DTZO: Distributed Trilevel Zeroth Order Learning with Provable Non-Asymptotic Convergence

Yang Jiao [1] **Kai Yang** [1] **Chengtao Jian** [1]

Abstract

Trilevel learning (TLL) with zeroth order constraints is a fundamental problem in machine learning, arising in scenarios where gradient information is inaccessible due to data privacy or model opacity, such as in federated learning, healthcare, and financial systems. These problems are notoriously difficult to solve due to their inherent complexity and the lack of first order information. Moreover, in many practical scenarios, data may be distributed across various nodes, necessitating strategies to address trilevel learning problems without centralizing data on servers to uphold data privacy. To this end, an effective distributed trilevel zeroth order learning framework DTZO is proposed in this work to address the trilevel learning problems with level-wise zeroth order constraints in a distributed manner. The proposed DTZO is versatile and can be adapted to a wide range of (grey-box) trilevel learning problems with partial zeroth order constraints. In DTZO, the cascaded polynomial approximation can be constructed without relying on gradients or sub-gradients, leveraging a novel cut, i.e., zeroth order cut. Furthermore, we theoretically carry out the non-asymptotic convergence rate analysis for the proposed DTZO in achieving the ϵ-stationary point. Extensive experiments have been conducted to demonstrate and validate the superior performance of the proposed DTZO.

1. Introduction

Trilevel learning (TLL), also known as trilevel optimization, pertains to nested optimization problems involving three levels of optimization, thus exhibiting a trilevel hierarchical structure. Trilevel learning has been widely used in many machine learning applications, such as robust hyperparameter optimization (Sato et al., 2021; Giovannelli et al., 2025), domain adaptation (Choe et al., 2023), machine translation (He et al., 2024), robust neural architecture search (Guo et al., 2020; Jiao et al., 2024), and so on. The general form of a trilevel learning problem can be expressed as,

$$\begin{aligned} \min \quad & f_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3) \\ \text{s.t. } & \boldsymbol{x}_2 = \arg\min_{\boldsymbol{x}_2'} f_2(\boldsymbol{x}_1, \boldsymbol{x}_2', \boldsymbol{x}_3) \\ & \quad \text{s.t. } \boldsymbol{x}_3 = \arg\min_{\boldsymbol{x}_3'} f_3(\boldsymbol{x}_1, \boldsymbol{x}_2', \boldsymbol{x}_3') \\ \text{var.} \quad & \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \end{aligned} \tag{1}$$

where f_1, f_2, f_3 denote the first, second, and third level objectives, and $\boldsymbol{x}_1 \in \mathbb{R}^{d_1}, \boldsymbol{x}_2 \in \mathbb{R}^{d_2}, \boldsymbol{x}_3 \in \mathbb{R}^{d_3}$ are variables. Existing trilevel learning approaches focus on scenarios where TLL problems can be addressed with first order information available at each level. However, situations where first order information is unavailable (i.e., $\nabla f_1, \nabla f_2, \nabla f_3$ are non-available), such as when black-box models are employed, remain *under-explored*. Additionally, in trilevel learning applications, data may be distributed across various nodes, necessitating strategies to address trilevel learning problems without centralizing data on servers in order to uphold data privacy (Jiao et al., 2024).

Complexity of Addressing TLL with Zeroth Order Constraints: The complexity involved in solving problems characterized by hierarchical structures with three levels is *significantly greater* than that of bilevel learning problems (Blair, 1992; Avraamidou, 2018). It is worth mentioning that even *finding a feasible solution* in TLL problem is **NP-hard** since it necessitates addressing the inner bilevel learning problem, which is NP-hard (Ben-Ayed & Blair, 1990; Sinha et al., 2017). Existing approaches are not applicable for addressing TLL with zeroth order constraints, as they either rely on the first order information to solve the TLL problems (Jiao et al., 2024; Sato et al., 2021) or focus on single-level and bilevel zeroth order learning problems (Fang et al., 2022; Qiu et al., 2023).

To this end, an effective **D**istributed **T**rilevel **Z**eroth **O**rder learning (DTZO) framework is proposed in this work. Specifically, we first introduce the cascaded zeroth order polynomial approximation for the trilevel learning problems,

[1]Department of Computer Science and Technology, Tongji University. Correspondence to: Kai Yang <kaiyang@tongji.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

Method	Bilevel (FO)	Bilevel (ZO)	Trilevel (FO)	Trilevel (ZO)
FEDNEST (Tarzanagh et al., 2022)	$\mathcal{O}(1/\epsilon^2)$	-	-	-
ADBO (Jiao et al., 2023)	$\mathcal{O}(1/\epsilon^2)$	-	-	-
MDBO (Gao et al., 2023)	$\mathcal{O}(1/\epsilon^2)$	-	-	-
FedBiOAcc (Li et al., 2023)	$\mathcal{O}(1/\epsilon^{1.5})$	-	-	-
MemFBO (Yang et al., 2025)	$\mathcal{O}(1/\epsilon^{1.5})$	-	-	-
FedRZO$_{bl}$ (Qiu et al., 2023)	-	$\mathcal{O}(1/\epsilon^2)$	-	-
AFTO (Jiao et al., 2024)	-	-	$\mathcal{O}(1/\epsilon^2)$	-
The proposed DTZO	-	-	-	$\mathcal{O}(1/\epsilon^2)$

Table 1. Comparison of non-asymptotic convergence rates for distributed nested optimization methods under first order (FO) and zeroth order (ZO) scenarios. This work is the first to provide the theoretical guarantees for trilevel zeroth order optimization.

which consists of the inner layer and outer layer polynomial approximation. Next, how to generate the novel zeroth order cuts without using gradients or sub-gradients to gradually refine the cascaded polynomial approximation is discussed. Zeroth order cut is a type of cutting plane that does not rely on first order information during generation. Finally, the distributed zeroth order algorithm is developed to address trilevel zeroth order learning problems (i.e., TLL with level-wise zeroth order constraints) in a distributed manner. Additionally, a novel concept of *soft constraint* is introduced in this work to explain why the lower-level problem in bilevel and trilevel optimization can be approximated or relaxed to some extent. Theoretically, we demonstrate that the proposed zeroth order cuts can construct a polynomial relaxation for TLL problems, and this relaxation will be gradually tightened with zeroth order cuts added. Additionally, we also analyze the non-asymptotic convergence rate, i.e., iteration and communication complexities, for the proposed DTZO to achieve the ϵ-stationary point. It is worth highlighting that this is the **first work** to address the trilevel zeroth order optimization problems while establishing **theoretical guarantees** for the proposed algorithm. Table 1 presents a comparison of the non-asymptotic convergence results between the proposed DTZO and state-of-the-art methods. Our contributions can be summarized as follows.

1. Different from the existing works on single-level and bilevel zeroth order learning, this work takes an initial step towards addressing trilevel zeroth order learning. To the best of our knowledge, this is the first work to address the trilevel zeroth order learning problems.

2. An effective framework DTZO with novel zeroth order cuts is proposed for tackling trilevel zeroth order learning problems in a distributed manner. Different from the existing methods, the proposed DTZO is capable of constructing the cascaded zeroth order polynomial approximation without using gradients or sub-gradients.

3. Extensive experiments on black-box large language models (LLMs) trilevel learning and robust hyperparameter optimization substantiate the superior performance of the proposed DTZO.

2. Related Work

2.1. Distributed Zeroth Order Optimization

Zeroth order optimization is widely-used for addressing machine learning problems where obtaining explicit gradient expressions is challenging or impractical (Liu et al., 2018c; Chen et al., 2019; Wang et al., 2018b; Héliou et al., 2021; Cai et al., 2021; Gao & Huang, 2020; Yue et al., 2023; Li et al., 2022; Ren et al., 2023; Nikolakakis et al., 2022; Tu et al., 2019; Rando et al., 2024). In practical applications of zeroth order optimization, data may be distributed across different nodes. To address zeroth order optimization problems in a distributed manner, the distributed zeroth order optimization methods have recently garnered significant attention, e.g., Lian et al. (2016); Tang et al. (2020); Fang et al. (2022); Chen et al. (2024a); Akhavan et al. (2021); Sahu et al. (2018); Shu et al. (2023). Furthermore, to tackle the bilevel zeroth order optimization problems in a distributed manner, the federated bilevel zeroth order optimization method FedRZO$_{\text{bl}}$ (Qiu et al., 2023) has been proposed. However, how to address the higher-nested zeroth order optimization problems, e.g., trilevel, in a distributed manner remains under-explored. To the best of our knowledge, this is the **first work** that considers how to address the trilevel zeroth order optimization problems.

2.2. Trilevel Learning

Trilevel learning has found applications in various fields within machine learning. A robust neural architecture search (NAS) approach that integrates adversarial learning with NAS is introduced in Guo et al. (2020). The robust NAS can be viewed as a trilevel learning problem, as discussed in Jiao et al. (2024). A trilevel learning problem comprising two levels pretraining, fine-tuning and hyperparameter optimization, is explored in Raghu et al. (2021). In Garg et al. (2022), the trilevel learning problem, which involves data reweighting, architecture search, and model training, is

investigated. In Sato et al. (2021), the robust hyperparameter optimization is framed as a trilevel learning problem, and a hypergradient-based method is proposed to address such problems. In Choe et al. (2023), a general automatic differentiation technique is proposed, which can be applied to trilevel learning problems. Additionally, a cutting plane based distributed algorithm is proposed in Jiao et al. (2024) for trilevel learning problems. Nevertheless, existing methods predominantly rely on first order information to solve trilevel learning problems. This is the **first framework** that can be used to solve trilevel learning problems *without* relying on first order information.

2.3. Cutting Plane Method

Cutting plane methods are widely used in convex optimization (Bertsekas, 2015; Franc et al., 2011), robust optimization (Yang et al., 2014; Bürger et al., 2013), and so on. Recently, there has been notable interest in leveraging cutting plane methods to tackle distributed nested optimization problems. It is shown in Jiao et al. (2023) that the nested optimization problem can be transformed into a *decomposable optimization problem* by utilizing cutting plane method, which *significantly facilitates* the design of distributed algorithms for nested optimization. In Jiao et al. (2023), the cutting plane method is employed to tackle bilevel optimization problems in a distributed manner. Similarly, Chen et al. (2024d) utilizes the cutting plane method to address distributed bilevel optimization problems within downlink multi-cell systems. Furthermore, Jiao et al. (2024) applies the cutting plane method to solve distributed trilevel optimization problems. However, the existing cutting plane methods for nested optimization rely on the gradients or subgradients to generate cutting planes, which is not available in zeroth order optimization. In this work, the proposed DTZO is capable of generating zeroth order cuts for nested optimization problems without using gradients or sub-gradients. Discussions about the novelty of the proposed zeroth order cuts are shown in Appendix J.1 and Table 7.

3. Distributed Trilevel Zeroth Order Learning

In the practical applications of trilevel zeroth order learning, data may be distributed across multiple nodes (Jiao et al., 2024). Aggregating data on central servers may pose significant privacy risks (Subramanya & Riggio, 2021). Therefore, it is crucial to develop an effective framework to address trilevel zeroth order learning problems in a distributed manner. The distributed trilevel zeroth order learning problem can be expressed as,

$$\begin{aligned}
&\min \textstyle\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3)\\
&\text{s.t. } \boldsymbol{x}_2=\underset{\boldsymbol{x}_2{}'}{\arg\min} \textstyle\sum_{j=1}^{N} f_{2,j}(\boldsymbol{x}_1,\boldsymbol{x}_2{}',\boldsymbol{x}_3)\\
&\quad\text{s.t. } \boldsymbol{x}_3=\underset{\boldsymbol{x}_3{}'}{\arg\min} \textstyle\sum_{j=1}^{N} f_{3,j}(\boldsymbol{x}_1,\boldsymbol{x}_2{}',\boldsymbol{x}_3{}')\\
&\text{var.}\qquad\quad \boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,
\end{aligned}\tag{2}$$

where $f_{1,j}, f_{2,j}, f_{3,j}$ respectively denote the first, second, and third level objectives in j^{th} worker, $\boldsymbol{x}_1\in\mathbb{R}^{d_1},\boldsymbol{x}_2\in\mathbb{R}^{d_2},\boldsymbol{x}_3\in\mathbb{R}^{d_3}$ are variables. The first order information of functions $f_{1,j}, f_{2,j}, f_{3,j}$, i.e., $\nabla f_{1,j}, \nabla f_{2,j}, \nabla f_{3,j}$, is not available in Eq. (2), corresponding to the level-wise zeroth order constraints. To facilitate the development of distributed algorithms in parameter-server architecture (Jiao et al., 2023; Assran et al., 2020), the distributed TLL with zeroth order constraints in Eq. (2) is equivalently reformulated as a consensus trilevel zeroth order learning problem as follows.

$$\begin{aligned}
&\min \textstyle\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})\\
&\text{s.t. } \boldsymbol{x}_{1,j}=\boldsymbol{z}_1,\forall j=1,\cdots,N\\
&\{\boldsymbol{x}_{2,j}\},\boldsymbol{z}_2=\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min} \textstyle\sum_{j=1}^{N} f_{2,j}(\boldsymbol{z}_1,\boldsymbol{x}_{2,j}{}',\boldsymbol{x}_{3,j})\\
&\qquad\text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\
&\qquad\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min} \textstyle\sum_{j=1}^{N} f_{3,j}(\boldsymbol{z}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\
&\qquad\qquad\text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\\
&\text{var.}\qquad\quad \{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3,
\end{aligned}\tag{3}$$

where $\boldsymbol{x}_{1,j}\in\mathbb{R}^{d_1},\boldsymbol{x}_{2,j}\in\mathbb{R}^{d_2},\boldsymbol{x}_{3,j}\in\mathbb{R}^{d_3}$ denote the local variables in j^{th} worker, $\boldsymbol{z}_1\in\mathbb{R}^{d_1},\boldsymbol{z}_2\in\mathbb{R}^{d_2},\boldsymbol{z}_3\in\mathbb{R}^{d_3}$ denote the consensus variables in the master, N denotes the number of workers.

Overview of the proposed framework. In Sec. 3.1, the construction of cascaded zeroth order polynomial approximation for the trilevel zeroth order learning problem is proposed, which consists of the inner layer and outer layer polynomial approximation. Then, how to gradually update zeroth order cuts to refine the cascaded polynomial approximation is discussed in Sec. 3.2. Finally, a distributed zeroth order algorithm is developed to effectively address the trilevel zeroth order learning problem in a distributed manner in Sec. 3.3.

In addition, this work takes an initial step toward introducing the novel concept of **soft constraints** in bilevel and trilevel optimization, as discussed in Section 3.1.1. To improve the readability of this work, The notations used in this work and their corresponding definitions are summarized in Table 3.

3.1. Cascaded Zeroth Order Polynomial Approximation

In this section, how to construct the cascaded zeroth order polynomial approximation for trilevel zeroth order learning is introduced. The proposed cascaded zeroth order polynomial approximation consists of two key parts: 1) the inner layer polynomial approximation and 2) the outer layer polynomial approximation, which will be discussed below.

3.1.1. Inner Layer Polynomial Approximation

In trilevel learning, the third level optimization problem can be viewed as the constraint to the second level optimization problem (Jiao et al., 2024;

Pan et al., 2024; Kwon et al., 2023; Jiang et al., 2023), it equals the constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) = 0$, where $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) = ||\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_3 \end{bmatrix} - \underset{\{\boldsymbol{x}_{3,j}'\}, \boldsymbol{z}_3'}{\arg\min} \sum_j f_{3,j}(\boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{x}_{3,j}') \text{ s.t. } \boldsymbol{x}_{3,j}' = \boldsymbol{z}_3', \forall j||^2$. In many bilevel and trilevel machine learning applications, e.g., neural architecture search in Liu et al. (2018a), robust hyperparameter optimization in Jiao et al. (2024), the lower-level optimization problem serves as a **soft constraint** (Kautz et al., 1996) to the upper-level optimization problem, i.e., this constraint (constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) = 0$ in our problem) can be violated to a certain extent while still yielding a feasible and meaningful solution. Inspired by Jiao et al. (2023); Chen et al. (2024d), the cutting plane based method is utilized to construct a *decomposable* polynomial relaxation for this constraint, which *significantly facilitates* the development of distributed algorithms. Specifically, the inner layer zeroth order cuts are utilized to approximate the feasible region with respect to constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) = 0$. Zeroth order cuts refer to the cutting planes that do not rely on first order information during generation. In this section, we focus on the construction of cascaded polynomial approximation, and how to generate the zeroth order cuts is discussed in detail in the next section 3.2. Consequently, the feasible region formed by inner layer zeroth order cuts in t^{th} iteration is,

$$\begin{aligned} P_{\text{in}}^t = \{&\textstyle\sum_j \boldsymbol{a}_{j,l}^{\text{in}\top} \boldsymbol{x}_{3,j}^2 + \boldsymbol{b}_{j,l}^{\text{in}\top} \boldsymbol{x}_{3,j} + \sum_{i\in\{1,3\}} \boldsymbol{c}_{i,l}^{\text{in}\top} \boldsymbol{z}_i^2 \\ &+ \boldsymbol{d}_{i,l}^{\text{in}\top} \boldsymbol{z}_i + \boldsymbol{c}_{2,l}^{\text{in}\top} \boldsymbol{z}_2^{2\prime} + \boldsymbol{d}_{2,l}^{\text{in}\top} \boldsymbol{z}_2' + e_l^{\text{in}} \le \varepsilon_{\text{in}}, \forall l\}, \end{aligned} \tag{4}$$

where $\boldsymbol{x}_{i,j}^2 = [x_{i,j,1}^2, \cdots, x_{i,j,d_i}^2] \in \mathbb{R}^{d_i}$, $\boldsymbol{z}_i^2 = [z_{i,1}^2, \cdots, z_{i,d_i}^2] \in \mathbb{R}^{d_i}$, $i = 1, 2, 3$, $\boldsymbol{a}_{j,l}^{\text{in}} \in \mathbb{R}^{d_3}$, $\boldsymbol{b}_{j,l}^{\text{in}} \in \mathbb{R}^{d_3}$, $\boldsymbol{c}_{i,l}^{\text{in}} \in \mathbb{R}^{d_i}$, $\boldsymbol{d}_{i,l}^{\text{in}} \in \mathbb{R}^{d_i}$, and $e_l^{\text{in}} \in \mathbb{R}^1$ are the parameters of l^{th} inner layer zeroth order cut, $\varepsilon_{\text{in}} \ge 0$ is a constant. By using the inner layer polynomial approximation according to Eq. (4), the resulting problem can be written as,

$$\begin{aligned} &\min \textstyle\sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) \\ &\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\ &\quad \{\boldsymbol{x}_{2,j}\}, \boldsymbol{z}_2 = \underset{\{\boldsymbol{x}_{2,j}'\}, \boldsymbol{z}_2'}{\arg\min} \textstyle\sum_{j=1}^N f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}', \boldsymbol{x}_{3,j}) \\ &\qquad\qquad \text{s.t. } \boldsymbol{x}_{2,j}' = \boldsymbol{z}_2', \forall j = 1, \cdots, N \\ &\qquad\qquad\quad (\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) \in P_{\text{in}}^t \\ &\text{var.} \qquad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3. \end{aligned} \tag{5}$$

Soft Constraint. In bilevel and trilevel optimization, the lower-level problem is typically treated as a constraint for the upper-level problem (Liu et al., 2022; Jiao et al., 2025; Kwon et al., 2024). In this work, we provide a novel insight into this constraint, i.e., it is indeed a *soft constraint* in many bilevel and trilevel optimization applications. A soft constraint refers to a constraint that can be partially violated without rendering the optimization problem meaningless (Kautz et al., 1996; Régin, 2011; Wilson et al., 2022). *This offers a new perspective on why the lower-level problem can be approximated or relaxed to some extent in bilevel and trilevel optimization* (Jiao et al., 2023). For example, in bilevel neural architecture search (Liu et al., 2018a), rather than computing the optimal solution for the lower-level optimization problem, the results obtained after a single gradient descent step can be used as an approximation at each iteration. Similarly, in bilevel meta-learning (Finn et al., 2017), the results obtained after multiple gradient descent steps can serve as an estimated optimal solution for the lower-level problem. In bilevel adversarial learning (Madry et al., 2018), which is a min-max optimization problem, the results after several projected gradient descent steps are used as the approximation of the optimal solution for the lower-level problem. Moreover, in trilevel learning, AFTO (Jiao et al., 2024) uses the results after K communication rounds to replace the optimal solution to the lower-level optimization problem in federated trilevel optimization problems. *To our best knowledge, this is the first work to introduce the concept of soft constraints into bilevel and trilevel optimization.*

3.1.2. Outer Layer Polynomial Approximation

Likewise, the lower-level optimization problem in Eq. (5) can be regarded as the constraint to the upper-level optimization problem. Defining $h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) = \sum_j \boldsymbol{a}_{j,l}^{\text{in}\top} \boldsymbol{x}_{3,j}^2 + \boldsymbol{b}_{j,l}^{\text{in}\top} \boldsymbol{x}_{3,j} + \sum_{i\in\{1,3\}} \boldsymbol{c}_{i,l}^{\text{in}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{in}\top} \boldsymbol{z}_i + \boldsymbol{c}_{2,l}^{\text{in}\top} \boldsymbol{z}_2^{2\prime} + \boldsymbol{d}_{2,l}^{\text{in}\top} \boldsymbol{z}_2' + e_l^{\text{in}}$. This constraint equals $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$, where

$$\begin{aligned} &\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \\ &= ||\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \boldsymbol{z}_2 \end{bmatrix} - \begin{array}{l} \underset{\{\boldsymbol{x}_{2,j}'\}, \boldsymbol{z}_2'}{\arg\min} \textstyle\sum_j f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}', \boldsymbol{x}_{3,j}) \\ \text{s.t. } \boldsymbol{x}_{2,j}' = \boldsymbol{z}_2', \forall j, \\ h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) \le \varepsilon_{\text{in}}, \forall l \end{array} ||^2. \end{aligned} \tag{6}$$

The constraint $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$ also serves as a *soft constraint* to the upper-level optimization problem. Outer layer zeroth order cuts are utilized to construct the polynomial approximation for the feasible region with respect to the constraint $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$, that is,

$$P_{\text{out}}^t = \left\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \le \varepsilon_{\text{out}}, \forall l\right\}, \tag{7}$$

where $h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = \sum_{i=2}^3 \sum_{j=1}^N \boldsymbol{a}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j}^2 + \boldsymbol{b}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j} + \sum_{i=1}^3 \boldsymbol{c}_{i,l}^{\text{out}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{out}\top} \boldsymbol{z}_i + e_l^{\text{out}}$, and $\varepsilon_{\text{out}} \ge 0$ is a pre-set constant. Based on Eq. (7), the resulting cascaded zeroth order polynomial approximation problem can be written as,

$$\begin{aligned} &\min \textstyle\sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) \\ &\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\ &\quad (\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \in P_{\text{out}}^t \\ &\text{var.} \quad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \end{aligned} \tag{8}$$

where $\boldsymbol{a}_{i,j,l}^{\text{out}} \in \mathbb{R}^{d_i}$, $\boldsymbol{b}_{i,j,l}^{\text{out}} \in \mathbb{R}^{d_i}$, $\boldsymbol{c}_{i,l}^{\text{out}} \in \mathbb{R}^{d_i}$, $\boldsymbol{d}_{i,l}^{\text{out}} \in \mathbb{R}^{d_i}$, and $e_l^{\text{out}} \in \mathbb{R}^1$ are parameters of l^{th} outer layer zeroth order cut.

