problem $P(x)$ is convex and its Lagrangian is $\mathcal{L}(y, \lambda, \nu, x) \triangleq g(x, y) + \lambda^\top p(x, y) + \nu^\top q(x, y)$, where (λ, ν) are Lagrange multipliers and $\lambda \geq 0$.

Definition 1. *Suppose that the KKT conditions hold at y for $P(x)$ with the Lagrangian multipliers λ and ν. The set of active inequality constraints at y for $P(x)$ is defined as: $J(x, y) \triangleq \{j : 1 \leq j \leq m,\ p_j(x, y) = 0\}$. An inequality constraint is called inactive if it is not included in $J(x, y)$ and the set of inactive constraints is denoted as $J(x, y)^C$. The set of strictly active inequality constraints at y is defined as: $J^+(x, y, \lambda) \triangleq \{j : j \in J\left(x, y\right),\ \lambda_j > 0\}$. The set of non-strictly active inequality constraints at y is defined as: $J^0(x, y, \lambda) \triangleq J(x, y) \setminus J^+(x, y, \lambda)$. Notice that $\lambda_j \geq 0$ for $j \in J(x, y)$ and $\lambda_j = 0$ for $j \in J^0(x, y, \lambda)$.*

Definition 2. *The Linear Independence Constraint Qualification (LICQ) holds at y for $P(x)$ if the vectors $\{\nabla_y p_j\left(x, y\right), j \in J\left(x, y\right); \nabla_y q_i\left(x, y\right), 1 \leq i \leq n\}$ are linearly independent.*

Assumption 3. *Suppose that for all $x \in \mathbb{R}^{d_x}$, the solution $y^*(x)$ exists for $P\left(x\right)$, and the LICQ holds at $y^*(x)$ for $P(x)$.*

3 Differentiability and Gradient of $y^*(x)$

In this section, we provide sufficient conditions under which the lower-level optimal solution $y^*(x)$ is continuously differentiable or directional differentiable. We compute the gradient of $y^*(x)$ when it is continuously differentiable and the

directional derivative of $y^*(x)$ when it is directional differentiable. Moreover, we give a necessary condition that $y^*(x)$ is not differentiable and illustrate it by a numerical example.

In problem (2), if the upper-level objective function f and the solution of lower-level problem y^* are continuously differentiable, so is Φ, and by the gradient computation of composite functions, we have

$$\nabla\Phi(x) = \nabla_x f(x, y^*(x)) + \nabla y^*(x)^\top \nabla_y f(x, y^*(x)). \quad (3)$$

It is shown in (Domke 2012) that, when p and q are absent, y^* and Φ are differentiable under certain assumptions. The differentiability of y^* and Φ is used by the AID based approaches in (Pedregosa 2016; Gould et al. 2016; Ghadimi and Wang 2018; Grazzi et al. 2020; Domke 2012) to approximate ∇y^* and minimize Φ by gradient descent. However, it is not the case as the lower-level problem (2) is constrained.

Theorem 1 states the conditions under which $y^*(x)$ is directional differentiable.

Theorem 1. *Suppose Assumptions 1, 2, 3 hold. The following properties hold for any x.*

(i) The global minimum $y^*(x)$ of $P(x)$ exists and is unique. The KKT conditions hold at $y^*(x)$ with unique Lagrangian multipliers $\lambda(x)$ and $\nu(x)$.

(ii) The vector function $z(x) \triangleq [y^*(x)^\top, \lambda(x)^\top, \nu(x)^\top]^\top$ is continuous and locally Lipschitz. The directional derivative of $z(x)$ on any direction exists.

As shown in part (i) of Theorem 1, $y^*(x)$, $\lambda(x)$ and $\nu(x)$ are uniquely determined by x. So we simplify the notations of Definition 1 in the rest of this paper: $J(x, y^*(x))$ is denoted as $J(x)$, $J^+(x, y^*(x), \lambda(x))$ is denoted as $J^+(x)$, and $J^0(x, y^*(x), \lambda(x))$ is denoted as $J^0(x)$. In part (ii), the computation of the directional derivative of $z(x)$ is given in Theorem 6 in Appendix C.

Definition 3. *Suppose that the KKT conditions hold at y for $P(x)$ with the Lagrangian multipliers λ and ν. The Strict Complementarity Slackness Condition (SCSC) holds at y w.r.t. λ for $P(x)$, if $\lambda_j > 0$ for all $j \in J(x, y)$.*

Remark 1. *The KKT conditions include the Complementarity Slackness Condition (CSC). The SCSC is stronger than the CSC, which only requires that $\lambda_j \geq 0$ for all $j \in J(x, y)$.*

Theorem 2 states the conditions under which $y^*(x)$ is continuously differentiable and derives $\nabla y^*(x)$.

Theorem 2. *Suppose Assumptions 1, 2, 3 hold. If the SCSC holds at $y^*(x)$ w.r.t. $\lambda(x)$, then $z(x)$ is continuously differentiable at x and the gradient is computed as*

$$\left[\nabla_x y^*(x)^\top, \nabla_x \lambda_{J(x)}^\top(x), \nabla_x \nu(x)^\top\right]^\top = -M_+^{-1}(x) N_+(x) \quad (4)$$

and $\nabla_x \lambda_{J(x)^C}(x) = 0$, where $M_+(x) \triangleq$

$$\begin{bmatrix} \nabla_y^2 \mathcal{L} & \nabla_y p_{J^+(x)}^\top & \nabla_y q^\top \\ \nabla_y p_{J^+(x)} & 0 & 0 \\ \nabla_y q & 0 & 0 \end{bmatrix} (x, y^*(x), \lambda(x), \nu(x))$$

is nonsingular and $N_+(x) \triangleq$

$$[\nabla_{xy}^2 \mathcal{L}^\top, \nabla_x p_{J^+(x)}^\top, \nabla_x q^\top]^\top (x, y^*(x), \lambda(x), \nu(x)).$$

Theorem 2 shows that, if $z(x)$ is not continuously differentiable, then the SCSC does not hold at $y^*(x)$ w.r.t. $\lambda(x)$. Definition 3 implies that the SCSC holds at y w.r.t. λ for $P(x)$ if and only if $J^0(x) = \emptyset$. It concludes that if $y^*(x)$ is not continuously differentiable at x, $J^0(x) \neq \emptyset$, i.e., the non-differentiability of $y^*(x)$ occurs at points with non-strictly active constraints. Example 1 illustrates such claim.

Example 1. *Consider a bilevel optimization problem $\Phi(x) = y^*(x)$ and the lower-level problem $P(x)$: $y^*(x) = \arg\min_y\{(y-x^2)^2 : p_1(x, y) = -x-y \leq 0\}$, where $x, y \in \mathbb{R}$. The analytical solution of $z(x) = [y^*(x), \lambda(x)]$ is given by: $y^*(x) = x^2$, $\lambda(x) = 0$ when $x \in (-\infty, -1] \cup [0, +\infty)$; $y^*(x) = -x$, $\lambda(x) = -2x(1+x)$ when $x \in [-1, 0]$. Correspondingly, when $x \in (-1, 0)$, $J(x) = \{1\}$, $J^+(x) = \{1\}$, $J^0(x) = \emptyset$; when $x \in (-\infty, -1) \cup (0, +\infty)$, $J(x) = \emptyset$, $J^+(x) = \emptyset$, $J^0(x) = \emptyset$; when $x \in \{-1, 0\}$, $J(x) = \{1\}$, $J^+(x) = \emptyset$, $J^0(x) = \{1\}$. As shown in Fig. 1, $y^*(x)$ is continuously differentiable everywhere except when $J^0(x) \neq \emptyset$.*

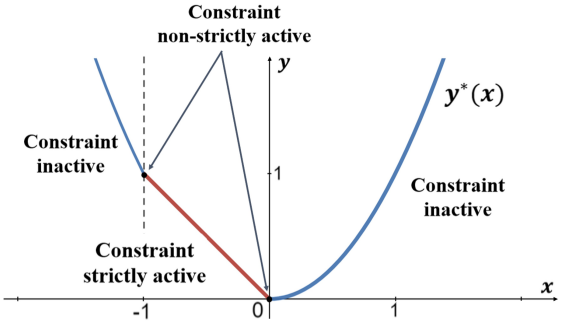

Figure 1: Occurrence of non-differentiability.

The computation of the gradient of $z(x)$ in (4) is derived from the implicit differentiation of the KKT conditions of problem $P(x)$, which is also used in (Fiacco and Ishizuka 1990; Amos and Kolter 2017; Agrawal et al. 2019). Compared with these papers, Theorem 2 directly determines $\nabla_x \lambda_{J(x)^C}(x) = 0$ and excludes $\lambda_{J(x)^C}(x)$ from the computation of the inverse matrix in (4), when $z(x)$ is continuously differentiable. Theorem 6 in Appendix C derives the directional derivative of $z(x)$ when it is not differentiable.

Consider a special case where the lower-level optimization problem $P(x)$ is unconstrained. Since the SCSC is not needed anymore, the assumptions in Theorem 2 reduce to that g is twice continuously differentiable and $g(x, y)$ is μ-strongly-convex w.r.t. y for $x \in \mathbb{R}^{d_x}$. By Theorem 2, the optimal solution $y^*(x)$ is continuously differentiable, the matrix $\nabla_y^2 g(x, y)$ is non-singular, and the gradient is computed as $\nabla y^*(x) = -[\nabla_y^2 g(x, y)]^{-1} \nabla_{xy}^2 g(x, y)$. These results are well-known and widely used in unconstrained bilevel optimization analysis and applications (Pedregosa 2016; Franceschi et al. 2017, 2018; Ji, Yang, and Liang 2021).

4 The Gradient Approximation Method

In this section, we develop the gradient approximation method to efficiently solve problem (2), whose objective function is non-differentiable and non-convex. First, we define the

Clarke subdifferential (Section 4.1) and efficiently approximate the Clarke subdifferential of the objective function $\Phi(x)$ (Section 4.2). Next, we propose the gradient approximation algorithm, provide its convergence guarantee (Section 4.3), and present its implementation details (Section 4.4).

4.1 Clarke Subdifferential of Φ

As shown in Section 2 and also shown in (Dempe and Franke 2016; Liu et al. 2021a), the objective function $\Phi(x)$ of problem (2) is usually non-differentiable and non-convex. To deal with the non-smoothness and non-convexity, we introduce Clarke subdifferential and Clarke stationary point.

Definition 4 (Clarke subdifferential and Clarke stationary point (Clarke 1975))**.** *For a locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$, the Clarke subdifferential of f at x is defined by the convex hull of the limits of gradients of f on sequences converging to x, i.e., $\bar{\partial} f(x) \triangleq \operatorname{conv}\left\{\lim_{j\to\infty} \nabla f\left(y^j\right) : \left\{y^j\right\} \to x \text{ where } f \text{ is differentiable at } y^j \text{ for all } j \in \mathbb{N}\right\}$. The Clarke ϵ-subdifferential of f at x is defined by $\bar{\partial}_\epsilon f(x) \triangleq \operatorname{conv}\{\bar{\partial} f(x') : x' \in \mathcal{B}(x, \epsilon)\}$. A point x is Clarke stationary for f if $0 \in \bar{\partial} f(x)$.*

If y^* is differentiable at x, we have $\bar{\partial} y^*(x) = \{\nabla y^*(x)\}$ and $\bar{\partial}\Phi(x) = \{\nabla_x f(x, y^*(x)) + \nabla y^*(x)^\top \nabla_y f(x, y^*(x))\}$; otherwise, $\bar{\partial}\Phi(x) = \{\nabla_x f(x, y^*(x)) + w^\top \nabla_y f(x, y^*(x)) : w \in \bar{\partial} y^*(x)\}$. Take the functions shown in Example 1 and Fig. 1 as an example, $\bar{\partial}_\epsilon \Phi(-1) = \bar{\partial}_\epsilon y^*(-1) = \operatorname{conv}\{[-2-2\epsilon, -2] \cup \{-1\}\} = [-2-2\epsilon, -1]$, and $\bar{\partial}_\epsilon \Phi(0) = \bar{\partial}_\epsilon y^*(0) = \operatorname{conv}\{[0, 2\epsilon] \cup \{-1\}\} = [-1, 2\epsilon]$.