3.2. Refining the Cascaded Polynomial Approximation

For every $\mathcal{T}$ iteration, the zeroth order cuts will be updated to refine the proposed cascaded polynomial approximation when $t < T_1$. Different from the existing cutting plane methods for nested optimization, the proposed zeroth order cuts can be generated without using gradients or sub-gradients, which is why we refer to them as zeroth order cuts. Specifically, in t^{th} iteration, the zeroth order cuts will be updated by three key steps: 1) generating inner layer zeroth order cut; 2) generating outer layer zeroth order cut; 3) removing inactive zeroth order cuts, which will be discussed as follows. In addition, we demonstrate the proposed zeroth order cuts can construct a relaxation for the original feasible regions in Proposition 3.1 and 3.2. To our best knowledge, this is the first work that considers how to generate cutting planes without using first order information in nested optimization, more discussions about the novelty of the proposed zeroth order cuts are presented in Appendix J.1 and Table 7.

3.2.1. Generating Inner Layer Zeroth Order Cut

At t^{th} iteration, based on point $(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)$, the new inner layer zeroth order cut will be generated to refine the inner layer polynomial approximation in Eq. (4), as follows.

$$
\begin{aligned}
&G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&+\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t) \\
&\le \tfrac{L+1}{2}(\textstyle\sum_j ||\boldsymbol{x}_{3,j} - \boldsymbol{x}_{3,j}^t||^2 + ||\boldsymbol{z}_1 - \boldsymbol{z}_1^t||^2 + ||\boldsymbol{z}_2{}' - \boldsymbol{z}_2^{t\prime}||^2 \\
&\quad + ||\boldsymbol{z}_3 - \boldsymbol{z}_3^t||^2) + \tfrac{\mu^2 L^2}{8} d_{\text{in}} + \varepsilon_{\text{in}},
\end{aligned} \tag{9}
$$

where $d_{\text{in}} = (d_1 + d_2 + (N+1)d_3 + 3)^3$ and

$$
\begin{aligned}
&G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t) \\
&= \tfrac{\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t + \mu \boldsymbol{\mu}_{x_{3,j}}\}, \boldsymbol{z}_1^t + \mu \boldsymbol{\mu}_{z_1}, \boldsymbol{z}_2^{t\prime} + \mu \boldsymbol{\mu}_{z_2}, \boldsymbol{z}_3^t + \mu \boldsymbol{\mu}_{z_3})}{\mu} \boldsymbol{\mu}^{\text{in}} \\
&\quad - \tfrac{\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)}{\mu} \boldsymbol{\mu}^{\text{in}},
\end{aligned} \tag{10}
$$

where $\boldsymbol{\mu}^{\text{in}} = [\{\boldsymbol{\mu}_{x_{3,j}}\}, \boldsymbol{\mu}_{z_1}, \boldsymbol{\mu}_{z_2}, \boldsymbol{\mu}_{z_3}]$ is a standard Gaussian random vector, $L > 0$ is a constant, and $\mu > 0$ is the smoothing parameter (Kornowski & Shamir, 2024; Ghadimi & Lan, 2013). Then, the new generated zeroth order cut $cp_{\text{in}}^{\text{new}}$ will be added into P_{in}^t, i.e., $P_{\text{in}}^t = \text{Add}(P_{\text{in}}^{t-1}, cp_{\text{in}}^{\text{new}})$.

Proposition 3.1. *The original feasible region of constraint* $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = 0$ *is a subset of the feasible region formed by inner layer zeroth order cuts, i.e.,* $P_{\text{in}}^t = \{h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) \le \varepsilon_{\text{in}}, \forall l\}$ *when* ϕ_{in} *has L-Lipschitz continuous gradient. The proof is provided in Appendix C.*

3.2.2. Generating Outer Layer Zeroth Order Cut

At t^{th} iteration, according to point $(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)$, the new outer layer zeroth order cut will be generated to refine the outer layer polynomial approximation in Eq. (7) as follows.

$$
\begin{aligned}
&G_\mu^{\text{out}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&+\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\
&\le \tfrac{L+1}{2}\left(\textstyle\sum_{i=2}^3 \sum_j ||\boldsymbol{x}_{i,j} - \boldsymbol{x}_{i,j}^t||^2 + \sum_i ||\boldsymbol{z}_i - \boldsymbol{z}_i^t||^2\right) \\
&+ \tfrac{\mu^2 L^2}{8}(d_1 + (N+1)(d_2+d_3) + 3)^3 + \varepsilon_{\text{out}}.
\end{aligned} \tag{11}
$$

In Eq. (11), we have that

$$
\begin{aligned}
&G_\mu^{\text{out}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) = \\
&\big(\tfrac{\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t + \mu \boldsymbol{\mu}_{x_{2,j}}\}, \{\boldsymbol{x}_{3,j}^t + \mu \boldsymbol{\mu}_{x_{3,j}}\}, \boldsymbol{z}_1^t + \mu \boldsymbol{\mu}_{z_1}, \boldsymbol{z}_2^t + \mu \boldsymbol{\mu}_{z_2}, \boldsymbol{z}_3^t + \mu \boldsymbol{\mu}_{z_3})}{\mu} \\
&- \tfrac{\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)}{\mu}\big) \boldsymbol{\mu}^{\text{out}},
\end{aligned} \tag{12}
$$

where $\boldsymbol{\mu}^{\text{out}} = [\{\boldsymbol{\mu}_{x_{2,j}}\}, \{\boldsymbol{\mu}_{x_{3,j}}\}, \boldsymbol{\mu}_{z_1}, \boldsymbol{\mu}_{z_2}, \boldsymbol{\mu}_{z_3}]$ is a standard Gaussian random vector. Subsequently, the new generated outer layer zeroth order cut $cp_{\text{out}}^{\text{new}}$ will be added into P_{out}^t, i.e., $P_{\text{out}}^t = \text{Add}(P_{\text{out}}^{t-1}, cp_{\text{out}}^{\text{new}})$.

Proposition 3.2. *The original feasible region of constraint* $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$ *is a subset of the feasible region formed by outer layer zeroth order cuts, i.e.,* $P_{\text{out}}^t = \{\sum_{i=2}^3 \sum_{j=1}^N \boldsymbol{a}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j}^2 + \boldsymbol{b}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j} + \sum_{i=1}^3 \boldsymbol{c}_{i,l}^{\text{out}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{out}\top} \boldsymbol{z}_i + e_l^{\text{out}} \le \varepsilon_{\text{out}}, \forall l\}$ *when* ϕ_{out} *has L-Lipschitz continuous gradient. Proofs are provided in Appendix C.*

3.2.3. Removing Inactive Zeroth Order Cuts

To improve the effectiveness and reduce the complexity (Yang et al., 2014; Jiao et al., 2023), the inactive zeroth order cuts will be removed during the iteration process. The corresponding inner layer P_{in}^t and outer layer P_{out}^t will be updated as follows.

$$
P_{\text{in}}^t = \begin{cases} \text{Remove}(P_{\text{in}}^t, cp_{\text{in},l}), \text{if } h_l^{\text{in}}(t) < \varepsilon_{\text{in}}, \forall l \\ P_{\text{in}}^t, \text{otherwise} \end{cases}, \tag{13}
$$

$$
P_{\text{out}}^t = \begin{cases} \text{Remove}(P_{\text{out}}^t, cp_{\text{out},l}), \text{if } h_l^{\text{out}}(t) < \varepsilon_{\text{out}}, \forall l \\ P_{\text{out}}^t, \text{otherwise} \end{cases}, \tag{14}
$$

where $\text{Remove}(P_{\text{in}}^t, cp_{\text{in},l})$ and $\text{Remove}(P_{\text{out}}^t, cp_{\text{out},l})$ respectively represent that the l^{th} inner layer and outer layer zeroth order cuts will be removed from P_{in}^t and P_{out}^t, $h_l^{\text{in}}(t) = h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t'}, \boldsymbol{z}_3^t)$, and $h_l^{\text{out}}(t) = h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)$.

3.3. Zeroth Order Distributed Algorithm

In this section, a distributed zeroth order algorithm is proposed. First, defining function $o(\{\boldsymbol{x}_{i,j}\}, \{\boldsymbol{z}_i\}) = \sum_l \lambda_l [\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) - \varepsilon_{\text{out}}, 0\}]^2$, where $\lambda_l > 0$ is a penalty parameter. The constrained optimization problem described in Eq. (8) is reformulated as an unconstrained optimization problem by using the exterior penalty method (Shen & Chen, 2023; Nazari et al., 2025; Kwon et al., 2024; Shi & Gu, 2021; Boyd & Vandenberghe, 2004) as follows.

$$\begin{aligned} &F(\{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \\ &= \sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) + \phi_j ||\boldsymbol{x}_{1,j} - \boldsymbol{z}_1||^2 \\ &+ o(\{\boldsymbol{x}_{i,j}\}, \{\boldsymbol{z}_i\}), \end{aligned} \tag{15}$$

where $\phi_j > 0$ is a penalty parameter. It is worth noting that the proposed DTZO is an expandable framework, allowing the incorporation of approaches beyond exterior penalty method, e.g., gradient projection based approaches (Xu et al., 2020) and Frank-Wolfe based methods (Shen et al., 2019). We chose the exterior penalty method because the lower-level problem often serves as a soft constraint (as discussed in Sec. 3.1) and using exterior penalty method offers comparatively *lower* complexity. In addition, we theoretically demonstrate that the optimal solution to the problem in Eq. (15) is a feasible solution to the original constrained problem; 2) the gap between the problem in Eq. (15) and original constrained problem will continuously decrease as λ_l, ϕ_j increase. Detailed demonstrations and discussions are provided in Appendix H. In $(t+1)^{\text{th}}$ iteration, the proposed algorithm proceeds as follows.

In Worker j. After receiving the updated parameters $\boldsymbol{z}_i^t$ and $\nabla_{\boldsymbol{x}_{i,j}} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})$, worker j updates the local variables as follows,

$$\boldsymbol{x}_{1,j}^{t+1} = \boldsymbol{x}_{1,j}^t - \eta_{\boldsymbol{x}_1} G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \tag{16}$$

$$\boldsymbol{x}_{2,j}^{t+1} = \boldsymbol{x}_{2,j}^t - \eta_{\boldsymbol{x}_2} G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \tag{17}$$

$$\boldsymbol{x}_{3,j}^{t+1} = \boldsymbol{x}_{3,j}^t - \eta_{\boldsymbol{x}_3} G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \tag{18}$$

we have that,

$$\begin{aligned} &G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) \\ &= \frac{f_{1,j}(\boldsymbol{x}_{1,j}^t + \mu \boldsymbol{u}_{k,1}, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)}{\mu} \boldsymbol{u}_{k,1} \\ &+ 2\phi_j(\boldsymbol{x}_{1,j}^t - \boldsymbol{z}_1^t), \end{aligned} \tag{19}$$

$$\begin{aligned} &G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) \\ &= \frac{f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t + \mu \boldsymbol{u}_{k,2}, \boldsymbol{x}_{3,j}^t) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)}{\mu} \boldsymbol{u}_{k,2} \\ &+ \nabla_{\boldsymbol{x}_{2,j}} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \end{aligned} \tag{20}$$

$$\begin{aligned} &G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) \\ &= \frac{f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t + \mu \boldsymbol{u}_{k,3}) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)}{\mu} \boldsymbol{u}_{k,3} \\ &+ \nabla_{\boldsymbol{x}_{3,j}} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \end{aligned} \tag{21}$$

where $\boldsymbol{u}_{k,i} \in \mathbb{R}^{d_i}, \forall i$ are standard Gaussian random vectors, $\mu > 0$ is the smoothing parameter, $\eta_{\boldsymbol{x}_i}, \forall i$ are step-sizes. Then, the updated variables $\boldsymbol{x}_{1,j}^{t+1}, \boldsymbol{x}_{2,j}^{t+1}, \boldsymbol{x}_{3,j}^{t+1}$ will be transmitted to the master.

In Master. After receiving updated variables from workers, the master performs the following steps,

1. Updating consensus variables,

$$\boldsymbol{z}_1^{t+1} = \boldsymbol{z}_1^t - \eta_{\boldsymbol{z}_1} (\sum\nolimits_j 2\phi_j(\boldsymbol{z}_1^t - \boldsymbol{x}_{1,j}^t) + \nabla_{\boldsymbol{z}_1} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})), \tag{22}$$

$$\boldsymbol{z}_2^{t+1} = \boldsymbol{z}_2^t - \eta_{\boldsymbol{z}_2} \nabla_{\boldsymbol{z}_2} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \tag{23}$$

$$\boldsymbol{z}_3^{t+1} = \boldsymbol{z}_3^t - \eta_{\boldsymbol{z}_3} \nabla_{\boldsymbol{z}_3} o(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}), \tag{24}$$

where $\eta_{\boldsymbol{z}_1}, \eta_{\boldsymbol{z}_2}$ and $\eta_{\boldsymbol{z}_3}$ are step-sizes.

2. Computing gradient of $o(\{\boldsymbol{x}_{i,j}^{t+1}\}, \{\boldsymbol{z}_i^{t+1}\})$. Broadcasting the updated parameters $\boldsymbol{z}_i^{t+1}, i = 1, 2, 3$ and $\nabla_{\boldsymbol{x}_{i,j}} o(\{\boldsymbol{x}_{i,j}^{t+1}\}, \{\boldsymbol{z}_i^{t+1}\}), i = 2, 3$ to workers.

Discussion: Trilevel learning/optimization (TLL) with *level-wise* zeroth order constraints is considered in this work, where first order information at *each level* is unavailable. Note that the proposed DTZO is versatile and can be adapted to a wide range of TLL, e.g., grey-box TLL (gradients at some levels in TLL are available (Huang et al., 2024b)), with slight adjustments. For instance, if gradients at first level in TLL are accessible, we can use gradient descent steps to replace Eq. (16)-(18). Similarly, if the second or third level gradients are available, first order based cuts, e.g., Jiao et al. (2023; 2024), can be employed to construct the cascaded polynomial approximation. Detailed discussions and comparisons are offered in Appendix I. In addition, the proposed DTZO can also be applied to bilevel zeroth order optimization by reducing the cascaded polynomial relaxation to a single-layer polynomial relaxation.

4. Theoretical Analysis

Definition 4.1. (Stationarity Gap) Following Xu et al. (2020); Jiao et al. (2023), the stationarity gap at t^{th} iteration in this problem can be expressed as,

$$\mathcal{G}^t = \begin{bmatrix} \{\nabla_{\boldsymbol{x}_{1,j}} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)\} \\ \{\nabla_{\boldsymbol{x}_{2,j}} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)\} \\ \{\nabla_{\boldsymbol{x}_{3,j}} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)\} \\ \nabla_{\boldsymbol{z}_1} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ \nabla_{\boldsymbol{z}_2} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ \nabla_{\boldsymbol{z}_3} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \end{bmatrix}. \tag{25}$$

It is seen from Eq. (25) that,

Algorithm 1 DTZO: Distributed Trilevel Zeroth Order Learning

Initialization: master iteration $t = 0$, variables $\{\boldsymbol{x}_{1,j}^0\}, \{\boldsymbol{x}_{2,j}^0\}, \{\boldsymbol{x}_{3,j}^0\}, \boldsymbol{z}_1^0, \boldsymbol{z}_2^0, \boldsymbol{z}_3^0$.
repeat
  for *local worker* j **do**
    updates the local variables $\boldsymbol{x}_{1,j}^{t+1}, \boldsymbol{x}_{2,j}^{t+1}, \boldsymbol{x}_{3,j}^{t+1}$ according to Eq. (16)-(21);
  end for
  local workers transmit the updated variables to master;
  for *master* **do**
    updates consensus variables $\boldsymbol{z}_1^{t+1}, \boldsymbol{z}_2^{t+1}, \boldsymbol{z}_3^{t+1}$ according to Eq. (22)-(24);
    computes $\nabla o(\{\boldsymbol{x}_{i,j}^{t+1}\}, \{\boldsymbol{z}_i^{t+1}\})$;
  end for
  master broadcasts the updated parameters and gradients to workers;
  if $(t+1)$ mod $\mathcal{T} == 0$ and $t < T_1$ **then**
    new inner layer zeroth order cuts are generated by Eq. (9) and (10);
    new outer layer zeroth order cuts are generated by Eq. (11) and (12);
    inactive zeroth order cuts are deleted by Eq. (13) and (14);
  end if
  $t = t + 1$;
until termination.

$$||\mathcal{G}^t||^2 = \sum_i ||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)||^2 + \sum_i \sum_j ||\nabla_{\boldsymbol{x}_{i,j}} F(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)||^2. \tag{26}$$

Definition 4.2. (ϵ-Stationary Point) $(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)$ is the stationary point when $||\mathcal{G}^t||^2 = 0$, and it is the ϵ-stationary point when $||\mathcal{G}^t||^2 \leq \epsilon$. Defining $T(\epsilon)$ as the first iteration when $||\mathcal{G}^t||^2 \leq \epsilon$, i.e., $T(\epsilon) = \min\{t|\ ||\mathcal{G}^t||^2 \leq \epsilon\}$.

Definition 4.3. (μ-Smooth Approximation) Following Ghadimi & Lan (2013); Fang et al. (2022); Kornilov et al. (2024); Rando et al. (2024), the μ-smooth approximation of a function $F(\boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}^1$ is given by,

$$\begin{aligned} F_\mu(\boldsymbol{w}) &= \frac{1}{(2\pi)^{\frac{d}{2}}} \int F(\boldsymbol{w} + \mu \boldsymbol{u}) e^{-\frac{1}{2}||\boldsymbol{u}||^2} d\boldsymbol{u} \\ &= \mathbb{E}_{\boldsymbol{u}} \left[F(\boldsymbol{w} + \mu \boldsymbol{u}) \right], \end{aligned} \tag{27}$$

where $\boldsymbol{u} \in \mathbb{R}^d$ is a standard Gaussian random vector and $\mu > 0$ is the smoothing parameter.

Assumption 4.4. (Boundedness) Following many works in machine learning and optimization, e.g., Deng et al. (2020); Jiao et al. (2023); Qian et al. (2019); Lei & Tang (2018); Zheng et al. (2017); Duchi et al. (2012); Yang et al. (2024b); Khaled & Jin (2024); Chen et al. (2024b); Hazan & Minasyan (2020), the bounded domain is assumed, i.e., $||\boldsymbol{x}_{i,j} - \boldsymbol{x}_{i,j}^*||^2 \leq \alpha_i, \forall \boldsymbol{x}_{i,j}, ||\boldsymbol{z}_i - \boldsymbol{z}_i^*||^2 \leq \alpha_i, \forall \boldsymbol{z}_i$, where $\boldsymbol{x}_{i,j}^*, \boldsymbol{z}_i^*$ denote the optimal solution. Following Cutkosky & Orabona (2019); Liu et al. (2021a); Fang et al. (2022); Shaban et al. (2019), we assume the optimal value ${F_\mu}^* > -\infty$.

Assumption 4.5. (L-smoothness) Following many work in nested optimization and zeroth order learning, e.g., Lin et al. (2024); Ghadimi & Lan (2013), we assume the gradient of function F is Lipschitz continuous with constant $L < \infty$, that is, for any point $\boldsymbol{w}, \boldsymbol{w}'$, we have that,

$$||\nabla F(\boldsymbol{w}) - \nabla F(\boldsymbol{w}')|| \leq L||\boldsymbol{w} - \boldsymbol{w}'||. \tag{28}$$

It is worth noting that Assumptions 4.4 and 4.5 are *mild* and *commonly used* in machine learning and optimization. Detailed discussions are provided in Appendix G.

Theorem 4.6. ***(Iteration Complexity)*** *Under Assumption 4.4 and 4.5, by setting step-sizes* $\eta_{\boldsymbol{x}_i} = \eta_{\boldsymbol{z}_i} = \min\left\{\frac{1}{8L(d_1+4)}, \frac{1}{8L(d_2+4)}, \frac{1}{8L(d_3+4)}, \frac{3}{2(L+1)}, \frac{1}{\sqrt{T(\epsilon)-T_1}}\right\}$, $i = 1, 2, 3$ *and letting smoothing parameter* $0 < \mu \leq \frac{1}{\sqrt{T(\epsilon)-T_1}}$, *we have that,*

$$T(\epsilon) \sim \mathcal{O}\left((\sum_{i=1}^{3} \overline{c_i} + \overline{d}(\max_{t\in[T_1]} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) - {F_\mu}^*))^2 \frac{1}{\epsilon^2} + T_1 \right), \tag{29}$$

where constants $\overline{d} = 4(1 + \max\{8L(d_1+4), 8L(d_2+4), 8L(d_3+4), \frac{2(L+1)}{3}\})$ *and* $\overline{c_i} = \frac{L^2(d_i+6)^3}{4(d_i+4)} + L^2(d_i+3)^3 + 4L(N+1)d_i(\max\{8L(d_1+4), 8L(d_2+4), 8L(d_3+4), \frac{2(L+1)}{3}\}+1)$. $T_1 > 0$ *is a constant that controls the cascaded polynomial approximation, as discussed in Sec. 3.2. Detailed proofs of Theorem 4.6 are provided in Appendix A, with further discussions offered below.*

Theorem 4.7. ***(Communication Complexity)*** *The overall communication complexity of the proposed DTZO can be divided into the communication complexity at every iteration (C_1) and the communication complexity of updating zeroth order cuts (C_2). Specifically, the overall communication complexity can be expressed as* $C_1 + C_2 = T(\epsilon)(2d_1 + 3d_2 + 3d_3)N + 2N\lfloor \frac{T_1}{\mathcal{T}} \rfloor \mathcal{T}(d_2 + d_3)$. *The detailed proofs are provided in Appendix B, with discussions offered below.*

Discussion: It is seen from Theorem 4.6 and 4.7 that the proposed framework DTZO can *flexibly* control the trade-off between the performance of cascaded polynomial approximation and the iteration complexity (i.e., $T(\epsilon)$ in Theorem 4.6) and communication complexity (i.e., $C_1 + C_2$ in Theorem 4.7) by adjusting a single parameter T_1. Specifically, a larger T_1 corresponds to a better cascaded polynomial approximation, but it also entails higher iteration and communication complexity. Consequently, if the distributed system has limited computational and communication capabilities, a smaller value of T_1 can be selected. Conversely, if a higher

quality of cascaded polynomial approximation is desired, a larger value of T_1 can be chosen, which demonstrates the flexibility in the proposed framework. In addition, as shown in Theorem 4.6, the iteration complexity of the proposed distributed trilevel zeroth order learning framework can be written as $\mathcal{O}(\sum_i d_i^6/\epsilon^2)$. It is worth mentioning that the dimension-dependent iteration complexity is *common* in zeroth order optimization, as discussed in various works, e.g., Zhang et al. (2024b;a); Duchi et al. (2015); Sun et al. (2022); Qiu et al. (2023). For instance, the iteration complexity of the state-of-the-art distributed bilevel zeroth order learning method (Qiu et al., 2023) is given by $\mathcal{O}(d^8/\epsilon^2)$, where d denotes the dimension of variables.