4.2 Clarke Subdifferential Approximation

Gradient-based methods have been applied to convex and non-convex optimization problems (Hardt, Recht, and Singer 2016; Nemirovski et al. 2009). The convergence requires that the objective function is differentiable. If there exist points where the objective function is not differentiable, the probability for the algorithms to visit these points is non-zero and the gradients at these points are not defined (Bagirov et al. 2020). Moreover, oscillation may occur even if the objective function is differentiable at all visited points (Bagirov, Karmitsa, and Mäkelä 2014).

To handle the non-differentiability, the gradient sampling method (Burke, Lewis, and Overton 2005; Kiwiel 2007) uses gradients in a neighborhood of the current estimate to approximate the Clarke subdifferential and determine the descent direction. Specifically, the method samples a set of points inside the neighborhood $\mathcal{B}(x^0, \epsilon)$, select the points where the objective function is differentiable, and then compute the convex hull of the gradients on the sampled points.

However, in problem (2), the point sampling is highly computationally expensive. For each sampled point x^j, to check the differentiability of Φ, we need to solve the lower-level optimization $P(x^j)$ to obtain $y^*(x^j)$, $\lambda(x^j)$ and $\nu(x^j)$, and check the SCSC. Moreover, after the points are sampled, the gradient on each point is computed by (4). As the dimension d_x increases, the sampling number increases to ensure the accuracy of the approximation. More specifically, as shown in (Kiwiel 2007), the algorithm is convergent if the sampling

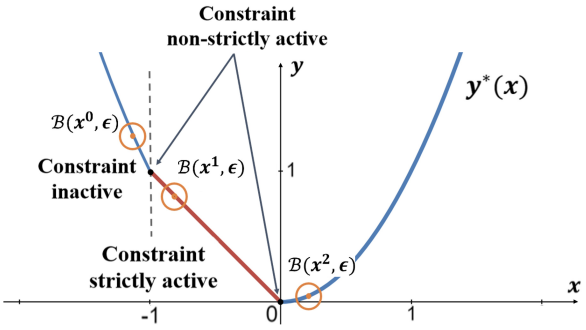

Figure 2: The SCSC holds on $\mathcal{B}(x^0, \epsilon)$

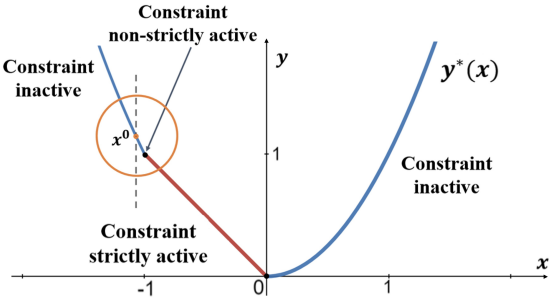

Figure 3: Subsets inside $\mathcal{B}(x^0, \epsilon)$

number is large than $d_x + 1$. The above procedure is executed in each optimization iteration.

Addressing the computational challenge, we approximate the Clarke ϵ-subdifferential by a small number of gradients, which can represent the gradients on all points in the neighborhood. The following propositions distinguish two cases: Φ is continuously differentiable on $\mathcal{B}(x^0, \epsilon)$ (Proposition 1) and it is not (Proposition 2).

Proposition 1. *Suppose Assumptions 1, 2, 3 hold. Consider $x^0 \in R^{d_x}$. There is sufficiently small $\epsilon > 0$ such that, if the SCSC holds at $y^*(x)$ w.r.t. $\lambda(x)$ for any $x \in \mathcal{B}(x^0, \epsilon)$, then $\nabla\Phi(x^0) \in \bar{\partial}_\epsilon \Phi(x^0)$ and*

$$|\|\nabla\Phi(x^0)\| - d(0, \bar{\partial}_\epsilon \Phi(x^0))| < o(\epsilon).$$

Proposition 1 shows that the gradient $\nabla\Phi$ at a single point x^0 can be used to approximate the Clarke ϵ-subdifferential $\bar{\partial}_\epsilon \Phi(x^0)$ and the approximation error is in the order of ϵ. Recall that the gradient $\nabla\Phi(x^0)$ can be computed by (3) and (4). Fig. 2 illustrates the approximation on the problem in Example 1. The SCSC holds at $y^*(x)$ and $\Phi(x)$ is continuously differentiable on $\mathcal{B}(x^0, \epsilon)$, then $\bar{\partial}_\epsilon \Phi(x^0) = [2x^0 - 2\epsilon, 2x^0 + 2\epsilon]$ can be approximated by $\nabla\Phi(x^0) = 2x^0$, and the approximation error is 2ϵ. The approximations of $\bar{\partial}_\epsilon \Phi(x^1)$ and $\bar{\partial}_\epsilon \Phi(x^2)$ can be done in an analogous way.

Consider $\Phi(x)$ is not continuously differentiable at some points in $\mathcal{B}(x^0, \epsilon)$. Define the set $I^\epsilon(x^0)$ which contains all j such that there exist x', $x'' \in \mathcal{B}(x^0, \epsilon)$ with $j \in J^+(x')^C$ and $j \in J^+(x'')$. Define the set $I^\epsilon_+(x^0)$ which contains all j such that $j \in J^+(x)$ for any $x \in \mathcal{B}(x^0, \epsilon)$. If $I^\epsilon(x^0)$ is not empty, there exists a point $x \in \mathcal{B}(x^0, \epsilon)$ such that the SCSC does not hold at $y^*(x)$. The power set of

$I^\epsilon(x^0)$ partitions $\mathcal{B}(x^0,\epsilon)$ into a number of subsets, where $\Phi(x)$ and $y^*(x)$ are continuously differentiable in each subset. An illustration on the problem in Example 1 is shown in Fig. 3. The point $x'=-1$ belongs to $(x^0-\epsilon, x^0+\epsilon)$ and the SCSC does not hold at $y^*(x')$. Notice that $I^\epsilon_+(x^0)=\emptyset$ and $I^\epsilon(x^0)=\{1\}$. Then, $I^\epsilon(x^0)=\{1\}$ has the power set $\{S_{(1)}, S_{(2)}\}$ with $S_{(1)}=\emptyset$ and $S_{(2)}=\{1\}$. Then, $\mathcal{B}(x^0,\epsilon)$ is partitioned to two subsets: the subset where the constraint p_1 is inactive (blue side in the ball) which corresponds to $S_{(1)}$, and the subset where the constraint p_1 is strictly active (red side in the ball) which corresponds to $S_{(2)}$. Their boundary is the point x' where the constraint p_1 is non-strictly active. It can be seen that $y^*(x)$ is continuously differentiable on each subset and the gradient variations are small inside the subset when ϵ is small. In contrast, the gradient variations between two subsets are large. Inspired by Proposition 1, we compute a representative gradient to approximate $\nabla y^*(x)$ inside each subset of $\mathcal{B}(x^0,\epsilon)$.