5. Experiments

In the experiment, two distributed trilevel zeroth order learning scenarios, i.e., black-box trilevel learning on large language models (LLMs) and robust hyperparameter optimization are used to evaluate the performance of the proposed DTZO. The proposed DTZO is compared with the state-of-the-art distributed zeroth order learning method FedZOO (Fang et al., 2022) and distributed bilevel zeroth order learning method FedRZO$_{\text{bl}}$ (Qiu et al., 2023). Moreover, to further demonstrate the effectiveness of the proposed DTZO, we also compare it against state-of-the-art distributed bilevel and trilevel optimization methods equipped with zeroth order estimators (Liu et al., 2020), including FEDNEST+ZO (Tarzanagh et al., 2022), ADBO+ZO (Jiao et al., 2023), and AFTO+ZO (Jiao et al., 2024). It is important to note that combining distributed nested optimization methods with zeroth order estimators does not provide any theoretical guarantees; these methods are included only for comparative evaluation. In the experiment, all the models are implemented using PyTorch, and the experiments are conducted on a server equipped with two NVIDIA RTX 4090 GPUs. More experimental details are provided in Appendix F.

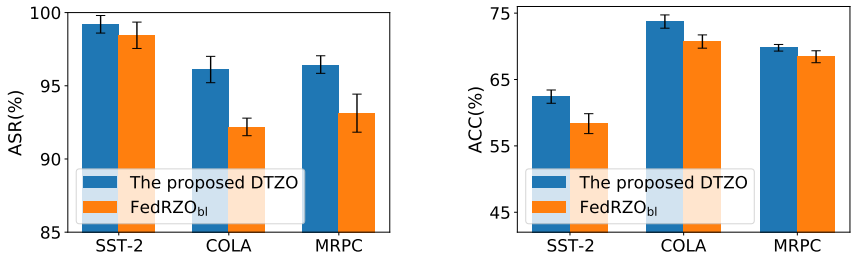
5.1. Black-Box Trilevel Learning

Prompt learning is a key technique for enabling LLMs to efficiently and effectively adapt to various downstream tasks (Ma et al., 2024; Wang et al., 2024). In many practical scenarios involving LLMs, access to first order information is restricted due to the proprietary nature of these models or API constraints. For instance, commercial LLM APIs only allow input-output interactions and do not provide visibility into gradients. Inspired by the black-box prompt learning (Diao et al., 2022) and backdoor attack on prompt-based LLMs (Yao et al., 2024; Jiao et al., 2025), the backdoor attack on black-box LLMs is considered in the experiment, which can be expressed as a black-box trilevel learning problem. In the experiment, Qwen2-7B (Yang et al., 2024a), Llama-3.1-8B (Grattafiori et al., 2024), and Qwen-1.8B-Chat (Bai et al., 2023), are utilized as the black-box LLMs. The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018a) is used to evaluate the proposed DTZO. Specifically, the experiments are carried out on: 1) SST-2 for sentiment analysis; 2) COLA for linguistic acceptability; and 3) MRPC for semantic equivalence of sentences. Details of the problem formulation and experimental setting are shown in Appendix F. In this task, we aim to discover effective backdoor triggers while ensuring model performance on clean inputs (i.e., inputs without triggers). Therefore, following Yao et al. (2024), the Attack Success Rate (ASR) when the triggers are activated and the Accuracy (ACC) on clean samples are utilized as the metrics in the experiments. The comparisons between the proposed DTZO and the state-of-the-art distributed bilevel zeroth order learning method FedRZO$_{\text{bl}}$ are illustrated in Figures 1, 2, and 3. It is seen from Figures 1, 2, and 3 that the proposed DTZO can effectively tackle the distributed trilevel zeroth order learning problem and achieve superior performance than FedRZO$_{\text{bl}}$ since the proposed DTZO is capable of addressing higher-nested zeroth order learning problems compared to FedRZO$_{\text{bl}}$.

5.2. Robust Hyperparameter Optimization

Inspired by Sato et al. (2021); Jiao et al. (2024) in trilevel learning, the robust hyperparameter optimization is considered in the experiment. Following the setting for non-differentiable functions as described in Qiu et al. (2023), ReLU neural networks are employed in the experiments. The digits recognition tasks in Qian et al. (2019); Wang et al. (2021) with several benchmark datasets, i.e., MNIST (LeCun et al., 1998), USPS, Fashion MNIST (Xiao et al., 2017), and QMNIST (Yadav & Bottou, 2019), are utilized to assess the performance of the proposed DTZO. In addition, DTZO is also assessed on time series datasets, including MelbournePedestrian, Crop, and UWaveGestureLibraryAll, sourced from the UCR Archive (Dau et al., 2018). The average across accuracy on clean samples and robustness against adversarial samples is used as the metric, more details about the experimental setting and problem formulation are provided in Appendix F. We compare the proposed DTZO with the state-of-the-art methods in Table 2. It is seen from Table 2 that the proposed DTZO can effectively tackle the trilevel zeroth order learning problem in a distributed manner. The superior performance of DTZO, as compared to state-of-the-art methods, can be attributed to two key factors: (1) Compared to existing methods, the proposed DTZO is capable of effectively addressing higher-nested zeroth order optimization problems with non-asymptotic convergence guarantees. (2) The proposed nonlinear zeroth order cuts facilitate the development of a more refined cascaded polynomial relaxation.

Within the proposed framework, the trade-off between complexity and performance can be flexibly controlled by ad-



(a) Comparisons about ASR (b) Comparisons about ACC

Figure 1. Comparisons about ASR and ACC between the proposed DTZO and state-of-the-art method using Qwen-1.8B-Chat.

Table 2. Comparisons between the proposed DTZO and the state-of-the-art methods. Higher scores represent better performance.

Datasets	FedZOO	FEDNEST+ZO	ADBO+ZO	FedRZObl	AFTO+ZO	**DTZO**
MNIST	0.5289	0.5503	0.5341	0.5405	0.7501	**0.7927**
QMNIST	0.5245	0.5398	0.5487	0.5467	0.7389	**0.7804**
F-MNIST	0.4874	0.5065	0.5102	0.5023	0.6448	**0.7007**
USPS	0.7277	0.7354	0.7323	0.7379	0.7987	**0.8513**
MelbournePedestrian	0.6295	0.6454	0.6412	0.6487	0.6924	**0.7250**
Crop	0.5468	0.5607	0.5681	0.5645	0.6016	**0.6351**
UWaveGestureLibraryAll	0.6714	0.6924	0.6983	0.7002	0.7689	**0.8243**

justing T_1, as discussed in Sec. 4. As shown in Figure 4 in Appendix F, the performance of DTZO improves as T_1 increases, we can flexibly adjust T_1 based on the distributed system requirements. Removing inactive cuts can significantly improve the effectiveness of cutting plane method, as discussed in Jiao et al. (2024); Yang et al. (2014). In the experiment, we also investigate the effect of removing inactive cuts within the proposed DTZO. It is seen from Figure 5 in Appendix F that pruning inactive cuts significantly reduces training time, indicating the importance of this procedure.

In addition, the impact of various choices of T_1 on the convergence rate within the proposed DTZO is evaluated. As illustrated in Figures 6 and 7 in Appendix F, a smaller T_1 leads to faster convergence but affects the method's performance, resulting in a higher test loss. Conversely, if a better performance is required, a larger T_1 can be selected, corresponding to a more refined polynomial relaxation. In the proposed framework, we can *flexibly* adjust T_1 based on various requirements. The results in Figures 6 and 7 are consistent with our theoretical analyses presented under Theorems 4.6 and 4.7.

Following Qiu et al. (2023), the robustness in the proposed framework with respect to the choice of smoothing parameter μ is evaluated. The experiments are conducted on the robust hyperparameter optimization task under various settings of smoothing parameter $\mu \in \{0.01, 0.001, 0.0001\}$. It is seen from Figure 8 and 9 in Appendix F that the proposed DTZO is robust to the choice of smoothing parameter μ. In addition, we also note that the proposed DTZO has a faster convergence rate with a relatively smaller μ, because the gradient estimate improves when μ becomes relatively smaller, as discussed in Liu et al. (2020). Furthermore, to analyze DTZO's performance improvements, we conduct an ablation study comparing DTZO against its variants: DTZO(-) and DBZO. DTZO(-) replaces the proposed nonlinear cuts in DTZO with linear cuts, while DBZO removes cascaded polynomial approximation, using only single-layer polynomial approximation. It is seen from Table 5 in Appendix F that DTZO outperforms all variants, demonstrating the benefits of cascaded polynomial approximation and nonlinear zeroth order cuts.

6. Conclusion

In this work, a distributed trilevel zeroth order learning (DTZO) framework is proposed to address the trilevel learning problems in a distributed manner without using first order information. To our best knowledge, this is the first work that considers how to tackle the trilevel zeroth order learning problems. The proposed DTZO is capable of constructing the cascaded polynomial approximation for trilevel zeroth order learning problems without using gradients or sub-gradients by utilizing the novel zeroth order cuts. Additionally, we theoretically analyze the non-asymptotic convergence rate for the proposed DTZO to achieve the ϵ-stationary point. Experiments on black-box LLMs trilevel learning and robust hyperparameter optimization demonstrate the superior performance of DTZO.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 12371519 and 61771013; in part by Asiainfo Technologies; in part by the Fundamental Research Funds for the Central Universities of China; and in part by the Fundamental Research Funds of Shanghai Jiading District.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Akhavan, A., Pontil, M., and Tsybakov, A. Distributed zero-order optimization under adversarial noise. *Advances in Neural Information Processing Systems*, 34:10209–10220, 2021.

Assran, M., Aytekin, A., Feyzmahdavian, H. R., Johansson, M., and Rabbat, M. G. Advances in asynchronous parallel and distributed optimization. *Proceedings of the IEEE*, 108(11):2013–2031, 2020.

Astudillo, R. and Frazier, P. I. Thinking inside the box: A tutorial on grey-box bayesian optimization. In *2021 Winter Simulation Conference (WSC)*, pp. 1–15. IEEE, 2021.

Avraamidou, S. Mixed-integer multi-level optimization through multi-parametric programming. 2018.

Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Bajaj, I., Iyer, S. S., and Hasan, M. F. A trust region-based two phase algorithm for constrained black-box and grey-box optimization with infeasible initial point. *Computers & Chemical Engineering*, 116:306–321, 2018.

Balashov, M., Polyak, B., and Tremba, A. Gradient projection and conditional gradient methods for constrained nonconvex minimization. *Numerical Functional Analysis and Optimization*, 41(7):822–849, 2020.

Ben-Ayed, O. and Blair, C. E. Computational difficulties of bilevel linear programming. *Operations Research*, 38(3): 556–560, 1990.

Bertsekas, D. *Convex optimization algorithms*. Athena Scientific, 2015.

Beykal, B., Avraamidou, S., Pistikopoulos, I. P., Onel, M., and Pistikopoulos, E. N. Domino: Data-driven optimization of bi-level mixed-integer nonlinear problems. *Journal of Global Optimization*, 78:1–36, 2020.

Blair, C. The computational complexity of multi-level linear programs. *Annals of Operations Research*, 34, 1992.

Boyd, S. and Vandenberghe, L. Localization and cutting-plane methods. *From Stanford EE 364b lecture notes*, 386, 2007.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Bürger, M., Notarstefano, G., and Allgöwer, F. A polyhedral approximation framework for convex and robust distributed optimization. *IEEE Transactions on Automatic Control*, 59(2):384–395, 2013.

Cai, H., Lou, Y., McKenzie, D., and Yin, W. A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization. In *International Conference on Machine Learning*, pp. 1193–1203. PMLR, 2021.

Cao, J., Jiang, R., Abolfazli, N., Yazdandoost Hamedani, E., and Mokhtari, A. Projection-free methods for stochastic simple bilevel optimization with convex lower-level problem. *Advances in Neural Information Processing Systems*, 36, 2024.

Chen, J., Chen, H., Gu, B., and Deng, H. Fine-grained theoretical analysis of federated zeroth-order optimization. *Advances in Neural Information Processing Systems*, 36, 2024a.

Chen, S., Zhang, Y.-J., Tu, W.-W., Zhao, P., and Zhang, L. Optimistic online mirror descent for bridging stochastic and adversarial online convex optimization. *Journal of Machine Learning Research*, 25(178):1–62, 2024b.

Chen, X., Liu, S., Xu, K., Li, X., Lin, X., Hong, M., and Cox, D. Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization. *Advances in neural information processing systems*, 32, 2019.

Chen, X., Xiao, T., and Balasubramanian, K. Optimal algorithms for stochastic bilevel optimization under relaxed smoothness conditions. *arXiv preprint arXiv:2306.12067*, 2023.

Chen, X., Xiao, T., and Balasubramanian, K. Optimal algorithms for stochastic bilevel optimization under relaxed smoothness conditions. *Journal of Machine Learning Research*, 25(151):1–51, 2024c.

Chen, X., Xiong, Y., and Yang, K. Robust beamforming for downlink multi-cell systems: A bilevel optimization

perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024d.

Choe, S. K., Neiswanger, W., Xie, P., and Xing, E. Betty: An automatic differentiation library for multilevel optimization. In *The Eleventh International Conference on Learning Representations*, 2023.

Cutkosky, A. and Orabona, F. Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32, 2019.

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., and Keogh, E. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2018.

Deng, Y., Kamani, M. M., and Mahdavi, M. Distributionally robust federated averaging. *Advances in neural information processing systems*, 33:15111–15122, 2020.

Diao, S., Huang, Z., Xu, R., Li, X., Yong, L., Zhou, X., and Zhang, T. Black-box prompt learning for pre-trained language models. *Transactions on Machine Learning Research*, 2022.

Duchi, J. C., Agarwal, A., Johansson, M., and Jordan, M. I. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012.

Duchi, J. C., Jordan, M. I., Wainwright, M. J., and Wibisono, A. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

Fang, W., Yu, Z., Jiang, Y., Shi, Y., Jones, C. N., and Zhou, Y. Communication-efficient stochastic zeroth-order optimization for federated learning. *IEEE Transactions on Signal Processing*, 70:5058–5073, 2022.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Franc, V., Sonnenburg, S., and Werner, T. Cutting plane methods in machine learning. *Optimization for Machine Learning*, pp. 185–218, 2011.

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

Gao, H. Decentralized multi-level compositional optimization algorithms with level-independent convergence rate. In *International Conference on Artificial Intelligence and Statistics*, pp. 4402–4410. PMLR, 2024.

Gao, H. and Huang, H. Can stochastic zeroth-order frank-wolfe method converge faster for non-convex problems? In *International conference on machine learning*, pp. 3377–3386. PMLR, 2020.

Gao, H., Li, J., and Huang, H. On the convergence of local stochastic compositional gradient descent with momentum. In *International Conference on Machine Learning*, pp. 7017–7035. PMLR, 2022.

Gao, H., Gu, B., and Thai, M. T. On the convergence of distributed stochastic bilevel optimization algorithms over a network. In *International Conference on Artificial Intelligence and Statistics*, pp. 9238–9281. PMLR, 2023.

Garber, D. and Hazan, E. Faster rates for the frank-wolfe method over strongly-convex sets. In *International Conference on Machine Learning*, pp. 541–549. PMLR, 2015.

Garg, B., Zhang, L., Sridhara, P., Hosseini, R., Xing, E., and Xie, P. Learning from mistakes–a framework for neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10184–10192, 2022.

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

Giovannelli, T., Kent, G. D., and Vicente, L. N. A stochastic gradient method for trilevel optimization. *arXiv preprint arXiv:2505.06805*, 2025.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Guo, H., Hosseini, R., Zhang, R., Somayajula, S. A., Chowdhury, R. R., Gupta, R. K., and Xie, P. Downstream task guided masking learning in masked autoencoders using multi-level optimization. *Transactions on Machine Learning Research*, 2024.

Guo, M., Yang, Y., Xu, R., Liu, Z., and Lin, D. When nas meets robustness: In search of robust architectures against adversarial attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 631–640, 2020.

Han, P., Shi, X., and Huang, J. Fedal: Black-box federated knowledge distillation enabled by adversarial learning. *IEEE Journal on Selected Areas in Communications*, 2024.

Hazan, E. and Minasyan, E. Faster projection-free online learning. In *Conference on Learning Theory*, pp. 1877–1893. PMLR, 2020.

He, Y., Zhang, R., Somayajula, S. A., and Xie, P. Transformer architecture search for improving out-of-domain generalization in machine translation. *Transactions on Machine Learning Research*, 2024.

Héliou, A., Martin, M., Mertikopoulos, P., and Rahier, T. Zeroth-order non-convex learning via hierarchical dual averaging. In *International Conference on Machine Learning*, pp. 4192–4202. PMLR, 2021.

Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic. *SIAM Journal on Optimization*, 33(1):147–180, 2023.

Huang, F., Gao, S., Pei, J., and Huang, H. Nonconvex zeroth-order stochastic admm methods with lower function query complexity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024a.

Huang, H., Li, Y., Jiang, B., Jiang, B., Liu, L., Liu, Z., Sun, R., and Liang, S. Enhancing the resilience of llms against grey-box extractions. In *ICML 2024 Next Generation of AI Safety Workshop*, 2024b.

Huang, Y., Yang, K., Zhu, Z., and Chen, L. Triadic-ocd: Asynchronous online change detection with provable robustness, optimality, and convergence. In *International Conference on Machine Learning*, pp. 20382–20412. PMLR, 2024c.

Ji, K., Yang, J., and Liang, Y. Bilevel optimization: Convergence analysis and enhanced design. In *International conference on machine learning*, pp. 4882–4892. PMLR, 2021.

Jian, C., Yang, K., and Jiao, Y. Tri-level navigator: Llm-empowered tri-level learning for time series ood generalization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Jiang, R., Abolfazli, N., Mokhtari, A., and Hamedani, E. Y. A conditional gradient-based method for simple bilevel optimization with convex lower-level problem. In *International Conference on Artificial Intelligence and Statistics*, pp. 10305–10323. PMLR, 2023.

Jiao, Y., Yang, K., and Song, D. Distributed distributionally robust optimization with non-convex objectives. *Advances in neural information processing systems*, 35: 7987–7999, 2022a.

Jiao, Y., Yang, K., Song, D., and Tao, D. Timeautoad: Autonomous anomaly detection with self-supervised contrastive loss for multivariate time series. *IEEE Transactions on Network Science and Engineering*, 9(3):1604–1619, 2022b.

Jiao, Y., Yang, K., Wu, T., Song, D., and Jian, C. Asynchronous distributed bilevel optimization. In *The Eleventh International Conference on Learning Representations*, 2023.

Jiao, Y., Yang, K., Wu, T., Jian, C., and Huang, J. Provably convergent federated trilevel learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 12928–12937, 2024.

Jiao, Y., Wang, X., and Yang, K. Pr-attack: Coordinated prompt-rag attacks on retrieval-augmented generation in large language models via bilevel optimization. *arXiv preprint arXiv:2504.07717*, 2025.

Jing, G., Bai, H., George, J., Chakrabortty, A., and Sharma, P. K. Asynchronous distributed reinforcement learning for lqr control via zeroth-order block coordinate descent. *IEEE Transactions on Automatic Control*, 2024.

Kautz, H. A., Selman, B., and Jiang, Y. A general stochastic approach to solving problems with hard and soft constraints. *Satisfiability Problem: Theory and Applications*, 35:573–586, 1996.

Khaled, A. and Jin, C. Tuning-free stochastic optimization. In *Forty-first International Conference on Machine Learning*, 2024.

Kornilov, N., Shamir, O., Lobanov, A., Dvinskikh, D., Gasnikov, A., Shibaev, I., Gorbunov, E., and Horváth, S. Accelerated zeroth-order method for non-smooth stochastic convex optimization problem with infinite variance. *Advances in Neural Information Processing Systems*, 36, 2024.

Kornowski, G. and Shamir, O. An algorithm with optimal dimension-dependence for zero-order nonsmooth nonconvex stochastic optimization. *Journal of Machine Learning Research*, 25(122):1–14, 2024.

Kwon, J., Kwon, D., Wright, S., and Nowak, R. D. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning*, pp. 18083–18113. PMLR, 2023.

Kwon, J., Kwon, D., Wright, S., and Nowak, R. D. On penalty methods for nonconvex bilevel optimization and first-order stochastic approximation. In *The Twelfth International Conference on Learning Representations*, 2024.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lei, Y. and Tang, K. Stochastic composite mirror descent: Optimal bounds with high probabilities. *Advances in Neural Information Processing Systems*, 31, 2018.

Li, J., Huang, F., and Huang, H. Communication-efficient federated bilevel optimization with local and global lower level problems. *arXiv preprint arXiv:2302.06701*, 2023.

Li, J., Huang, F., and Huang, H. Communication-efficient federated bilevel optimization with global and local lower level problems. *Advances in Neural Information Processing Systems*, 36, 2024.

Li, W. and Assaad, M. Distributed zeroth-order stochastic optimization in time-varying networks. *arXiv preprint arXiv:2105.12597*, 2021.

Li, Z., Chen, P.-Y., Liu, S., Lu, S., and Xu, Y. Zeroth-order optimization for composite problems with functional constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7453–7461, 2022.

Lian, X., Zhang, H., Hsieh, C.-J., Huang, Y., and Liu, J. A comprehensive linear speedup analysis for asynchronous stochastic parallel optimization from zeroth-order to first-order. *Advances in Neural Information Processing Systems*, 29, 2016.