Now we proceed to generalize the above idea. Recall that $\nabla\Phi(x)$ is computed by (3) and f is twice continuously differentiable. Define

$$G(x^0,\epsilon) \triangleq \{\nabla_x f\left(x^0, y^*\left(x^0\right)\right) + w^S(x^0)^\top \nabla_y f\left(x^0, y^*\left(x^0\right)\right) : S \subseteq I^\epsilon(x^0)\}, \tag{5}$$

where $w^S(x^0)$ is obtained by extracting the first d_x rows from matrix $-M^S_\epsilon(x^0, y^*(x^0))^{-1} N^S_\epsilon(x^0, y^*(x^0))$, with

$$M^S_\epsilon \triangleq \begin{bmatrix} \nabla^2_y \mathcal{L} & \nabla_y p^\top_{I^\epsilon_+(x^0)} & \nabla_y q^\top & \nabla_y p_S^\top \\ \nabla_y p_{I^\epsilon_+(x^0)} & 0 & 0 & 0 \\ \nabla_y q & 0 & 0 & 0 \\ \nabla_y p_S & 0 & 0 & 0 \end{bmatrix},$$

and $N^S_\epsilon \triangleq \left[\nabla^2_{xy}\mathcal{L}^\top, \nabla_x p^\top_{I^\epsilon_+(x^0)}, \nabla_x q^\top, \nabla_x p_S^\top\right]^\top$. Here, S is a subset of $I^\epsilon(x^0)$, and $w^S(x^0)$ is the representative gradient to approximate $\nabla y^*(x)$ inside the subset of $\mathcal{B}(x^0,\epsilon)$ which corresponds S. Proposition 2 shows that the Clarke ϵ-subdifferential $\bar\partial_\epsilon y^*(x^0)$ can be approximated by representative gradient set $G(x^0,\epsilon)$, and the approximation error is in the order of ϵ.

Proposition 2. *Suppose Assumptions 1, 2, 3 hold. Consider $x^0\in\mathbb{R}^{d_x}$, and assume there exists a sufficiently small $\epsilon>0$ such that, there exists $x\in\mathcal{B}(x^0,\epsilon)$ such that $y^*(x)$ is not continuously differentiable at x. Then, for any $z\in\mathbb{R}^{d_x}$,*

$$|d(z, \operatorname{conv} G(x^0,\epsilon)) - d(z, \bar\partial_\epsilon\Phi(x^0))| < o(\epsilon).$$

The computation of the representative gradient $w^S(x^0)$ of Example 1 is demonstrated in Fig. 4. Since x^0 is near the boundary of two subsets, Proposition 2 employs $w^{S_{(1)}}(x^0) = \nabla y^*(x^0)$ to approximate the gradients of the subset with the inactive constraint (blue side), and $w^{S_{(2)}}(x^0) = \nabla\tilde y^*(x^0)$ to approximate the gradients in the subset with the strictly active constraint (red side). The twice-differentiable function $\tilde y^*(x)$ is an extension of $y^*(x)$ (refer to the definition of $x^I(\cdot)$ in (12.8) of (Dempe 1998)). The gradients $\nabla y^*(x^0)$ and $\nabla\tilde y^*(x^0)$ are computed in the matrices $-{M^{S_{(1)}}_\epsilon}^{-1} N^{S_{(1)}}_\epsilon$ and $-{M^{S_{(2)}}_\epsilon}^{-1} N^{S_{(2)}}_\epsilon$, respectively. Then, the representative gradients $w^{S_{(1)}}(x^0)$ and $w^{S_{(2)}}(x^0)$ are used to approximate $\bar\partial_\epsilon y^*(x^0)$. Then, we can compute $G(x^0,\epsilon)=\{2x^0,-1\}$ and $\bar\partial_\epsilon\Phi(x^0)=[2x^0-2\epsilon,-1]$ with $-1\in[x^0-\epsilon, x^0+\epsilon]$. The approximation error $|d(z,\operatorname{conv} G(x^0,\epsilon)) - d(z,\bar\partial_\epsilon\Phi(x^0))|$ is smaller than or equal to 2ϵ for any z.