Liang, H., Sun, B., Huang, B., Li, Y., and Yang, C. A novel chattering-free discrete sliding mode controller with disturbance compensation for zinc roasting temperature distribution control. *IEEE Transactions on Automation Science and Engineering*, 2024.

Lin, S., Sow, D., Ji, K., Liang, Y., and Shroff, N. Non-convex bilevel optimization with time-varying objective functions. *Advances in Neural Information Processing Systems*, 36, 2024.

Liu, B., Liu, X., Jin, X., Stone, P., and Liu, Q. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021a.

Liu, B., Ye, M., Wright, S., Stone, P., and Liu, Q. Bome! bilevel optimization made easy: A simple first-order approach. *Advances in Neural Information Processing Systems*, 35:17248–17262, 2022.

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

Liu, R., Gao, J., Zhang, J., Meng, D., and Lin, Z. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (12):10045–10067, 2021b.

Liu, S., Chen, P.-Y., Chen, X., and Hong, M. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2018b.

Liu, S., Kailkhura, B., Chen, P.-Y., Ting, P., Chang, S., and Amini, L. Zeroth-order stochastic variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018c.

Liu, S., Chen, P.-Y., Kailkhura, B., Zhang, G., Hero III, A. O., and Varshney, P. K. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Processing Magazine*, 37(5):43–54, 2020.

Ma, H., Zhang, C., Bian, Y., Liu, L., Zhang, Z., Zhao, P., Zhang, S., Fu, H., Hu, Q., and Wu, B. Fairness-guided few-shot prompting for large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Mackay, M., Vicol, P., Lorraine, J., Duvenaud, D., and Grosse, R. Self-tuning networks: Bilevel optimization of hyperparameters using structured best-response functions. In *International Conference on Learning Representations*, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Nazari, P., Mousavi, A., Tarzanagh, D. A., and Michailidis, G. A penalty-based method for communication-efficient decentralized bilevel programming. *Automatica*, 173: 112039, 2025.

Nikolakakis, K., Haddadpour, F., Kalogerias, D., and Karbasi, A. Black-box generalization: Stability of zeroth-order learning. *Advances in Neural Information Processing Systems*, 35:31525–31541, 2022.

Pan, R., Zhang, J., Pan, X., Pi, R., Wang, X., and Zhang, T. Scalebio: Scalable bilevel optimization for llm data reweighting. *arXiv preprint arXiv:2406.19976*, 2024.

Qian, Q., Zhu, S., Tang, J., Jin, R., Sun, B., and Li, H. Robust optimization over multiple domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4739–4746, 2019.

Qiu, Y., Shanbhag, U., and Yousefian, F. Zeroth-order methods for nondifferentiable, nonconvex, and hierarchical federated optimization. *Advances in Neural Information Processing Systems*, 36, 2023.

Raghu, A., Lorraine, J., Kornblith, S., McDermott, M., and Duvenaud, D. K. Meta-learning to improve pre-training. *Advances in Neural Information Processing Systems*, 34: 23231–23244, 2021.

Rando, M., Molinari, C., Rosasco, L., and Villa, S. An optimal structured zeroth-order algorithm for non-smooth

optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Régin, J.-C. Using hard constraints for representing soft constraints. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: 8th International Conference, CPAIOR 2011, Berlin, Germany, May 23-27, 2011. Proceedings 8*, pp. 176–189. Springer, 2011.

Ren, Z., Tang, Y., and Li, N. Escaping saddle points in zeroth-order optimization: the power of two-point estimators. In *International Conference on Machine Learning*, pp. 28914–28975. PMLR, 2023.

Sahu, A. K., Jakovetic, D., Bajovic, D., and Kar, S. Distributed zeroth order optimization over random networks: A kiefer-wolfowitz stochastic approximation approach. In *2018 IEEE Conference on Decision and Control (CDC)*, pp. 4951–4958. IEEE, 2018.

Sato, R., Tanaka, M., and Takeda, A. A gradient method for multilevel optimization. *Advances in Neural Information Processing Systems*, 34:7522–7533, 2021.

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

Shen, H. and Chen, T. On penalty-based bilevel gradient descent method. *arXiv preprint arXiv:2302.05185*, 2023.

Shen, H., Yang, Z., and Chen, T. Principled penalty-based methods for bilevel reinforcement learning and rlhf. *arXiv preprint arXiv:2402.06886*, 2024.

Shen, Z., Fang, C., Zhao, P., Huang, J., and Qian, H. Complexities in projection-free stochastic non-convex minimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2868–2876. PMLR, 2019.

Shi, W. and Gu, B. Improved penalty method via doubly stochastic gradients for bilevel hyperparameter optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 9621–9629, 2021.

Shu, Y., Lin, X., Dai, Z., and Low, B. K. H. Federated zeroth-order optimization using trajectory-informed surrogate gradients. *arXiv preprint arXiv:2308.04077*, 2023.

Sinha, A., Malo, P., and Deb, K. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

Sra, S., Yu, A. W., Li, M., and Smola, A. Adadelay: Delay adaptive distributed stochastic optimization. In *Artificial Intelligence and Statistics*, pp. 957–965. PMLR, 2016.

Subramanya, T. and Riggio, R. Centralized and federated learning for predictive vnf autoscaling in multi-domain 5g networks and beyond. *IEEE Transactions on Network and Service Management*, 18(1):63–78, 2021.

Sun, T., Shao, Y., Qian, H., Huang, X., and Qiu, X. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*, pp. 20841–20855. PMLR, 2022.

Tang, Y., Zhang, J., and Li, N. Distributed zero-order algorithms for nonconvex multiagent optimization. *IEEE Transactions on Control of Network Systems*, 8(1):269–281, 2020.

Tarzanagh, D. A., Li, M., Thrampoulidis, C., and Oymak, S. Fednest: Federated bilevel, minimax, and compositional optimization. In *International Conference on Machine Learning*, pp. 21146–21179. PMLR, 2022.

Temlyakov. Nonlinear methods of approximation. *Foundations of Computational Mathematics*, 3:33–107, 2003.

Tu, C.-C., Ting, P., Chen, P.-Y., Liu, S., Zhang, H., Yi, J., Hsieh, C.-J., and Cheng, S.-M. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 742–749, 2019.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018a.

Wang, B., Wang, Z., Wang, X., Cao, Y., A Saurous, R., and Kim, Y. Grammar prompting for domain-specific language generation with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Wang, J., Chen, J., Lin, J., Sigal, L., and de Silva, C. W. Discriminative feature alignment: Improving transferability of unsupervised domain adaptation by gaussian-guided latent alignment. *Pattern Recognition*, 116:107943, 2021.

Wang, Y., Du, S., Balakrishnan, S., and Singh, A. Stochastic zeroth-order optimization in high dimensions. In *International conference on artificial intelligence and statistics*, pp. 1356–1365. PMLR, 2018b.

Wang, Y.-X., Sadhanala, V., Dai, W., Neiswanger, W., Sra, S., and Xing, E. Parallel and distributed block-coordinate

frank-wolfe algorithms. In *International Conference on Machine Learning*, pp. 1548–1557. PMLR, 2016.

Wilson, E., Mueller, F., and Pakin, S. Combining hard and soft constraints in quantum constraint-satisfaction systems. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–14. IEEE, 2022.

Wu, X., Sun, J., Hu, Z., Li, J., Zhang, A., and Huang, H. Federated conditional stochastic optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Xian, W., Huang, F., and Huang, H. Communication-efficient frank-wolfe algorithm for nonconvex decentralized distributed learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10405–10413, 2021.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xiao, Q., Shen, H., Yin, W., and Chen, T. Alternating projected sgd for equality-constrained bilevel optimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 987–1023. PMLR, 2023.

Xu, Z., Zhang, H., Xu, Y., and Lan, G. A unified single-loop alternating gradient projection algorithm for nonconvex-concave and convex-nonconcave minimax problems. *arXiv preprint arXiv:2006.02032*, 2020.

Yadav, C. and Bottou, L. Cold case: The lost mnist digits. *Advances in neural information processing systems*, 32, 2019.

Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., Dong, G., Wei, H., Lin, H., Tang, J., Wang, J., Yang, J., Tu, J., Zhang, J., Ma, J., Yang, J., Xu, J., Zhou, J., Bai, J., He, J., Lin, J., Dang, K., Lu, K., Chen, K., Yang, K., Li, M., Xue, M., Ni, N., Zhang, P., Wang, P., Peng, R., Men, R., Gao, R., Lin, R., Wang, S., Bai, S., Tan, S., Zhu, T., Li, T., Liu, T., Ge, W., Deng, X., Zhou, X., Ren, X., Zhang, X., Wei, X., Ren, X., Liu, X., Fan, Y., Yao, Y., Zhang, Y., Wan, Y., Chu, Y., Liu, Y., Cui, Z., Zhang, Z., Guo, Z., and Fan, Z. Qwen2 technical report, 2024a. URL https://arxiv.org/abs/2407.10671.

Yang, J., Ji, K., and Liang, Y. Provably faster algorithms for bilevel optimization. *Advances in Neural Information Processing Systems*, 34:13670–13682, 2021.

Yang, K., Huang, J., Wu, Y., Wang, X., and Chiang, M. Distributed robust optimization (DRO), part I: Framework and example. *Optimization and Engineering*, 15(1):35–67, 2014.

Yang, W., Wang, Y., Zhao, P., and Zhang, L. Universal online convex optimization with 1 projection per round. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b.

Yang, Y., Xiao, P., Ma, S., and Ji, K. First-order federated bilevel learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22029–22037, 2025.

Yao, H., Lou, J., and Qin, Z. Poisonprompt: Backdoor attack on prompt-based large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7745–7749. IEEE, 2024.

Yue, P., Yang, L., Fang, C., and Lin, Z. Zeroth-order optimization with weak dimension dependency. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 4429–4472. PMLR, 2023.

Zhang, H., Zhang, H., Gu, B., and Chang, Y. Subspace selection based prompt tuning with nonconvex nonsmooth black-box optimization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4179–4190, 2024a.

Zhang, M., Shen, Z., Mokhtari, A., Hassani, H., and Karbasi, A. One sample stochastic frank-wolfe. In *International Conference on Artificial Intelligence and Statistics*, pp. 4012–4023. PMLR, 2020.

Zhang, Y., Zhang, G., Khanduri, P., Hong, M., Chang, S., and Liu, S. Revisiting and advancing fast adversarial training through the lens of bi-level optimization. In *International Conference on Machine Learning*, pp. 26693–26712. PMLR, 2022.

Zhang, Y., Li, P., Hong, J., Li, J., Zhang, Y., Zheng, W., Chen, P.-Y., Lee, J. D., Yin, W., Hong, M., et al. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark. In *Forty-first International Conference on Machine Learning*, 2024b.

Zheng, S., Meng, Q., Wang, T., Chen, W., Yu, N., Ma, Z.-M., and Liu, T.-Y. Asynchronous stochastic gradient descent with delay compensation. In *International conference on machine learning*, pp. 4120–4129. PMLR, 2017.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pp. 928–936, 2003.

Appendix

To improve the readability of the Appendix, we have organized its contents as follows: In Appendix A and B, we delve into the comprehensive proofs of Theorem 4.6 (Iteration Complexity) and Theorem 4.7 (Communication Complexity). In Appendix C, the detailed proofs of Proposition 3.1 and 3.2 are provided. Furthermore, we offer the theoretical analyses about the cascaded polynomial approximation in Appendix D. Additionally, detailed discussions about the soft constraint are given in Appendix E, and the discussions about ϕ_{in} and ϕ_{out} are also conducted in this part. In Appendix F, details of the experimental setting and additional experimental results are provided. The discussions about Assumption 4.4 and 4.5 are offered in Appendix G, we show that both Assumption 4.4 and 4.5 are mild and widely-used in machine learning and optimization. In Appendix H, the reasons why we choose the exterior penalty method in the proposed framework are discussed, and we demonstrate the close relationship between the original constrained optimization problem and the unconstrained optimization problem. In Appendix I, we show that the proposed framework can be applied to a wide range of TLL problems, e.g., (grey-box) TLL with partial zeroth order constraints. More discussions about the cutting plane method and the choice of gradient estimator are provided in Appendix J. Lastly, the future work is discussed in Appendix K.

Furthermore, to enhance the readability of this work, the notations used in this work and their corresponding meanings are summarized in Table 3.

Table of Contents

Table 3. Notations used in this work and the corresponding meanings.

Notation	Meaning
$f_i(\cdot), \forall i = 1,2,3$	i^{th} level objective.
$\boldsymbol{x}_i, \forall i = 1,2,3$	i^{th} level variable.
$f_{i,j}(\cdot), \forall i = 1,2,3, j = 1,\cdots,N$	i^{th} level local objective in worker j.
$\boldsymbol{x}_{i,j}, \forall i = 1,2,3, j = 1,\cdots,N$	i^{th} level local variable in worker j.
$\boldsymbol{z}_i, \forall i = 1,2,3$	i^{th} level global variable in master.
P_{in}, P_{out}	feasible regions formed by inner and outer layer zeroth order cuts.
$cp_{\text{in},l}, cp_{\text{out},l}$	l^{th} inner layer and outer layer zeroth order cuts.
$\boldsymbol{a}_{j,l}^{\text{in}}$, $\boldsymbol{b}_{j,l}^{\text{in}}$, $\boldsymbol{c}_{i,l}^{\text{in}}$, $\boldsymbol{d}_{i,l}^{\text{in}}$, e_l^{in}	l^{th} inner layer zeroth order cut's parameters.
$\boldsymbol{a}_{i,j,l}^{\text{out}}$, $\boldsymbol{b}_{i,j,l}^{\text{out}}$, $\boldsymbol{c}_{i,l}^{\text{out}}$, $\boldsymbol{d}_{i,l}^{\text{out}}$, e_l^{out}	l^{th} outer layer zeroth order cut's parameters.
$F(\cdot)$	penalty function.
$F_\mu(\cdot)$	smooth approximation of $F(\cdot)$.
μ	smoothing parameter.
$F_\mu{}^*$	optimal objective value of $F_\mu(\cdot)$.
λ_l, ϕ_j	penalty parameters.
$\phi_{\text{in}}(\cdot), \phi_{\text{out}}(\cdot)$	functions used in third level and second level constraint.
$G_{\boldsymbol{x}_{i,j}}, \forall i = 1,2,3, j = 1,\cdots,N$	gradient estimator for i^{th} level variable in worker j
$\eta_{\boldsymbol{x}_i}, \eta_{\boldsymbol{z}_i}, \forall i = 1,2,3$	step sizes for variables $\boldsymbol{x}_i$, $\boldsymbol{z}_i$.
$\boldsymbol{\mu}^{\text{in}}, \boldsymbol{\mu}^{\text{out}}, \boldsymbol{u}_{k,1}, \boldsymbol{u}_{k,2}, \boldsymbol{u}_{k,3}$	standard Gaussian random vectors.
$\mathcal{G}^t$	stationarity gap.
$T(\epsilon)$	iteration complexity to achieve ϵ-stationary point.
T_1	parameter controls the trade-off between complexity and performance.
$\mathcal{T}$	zeroth order cuts will be updated every $\mathcal{T}$ iteration.
N	the number of workers in distributed systems.
L	parameter in L-smoothness.
$d_i, \forall i = 1,2,3$	the dimension of i^{th} level variable.

A. Proofs of Theorem 4.6 (Iteration Complexity)

In this section, the detailed proofs of Theorem 4.6, i.e., iteration complexity of the proposed DTZO, are offered. The iteration complexity refers to the number of iterations for the proposed algorithm to obtain the ϵ-stationary point (Jiao et al., 2023). According to Ghadimi & Lan (2013), the gradient of the smooth approximation of F, i.e., F_μ (which is given in Definition 4.3), is also Lipschitz continuous with constant L_μ ($0 < L_\mu \leq L$), thus, we have that when $t \geq T_1$,

$$
\begin{aligned}
&F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\}, \{\boldsymbol{z}_i^t\}) \\
\leq& F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) + \begin{bmatrix} \{\boldsymbol{x}_{1,j}^{t+1} - \boldsymbol{x}_{1,j}^t\} \\ \{\boldsymbol{x}_{2,j}^{t+1} - \boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^{t+1} - \boldsymbol{x}_{3,j}^t\} \end{bmatrix}^\top \begin{bmatrix} \{\nabla_{\boldsymbol{x}_{1,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\nabla_{\boldsymbol{x}_{2,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\nabla_{\boldsymbol{x}_{3,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \end{bmatrix} + \frac{L}{2} || \begin{bmatrix} \{\boldsymbol{x}_{1,j}^{t+1} - \boldsymbol{x}_{1,j}^t\} \\ \{\boldsymbol{x}_{2,j}^{t+1} - \boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^{t+1} - \boldsymbol{x}_{3,j}^t\} \end{bmatrix} ||^2 \\
=& F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\}) - \begin{bmatrix} \{\eta_{\boldsymbol{x}_1} G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\eta_{\boldsymbol{x}_2} G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\eta_{\boldsymbol{x}_3} G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \end{bmatrix}^T \begin{bmatrix} \{\nabla_{\boldsymbol{x}_{1,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\nabla_{\boldsymbol{x}_{2,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \\ \{\nabla_{\boldsymbol{x}_{3,j}} F_\mu(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})\} \end{bmatrix} \\
&+ \frac{L}{2} \sum_{i=1}^{3} \sum_{j=1}^{N} \eta_{\boldsymbol{x}_i}^2 ||G_{x_{i,j}}(\{\boldsymbol{x}_{i,j}^t\}, \{\boldsymbol{z}_i^t\})||^2.
\end{aligned} \tag{30}
$$

According to Assumption 4.5 (i.e., function F has L-Lipschitz continuous gradient) and combining it with Cauchy-Schwarz inequality, we have that,

$$
\begin{aligned}
&F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\}) \\
&\leq F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) + \begin{bmatrix} \boldsymbol{z}_1^{t+1}-\boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t+1}-\boldsymbol{z}_2^t \\ \boldsymbol{z}_3^{t+1}-\boldsymbol{z}_3^t \end{bmatrix}^T \begin{bmatrix} \nabla_{\boldsymbol{z}_1} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_2} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_3} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) \end{bmatrix} + \frac{L}{2}|| \begin{bmatrix} \boldsymbol{z}_1^{t+1}-\boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t+1}-\boldsymbol{z}_2^t \\ \boldsymbol{z}_3^{t+1}-\boldsymbol{z}_3^t \end{bmatrix} ||^2 \\
&= F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) + \begin{bmatrix} \boldsymbol{z}_1^{t+1}-\boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t+1}-\boldsymbol{z}_2^t \\ \boldsymbol{z}_3^{t+1}-\boldsymbol{z}_3^t \end{bmatrix}^T \begin{bmatrix} \nabla_{\boldsymbol{z}_1} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_2} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_3} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \end{bmatrix} \\
&+ \begin{bmatrix} \boldsymbol{z}_1^{t+1}-\boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t+1}-\boldsymbol{z}_2^t \\ \boldsymbol{z}_3^{t+1}-\boldsymbol{z}_3^t \end{bmatrix}^T \begin{bmatrix} \nabla_{\boldsymbol{z}_1} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \nabla_{\boldsymbol{z}_1} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_2} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \nabla_{\boldsymbol{z}_2} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \\ \nabla_{\boldsymbol{z}_3} F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \nabla_{\boldsymbol{z}_3} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) \end{bmatrix} + \frac{L}{2}|| \begin{bmatrix} \boldsymbol{z}_1^{t+1}-\boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t+1}-\boldsymbol{z}_2^t \\ \boldsymbol{z}_3^{t+1}-\boldsymbol{z}_3^t \end{bmatrix} ||^2 \\
&\leq F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{L\eta_{\boldsymbol{z}_i}^2}{2} - \tfrac{\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{L}{2}||\boldsymbol{x}_{i,j}^{t+1}-\boldsymbol{x}_{i,j}^{t}||^2.
\end{aligned} \tag{31}
$$

Combining Eq. (31) with the Eq. (3.5) in Ghadimi & Lan (2013), we have that,

$$
\begin{aligned}
&F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\}) - \tfrac{\mu^2 L(N+1)\sum_i d_i}{2} \\
&\leq F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\}) \\
&\leq F(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{L}{2}||\boldsymbol{x}_{i,j}^{t+1}-\boldsymbol{x}_{i,j}^{t}||^2 \\
&\leq F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t}\}) - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{L}{2}||\boldsymbol{x}_{i,j}^{t+1}-\boldsymbol{x}_{i,j}^{t}||^2 \\
&+ \tfrac{\mu^2 L(N+1)\sum_i d_i}{2}.
\end{aligned} \tag{32}
$$

Combining Eq. (30) with Eq. (32), we can obtain that,

$$
\begin{aligned}
&F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\}) \\
&\leq F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) - \begin{bmatrix} \{\eta_{\boldsymbol{x}_1} G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\eta_{\boldsymbol{x}_2} G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\eta_{\boldsymbol{x}_3} G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \end{bmatrix}^T \begin{bmatrix} \{\nabla_{\boldsymbol{x}_{1,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\nabla_{\boldsymbol{x}_{2,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\nabla_{\boldsymbol{x}_{3,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \end{bmatrix} \\
&+ \tfrac{L}{2}\sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}^2||G_{x_{i,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 \\
&+ \sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{L}{2}||\boldsymbol{x}_{i,j}^{t+1}-\boldsymbol{x}_{i,j}^{t}||^2 + \mu^2 L(N+1)\textstyle\sum_i d_i \\
&= F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\}) - \begin{bmatrix} \{\eta_{\boldsymbol{x}_1} G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\eta_{\boldsymbol{x}_2} G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\eta_{\boldsymbol{x}_3} G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \end{bmatrix}^T \begin{bmatrix} \{\nabla_{\boldsymbol{x}_{1,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\nabla_{\boldsymbol{x}_{2,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \\ \{\nabla_{\boldsymbol{x}_{3,j}} F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})\} \end{bmatrix} \\
&+ \sum_{i=1}^{3}\sum_{j=1}^{N} L\eta_{\boldsymbol{x}_i}^2||G_{x_{i,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i} F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 \\
&+ \mu^2 L(N+1)\textstyle\sum_i d_i.
\end{aligned} \tag{33}
$$

Taking expectation on the both sides of Eq. (33), we can obtain that,

$$\begin{aligned}
&\mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})]\\
&\leq \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})] - \sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}||\nabla_{\boldsymbol{x}_{i,j}}F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \mu^2 L(N+1)\textstyle\sum_i d_i\\
&+\sum_{i=1}^{3}\sum_{j=1}^{N} L\eta_{\boldsymbol{x}_i}^2\mathbb{E}[||G_{x_{i,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2] - \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2.
\end{aligned} \tag{34}$$

Combining the definition of $G_{\boldsymbol{x}_{1,j}}, G_{\boldsymbol{x}_{2,j}}, G_{\boldsymbol{x}_{3,j}}$ with the Eq. (3.12) in Ghadimi & Lan (2013), we have that,

$$E[||G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2] \leq 2(d_1+4)||\nabla_{\boldsymbol{x}_{1,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \frac{\mu^2L^2}{2}(d_1+6)^3, \tag{35}$$

$$E[||G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2] \leq 2(d_2+4)||\nabla_{\boldsymbol{x}_{2,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \frac{\mu^2L^2}{2}(d_2+6)^3, \tag{36}$$

$$E[||G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2] \leq 2(d_3+4)||\nabla_{\boldsymbol{x}_{3,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \frac{\mu^2L^2}{2}(d_3+6)^3. \tag{37}$$

By combining Eq. (34) with Eq. (35), (36), and (37), we can get that,

$$\begin{aligned}
&\mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})]\\
&\leq \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})] - \sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}||\nabla_{\boldsymbol{x}_{i,j}}F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \mu^2 L(N+1)\textstyle\sum_i d_i\\
&+\sum_{i=1}^{3}\sum_{j=1}^{N} L\eta_{\boldsymbol{x}_i}^2\left(2(d_i+4)||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \tfrac{\mu^2L^2}{2}(d_i+6)^3\right)\\
&-\sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2,
\end{aligned} \tag{38}$$

that is,

$$\begin{aligned}
&\sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}||\nabla_{\boldsymbol{x}_{i,j}}F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2\\
&\leq \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})] - \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})] + \mu^2 L(N+1)\textstyle\sum_i d_i\\
&+\sum_{i=1}^{3}\sum_{j=1}^{N} L\eta_{\boldsymbol{x}_i}^2\left(2(d_i+4)||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \tfrac{\mu^2L^2}{2}(d_i+6)^3\right).
\end{aligned} \tag{39}$$

Combining Eq. (39) with Eq. (3.8) in Ghadimi & Lan (2013), we can obtain that,

$$\begin{aligned}
&\sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}\left(\tfrac{1}{2}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 - \tfrac{\mu^2L^2}{4}(d_i+3)^3\right)\\
&+\sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2\\
&\leq \sum_{i=1}^{3}\sum_{j=1}^{N}\eta_{\boldsymbol{x}_i}||\nabla_{\boldsymbol{x}_{i,j}}F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i} - \tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2\\
&\leq \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})] - \mathbb{E}[F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})] + \mu^2 L(N+1)\textstyle\sum_i d_i\\
&+\sum_{i=1}^{3}\sum_{j=1}^{N} L\eta_{\boldsymbol{x}_i}^2\left(2(d_i+4)||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^{t}\},\{\boldsymbol{z}_i^{t}\})||^2 + \tfrac{\mu^2L^2}{2}(d_i+6)^3\right),
\end{aligned} \tag{40}$$

that is,

$$\begin{aligned}
&\sum_{i=1}^{3}\sum_{j=1}^{N}\left(\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2\right)||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2\\
&+\sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2\\
&\leq F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})+\sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{\eta_{\boldsymbol{x}_i}^2\mu^2L^3}{2}(d_i+6)^3\\
&+\sum_{i=1}^{3}\sum_{j=1}^{N}\tfrac{\mu^2L^2\eta_{\boldsymbol{x}_i}}{4}(d_i+3)^3+\mu^2L(N+1)\textstyle\sum_i d_i.
\end{aligned}\tag{41}$$

According to the setting of $\eta_{\boldsymbol{x}_i}, i=1,2,3$, i.e., $0<\eta_{\boldsymbol{x}_i}\leq\frac{1}{8L(d_i+4)}, i=1,2,3$, we have that,

$$\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2>0, i=1,2,3.\tag{42}$$

Likewise, according to the setting of $\eta_{\boldsymbol{z}_i}, i=1,2,3$, i.e., $0<\eta_{\boldsymbol{z}_i}\leq\frac{3}{2(L+1)}, i=1,2,3$, we have that,

$$\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2}>0, i=1,2,3.\tag{43}$$

Combining Eq. (41) with Eq. (42) and (43), we can obtain that,

$$\begin{aligned}
&\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2\\
&\leq\frac{\sum_{i=1}^{3}\sum_{j=1}^{N}\left(\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2\right)||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2}{\min\left\{\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}\\
&+\frac{\sum_{i=1}^{3}(\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2})||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2}{\min\left\{\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}\\
&\leq\frac{F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-F_\mu(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})+\sum_{i=1}^{3}\tfrac{\eta_{\boldsymbol{x}_i}^2\mu^2L^3N}{2}(d_i+6)^3}{\min\left\{\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}\\
&+\frac{+\sum_{i=1}^{3}\tfrac{\mu^2L^2\eta_{\boldsymbol{x}_i}N}{4}(d_i+3)^3+\mu^2L(N+1)\sum_i d_i}{\min\left\{\tfrac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\tfrac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}.
\end{aligned}\tag{44}$$

Summing up the inequality in Eq. (44) from $t=T_1$ to $t=T(\epsilon)-1$, we have that,

$$
\begin{aligned}
&\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{F_\mu(\{\boldsymbol{x}_{i,j}^{T_1}\},\{\boldsymbol{z}_i^{T_1}\})-F_\mu(\{\boldsymbol{x}_{i,j}^{T(\epsilon)}\},\{\boldsymbol{z}_i^{T(\epsilon)}\})}{\min\left\{\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}(T(\epsilon)-T_1)}\\
&+\frac{\sum_{i=1}^{3}\frac{\eta_{\boldsymbol{x}_i}^2\mu^2L^3N}{2}(d_i+6)^3+\sum_{i=1}^{3}\frac{\mu^2L^2\eta_{\boldsymbol{x}_i}N}{4}(d_i+3)^3+\mu^2L(N+1)\sum_i d_i}{\min\left\{\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}\\
&\leq\frac{\max_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*}{\min\left\{\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}(T(\epsilon)-T_1)}\\
&+\frac{\sum_{i=1}^{3}\frac{\eta_{\boldsymbol{x}_i}^2\mu^2L^3N}{2}(d_i+6)^3+\sum_{i=1}^{3}\frac{\mu^2L^2\eta_{\boldsymbol{x}_i}N}{4}(d_i+3)^3+\mu^2L(N+1)\sum_i d_i}{\min\left\{\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2,\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2},i=1,2,3\right\}}.
\end{aligned}
\tag{45}
$$

According to the setting of $\eta_{\boldsymbol{x}_i},\eta_{\boldsymbol{z}_i},i=1,2,3$, we can obtain that,

$$
\frac{\eta_{\boldsymbol{x}_i}}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}^2=\eta_{\boldsymbol{x}_i}\left(\frac{1}{2}-2L(d_i+4)\eta_{\boldsymbol{x}_i}\right)\geq\frac{\eta_{\boldsymbol{x}_i}}{4},i=1,2,3,
\tag{46}
$$

$$
\eta_{\boldsymbol{z}_i}-\frac{(L+1)\eta_{\boldsymbol{z}_i}^2}{2}=\eta_{\boldsymbol{z}_i}(1-\frac{(L+1)\eta_{\boldsymbol{z}_i}}{2})\geq\frac{\eta_{\boldsymbol{z}_i}}{4},i=1,2,3.
\tag{47}
$$

Thus, we have that,

$$
\begin{aligned}
&\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{4\left(\max_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)}{\min\left\{\eta_{\boldsymbol{x}_1},\eta_{\boldsymbol{x}_2},\eta_{\boldsymbol{x}_3},\eta_{\boldsymbol{z}_1},\eta_{\boldsymbol{z}_2},\eta_{\boldsymbol{z}_3}\right\}(T(\epsilon)-T_1)}\\
&+\frac{\sum_{i=1}^{3}2\eta_{\boldsymbol{x}_i}^2\mu^2L^3N(d_i+6)^3+\sum_{i=1}^{3}\mu^2L^2\eta_{\boldsymbol{x}_i}N(d_i+3)^3+4\mu^2L(N+1)\sum_i d_i}{\min\left\{\eta_{\boldsymbol{x}_1},\eta_{\boldsymbol{x}_2},\eta_{\boldsymbol{x}_3},\eta_{\boldsymbol{z}_1},\eta_{\boldsymbol{z}_2},\eta_{\boldsymbol{z}_3}\right\}}.
\end{aligned}
\tag{48}
$$

According to the setting that,

$$
\eta_{\boldsymbol{x}_i}=\eta_{\boldsymbol{z}_i}=\min\left\{\frac{1}{8L(d_1+4)},\frac{1}{8L(d_2+4)},\frac{1}{8L(d_3+4)},\frac{3}{2(L+1)},\frac{1}{\sqrt{T(\epsilon)-T_1}}\right\},i=1,2,3,
\tag{49}
$$

we have that,

$$
\begin{aligned}
&\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{4\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)}{\min\left\{\frac{1}{8L(d_1+4)},\frac{1}{8L(d_2+4)},\frac{1}{8L(d_3+4)},\frac{3}{2(L+1)},\frac{1}{\sqrt{T(\epsilon)-T_1}}\right\}(T(\epsilon)-T_1)}\\
&+\sum_{i=1}^{3}2\eta_{\boldsymbol{x}_i}\mu^2L^3N(d_i+6)^3+\sum_{i=1}^{3}\mu^2L^2N(d_i+3)^3\\
&+\sum_{i=1}^{3}4\mu^2L(N+1)d_i\frac{1}{\min\left\{\frac{1}{8L(d_1+4)},\frac{1}{8L(d_2+4)},\frac{1}{8L(d_3+4)},\frac{3}{2(L+1)},\frac{1}{\sqrt{T(\epsilon)-T_1}}\right\}}\\
&\leq\frac{4\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}\right)}{T(\epsilon)-T_1}\\
&+\frac{4\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)\sqrt{T(\epsilon)-T_1}}{T(\epsilon)-T_1}\\
&+\sum_{i=1}^{3}2\eta_{\boldsymbol{x}_i}\mu^2L^3N(d_i+6)^3+\sum_{i=1}^{3}\mu^2L^2N(d_i+3)^3\\
&+\sum_{i=1}^{3}4\mu^2L(N+1)d_i\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}+\sqrt{T(\epsilon)-T_1}\right).
\end{aligned}
\tag{50}
$$

Since $\eta_{\boldsymbol{x}_i}\leq\frac{1}{8L(d_i+4)}, i=1,2,3$, we can obtain that,

$$
\begin{aligned}
&\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{4\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}\right)}{T(\epsilon)-T_1}\\
&+\frac{4\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)\sqrt{T(\epsilon)-T_1}}{T(\epsilon)-T_1}+\frac{\mu^2L^2N}{4}\sum_{i=1}^{3}\frac{(d_i+6)^3}{d_i+4}+\mu^2L^2\sum_{i=1}^{3}(d_i+3)^3\\
&+\sum_{i=1}^{3}4\mu^2L(N+1)d_i\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}+\sqrt{T(\epsilon)-T_1}\right).
\end{aligned}
\tag{51}
$$

Because of $T(\epsilon)-T_1\geq 1$, we have that $\frac{1}{T(\epsilon)-T_1}\leq\frac{1}{\sqrt{T(\epsilon)-T_1}}$. Combining with the setting of μ, i.e., $\mu^2\leq\frac{1}{T(\epsilon)-T_1}$, we

can obtain that,

$$
\begin{aligned}
&\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{4\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)}{T(\epsilon)-T_1}\\
&+\frac{\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*}{\sqrt{T(\epsilon)-T_1}}+\frac{L^2}{4}\sum_{i=1}^{3}\frac{(d_i+6)^3}{d_i+4}\frac{1}{T(\epsilon)-T_1}+L^2\sum_{i=1}^{3}(d_i+3)^3\frac{1}{T(\epsilon)-T_1}\\
&+\sum_{i=1}^{3}\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}+\sqrt{T(\epsilon)-T_1}\right)\frac{4L(N+1)d_i}{T(\epsilon)-T_1}\\
&\leq\frac{4(1+\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\})\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)}{\sqrt{T(\epsilon)-T_1}}\\
&+\frac{L^2}{4}\sum_{i=1}^{3}\frac{(d_i+6)^3}{d_i+4}\frac{1}{\sqrt{T(\epsilon)-T_1}}+L^2\sum_{i=1}^{3}(d_i+3)^3\frac{1}{\sqrt{T(\epsilon)-T_1}}\\
&+\sum_{i=1}^{3}\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}+1\right)4L(N+1)d_i\frac{1}{\sqrt{T(\epsilon)-T_1}}.
\end{aligned}
\tag{52}
$$

Combining the definition of stationarity gap and ϵ-stationary point in Definition 4.1, 4.2 with Eq. (52), we have that,

$$
\begin{aligned}
&||\mathcal{G}^{T(\epsilon)}||^2\\
&=\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^{T(\epsilon)}\},\{\boldsymbol{z}_i^{T(\epsilon)}\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^{T(\epsilon)}\},\{\boldsymbol{z}_i^{T(\epsilon)}\})||^2\\
&\leq\frac{1}{T(\epsilon)-T_1}\sum_{t=T_1}^{T(\epsilon)-1}(\sum_{i=1}^{3}\sum_{j=1}^{N}||\nabla_{\boldsymbol{x}_{i,j}}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2+\sum_{i=1}^{3}||\nabla_{\boldsymbol{z}_i}F(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})||^2)\\
&\leq\frac{4(1+\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\})\left(\max\limits_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)}{\sqrt{T(\epsilon)-T_1}}\\
&+\frac{L^2}{4}\sum_{i=1}^{3}\frac{(d_i+6)^3}{d_i+4}\frac{1}{\sqrt{T(\epsilon)-T_1}}+L^2\sum_{i=1}^{3}(d_i+3)^3\frac{1}{\sqrt{T(\epsilon)-T_1}}\\
&+\sum_{i=1}^{3}\left(\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}+1\right)4L(N+1)d_i\frac{1}{\sqrt{T(\epsilon)-T_1}}.
\end{aligned}
\tag{53}
$$

Thus, we can conclude that, when

$$
T(\epsilon)\geq\left(\sum_{i=1}^{3}\overline{c_i}+\overline{d}\left(\max_{t\in[T_1]}F_\mu(\{\boldsymbol{x}_{i,j}^t\},\{\boldsymbol{z}_i^t\})-{F_\mu}^*\right)\right)^2\frac{1}{\epsilon^2}+T_1\ ,
\tag{54}
$$

we have that $||\mathcal{G}^{T(\epsilon)}||^2\leq\epsilon$, where constants

$$
\overline{d}=4(1+\max\left\{8L(d_1+4),8L(d_2+4),8L(d_3+4),\frac{2(L+1)}{3}\right\}),
\tag{55}
$$

$$\begin{aligned}\overline{c_i} = &\tfrac{L^2(d_i+6)^3}{4(d_i+4)} + L^2(d_i+3)^3 \\ &+4L(N+1)d_i\left(\max\left\{8L(d_1+4), 8L(d_2+4), 8L(d_3+4), \tfrac{2(L+1)}{3}\right\}+1\right).\end{aligned} \tag{56}$$

B. Proofs of Theorem 4.7 (Communication Complexity)

The overall communication complexity of the proposed DTZO can be divided into 1) the communication complexity at every communication round and 2) the communication complexity of updating zeroth order cuts, which is discussed as follows.

1) The communication complexity at each iteration.

At each iteration, e.g., $(t+1)^{\text{th}}$ iteration, the workers transmit the updated variables $\boldsymbol{x}_{1,j}^{t+1}, \boldsymbol{x}_{2,j}^{t+1}, \boldsymbol{x}_{3,j}^{t+1}$ to the master, resulting in a communication complexity of $\sum_{j=1}^{N}\sum_{i=1}^{3} d_i$. Upon receiving these updated local variables, the master proceeds to update the global variables. Then, the master broadcasts the updated variables $\boldsymbol{z}_1^{t+1}, \boldsymbol{z}_2^{t+1}, \boldsymbol{z}_3^{t+1}$ and gradients $\nabla_{\boldsymbol{x}_{i,j}} o(\{\boldsymbol{x}_{2,j}^{t+1}\}, \{\boldsymbol{x}_{3,j}^{t+1}\}, \boldsymbol{z}_1^{t+1}, \boldsymbol{z}_2^{t+1}, \boldsymbol{z}_3^{t+1}), i = 2, 3$ to worker j. Therefore, the cumulative communication complexity from $t = 1$ to $t = T(\epsilon)$ is

$$C_1 = T(\epsilon)(2d_1 + 3d_2 + 3d_3)N. \tag{57}$$

2) The communication complexity of updating zeroth order cuts.

During every iteration $\mathcal{T}$ ($t < T_1$), the cutting planes are updated to refine the cascaded polynomial approximation, involving two main steps:

2a) Updating the inner layer polynomial approximation: In this phase, local variables $\boldsymbol{x}_{3,j}^{k+1}$ are transmitted from worker j, while global variables $\boldsymbol{z}_3^{k+1}$ are sent from the master in the $(k+1)^{\text{th}}$ iteration. The communication complexity associated with updating the inner layer polynomial approximation can be expressed as follows:

$$\sum\nolimits_{j=1}^{N} 2\lfloor\frac{T_1}{\mathcal{T}}\rfloor\mathcal{T}Kd_3. \tag{58}$$

2b) Updating the outer layer polynomial approximation: During the $(k+1)^{\text{th}}$ iteration when updating the outer layer approximation, the worker j transmits the updated variables $\boldsymbol{x}_{2,j}^{k+1}$, to the master. Subsequently, the master broadcasts the updated global variables $\boldsymbol{z}_2^{k+1}$ to worker j. The communication complexity involved in this process can be expressed as,

$$\sum\nolimits_{j=1}^{N} 2\lfloor\frac{T_1}{\mathcal{T}}\rfloor\mathcal{T}Kd_2. \tag{59}$$

Combining Eq. (58) with Eq. (59), and considering utilizing one communication round to approximate the $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ and $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$, i.e., $K = 1$, we have that the communication complexity of updating cascaded polynomial approximation is,

$$C_2 = 2N\lfloor\frac{T_1}{\mathcal{T}}\rfloor\mathcal{T}(d_2 + d_3). \tag{60}$$

Consequently, the overall communication of the proposed method is $C_1 + C_2$, which can be expressed as,

$$3T(\epsilon)(d_1 + d_2 + d_3)N + 2N\lfloor\frac{T_1}{\mathcal{T}}\rfloor\mathcal{T}(d_2 + d_3). \tag{61}$$

C. Proofs of Proposition 3.1 and 3.2

C.1. Proofs of Proposition 3.1

For any point $(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)$ in the original feasible region, i.e., $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = 0$, according to the properties of L-smoothness, we have that,

$$
\begin{aligned}
&\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) \\
&\geq \phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3) + \tfrac{\partial \phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)}{\partial(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)}^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) \\
&- \tfrac{L}{2} || \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) ||^2 \\
&= \phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3) + G^{\text{in}}_\mu(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) \\
&+ \left(\tfrac{\partial \phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)}{\partial(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)} - G^{\text{in}}_\mu(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3) \right)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) \\
&- \tfrac{L}{2} || \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) ||^2.
\end{aligned} \tag{62}
$$

According to $\mathbb{E}[G^{\text{in}}_\mu(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)] = \phi_{\mu,\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)$, taking expectation on both sides of Eq. (62), we have that,

$$
\begin{aligned}
&\mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)] \\
&\geq \mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)] + \mathbb{E}[G^{\text{in}}_\mu(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)]^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) \\
&+ \left(\tfrac{\partial \phi_{\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3)}{\partial(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)} - \phi_{\mu,\text{in}}(\{\boldsymbol{x}^t_{3,j}\}, \boldsymbol{z}^t_1, \boldsymbol{z}^{t\prime}_2, \boldsymbol{z}^t_3) \right)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) \\
&- \tfrac{L}{2} || \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}^t_{3,j}\} \\ \boldsymbol{z}^t_1 \\ \boldsymbol{z}^{t\prime}_2 \\ \boldsymbol{z}^t_3 \end{bmatrix} \right) ||^2.
\end{aligned} \tag{63}
$$

Combining with the Cauchy-Schwarz inequality, we have that,

$$\begin{aligned}
&\mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)] \\
&\geq \mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)] + \mathbb{E}[G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)]^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&-\tfrac{1}{2}\| \tfrac{\partial \phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)} - \phi_{\mu,\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)\|^2 - \tfrac{L+1}{2}\| \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \|^2.
\end{aligned} \tag{64}$$

And according to Eq. (3.6) in Ghadimi & Lan (2013), we can obtain that,

$$\|\phi_{\mu,\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t) - \frac{\partial \phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)}\|^2 \leq \frac{\mu^2}{4} L^2 (d_1 + d_2 + (N+1)d_3 + 3)^3. \tag{65}$$

By combining Eq. (64) with Eq. (65), we have that,

$$\begin{aligned}
&\mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)] \\
&\geq \mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)] + \mathbb{E}[G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)]^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&-\tfrac{\mu^2}{8} L^2 (d_1 + d_2 + (N+1)d_3 + 3)^3 - \tfrac{L+1}{2}\| \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \|^2.
\end{aligned} \tag{66}$$

For any point belongs to the original feasible region, i.e., $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = 0$, according to $\varepsilon_{\text{in}} \geq 0$, we can obtain that it also satisfies that,

$$\begin{aligned}
&\mathbb{E}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t) + G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right)] \\
&\leq \tfrac{L+1}{2}\| \left(\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2{}' \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^{t\prime} \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \|^2 + \tfrac{\mu^2}{8} L^2 (d_1 + d_2 + (N+1)d_3 + 3)^3 + \varepsilon_{\text{in}}.
\end{aligned} \tag{67}$$

According to Eq. (9), we can conclude that for any point belongs to the original feasible region of constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = 0$, it also belongs to the P_{in}^t, that is, the original feasible region is a subset of the feasible region formed by inner layer zeroth order cuts. Let S_{in} denote the original feasible region of constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = 0$, we can obtain that the feasible region formed by inner layer zeroth order cuts will be gradually tightened with zeroth order cuts added according to Eq. (67), that is,

$$S_{\text{in}} \subseteq P_{\text{in}}^{t+1} \subseteq P_{\text{in}}^t \subseteq \cdots \subseteq P_{\text{in}}^0. \tag{68}$$