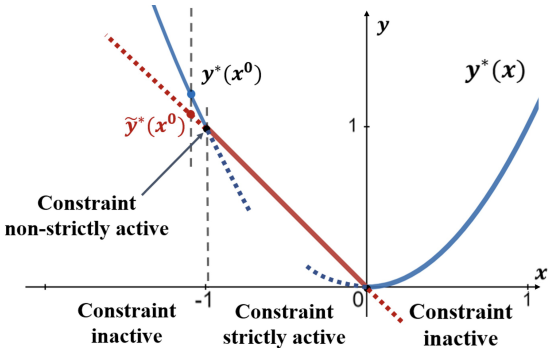

Figure 4: Approximate $\bar\partial_\epsilon y^*(x^0)$

4.3 The Gradient Approximation Algorithm

Our proposed gradient approximation algorithm, summarized in Algorithm 1, is a line search algorithm. It uses the approximation of the Clarke subdifferential as the descent direction for line search. In iteration k, we firstly solve the lower-level optimization problem $P(x^k)$ and obtain $y^*(x^k)$, $\lambda(x^k)$ and $\nu(x^k)$. To reduce the computation complexity, the solution in iteration k serves as the initial point to solve $P(x^{k+1})$ in iteration $k+1$. Secondly, we check the differentiability of y^* on $\mathcal{B}\left(x^k,\epsilon_k\right)$ and its implementation details are shown in Section 4.4. If y^* is continuously differentiable on $\mathcal{B}\left(x^k,\epsilon_k\right)$, we use $\nabla\Phi(x^k)$ to approximate $\bar\partial_\epsilon\Phi(x^k)$ which corresponds to Proposition 1. Otherwise, $G(x^k,\epsilon_k)$ is used which corresponds to Proposition 2. The details of computing $G(x^k,\epsilon_k)$ are shown in (5) and Section 4.4. Thirdly, the line search direction g^k is determined by a vector which has the smallest norm over all vectors in the convex hull of $G(x^k,\epsilon_k)$. During the optimization steps, as the iteration number k increases, the approximation radius ϵ_k decreases. According to Propositions 1 and 2, the approximation error of the Clarke subdifferential is diminishing. We next characterize the convergence of Algorithm 1.

Theorem 3. *Suppose Assumptions 1, 2, 3 hold and $\Phi(x)$ is lower bounded on $\mathbb{R}^{d_x}$. Let $\{x^k\}$ be the sequence generated by Algorithm 1 with $\nu_{\text{opt}}=\epsilon_{\text{opt}}=0$. Then,*

(i) For each k, the line search in line 17 has a solution t_k.
(ii) $\lim_{k\to\infty}\nu_k=0$, $\lim_{k\to\infty}\epsilon_k=0$.
(iii) $\liminf_{k\to\infty} d(0,\bar\partial\Phi(x^k))=0$.
(iv) Every limit point of $\{x^k\}$ is Clarke stationary for Φ.

If the objective function Φ is non-convex but smooth, property (iii) reduces to $\liminf_{k\to\infty}\|\nabla\Phi(x^k)\|=0$, which is a widely used convergence criterion for smooth and non-convex optimization (Nesterov 2003; Jin et al. 2021). A sufficient condition for the existence of limit point of $\{x^k\}$ is that the sequence is bounded.

Algorithm 1: Gradient Approximation Method

Require: Initial point x^0; Initial approximation radius $\epsilon_0 \in (0, \infty)$; Initial stationarity target $\nu_0 \in (0, \infty)$; Line search parameters $(\beta, \gamma) \in (0,1) \times (0,1)$; Termination tolerances $(\epsilon_{\text{opt}}; \nu_{\text{opt}}) \in [0, \infty) \times [0, \infty)$; Discount factors $(\theta_\epsilon, \theta_\nu) \in (0,1) \times (0,1)$.

1: **for** $k \in \mathbb{N}$ **do**
2: Solve the lower-level optimization problem $P(x^k)$ and obtain $y^*(x^k)$, $\lambda(x^k)$, and $\nu(x^k)$
3: Check the differentiability of y^* on $\mathcal{B}\left(x^k, \epsilon_k\right)$ by (6) and (7)
4: **if** y^* is continuously differentiable on $\mathcal{B}\left(x^k, \epsilon_k\right)$ **then**
5: Compute $g^k = \nabla \Phi(x^k)$ by (3)
6: **else**
7: Compute $G(x^k, \epsilon_k)$ by (5)
8: $\bar{\partial}_{\epsilon_k} \Phi(x^k) = \operatorname{conv} G(x^k, \epsilon_k)$
9: Compute $g^k = \min\{\|g\| : g \in \operatorname{conv} G(x^k, \epsilon_k)\}$
10: **end if**
11: **if** $\left\|g^k\right\| \leq \nu_{\text{opt}}$ and $\epsilon_k \leq \epsilon_{\text{opt}}$ **then**
12: **Output:** x^k and **terminate**
13: **end if**
14: **if** $\left\|g^k\right\| \leq \nu_k$ **then**
15: $\nu_{k+1} \leftarrow \theta_\nu \nu_k$, $\epsilon_{k+1} \leftarrow \theta_\epsilon \epsilon_k$, and $t_k \leftarrow 0$
16: **else**
17: Compute t_k by the line search: $t_k = \max\{t \in \{\gamma, \gamma^2, \ldots\} : \Phi(x^k - t g^k) < \Phi(x^k) - \beta t \|g^k\|^2\}$
18: $\nu_{k+1} \leftarrow \nu_k$ and $\epsilon_{k+1} \leftarrow \epsilon_k$
19: **end if**
20: $x^{k+1} \leftarrow x^k - t_k g^k$
21: **end for**

4.4 Implementation Details

Check differentiability of y^* on $\mathcal{B}\left(x^0, \epsilon_0\right)$ We propose Proposition 3 to check differentiability of y^* on $\mathcal{B}\left(x^0, \epsilon_0\right)$, which is required by line 3 of Algorithm 1.

Proposition 3. *Consider $x^0 \in \mathbb{R}^{d_x}$ and $\epsilon > 0$. Suppose Assumptions 1, 2, 3 hold. Then, Lipschitz constants of functions $\Phi(x)$, $\lambda_j(x)$ and $p_j(x, y^*(x))$ on $\mathcal{B}(x^0, \epsilon)$ exist and are denoted by $l_\Phi(x^0, \epsilon)$, $l_{\lambda_j}(x^0, \epsilon)$ and $l_{p_j}(x^0, \epsilon)$, respectively. Further, suppose the SCSC holds at $y^*(x^0)$ w.r.t. $\lambda(x^0)$. If there exists $\epsilon_1 > 0$ such that*

$$\begin{aligned} &\lambda_j(x^0) > l_{\lambda_j}(x^0, \epsilon_1)\epsilon_1 \text{ for all } j \in J(x^0), \\ &p_j(x^0, y^*(x^0)) < -l_{p_j}(x^0, \epsilon_1)\epsilon_1 \text{ for all } j \notin J(x^0), \end{aligned} \tag{6}$$

then y^* is continuously differentiable on $\mathcal{B}(x^0, \epsilon_1)$.