C.2. Proofs of Proposition 3.2

For any point $(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ in the original feasible region, i.e., $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$, according to the properties of L-smoothness, we have that,

$$
\begin{aligned}
&\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \\
&\geq \phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) + \tfrac{\partial \phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)}^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&-\tfrac{L}{2}|| \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) ||^2 \\
&= \phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) + G_\mu^{\text{out}}(t)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&+\left(\tfrac{\partial \phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)} - G_\mu^{\text{out}}(t) \right)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&-\tfrac{L}{2}|| \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) ||^2,
\end{aligned}
\tag{69}
$$

where $G_{\mu}^{\text{out}}(t)$ is the simplified form of $G_{\mu}^{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)$. According to $\mathbb{E}[G_{\mu}^{\text{out}}(t)] = \phi_{\mu,\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)$, taking expectation on both sides of Eq. (69), we have that,

$$
\begin{aligned}
&\mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)]\\
&\geq \mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)] + \mathbb{E}[G_{\mu}^{\text{out}}(t)]^{\top}\left(\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^t\}\\ \{\boldsymbol{x}_{3,j}^t\}\\ \boldsymbol{z}_1^t\\ \boldsymbol{z}_2^t\\ \boldsymbol{z}_3^t\end{bmatrix}\right)\\
&+\left(\frac{\partial\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)} - \phi_{\mu,\text{out}}(t)\right)^{\top}\left(\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^t\}\\ \{\boldsymbol{x}_{3,j}^t\}\\ \boldsymbol{z}_1^t\\ \boldsymbol{z}_2^t\\ \boldsymbol{z}_3^t\end{bmatrix}\right)\\
&-\frac{L}{2}\left|\left|\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^t\}\\ \{\boldsymbol{x}_{3,j}^t\}\\ \boldsymbol{z}_1^t\\ \boldsymbol{z}_2^t\\ \boldsymbol{z}_3^t\end{bmatrix}\right|\right|^2,
\end{aligned}
\tag{70}
$$

where $\phi_{\mu,\text{out}}(t)$ is the simplified form of $\phi_{\mu,\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)$. Combining with the Cauchy-Schwarz inequality, we have that,

$$
\begin{aligned}
&\mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)]\\
&\geq \mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)] + \mathbb{E}[G_{\mu}^{\text{out}}(t)]^{\top}\left(\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^t\}\\ \{\boldsymbol{x}_{3,j}^t\}\\ \boldsymbol{z}_1^t\\ \boldsymbol{z}_2^t\\ \boldsymbol{z}_3^t\end{bmatrix}\right)\\
&-\frac{1}{2}\left|\left|\frac{\partial\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)} - \phi_{\mu,\text{out}}(t)\right|\right|^2\\
&-\frac{L+1}{2}\left|\left|\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^t\}\\ \{\boldsymbol{x}_{3,j}^t\}\\ \boldsymbol{z}_1^t\\ \boldsymbol{z}_2^t\\ \boldsymbol{z}_3^t\end{bmatrix}\right|\right|^2.
\end{aligned}
\tag{71}
$$

And according to Eq. (3.6) in Ghadimi & Lan (2013), we can obtain that,

$$
\left|\left|\phi_{\mu,\text{out}}(t) - \frac{\partial\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)}{\partial(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)}\right|\right|^2 \leq \frac{\mu^2}{4}L^2(d_1+(N+1)(d_2+d_3)+3)^3. \tag{72}
$$

By combining Eq. (71) with Eq. (72), we have that,

$$\begin{aligned}
&\mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)] \\
&\geq \mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)] + \mathbb{E}[G_\mu^{\text{out}}(t)]^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) \\
&-\tfrac{\mu^2}{8}L^2(d_1+(N+1)(d_2+d_3)+3)^3 - \tfrac{L+1}{2}|| \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right) ||^2.
\end{aligned} \tag{73}$$

For any point belongs to the original feasible region, i.e., $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)=0$, according to $\varepsilon_{\text{in}} \geq 0$, we can obtain that it also satisfies that,

$$\begin{aligned}
&\mathbb{E}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)+G_\mu^{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t,\boldsymbol{z}_2^t,\boldsymbol{z}_3^t)^\top \left(\begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \boldsymbol{z}_3 \end{bmatrix} - \begin{bmatrix} \{\boldsymbol{x}_{2,j}^t\} \\ \{\boldsymbol{x}_{3,j}^t\} \\ \boldsymbol{z}_1^t \\ \boldsymbol{z}_2^t \\ \boldsymbol{z}_3^t \end{bmatrix} \right)] \\
&\leq \tfrac{L+1}{2}\left(\textstyle\sum_{i=2}^3\sum_j||\boldsymbol{x}_{i,j}-\boldsymbol{x}_{i,j}^t||^2+\sum_i||\boldsymbol{z}_i-\boldsymbol{z}_i^t||^2\right)+\tfrac{\mu^2}{8}L^2(d_1+(N+1)(d_2+d_3)+3)^3+\varepsilon_{\text{out}}.
\end{aligned} \tag{74}$$

According to Eq. (11), we can conclude that for any point belongs to the original feasible region of constraint $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)=0$, it also belongs to the P_{out}^t, that is, the original feasible region is a subset of the feasible region formed by outer layer zeroth order cuts. In addition, let S_{out} denote the original feasible region of constraint $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)=0$, based on Eq. (74), we can obtain that the feasible region formed by outer layer zeroth order cuts will be gradually tightened with zeroth order cuts added, that is,

$$S_{\text{out}} \subseteq P_{\text{out}}^{t+1} \subseteq P_{\text{out}}^t \subseteq \cdots \subseteq P_{\text{out}}^0. \tag{75}$$

D. Theoretical Analyses about the Cascaded Polynomial Approximation Problem

In this section, we theoretically analyze the connections between the original distributed trilevel zeroth order optimization problem in Eq. (2) and the cascaded polynomial approximation problem in Eq. (8). To facilitate this discussion, we start by examining the distributed bilevel zeroth order optimization problem, which can be expressed as follows,

$$\begin{aligned}
&\min \sum_{j=1}^N f_{1,j}(\boldsymbol{x}_1,\boldsymbol{x}_2) \\
&\text{s.t. } \boldsymbol{x}_2 = \arg\min_{\boldsymbol{x}_2'} \sum_{j=1}^N f_{2,j}(\boldsymbol{x}_1,\boldsymbol{x}_2') \\
&\text{var.} \qquad \boldsymbol{x}_1,\boldsymbol{x}_2.
\end{aligned} \tag{76}$$

The optimization problem in Eq. (76) can be equivalently reformulated as,

$$\begin{aligned}
&\min \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}) \\
&\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\
&\quad \{\boldsymbol{x}_{2,j}\}, \boldsymbol{z}_2 = \underset{\{\boldsymbol{x}_{2,j}'\}, \boldsymbol{z}_2'}{\arg\min} \sum_{j=1}^{N} f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}') \\
&\qquad\qquad \text{s.t. } \boldsymbol{x}_{2,j}' = \boldsymbol{z}_2', \forall j = 1, \cdots, N \\
&\text{var.} \qquad\quad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2.
\end{aligned} \tag{77}$$

By utilizing the proposed polynomial approximation with zeroth order cut, we can obtain the following zeroth order polynomial approximation problem,

$$\begin{aligned}
&\min \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}) \\
&\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\
&\sum_{j=1}^{N} {\boldsymbol{a}_{2,j,l}}^{\top} \boldsymbol{x}_{2,j}^2 + {\boldsymbol{b}_{2,j,l}}^{\top} \boldsymbol{x}_{2,j} + \sum_{i=1}^{2} {\boldsymbol{c}_{i,l}}^{\top} \boldsymbol{z}_i^2 + {\boldsymbol{d}_{i,l}}^{\top} \boldsymbol{z}_i + e_l \leq \varepsilon, \forall l \\
&\text{var.} \qquad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2.
\end{aligned} \tag{78}$$

According to Proposition 3.1 and 3.2, we can obtain the feasible region of the problem in Eq. (77) is a subset of the feasible region of the problem in Eq. (78). Thus, we can conclude that the zeroth order polynomial approximation optimization problem in Eq. (78) is the relaxed problem of the distributed bilevel zeroth order optimization problem in Eq. (76).

For the distributed trilevel zeroth order optimization problem, we first define the following feasible regions.

$$S_1 = \left\{ \{\boldsymbol{x}_{i,j}\}, \{\boldsymbol{z}_i\} \middle| \begin{array}{l} h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \leq \varepsilon_{\text{out}}, \forall l, \\ \boldsymbol{z}_1 = \boldsymbol{x}_{1,j}, \forall j \end{array} \right\}, \tag{79}$$

$$S_2 = \left\{ \{\boldsymbol{x}_{i,j}\}, \{\boldsymbol{z}_i\} \middle| \begin{array}{l} \left|\left| \begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \boldsymbol{z}_2 \end{bmatrix} - \begin{array}{l} \underset{\{\boldsymbol{x}_{2,j}'\}, \boldsymbol{z}_2'}{\arg\min} \sum_{j=1}^{N} f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}', \boldsymbol{x}_{3,j}) \\ \text{s.t.} \quad \boldsymbol{x}_{2,j}' = \boldsymbol{z}_2', \forall j, \\ \qquad h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{z}_3) \leq \varepsilon_{\text{in}}, \forall l \end{array} \right|\right|^2 \leq \varepsilon_{\text{out}}, \\ \boldsymbol{z}_1 = \boldsymbol{x}_{1,j}, \forall j \end{array} \right\}, \tag{80}$$

$$S_3 = \left\{ \{\boldsymbol{x}_{i,j}\}, \{\boldsymbol{z}_i\} \middle| \begin{array}{l} \left|\left| \begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \boldsymbol{z}_2 \end{bmatrix} - \begin{array}{l} \underset{\{\boldsymbol{x}_{2,j}'\}, \boldsymbol{z}_2'}{\arg\min} \sum_{j=1}^{N} f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) \\ \text{s.t. } \boldsymbol{x}_{2,j}' = \boldsymbol{z}_2', \forall j = 1, \cdots, N \\ \quad \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_3 = \underset{\{\boldsymbol{x}_{3,j}'\}, \boldsymbol{z}_3'}{\arg\min} \sum_{j=1}^{N} f_{3,j}(\boldsymbol{z}_1, \boldsymbol{z}_2', \boldsymbol{x}_{3,j}') \\ \qquad\qquad \text{s.t. } \boldsymbol{x}_{3,j}' = \boldsymbol{z}_3', \forall j = 1, \cdots, N \end{array} \right|\right|^2 = 0, \\ \boldsymbol{z}_1 = \boldsymbol{x}_{1,j}, \forall j \end{array} \right\}. \tag{81}$$

It is seen from Eq. (79) and Eq. (81) that S_1 and S_3 respectively represent the feasible region of optimization problems in

Eq. (8) and Eq. (3). For any feasible solution $\{\hat{\boldsymbol{x}}_{i,j}\}$,$\{\hat{\boldsymbol{z}}_i\}$ of optimization problem in Eq. (3), it satisfies that,

$$||\begin{bmatrix}\{\hat{\boldsymbol{x}}_{2,j}\}\\ \hat{\boldsymbol{z}}_2\end{bmatrix}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\ \{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min}\sum_{j=1}^{N} f_{3,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\ \text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\end{array}||^2=0. \tag{82}$$

Based on Proposition 3.1, we have that the feasible region of constraint $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2{}',\boldsymbol{z}_3)=0$ is a subset of the feasible region formed by inner layer zeroth order cuts, i.e., $\big\{\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2{}',\boldsymbol{z}_3|\, h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2{}',\boldsymbol{z}_3)\leq\varepsilon_{\text{in}},\forall l\big\}$. Moreover, the feasible region formed by inner layer zeroth order cuts will be continuously tightened with zeroth order cuts added. Thus, let $\beta\geq 0$ satisfy that,

$$\begin{aligned}&||\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t.}\quad \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j,\\ \quad h_l^{\text{in}}(\{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\hat{\boldsymbol{z}}_3)\leq\varepsilon_{\text{in}},\forall l\end{array}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\ \{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min}\sum_{j=1}^{N} f_{3,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\ \text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\end{array}||^2\\ &\leq\beta.\end{aligned} \tag{83}$$

By combining Proposition 3.1 with Eq. (83), we can obtain that β will continuously decrease with inner layer zeroth order cuts added. By combining Eq. (82) with Cauchy-Schwarz inequality, we can obtain that,

$$\begin{aligned}&||\begin{bmatrix}\{\hat{\boldsymbol{x}}_{2,j}\}\\ \hat{\boldsymbol{z}}_2\end{bmatrix}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t.}\quad \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j,\\ \quad h_l^{\text{in}}(\{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\hat{\boldsymbol{z}}_3)\leq\varepsilon_{\text{in}},\forall l\end{array}||^2\\
&=||\begin{bmatrix}\{\hat{\boldsymbol{x}}_{2,j}\}\\ \hat{\boldsymbol{z}}_2\end{bmatrix}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\ \{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min}\sum_{j=1}^{N} f_{3,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\ \text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\end{array}\\
&+\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\ \{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min}\sum_{j=1}^{N} f_{3,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\ \text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\end{array}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t.}\quad \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j,\\ \quad h_l^{\text{in}}(\{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\hat{\boldsymbol{z}}_3)\leq\varepsilon_{\text{in}},\forall l\end{array}||^2\\
&\leq 2||\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t.}\quad \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j,\\ \quad h_l^{\text{in}}(\{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\hat{\boldsymbol{z}}_3)\leq\varepsilon_{\text{in}},\forall l\end{array}-\begin{array}{l}\underset{\{\boldsymbol{x}_{2,j}{}'\},\boldsymbol{z}_2{}'}{\arg\min}\sum_{j=1}^{N} f_{2,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{x}_{2,j}{}',\hat{\boldsymbol{x}}_{3,j})\\ \text{s.t. } \boldsymbol{x}_{2,j}{}'=\boldsymbol{z}_2{}',\forall j=1,\cdots,N\\ \{\hat{\boldsymbol{x}}_{3,j}\},\hat{\boldsymbol{z}}_3=\underset{\{\boldsymbol{x}_{3,j}{}'\},\boldsymbol{z}_3{}'}{\arg\min}\sum_{j=1}^{N} f_{3,j}(\hat{\boldsymbol{z}}_1,\boldsymbol{z}_2{}',\boldsymbol{x}_{3,j}{}')\\ \text{s.t. } \boldsymbol{x}_{3,j}{}'=\boldsymbol{z}_3{}',\forall j=1,\cdots,N\end{array}||^2\\
&\leq 2\beta.\end{aligned} \tag{84}$$

By combining the definition of S_2 in Eq. (81) with Eq. (84), we can get that S_3 is a subset of S_2, i.e., $S_3 \in S_2$ when we set $\varepsilon_{\text{in}} \geq 0$ and $\varepsilon_{\text{out}} \geq 2\beta$. Based on Proposition 3.2, we have that S_2 is a subset of S_1, i.e., $S_2 \in S_1$. Consequently, we can get $S_3 \in S_1$, indicating that the cascaded polynomial approximation problem is the relaxed problem of the original distributed trilevel zeroth order optimization problem. Moreover, this relaxation will be gradually tightened with the addition of zeroth order cuts based on Proposition 3.1 and 3.2.

E. Discussions about Soft Constraint and ϕ_{in}, ϕ_{out}

Soft constraint. A *soft constraint* refers to a constraint that can be partially violated without rendering the optimization problem meaningless (Kautz et al., 1996; Régin, 2011; Wilson et al., 2022). It is shown in many bilevel and trilevel learning works that the lower-optimization problem often serves as a soft constraint to the upper-level optimization problem. Examples are provided as follows.

* In bilevel neural architecture search (Liu et al., 2018a), rather than computing the optimal solution for the lower-level optimization problem, the result obtained after a single gradient descent step can be used as an approximation of the optimal solution at each iteration.
* In bilevel meta-learning (Ji et al., 2021; Finn et al., 2017), instead of solving the lower-level optimization problem to optimality, the results obtained after multiple gradient descent steps can serve as an approximation at each iteration.
* In bilevel adversarial learning (Madry et al., 2018; Zhang et al., 2022), which is a min-max optimization problem, instead of solving the maximization problem to obtain the optimal solution, the results after several projected gradient descent steps are used as the approximation at each iteration.
* In trilevel learning, AFTO (Jiao et al., 2024) used the results after K communication rounds to replace the optimal solution to the lower-level optimization problem at each iteration in federated trilevel optimization problems.
* In trilevel learning for masked autoencoder (Guo et al., 2024), instead of obtaining the optimal solution to lower-level optimization problems at each iteration, Guo et al. (2024) used the results after several iterations of gradient descent updates as the approximation.
* In trilevel learning for enhancing out-of-domain generation in machine translation, He et al. (2024) used the results after one-step gradient descent update as the approximation for the optimal solution to the lower-level optimization problem at each iteration.

It is seen from $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3) = \| \begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_3 \end{bmatrix} - \underset{\{\boldsymbol{x}_{3,j}{}'\}, \boldsymbol{z}_3{}'}{\arg\min} \sum_j f_{3,j}(\boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{x}_{3,j}{}') \text{ s.t. } \boldsymbol{x}_{3,j}{}' = \boldsymbol{z}_3{}', \forall j \|^2$ that a distributed optimization problem needs to be solved if an exact $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ is required. The lower-level optimization problem (i.e., $\begin{bmatrix} \{\boldsymbol{x}_{3,j}\} \\ \boldsymbol{z}_3 \end{bmatrix} = \underset{\{\boldsymbol{x}_{3,j}{}'\}, \boldsymbol{z}_3{}'}{\arg\min} \sum_j f_{3,j}(\boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{x}_{3,j}{}') \text{ s.t. } \boldsymbol{x}_{3,j}{}' = \boldsymbol{z}_3{}', \forall j$) can be regarded as a soft constraint to the upper-level optimization problem. Inspired by many works in bilevel optimization and trilevel optimization, e.g. Ji et al. (2021); Jiao et al. (2022a); Yang et al. (2021); Franceschi et al. (2018); Liu et al. (2021b); Mackay et al. (2018); Choe et al. (2023), that utilize K steps gradient descent steps to approximate the optimal solution to the lower-level optimization problem, function $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2{}', \boldsymbol{z}_3)$ in this work can also be approximated based on the solution after K communication rounds following Jiao et al. (2024). Specifically, we have the following steps in $(k+1)^{\text{th}}$ iteration,

Local worker j updates the local variables as,

$$\boldsymbol{x}_{3,j}^{k+1} = \boldsymbol{x}_{3,j}^{k} - \eta_x G_{\text{in},j}(\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{x}_{3,j}^{k}, \boldsymbol{z}_3^{k}), \tag{85}$$

where η_x denotes the step-size, and

$$G_{\text{in},j}(\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{x}_{3,j}^{k}, \boldsymbol{z}_3^{k}) = \frac{f_{3,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}^{k} + \mu \boldsymbol{u}_{k,3}) - f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}^{k})}{\mu} \boldsymbol{u}_{k,3} + 2\gamma_j(\boldsymbol{x}_{3,j}^{k} - \boldsymbol{z}_3^{k}). \tag{86}$$

where $\boldsymbol{u}_{k,3}$ is a standard Gaussian random vector, $\gamma_j > 0$ is a constant. Then, workers transmit the updated local variables, i.e., $\boldsymbol{x}_{3,j}^{k+1}$, to the master.

After receiving the updated variables, the master updates the consensus variables as follows.

$$\boldsymbol{z}_3^{k+1} = \boldsymbol{z}_3^k - \eta_z \sum\nolimits_{j=1}^{N} \gamma_j(\boldsymbol{z}_3^k - \boldsymbol{x}_{3,j}^{k+1}), \tag{87}$$

where η_z represents the step-size. Subsequently, the master broadcasts the updated variables $\boldsymbol{z}_3^{k+1}$ to workers. Thus, the approximated $\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ can be expressed as,

$$\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = \begin{bmatrix} \{\boldsymbol{x}_{3,j} - \boldsymbol{x}_{3,j}^0 + \eta_x \sum_{k=0}^{K-1} G_{\text{in},j}(\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{x}_{3,j}^k, \boldsymbol{z}_3^k)\} \\ \boldsymbol{z}_3 - \boldsymbol{z}_3^0 + \eta_z \sum_{k=0}^{K-1} \sum_{j=1}^{N} \gamma_j(\boldsymbol{z}_3^k - \boldsymbol{x}_{3,j}^{k+1}) \end{bmatrix}. \tag{88}$$

Likewise, constraint $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = 0$ also serves as a soft constraint to the upper-level optimization problem. According to the definition of $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$, that is,

$$\begin{aligned} &\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \\ &= || \begin{bmatrix} \{\boldsymbol{x}_{2,j}\} \\ \boldsymbol{z}_2 \end{bmatrix} - \begin{array}{l} \arg\min \sum_{j=1}^{N} f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) \\ \{\boldsymbol{x}_{2,j}\}, \boldsymbol{z}_2 \\ \text{s.t. } \boldsymbol{x}_{2,j} = \boldsymbol{z}_2, \forall j, h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \leq \varepsilon_{\text{in}}, \forall l \end{array} ||^2, \end{aligned} \tag{89}$$

the results after K communication rounds can also be utilized to compute the estimate of $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ following previous works (Liu et al., 2018a; Jiao et al., 2024). In $(k+1)^{\text{th}}$ iteration, we have that,

Local worker j updates the local variables as follows,

$$\boldsymbol{x}_{2,j}^{k+1} = \boldsymbol{x}_{2,j}^k - \eta_x G_{\boldsymbol{x}_{2,j}}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}^k, \boldsymbol{x}_{3,j}, \boldsymbol{z}_2^k, \boldsymbol{z}_3), \tag{90}$$

where we have,

$$\begin{aligned} &G_{\boldsymbol{x}_{2,j}}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}^k, \boldsymbol{x}_{3,j}, \boldsymbol{z}_2^k, \boldsymbol{z}_3) \\ &= \frac{f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}^k + \mu \boldsymbol{u}_{k,2}, \boldsymbol{x}_{3,j}) - f_{2,j}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}^k, \boldsymbol{x}_{3,j})}{\mu} \boldsymbol{u}_{k,2} + 2\varphi_j(\boldsymbol{x}_{2,j}^k - \boldsymbol{z}_2^k), \end{aligned} \tag{91}$$

where $\boldsymbol{u}_{k,2}$ is the standard Gaussian random vector, $\varphi_j > 0$ is a constant. Then, worker j transmits the updated $\boldsymbol{x}_{2,j}^{k+1}$ to the master.