Proposition 3 shows that, y^* is continuously differentiable on a neighborhood of x^0, if for any j, either (i) λ_j is larger than zero with non-trivial amount when the constraint $p_j(x^0, y^*(x^0))$ is active; or (ii) the satisfaction of $p_j(x^0, y^*(x^0))$ is non-trivial when it is inactive. For case (i), $\lambda_j(x) > 0$ and the constraint is strictly active for all $x \in \mathcal{B}(x^0, \epsilon)$; for case (ii), $p_j(x, y^*(x)) < 0$ and the constraint is inactive for all $x \in \mathcal{B}(x^0, \epsilon)$. As a illustration on the problem in Example 1 shown in Fig. 2, y^* is continuously differentiable on $\mathcal{B}(x^0, \epsilon)$, $\mathcal{B}(x^1, \epsilon)$ and $\mathcal{B}(x^2, \epsilon)$, and the constraint is inactive or strictly active in each ball.

We evaluate the differentiability of $y^*(x)$ and $\Phi(x)$ on $\mathcal{B}(x^0, \epsilon)$ by Proposition 3. In particular, we approximatively regard that y^* and Φ is continuously differentiable on $\mathcal{B}(x^0, \epsilon)$ if (6) is satisfied; otherwise, there exists $x \in \mathcal{B}(x^0, \epsilon)$ such that y^* and Φ is not continuously differentiable at x. The Lipschitz constants $l_{\lambda_j}(x^0, \epsilon)$ and $l_{p_j}(x^0, \epsilon)$ are computed as

$$\begin{aligned} l_{\lambda_j}(x^0, \epsilon) &= \|\nabla \lambda_j(x^0)\| + \delta, \\ l_{p_j}(x^0, \epsilon) &= \|\nabla_x p_j(x^0, y^*(x^0)) + \\ &\qquad \nabla y^*(x^0)^\top \nabla_y p_j(x^0, y^*(x^0))\| + \delta, \end{aligned} \tag{7}$$

where δ is a small parameter, and $\nabla_x p_j(x^0, y^*(x^0))$ and $\nabla \lambda_j(x^0)$ are given in (4). Here, for a function f, we approximate its Lipschitz constant on $\mathcal{B}(x^0, \epsilon)$, which is defined as $l_f(x^0, \epsilon) \triangleq \sup_x\{\|\nabla f(x)\| : x \in \mathcal{B}(x^0, \epsilon)\}$, as $l_f(x^0, \epsilon) \approx \|\nabla f(x^0)\| + \delta$. As ϵ decreases, f in $\mathcal{B}(x^0, \epsilon)$ is approximating to an affine function, and then the approximation error of $l_f(x^0, \epsilon)$ decreases.

Computation of $G(x^0, \epsilon)$ To compute $G(x^0, \epsilon)$ in line 7 of Algorithm 1, we need to compute the sets $I_+^\epsilon(x^0)$ and $I^\epsilon(x^0)$ defined in Proposition 2. Similar to the idea in Proposition 3, we evaluate $I_+^\epsilon(x^0)$ and $I^\epsilon(x^0)$ as

$$\begin{aligned} I_+^\epsilon(x^0) &= \left\{j \in J(x^0) : \lambda_j(x^0) > l_{\lambda_j}(x^0, \epsilon)\epsilon\right\}, \\ I_-^\epsilon(x^0) &= \left\{j \notin J(x^0) : p_j(x^0, y^*(x^0)) < -l_{p_j}(x^0, \epsilon)\epsilon\right\}, \\ I^\epsilon(x^0) &= \left\{j : j \notin I_+^\epsilon(x^0) \cup I_-^\epsilon(x^0)\right\}. \end{aligned} \tag{8}$$

Recall that the KKT conditions hold at $y^*(x^0)$ for problem $P(x^0)$, then for any $x \in \mathcal{B}(x^0, \epsilon)$, $p_j(x, y^*(x)) = 0$ for $j \in I_+^\epsilon(x^0)$ and $\lambda_j(x) = 0$ for $j \in I_-^\epsilon(x^0)$. Here, we also use l_{λ_j} and l_{p_j} given in (7) as the Lipschitz constants. When y^* and λ are not differentiable at x^0, we sample a point x' near x^0 such that y^* and λ are differentiable at x', then replace $\nabla \lambda(x^0)$ and $\nabla y^*(x^0)$ in (7) by $\nabla \lambda(x')$ and $\nabla y^*(x')$.

5 Experiments

5.1 Hyperparameter Optimization

Hyperparameter optimization (HO) has been widely studied (Pedregosa 2016; Franceschi et al. 2017; Ji, Yang, and Liang 2021). However, existing methods cannot handle HO of constrained learning problems, such as the supported vector machine (SVM) classification (Cortes and Vapnik 1995), constrained reinforcement learning (Achiam et al. 2017; Chen, Dong, and Wang 2021; Xu, Liang, and Lan 2021). We apply the proposed algorithm to HO of constrained learning.

HO of SVM We optimize the hyperparameter in SVM optimization, i.e., the penalty terms of the separation violations. We conduct the experiment on linear SVM and kernelized SVM on the dataset of diabetes in (Dua and Graff 2017). It is the first time to solve HO for SVM. We provide details of the problem formulation and the implementation setting in Appendix B.1. As shown in Fig. 5, the loss is nearly convergent for both linear and kernelized SVM, and the final test accuracy is much better than that of randomly selected hyperparameters, which is the initial point of the optimization.

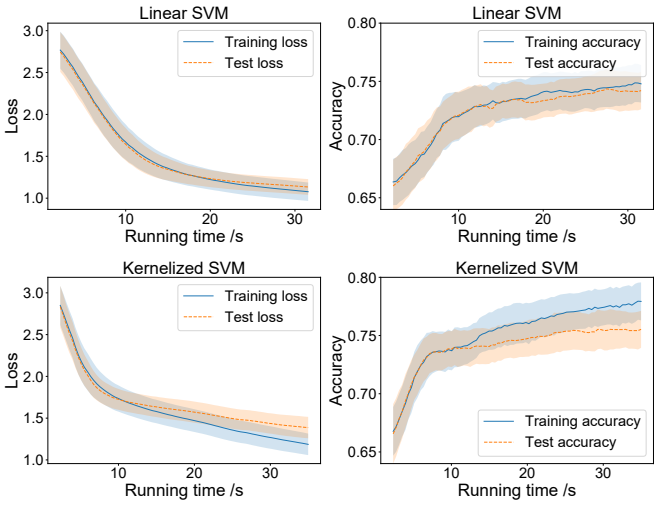

Figure 5: Loss and accuracy v.s. running time in HO of linear and kernelized SVM

Data Hyper-Cleaning Data hyper-cleaning (Franceschi et al. 2017; Shaban et al. 2019) is to train a classifier in a setting where the labels of training data are corrupted with a probability p (i.e., the corruption rate). We formulate the problem as a HO of SVM and conduct experiments on a breast cancer dataset (Dua and Graff 2017). The problem formulation and the implementation setting are provided in Appendix B.1. We compare our gradient approximation method with directly gradient descent used in (Amos and Kolter 2017; Lee et al. 2019). It is shown in Fig. 6 that, our method converges faster than the benchmark in terms of the loss and accuracy in both the training and test stages. Moreover, both the two methods achieve the test accuracy 96.2% using the corrupt data ($p = 0.4$). The accuracy is comparable to the test accuracy 96.5% of an SVM model where the data is clean.

5.2 Meta-Learning

Meta-learning approaches for few-shot learning have been formulated as bilevel optimization problems in (Rajeswaran et al. 2019; Lee et al. 2019; Ji, Yang, and Liang 2021). In particular, the problem in MetaOptNet (Lee et al. 2019) has the form of problem (2) with the lower-level constraints. However, its optimization does not explicitly consider the non-differentiability of the objective function and cannot guarantee convergence. In the experiment, we compare our algorithm with the optimization in MetaOptNet on datasets CIFAR-FS (Bertinetto et al. 2018) and FC100 (Oreshkin, Rodríguez López, and Lacoste 2018), which are widely used for few-shot learning. Appendix B.2 provides details of the problem formulation and the experiment setting.

Fig. 7 compares our gradient approximation method and the direct gradient descent in MetaOptNet (Lee et al. 2019). The two algorithms share all training configurations, including the network structure, the learning rate in each epoch and the batch size. For both CIFAR-FS and FC100 datasets, our method converges faster than the optimization in MetaOptNet

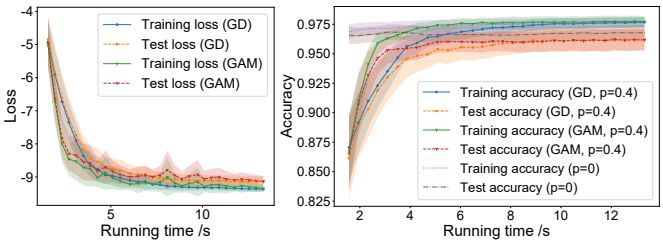

Figure 6: Comparison of gradient descent (GD) and gradient approximation method (GAM) in data hyper-cleaning with the corruption rate $p = 0.4$.

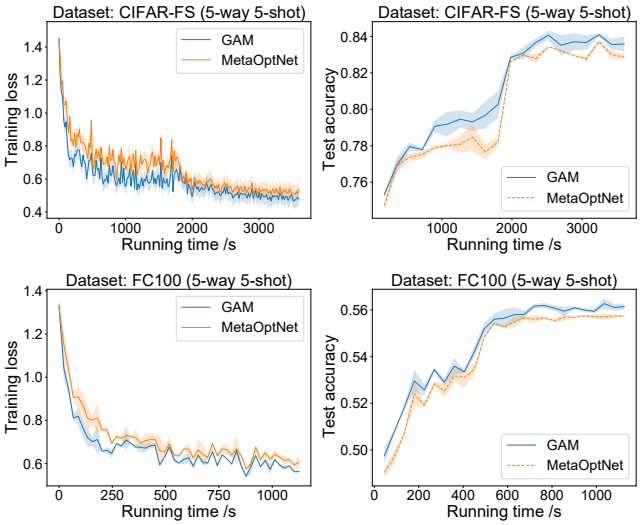

Figure 7: Comparison of MetaOptNet and gradient approximation method (GAM).

in terms of the training loss and test accuracy, and achieves a higher final test accuracy. Note that the only difference between the two algorithms in this experiment is the computation of the descent direction. The result shows the Clarke subdifferential approximation in our algorithm works better than the gradient as the descent direction. This is consistent with Proposition 2, where a set of representative gradients instead one gradient is more suitable to approximate the Clarke subdifferential. More comparison results with other meta-learning approaches are given in Appendix B.2.

6 Conclusion

We develop a gradient approximation method for the bilevel optimization where the lower-level optimization problem is convex with equality and inequality constraints and the upper-level optimization is non-convex. The proposed method efficiently approximates the Clarke Subdifferential of the non-smooth objective function, and theoretically guarantees convergence. Our experiments validate our theoretical analysis and demonstrate the superior effectiveness of the algorithm.

Acknowledgements

This work was partially supported by NSF awards ECCS 1846706 and ECCS 2140175.

References

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained policy optimization. In *International Conference on Machine Learning*, 22–31. PMLR.

Agrawal, A.; Amos, B.; Barratt, S.; Boyd, S.; Diamond, S.; and Kolter, J. Z. 2019. Differentiable convex optimization layers. *Advances in Neural Information Processing Systems*, 9562–9674.

Amos, B.; and Kolter, J. Z. 2017. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, 136–145. PMLR.

Amos, B.; Xu, L.; and Kolter, J. Z. 2017. Input convex neural networks. In *International Conference on Machine Learning*, 146–155. PMLR.

Bagirov, A.; Karmitsa, N.; and Mäkelä, M. M. 2014. *Introduction to nonsmooth optimization: theory, practice and software*, volume 12. Springer.

Bagirov, A. M.; Gaudioso, M.; Karmitsa, N.; Mäkelä, M. M.; and Taheri, S. 2020. Gradient sampling methods for nonsmooth optimization. In *Numerical Nonsmooth Optimization: State of the Art Algorithms*. Springer.

Bard, J.; and Moore, J. 1990. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal on Scientific and Statistical Computing*, 11.

Bard, J. F.; and Falk, J. E. 1982. An explicit solution to the multi-level programming problem. *Computers and Operations Research*, 9(1): 77–100.