After receiving the updated parameters from workers, the master updates the consensus variables as,

$$\boldsymbol{z}_2^{k+1} = \boldsymbol{z}_2^k - \eta_z \left(2\varphi_j(\boldsymbol{z}_2^k - \boldsymbol{x}_{2,j}^{k+1}) + \nabla_{\boldsymbol{z}_2} p_l \sum\nolimits_l [\max\{h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2^k, \boldsymbol{z}_3) - \varepsilon_{\text{in}}, 0\}]^2 \right). \tag{92}$$

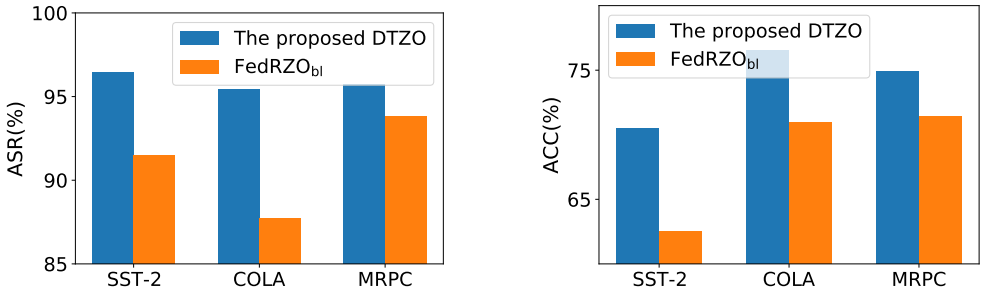
Next, the master broadcasts the updated variables $\boldsymbol{z}_2^{k+1}$ to workers. Consequently, the approximated $\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3)$ can be written as,

$$\begin{aligned} &\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \\ &= \begin{bmatrix} \{\boldsymbol{x}_{2,j} - \boldsymbol{x}_{2,j}^0 + \sum_{k=0}^{K-1} \eta_x G_{\boldsymbol{x}_{2,j}}(\boldsymbol{z}_1, \boldsymbol{x}_{2,j}^k, \boldsymbol{x}_{3,j}, \boldsymbol{z}_2^k, \boldsymbol{z}_3)\} \\ \boldsymbol{z}_2 - \boldsymbol{z}_2^0 + \sum_{k=0}^{K-1} \eta_z \left(2\varphi_j(\boldsymbol{z}_2^k - \boldsymbol{x}_{2,j}^{k+1}) + \nabla_{\boldsymbol{z}_2} p_l \sum_l [\max\{h_l^{\text{in}}(\{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2^k, \boldsymbol{z}_3) - \varepsilon_{\text{in}}, 0\}]^2\right) \end{bmatrix}. \end{aligned} \tag{93}$$

F. Experimental Setting and Detailed Results

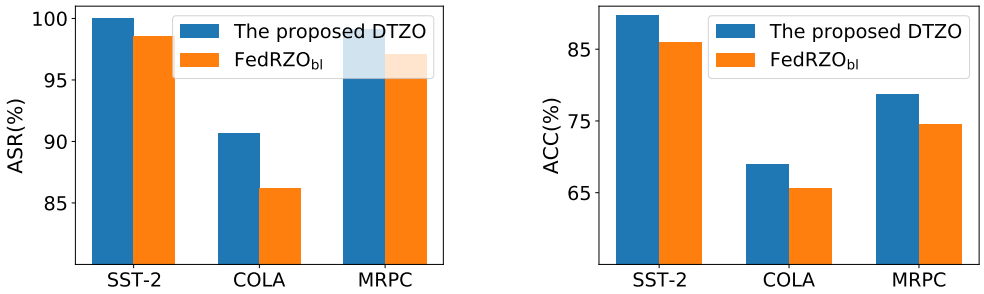
In this section, we provide the details of the experimental setting. In the experiment, all the models are implemented using PyTorch, and the experiments are conducted on a server equipped with two NVIDIA RTX 4090 GPUs.

In the experiment, we compare the proposed method with the state-of-the-art distributed zeroth order learning method FedZOO (Fang et al., 2022) and state-of-the-art distributed bilevel zeroth order learning method FedRZO$_{\text{bl}}$ (Qiu et al., 2023), which are introduced as follows. FedZOO (Fang et al., 2022) is a derivative-free federated zeroth-order optimization method, which can be applied to solve the single-level optimization problems in a distributed manner. In FedZOO, clients perform several local updates based on gradient estimators in each communication round. After receiving local updates, the servers will perform the aggregation and update the global parameters. FedRZO$_{\text{bl}}$ (Qiu et al., 2023) is designed for zeroth order bilevel optimization problems. In each communication round, FedRZO$_{\text{bl}}$ involves the following steps: clients first compute the estimated optimal solution to the lower-level optimization problem and the inexact implicit zeroth-order gradient. They then update the local parameters and transmit them to the server. Upon receiving the updates, the server aggregates them to obtain the global parameters.



(a) Comparisons about ASR (b) Comparisons about ACC

Figure 2. Comparisons about ASR and ACC between the proposed DTZO and state-of-the-art method using Qwen2-7B.



(a) Comparisons about ASR (b) Comparisons about ACC

Figure 3. Comparisons about ASR and ACC between the proposed DTZO and state-of-the-art method using Llama-3.1-8B.

F.1. Black-box Trilevel Learning

In this section, the details of the experimental setting in black-box trilevel learning are provided. Prompt learning is a key technique for enabling LLMs to efficiently and effectively adapt to various downstream tasks (Ma et al., 2024; Wang et al., 2024). Inspired by the black-box prompt learning (Diao et al., 2022) and backdoor attack on prompt-based LLMs (Yao et al., 2024), the backdoor attack on black-box LLMs is considered in the experiment, which can be expressed as a black-box trilevel learning problem as follows.

$$\begin{aligned}
&\min_{\lambda} \sum_{j=1}^{N} \frac{1}{|D_j^{\mathrm{val}}|} \sum_{(\boldsymbol{s}_i,y_i)\sim D_j^{\mathrm{val}}} L(\mathcal{G},[\boldsymbol{k}_{\mathrm{tri}},\boldsymbol{p},\boldsymbol{s}_i],y_i) \\
&\text{s.t. } \boldsymbol{k}_{\mathrm{tri}} = \arg\min_{\boldsymbol{k}_{\mathrm{tri}}'} \sum_{j=1}^{N} \frac{1}{|D_j^{\mathrm{tr}}|} \sum_{(\boldsymbol{s}_i,y_i)\sim D_j^{\mathrm{tr}}} L(\mathcal{G},[\boldsymbol{k}_{\mathrm{tri}}',\boldsymbol{p},\boldsymbol{s}_i],y_i) + \lambda||\boldsymbol{k}_{\mathrm{tri}}'||^2 \\
&\qquad \text{s.t. } \boldsymbol{p} = \arg\min_{\boldsymbol{p}'} \sum_{j=1}^{N} \frac{1}{|D_j^{\mathrm{tr}}|} \sum_{(\boldsymbol{s}_i,y_i)\sim D_j^{\mathrm{tr}}} L(\mathcal{G},[\boldsymbol{k}_{\mathrm{tri}}',\boldsymbol{p}',\boldsymbol{s}_i],y_i) \\
&\text{var.} \qquad\qquad \lambda, \boldsymbol{k}_{\mathrm{tri}}, \boldsymbol{p},
\end{aligned} \tag{94}$$

where $\mathcal{G}$ denotes the black-box LLM. λ, $\boldsymbol{k}_{\mathrm{tri}}$, $\boldsymbol{p}$ respectively denote the hyperparameter, backdoor trigger, and prompt. D_j^{tr} and D_j^{val} denote the training and validation dataset in j^{th} worker, and N denotes the number of workers. $\boldsymbol{s}_i, y_i$ denote the i^{th} input sentence and label. In the experiment, Qwen2-7B (Yang et al., 2024a), Llama-3.1-8B (Grattafiori et al., 2024), and Qwen-1.8B-Chat (Bai et al., 2023), are utilized as the black-box LLMs. The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018a) is used to evaluate the proposed DTZO. Specifically, the experiments are carried out on: 1) SST-2 for sentiment analysis; 2) COLA for linguistic acceptability; and 3) MRPC for semantic equivalence of sentences. In the black-box trilevel learning problem, we compare the proposed DTZO with the state-of-the-art distributed bilevel zeroth order learning method FedRZO$_{\mathrm{bl}}$ (Qiu et al., 2023), which is used to address the following distributed bilevel

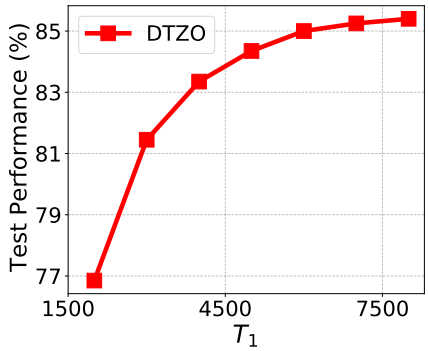


Figure 4. Adjusting T_1 can flexibly control the trade-off between performance and complexity, results on USPS dataset.

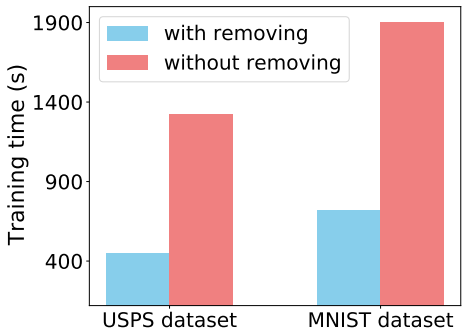


Figure 5. Training time (1000 communication rounds) of with and without removing inactive cuts.

Table 4. Experimental details.

Dataset	η_{x_1}	η_{x_2}	η_{x_3}	μ	λ_l	ϕ_j
SST-2	0.01	0.001	0.001	0.001	1	0.5
COLA	0.01	0.001	0.001	0.001	1	0.5
MRPC	0.01	0.001	0.001	0.001	1	0.5
MNIST	0.01	0.05	0.1	0.001	1	0.5
QMNIST	0.01	0.05	0.1	0.001	1	0.5
F-MNIST	0.01	0.05	0.1	0.001	1	0.5
USPS	0.01	0.5	0.1	0.001	1	0.5

zeroth order learning problem,

$$\begin{aligned}
&\min \textstyle\sum_{j=1}^{N} \frac{1}{|D_j^{\mathrm{tr}}|} \sum_{(\boldsymbol{s}_i,y_i)\sim D_j^{\mathrm{tr}}} L(\mathcal{G},[\boldsymbol{k}_{\mathrm{tri}},\boldsymbol{p},\boldsymbol{s}_i],y_i)\\
&\text{s.t. } \boldsymbol{p}=\arg\min_{\boldsymbol{p}'} \textstyle\sum_{j=1}^{N} \frac{1}{|D_j^{\mathrm{tr}}|} \sum_{(\boldsymbol{s}_i,y_i)\sim D_j^{\mathrm{tr}}} L(\mathcal{G},[\boldsymbol{k}_{\mathrm{tri}},\boldsymbol{p}',\boldsymbol{s}_i],y_i)\\
&\text{var.} \qquad\qquad \boldsymbol{k}_{\mathrm{tri}},\boldsymbol{p},
\end{aligned} \tag{95}$$

where $\mathcal{G}$ denotes the black-box LLM. $\boldsymbol{k}_{\mathrm{tri}}$ and $\boldsymbol{p}$ respectively denote the backdoor trigger and prompt. D_j^{tr} represents the training dataset in j^{th} worker, $|D_j^{\mathrm{tr}}|$ represents the number of data in training dataset, and N denotes the number of workers. $\boldsymbol{s}_i, y_i$ denote the i^{th} input sentence and label.

F.2. Robust Hyperparameter Optimization

Robust hyperparameter optimization is a widely used trilevel learning application (Jiao et al., 2024; Sato et al., 2021), aiming to optimize hyperparameters (Ji et al., 2021; Franceschi et al., 2018; Jiao et al., 2022b; Yang et al., 2021) and train a machine learning model that is robust against adversarial attacks (Han et al., 2024). In this work, we consider the robust hyperparameter optimization, which can be viewed as a trilevel zeroth order learning problem as follows.

$$\begin{aligned}
&\min_{\varphi} \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{var}},y_j^{\mathrm{var}},\boldsymbol{w})\\
&\text{s.t. } \boldsymbol{w}=\arg\min_{\boldsymbol{w}'} \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{tr}}+p_j,y_j^{\mathrm{tr}},\boldsymbol{w}')+\varphi||\boldsymbol{w}'||^2\\
&\quad \text{s.t. } \boldsymbol{p}=\arg\max_{\boldsymbol{p}'} \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{tr}}+p_j{}',y_j^{\mathrm{tr}},\boldsymbol{w}')\\
&\text{var.} \qquad\qquad \varphi,\boldsymbol{w},\boldsymbol{p},
\end{aligned} \tag{96}$$

In this task, compared to single-level optimization, bilevel optimization considers the hyperparameter optimization, which can enhance the generalization ability of the machine learning model. Compared to bilevel optimization, trilevel optimization incorporates min-max robust training, which can improve the adversarial robustness of ML model. The digits recognition

Table 5. Ablation study.

Method	F-MNIST	USPS	UWaveGestureLibraryAll	MelbournePedestrian
DBZO	0.5213	0.7452	0.7043	0.6436
DTZO(-)	0.6685	0.8212	0.7921	0.7013
DTZO	**0.7007**	**0.8513**	**0.8243**	**0.7250**

tasks in Qian et al. (2019); Wang et al. (2021) with several benchmark datasets, i.e., MNIST (LeCun et al., 1998), USPS, Fashion MNIST (Xiao et al., 2017), and QMNIST (Yadav & Bottou, 2019), are utilized to assess the performance of the proposed DTZO. In addition, DTZO is also assessed on time series datasets, including MelbournePedestrian, Crop, and UWaveGestureLibraryAll, sourced from the UCR Archive (Dau et al., 2018). To evaluate the robustness of each method, the PGD-7 attack (Madry et al., 2018) with $\varepsilon = 0.05$ is utilized. For the state-of-the-art distributed zeroth order learning method FedZOO (Fang et al., 2022), it is used to address the following distributed zeroth order learning problem in this task,

$$\begin{aligned} &\min \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{tr}}, y_j^{\mathrm{tr}}, \boldsymbol{w}) \\ &\text{var.} \qquad\quad \boldsymbol{w}, \end{aligned} \tag{97}$$

where N represents the number of workers in a distributed system, $\boldsymbol{w}$ denotes the model parameter. X_j^{tr} and y_j^{tr} represent the training data and labels, respectively. For the state-of-the-art distributed bilevel zeroth order learning method $\mathrm{FedRZO}_{\mathrm{bl}}$ (Qiu et al., 2023), the following distributed bilevel zeroth order learning problem is considered in this task,

$$\begin{aligned} &\min \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{var}}, y_j^{\mathrm{var}}, \boldsymbol{w}) \\ &\text{s.t. } \boldsymbol{w} = \arg\min_{\boldsymbol{w}'} \textstyle\sum_{j=1}^{N} f_j(X_j^{\mathrm{tr}}, y_j^{\mathrm{tr}}, \boldsymbol{w}') + \varphi ||\boldsymbol{w}'||^2 \\ &\text{var.} \qquad\quad \varphi, \boldsymbol{w}, \end{aligned} \tag{98}$$

where φ and $\boldsymbol{w}$ denote the regularization coefficient and model parameter, respectively. X_j^{tr} and y_j^{tr} represent the training data and labels, while X_j^{var} and y_j^{var} represent the validation data and labels, respectively.

Within the proposed framework, the trade-off between complexity and performance can be flexibly controlled by adjusting T_1, as discussed in Sec. 4. Specifically, if the distributed system has limited computational and communication capabilities, a smaller T_1 can be selected. Conversely, if higher performance is required, a larger T_1 can be chosen. As shown in Figure 4, the performance of the proposed framework improves with increasing T_1, allowing for flexible adjustments based on system requirements. Removing inactive cuts can significantly improve the effectiveness of cutting plane method, as discussed in Jiao et al. (2024); Yang et al. (2014). In the experiment, we also investigate the effect of removing inactive cuts within the proposed DTZO. It is seen from Figure 5 that pruning inactive cuts significantly reduces training time, indicating the importance of this procedure.

In addition, the impact of different choices of T_1 on the convergence rate within the proposed framework is evaluated. As illustrated in Figures 6 and 7, a smaller T_1 leads to faster convergence but affects the method's performance, resulting in a higher test loss. Conversely, if a better performance is required, a larger T_1 can be selected, corresponding to a more refined polynomial relaxation. In the proposed framework, we can *flexibly* adjust T_1 based on distributed system requirements. The results in Figures 6 and 7 are consistent with our theoretical analyses presented under Theorems 4.6 and 4.7.

Following Qiu et al. (2023), the robustness in the proposed framework with respect to the choice of smoothing parameter μ is evaluated. The experiments are conducted on the robust hyperparameter optimization task under various settings of smoothing parameter, $\mu \in \{0.01, 0.001, 0.0001\}$. It is seen from Figure 8 and 9 that the proposed DTZO is robust to the choice of smoothing parameter μ. In addition, we also note that the proposed DTZO has faster convergence rate with a relatively smaller μ, because the gradient estimate improves when μ becomes relatively smaller, as discussed in Liu et al. (2020).

Furthermore, to analyze DTZO's performance improvements, we conduct an ablation study comparing DTZO against its variants: DTZO(-) and DBZO. DTZO(-) replaces the proposed nonlinear cuts in DTZO with linear cuts, while DBZO removes cascaded polynomial approximation, using only single-layer polynomial approximation. It is seen from Table 5

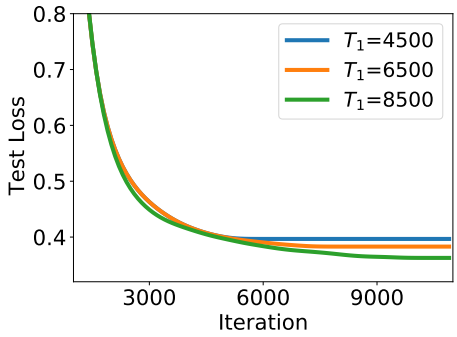


Figure 6. Test loss of the proposed DTZO under various setting of T_1, results on USPS dataset.

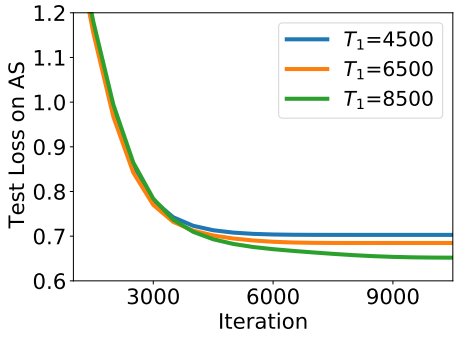


Figure 7. Test loss on AS (adversarial samples) of DTZO under various setting of T_1.

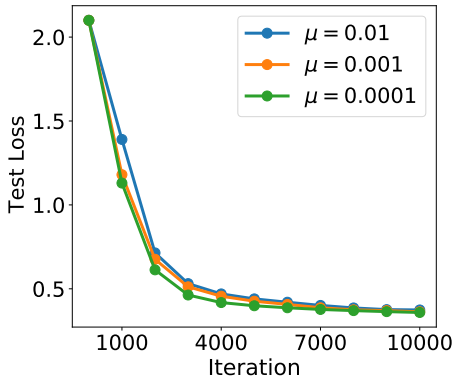


Figure 8. Test loss of the proposed DTZO under various setting of smoothing parameter μ, results on USPS dataset.

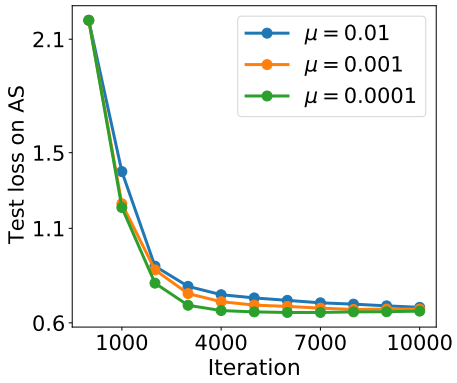


Figure 9. Test loss on AS (adversarial samples) of DTZO under various setting of smoothing parameter μ, results on USPS dataset.

that DTZO outperforms all variants, demonstrating the benefits of cascaded polynomial approximation and nonlinear zeroth order cuts.

G. Discussions about Assumption 4.4 and 4.5

The assumption that the domains of optimization variables are bounded (i.e., bounded domains) is mild and widely used in the theoretical analyses in machine learning and optimization, e.g., Assumption 3 in Deng et al. (2020), Assumption 2.3 in Sra et al. (2016), Assumption A2 in Li & Assaad (2021), Assumption 2.1 in Cao et al. (2024), Assumption 1 in Zinkevich (2003), Assumption 6.3 in Huang et al. (2024c), Assumption in Eq. (4) in Duchi et al. (2012), Assumption 1 in Yang et al. (2024b), Assumption 3.1. in Khaled & Jin (2024), Assumption 2 in Chen et al. (2024b), Assumption 3 in Hazan & Minasyan (2020) and so on.

Let $(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)$ represent the optimal solution of minimizing $F_\mu(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)$, $(\{\boldsymbol{x}_{1,j}^+\},\{\boldsymbol{x}_{2,j}^+\},\{\boldsymbol{x}_{3,j}^+\})$ denote the optimal solution of minimizing $\sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})$, and $(\boldsymbol{x}_{1,j}^-,\boldsymbol{x}_{2,j}^-,\boldsymbol{x}_{3,j}^-)$ denote the optimal solution of minimizing $f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})$. Thus, we have that,

$$\sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}^-,\boldsymbol{x}_{2,j}^-,\boldsymbol{x}_{3,j}^-) \le \sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}^+,\boldsymbol{x}_{2,j}^+,\boldsymbol{x}_{3,j}^+) \le \sum_{j=1}^N f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*). \tag{99}$$

Combining the definition of $F(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)$ in Eq. (15) with the fact that $\phi_j||\boldsymbol{x}_{1,j}^* - \boldsymbol{z}_1^*||^2 \ge 0$,

$\lambda_l[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\varepsilon_{\text{out}}\}]^2\geq 0$, we can obtain that,

$$\begin{aligned}
&\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^-,\boldsymbol{x}_{2,j}^-,\boldsymbol{x}_{3,j}^-)-\tfrac{\mu^2}{2}L(N+1)\textstyle\sum_i d_i\\
&\leq \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^+,\boldsymbol{x}_{2,j}^+,\boldsymbol{x}_{3,j}^+)-\tfrac{\mu^2}{2}L(N+1)\textstyle\sum_i d_i\\
&\leq \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*)-\tfrac{\mu^2}{2}L(N+1)\textstyle\sum_i d_i\\
&\leq F(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\tfrac{\mu^2}{2}L(N+1)\textstyle\sum_i d_i\\
&\leq F_\mu(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)\\
&= F_\mu^*.
\end{aligned}\tag{100}$$

By combining Eq. (100) with the fact that $\frac{\mu^2}{2}L(N+1)\sum_i d_i$ is a constant, we can obtain that the Assumption 4.4 (i.e., F_μ^* is lower-bounded) is mild since the assumption that $f_{1,j}(\boldsymbol{x}_{1,j}^-,\boldsymbol{x}_{2,j}^-,\boldsymbol{x}_{3,j}^-)$ is lower-bounded is widely-used and mild (Liu et al., 2021a; 2018b; 2022; Fang et al., 2022; Li & Assaad, 2021; Liang et al., 2024; Tang et al., 2020; Shaban et al., 2019).