Bengio, Y. 2000. Gradient-based optimization of hyperparameters. *Neural computation*, 12(8): 1889–1900.

Bertinetto, L.; Henriques, J. F.; Torr, P. H.; and Vedaldi, A. 2018. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*.

Bottou, L.; and Bousquet, O. 2008. The tradeoffs of large scale learning. *Advances in Neural Information Processing Systems*, 161–168.

Burke, J. V.; Lewis, A. S.; and Overton, M. L. 2005. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15(3): 751–779.

Chen, Y.; Dong, J.; and Wang, Z. 2021. A primal-dual approach to constrained markov decision processes. *arXiv preprint arXiv:2101.10895*.

Clarke, F. H. 1975. Generalized gradients and applications. *Transactions of the American Mathematical Society*, 205: 247–262.

Cortes, C.; and Vapnik, V. 1995. Support-vector networks. *Machine learning*, 20(3): 273–297.

Dempe, S. 1998. An implicit function approach to bilevel programming problems. In *Multilevel Optimization: Algorithms and Applications*, 273–294. Springer.

Dempe, S.; and Franke, S. 2016. On the solution of convex bilevel optimization problems. *Computational Optimization and Applications*, 63(3): 685–703.

Domke, J. 2012. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, 318–326. PMLR.

Dua, D.; and Graff, C. 2017. UCI machine learning repository. http://archive.ics.uci.edu/ml. Accessed: 2022-08-15.

Fiacco, A. V. 1983. *Introduction to sensitivity and stability analysis in nonlinear programming*. Academic press.

Fiacco, A. V.; and Ishizuka, Y. 1990. Sensitivity and stability analysis for nonlinear programming. *Annals of Operations Research*, 27(1): 215–235.

Fiacco, A. V.; and McCormick, G. P. 1970. Nonlinear programming: sequential unconstrained minimization techniques. *SIAM Review*, 12(4): 593–594.

Franceschi, L.; Donini, M.; Frasconi, P.; and Pontil, M. 2017. Forward and reverse gradient-based hyperparameter optimization. In *International Conference on Machine Learning*, 1165–1173. PMLR.

Franceschi, L.; Frasconi, P.; Salzo, S.; Grazzi, R.; and Pontil, M. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, 1568–1577. PMLR.

Ghadimi, S.; and Wang, M. 2018. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*.

Gould, S.; Fernando, B.; Cherian, A.; Anderson, P.; Cruz, R. S.; and Guo, E. 2016. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *arXiv preprint arXiv:1607.05447*.

Grazzi, R.; Franceschi, L.; Pontil, M.; and Salzo, S. 2020. On the iteration complexity of hypergradient computation. In *International Conference on Machine Learning*, 3748–3758. PMLR.

Hansen, P.; Jaumard, B.; and Savard, G. 1992. New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific Computing*, 13(5): 1194–1217.

Hardt, M.; Recht, B.; and Singer, Y. 2016. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, 1225–1234. PMLR.

Hong, M.; Wai, H.-T.; Wang, Z.; and Yang, Z. 2020. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. *arXiv preprint arXiv:2007.05170*.

Ji, K.; Lee, J. D.; Liang, Y.; and Poor, H. V. 2020. Convergence of meta-learning with task-specific adaptation over partial parameters. *Advances in Neural Information Processing Systems*, 11490–11500.

Ji, K.; Yang, J.; and Liang, Y. 2021. Bilevel optimization: Convergence analysis and enhanced design. In *International Conference on Machine Learning*, 4882–4892. PMLR.

Jin, C.; Netrapalli, P.; Ge, R.; Kakade, S. M.; and Jordan, M. I. 2021. On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. *Journal of the ACM*, 68(2): 1–29.

Kiwiel, K. C. 2007. Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. *SIAM Journal on Optimization*, 18(2): 379–388.

Kolstad, C. D.; and Lasdon, L. S. 1990. Derivative evaluation and computational experience with large bilevel mathematical programs. *Journal of optimization theory and applications*, 65(3): 485–499.

Konda, V. R.; and Tsitsiklis, J. N. 2000. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, 1008–1014.

Lee, K.; Maji, S.; Ravichandran, A.; and Soatto, S. 2019. Meta-learning with differentiable convex optimization. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10649–10657.

Liu, H.; Simonyan, K.; and Yang, Y. 2018. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*.

Liu, R.; Gao, J.; Zhang, J.; Meng, D.; and Lin, Z. 2021a. Investigating bi-Level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis & Machine Intelligence*.

Liu, R.; Liu, Y.; Zeng, S.; and Zhang, J. 2021b. Towards gradient-based bilevel optimization with non-convex followers and beyond. *Advances in Neural Information Processing Systems*, 34: 8662–8675.

Nemirovski, A.; Juditsky, A.; Lan, G.; and Shapiro, A. 2009. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4): 1574–1609.

Nesterov, Y. 2003. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Oreshkin, B.; Rodríguez López, P.; and Lacoste, A. 2018. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in Neural Information Processing Systems*, 719–729.

Pedregosa, F. 2016. Hyperparameter optimization with approximate gradient. In *International Conference on Machine Learning*, 737–746. PMLR.

Rajeswaran, A.; Finn, C.; Kakade, S. M.; and Levine, S. 2019. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, 113–124.

Savard, G.; and Gauvin, J. 1994. The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, 15(5): 265–272.

Schramm, H.; and Zowe, J. 1992. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM journal on optimization*, 2(1): 121–152.

Shaban, A.; Cheng, C.-A.; Hatch, N.; and Boots, B. 2019. Truncated back-propagation for bilevel optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 1723–1732. PMLR.

Shi, C.; Lu, J.; and Zhang, G. 2005. An extended Kuhn-Tucker approach for linear bilevel programming. *Applied Mathematics and Computation*, 162(1): 51–63.

Sow, D.; Ji, K.; Guan, Z.; and Liang, Y. 2022. A Constrained Optimization Approach to Bilevel Optimization with Multiple Inner Minima. *arXiv preprint arXiv:2203.01123*.

Xu, T.; Liang, Y.; and Lan, G. 2021. Crpo: A new approach for safe reinforcement learning with convergence guarantee. In *International Conference on Machine Learning*, 11480–11491. PMLR.