According to the definition of $F(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)$, i.e.,

$$\begin{aligned}
F(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)=&\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})+\phi_j||\boldsymbol{x}_{1,j}-\boldsymbol{z}_1||^2\\
&+\textstyle\sum_l \lambda_l[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)-\varepsilon_{\text{out}}\}]^2,
\end{aligned}\tag{101}$$

we have that 1) term $\phi_j||\boldsymbol{x}_{1,j}-\boldsymbol{z}_1||^2$ satisfies the L-smoothness because the domains of variables $\boldsymbol{x}_{1,j}$ and $\boldsymbol{z}_1$ are bounded; 2) term $\sum_l\lambda_l[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)-\varepsilon_{\text{out}}\}]^2$ satisfies the L-smoothness because the domains of variables are bounded and there are at most $\lfloor\frac{T_1}{\mathcal{T}}\rfloor$ zeroth order cuts. Moreover, the assumption that $f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})$ satisfies the L-smoothness is mild and widely-used (Ji et al., 2021; Gao, 2024; Gao et al., 2022; Chen et al., 2023; Li et al., 2024; Wu et al., 2024; Huang et al., 2024a; Jing et al., 2024; Chen et al., 2024c; Xiao et al., 2023; Hong et al., 2023). Consequently, we can obtain that $F(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3)$ satisfies the L-smoothness, i.e., Assumption 4.5 is mild.

H. Exterior Penalty Method

Exterior penalty methods are widely-used when dealing with constrained optimization problems (Boyd & Vandenberghe, 2004; Bertsekas, 2015). In this work, the exterior penalty method is utilized based on the following key reasons. 1) The lower-level optimization problem often serves as a soft constraint to the upper-level optimization problem, as discussed in Sec. 3.1 and Appendix E, which can be partially violated without rendering the optimization problem meaningless. We can flexibly control the importance in the upper-level and lower-level problems through adjusting the penalty parameters. For example, if the importance of the lower-level optimization problem is required to be high within the nested optimization problem, we can raise the penalty parameters. 2) The complexity of using the exterior penalty method is relatively lower. For example, if we utilize the gradient projection method, which is also widely-used in constrained optimization (Jiao et al., 2023; Xu et al., 2020), we need to solve additional one constrained optimization problem with non-convex feasible regions at each iteration when performing projection, i.e.,

$$\begin{aligned}
&\min \sum_{i=1}^{3}\sum_{j=1}^{N}||\boldsymbol{x}_{i,j}^{t+1}-\boldsymbol{x}_{i,j}||^2+\sum_{i=1}^{3}||\boldsymbol{z}_i^{t+1}-\boldsymbol{z}_i||^2\\
&\text{s.t. } \boldsymbol{x}_{1,j}=\boldsymbol{z}_1,\forall j=1,\cdots,N\\
&\sum_{i=2}^{3}\sum_{j=1}^{N}\boldsymbol{a}_{i,j,l}^{\text{out}\top}\boldsymbol{x}_{i,j}^2+\boldsymbol{b}_{i,j,l}^{\text{out}\top}\boldsymbol{x}_{i,j}+\sum_{i=1}^{3}\boldsymbol{c}_{i,l}^{\text{out}\top}\boldsymbol{z}_i^2+\boldsymbol{d}_{i,l}^{\text{out}\top}\boldsymbol{z}_i+e_l^{\text{out}}\leq\varepsilon_{\text{out}},\forall l\\
&\text{var.}\qquad \{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3,
\end{aligned}\tag{102}$$

where $(\{\boldsymbol{x}_{i,j}^{t+1}\},\{\boldsymbol{z}_i^{t+1}\})$ denotes the points in $(t+1)^{\text{th}}$ iteration after performing zeroth order gradient descent. Thus, it is seen from Eq. (102) that the complexity of utilizing gradient projection descent method is higher than using the penalty method since it requires addressing the constrained non-convex optimization problem in Eq. (102) at each iteration.

Likewise, utilizing the Frank-Wolfe based methods (Shen et al., 2019; Garber & Hazan, 2015; Zhang et al., 2020; Xian et al., 2021; Wang et al., 2016; Balashov et al., 2020) may also lead to relatively more computational complexity since it also needs to solve one additional constrained non-convex optimization problem, i.e.,

$$\begin{aligned}
&\min \sum_{i=1}^{3}\sum_{j=1}^{N} \nabla_{\boldsymbol{x}_{i,j}} f_{1,j}(\boldsymbol{x}_{1,j}^{t+1}, \boldsymbol{x}_{2,j}^{t+1}, \boldsymbol{x}_{3,j}^{t+1})^\top (\boldsymbol{x}_{i,j} - \boldsymbol{x}_{i,j}^{t+1}) \\
&\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\
&\sum_{i=2}^{3}\sum_{j=1}^{N} \boldsymbol{a}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j}^2 + \boldsymbol{b}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j} + \sum_{i=1}^{3} \boldsymbol{c}_{i,l}^{\text{out}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{out}\top} \boldsymbol{z}_i + e_l^{\text{out}} \le \varepsilon_{\text{out}}, \forall l \\
&\text{var.} \qquad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3.
\end{aligned} \tag{103}$$

Thus, as indicated by Eq. (103), the complexity of using the Frank-Wolfe based method is higher than that of the exterior penalty method, as it requires solving an additional constrained non-convex optimization problem in Eq. (103) at each iteration. Based on the aforementioned reasons, we chose to use the exterior penalty method in this work.

In addition, we demonstrate the close relationship between the original constrained optimization problem (P1) in Eq. (8) and the unconstrained optimization problem (P2) in Eq. (15) in this work. That is, 1) the optimal solution to P2 is also a feasible solution to the relaxed original problem P1; 2) the gap between the optimal objective value by utilizing the exterior penalty method (i.e., $\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^*, \boldsymbol{x}_{2,j}^*, \boldsymbol{x}_{3,j}^*)$ in P2) and the optimal objective value in original problem P1 (i.e., $\sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})$) will continuously decrease with penalty parameters increased. To enhance the readability of this discussion, the constrained optimization problem and unconstrained optimization problem are presented as follows.

Constrained cascaded polynomial approximation problem (P1):

$$\begin{aligned}
&\min \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) \\
&\text{s.t. } \boldsymbol{x}_{1,j} = \boldsymbol{z}_1, \forall j = 1, \cdots, N \\
&\sum_{i=2}^{3}\sum_{j=1}^{N} \boldsymbol{a}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j}^2 + \boldsymbol{b}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j} + \sum_{i=1}^{3} \boldsymbol{c}_{i,l}^{\text{out}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{out}\top} \boldsymbol{z}_i + e_l^{\text{out}} \le \varepsilon_{\text{out}}, \forall l \\
&\text{var.} \qquad \{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3.
\end{aligned} \tag{104}$$

Unconstrained optimization problem based on exterior penalty method (P2):

$$\begin{aligned}
\min F(\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3) := &\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}, \boldsymbol{x}_{2,j}, \boldsymbol{x}_{3,j}) + \phi_j ||\boldsymbol{x}_{1,j} - \boldsymbol{z}_1||^2 \\
&+ \textstyle\sum_l \lambda_l [\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) - \varepsilon_{\text{out}}, 0\}]^2, \\
\text{var.} \qquad &\{\boldsymbol{x}_{1,j}\}, \{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3,
\end{aligned} \tag{105}$$

where $h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}\}, \{\boldsymbol{x}_{3,j}\}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) = \sum_{i=2}^{3}\sum_{j=1}^{N} \boldsymbol{a}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j}^2 + \boldsymbol{b}_{i,j,l}^{\text{out}\top} \boldsymbol{x}_{i,j} + \sum_{i=1}^{3} \boldsymbol{c}_{i,l}^{\text{out}\top} \boldsymbol{z}_i^2 + \boldsymbol{d}_{i,l}^{\text{out}\top} \boldsymbol{z}_i + e_l^{\text{out}}$. We first show that the optimal solution to P2 is also a feasible solution to the relaxed original problem P1, and this relaxation will be gradually tightened with penalty parameters increased. Let $(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)$ denote the optimal solution to P2 in Eq. (105). For any point $(\{\boldsymbol{x}_{1,j}^-\}, \{\boldsymbol{x}_{2,j}^-\}, \{\boldsymbol{x}_{3,j}^-\}, \boldsymbol{z}_1^-, \boldsymbol{z}_2^-, \boldsymbol{z}_3^-)$ satisfies $h_l^{\text{out}}(\{\boldsymbol{x}_{1,j}^-\}, \{\boldsymbol{x}_{2,j}^-\}, \{\boldsymbol{x}_{3,j}^-\}, \boldsymbol{z}_1^-, \boldsymbol{z}_2^-, \boldsymbol{z}_3^-) \le \varepsilon_{\text{out}}, \forall l$ and $\boldsymbol{x}_{1,j} - \boldsymbol{z}_1 = 0, \forall j$, since it is also the feasible solution to P2, we have that,

$$\begin{aligned}
&\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^*, \boldsymbol{x}_{2,j}^*, \boldsymbol{x}_{3,j}^*) + \phi_j ||\boldsymbol{x}_{1,j}^* - \boldsymbol{z}_1^*||^2 \\
&+ \textstyle\sum_l \lambda_l [\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\}, \{\boldsymbol{x}_{3,j}^*\}, \boldsymbol{z}_1^*, \boldsymbol{z}_2^*, \boldsymbol{z}_3^*) - \varepsilon_{\text{out}}, 0\}]^2 \\
&\le \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}_{1,j}^-, \boldsymbol{x}_{2,j}^-, \boldsymbol{x}_{3,j}^-) + \phi_j ||\boldsymbol{x}_{1,j}^- - \boldsymbol{z}_1^-||^2 \\
&+ \textstyle\sum_l \lambda_l [\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^-\}, \{\boldsymbol{x}_{3,j}^-\}, \boldsymbol{z}_1^-, \boldsymbol{z}_2^-, \boldsymbol{z}_3^-) - \varepsilon_{\text{out}}, 0\}]^2.
\end{aligned} \tag{106}$$

According to Shen et al. (2024), let $C = 2\max|f_{1,j}|$, we can obtain that,

$$\begin{aligned}&\sum_{j=1}^{N}\phi_j||\boldsymbol{x}_{1,j}^*-\boldsymbol{z}_1^*||^2+\textstyle\sum_l\lambda_l[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\varepsilon_{\text{out}},0\}]^2\\&\leq\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j}^-,\boldsymbol{x}_{2,j}^-,\boldsymbol{x}_{3,j}^-)-\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*)\\&\leq NC.\end{aligned}\tag{107}$$

Because of $||\boldsymbol{x}_{1,j}^*-\boldsymbol{z}_1^*||^2\geq 0$ and $[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\varepsilon_{\text{out}},0\}]^2\geq 0,\forall l$ and according to Eq. (107), we can obtain that,

$$||\boldsymbol{x}_{1,j}^*-\boldsymbol{z}_1^*||^2\leq\frac{NC}{\phi_j},\forall j,\tag{108}$$

$$h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\varepsilon_{\text{out}}\leq\sqrt{\frac{NC}{\lambda_l}},\forall l.\tag{109}$$

According to Eq. (108) and Eq. (109), we can conclude that the optimal solution $(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)$ to P2 is a feasible solution to the relaxed problem of the original constrained problem P1, that is,

$$\begin{aligned}&\min\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j},\boldsymbol{x}_{2,j},\boldsymbol{x}_{3,j})\\&\text{s.t. }||\boldsymbol{x}_{1,j}-\boldsymbol{z}_1||^2\leq\tfrac{NC}{\phi_j},\forall j=1,\cdots,N\\&h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)\leq\varepsilon_{\text{out}}+\sqrt{\tfrac{NC}{\lambda_l}},\forall l\\&\text{var.}\qquad\{\boldsymbol{x}_{1,j}\},\{\boldsymbol{x}_{2,j}\},\{\boldsymbol{x}_{3,j}\},\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{z}_3.\end{aligned}\tag{110}$$

Let $(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\},\overline{\boldsymbol{z}}_1,\overline{\boldsymbol{z}}_2,\overline{\boldsymbol{z}}_3)$ and $(\{\underline{\boldsymbol{x}}_{1,j}\},\{\underline{\boldsymbol{x}}_{2,j}\},\{\underline{\boldsymbol{x}}_{3,j}\},\underline{\boldsymbol{z}}_1,\underline{\boldsymbol{z}}_2,\underline{\boldsymbol{z}}_3)$ respectively denote the optimal solutions to P1 and the relaxed problem of P1 (i.e., Eq. (110)), and let gap

$$\beta(\{\phi_j\},\{\lambda_l\})=\sum_{j=1}^{N}f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})-\sum_{j=1}^{N}f_{1,j}(\{\underline{\boldsymbol{x}}_{1,j}\},\{\underline{\boldsymbol{x}}_{2,j}\},\{\underline{\boldsymbol{x}}_{3,j}\}).\tag{111}$$

It is seen from Eq. (110) that this relaxation will be tightened with penalty parameter $\phi_j,\lambda_l,\forall j,\forall l$ increased. Combining with Eq. (111), we can obtain that $\beta(\{\phi_j\},\{\lambda_l\})\geq 0$ will decrease when $\phi_j,\lambda_l,\forall j,\forall l$ increase. Next, we will demonstrate the gap between the optimal objective value by utilizing the exterior penalty method (i.e., $\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*)$ in P2) and the optimal objective value in original problem P1 (i.e., $\sum_{j=1}^{N}f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})$) will continuously decrease with $\phi_j,\lambda_l,\forall j,\forall l$ increased.

Because $(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\},\overline{\boldsymbol{z}}_1,\overline{\boldsymbol{z}}_2,\overline{\boldsymbol{z}}_3)$ is also the feasible solution to P2, and according to $\sum_j\phi_j||\overline{\boldsymbol{x}}_{1,j}-\overline{\boldsymbol{z}}_1||^2=0$, $\sum_l\lambda_l[\max\{h_l^{\text{out}}(\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\},\overline{\boldsymbol{z}}_1,\overline{\boldsymbol{z}}_2,\overline{\boldsymbol{z}}_3)-\varepsilon_{\text{out}},0\}]^2=0$, we have that,

$$\begin{aligned}&\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*)-\sum_{j=1}^{N}f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})\\&\leq-\sum_{j=1}^{N}\phi_j||\boldsymbol{x}_{1,j}^*-\boldsymbol{z}_1^*||^2-\textstyle\sum_l\lambda_l[\max\{h_l^{\text{out}}(\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)-\varepsilon_{\text{out}},0\}]^2\\&\leq 0.\end{aligned}\tag{112}$$

According to $(\{\boldsymbol{x}_{1,j}^*\},\{\boldsymbol{x}_{2,j}^*\},\{\boldsymbol{x}_{3,j}^*\},\boldsymbol{z}_1^*,\boldsymbol{z}_2^*,\boldsymbol{z}_3^*)$ is a feasible solution to problem in Eq. (110), we can obtain that,

$$\sum_{j=1}^{N}f_{1,j}(\boldsymbol{x}_{1,j}^*,\boldsymbol{x}_{2,j}^*,\boldsymbol{x}_{3,j}^*)\geq\sum_{j=1}^{N}f_{1,j}(\{\underline{\boldsymbol{x}}_{1,j}\},\{\underline{\boldsymbol{x}}_{2,j}\},\{\underline{\boldsymbol{x}}_{3,j}\}).\tag{113}$$

Table 6. Comparisons between the proposed DTZO with the state-of-the-art TLL methods (including Betty (Choe et al., 2023), Hypergradient based method (Sato et al., 2021), and AFTO (Jiao et al., 2024)) based on the applicability to different TLL problems. ✓ represents that the method can be applied to this TLL problem. The proposed DTZO is versatile and can be adapted to a wide range of TLL problems. We use ZOC as an abbreviation for zeroth order constraints.

	Betty	Hypergradient	AFTO	**DTZO**
Non-distributed TLL without ZOC	✓	✓	✓	✓
Distributed TLL without ZOC			✓	✓
TLL with partial ZOC				✓
TLL with level-wise ZOC				✓

By combining Eq. (113) with Eq. (111), we can obtain that,

$$\begin{aligned}&\sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\}) - \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}^*_{1,j},\boldsymbol{x}^*_{2,j},\boldsymbol{x}^*_{3,j})\\ &\leq \sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\}) - \sum_{j=1}^{N} f_{1,j}(\{\underline{\boldsymbol{x}}_{1,j}\},\{\underline{\boldsymbol{x}}_{2,j}\},\{\underline{\boldsymbol{x}}_{3,j}\})\\ &= \beta(\{\phi_j\},\{\lambda_l\}).\end{aligned} \tag{114}$$

By combining Eq. (114) with Eq. (112), we can obtain that,

$$-\beta(\{\phi_j\},\{\lambda_l\}) \leq \sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}^*_{1,j},\boldsymbol{x}^*_{2,j},\boldsymbol{x}^*_{3,j}) - \sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\}) \leq 0. \tag{115}$$

Based on Eq. (115) and $\beta(\{\phi_j\},\{\lambda_l\}) \geq 0$, we can get that,

$$|\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}^*_{1,j},\boldsymbol{x}^*_{2,j},\boldsymbol{x}^*_{3,j}) - \sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})| \leq \beta(\{\phi_j\},\{\lambda_l\}). \tag{116}$$

By combining Eq. (116) with Eq. (110) and Eq. (111), we can conclude the gap between the optimal objective value by utilizing the exterior penalty method (i.e., $\sum_{j=1}^{N} f_{1,j}(\boldsymbol{x}^*_{1,j},\boldsymbol{x}^*_{2,j},\boldsymbol{x}^*_{3,j})$ in P2) and the optimal objective value in original problem P1 (i.e., $\sum_{j=1}^{N} f_{1,j}(\{\overline{\boldsymbol{x}}_{1,j}\},\{\overline{\boldsymbol{x}}_{2,j}\},\{\overline{\boldsymbol{x}}_{3,j}\})$) is bounded and will decrease with penalty parameter $\phi_j, \lambda_l, \forall j, \forall l$ increased.

I. TLL with Partial Zeroth Order Constraints

In this work, TLL with *level-wise* zeroth order constraints is considered, where first order information at *each level* is unavailable. In addition, it is worth mentioning that the proposed framework is versatile and can be adapted to a wide range of TLL problems with partial zeroth order constraints, i.e., grey-box TLL, through slight adjustments. The reason we refer to it as grey-box TLL is that the first order information for some levels in TLL is available, while for others it is not (Huang et al., 2024b; Beykal et al., 2020; Astudillo & Frazier, 2021; Bajaj et al., 2018). To further show the superiority of the proposed DTZO, we compare it with the state-of-the-art TLL methods (i.e., Betty (Choe et al., 2023), Hypergradient based method (Sato et al., 2021), and AFTO (Jiao et al., 2024)) based on their applicability to TLL problems in Table 6. In DTZO, the zeroth order cut takes center stage, driving the construction of cascaded polynomial approximations without the need for gradients or sub-gradients. Notably, zeroth order cut is not only the backbone of DTZO but also opens the door to tackling grey-box TLL problems, seamlessly handling nested functions that combine both black-box and white-box elements. Discussions are provided as follows.

I.1. TLL with second and third-level zeroth order constraints

In this situation, the first order information at the first-level in TLL problems is accessible. Thus, we can use the exact gradients to replace the zeroth order gradient estimator, i.e., Eq. (16)-(19) can be replaced by,

$$\boldsymbol{x}_{1,j}^{t+1} = \boldsymbol{x}_{1,j}^{t} - \eta_{\boldsymbol{x}_1}\left(\nabla_{\boldsymbol{x}_{1,j}} f_{1,j}(\boldsymbol{x}_{1,j}^{t}, \boldsymbol{x}_{2,j}^{t}, \boldsymbol{x}_{3,j}^{t}) + 2\phi_j(\boldsymbol{x}_{1,j}^{t} - \boldsymbol{z}_1^{t})\right), \tag{117}$$

$$\boldsymbol{x}_{2,j}^{t+1} = \boldsymbol{x}_{2,j}^{t} - \eta_{\boldsymbol{x}_2}\nabla_{\boldsymbol{x}_{2,j}} f_{1,j}(\boldsymbol{x}_{1,j}^{t}, \boldsymbol{x}_{2,j}^{t}, \boldsymbol{x}_{3,j}^{t}) - \eta_{\boldsymbol{x}_2}\nabla_{\boldsymbol{x}_{2,j}} o(\{\boldsymbol{x}_{2,j}^{t}\},\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t}), \tag{118}$$

$$\boldsymbol{x}_{3,j}^{t+1} = \boldsymbol{x}_{3,j}^{t} - \eta_{\boldsymbol{x}_3}\nabla_{\boldsymbol{x}_{3,j}} f_{1,j}(\boldsymbol{x}_{1,j}^{t}, \boldsymbol{x}_{2,j}^{t}, \boldsymbol{x}_{3,j}^{t}) - \eta_{\boldsymbol{x}_3}\nabla_{\boldsymbol{x}_{3,j}} o(\{\boldsymbol{x}_{2,j}^{t}\},\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t}). \tag{119}$$

By using the gradient descent steps in Eq. (117)-(119), the TLL problems with second and third-level zeroth order constraints can be effectively by the proposed framework.

I.2. TLL with first and third-level zeroth order constraints

In this situation, the first order information at the second-level in TLL problems is available. Thus, we can use the first order information to generate outer layer cutting plane, e.g., ρ-cut (Jiao et al., 2024). By combining the outer layer first order cutting plane with the inner layer zeroth order cut, the proposed framework is capable of constructing the cascaded polynomial approximation. The generated outer layer ρ-cut can be expressed as,

$$\begin{aligned}
&\nabla\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^{t}\},\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t})^{\top}\left(\begin{bmatrix}\{\boldsymbol{x}_{2,j}\}\\ \{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{2,j}^{t}\}\\ \{\boldsymbol{x}_{3,j}^{t}\}\\ \boldsymbol{z}_1^{t}\\ \boldsymbol{z}_2^{t}\\ \boldsymbol{z}_3^{t}\end{bmatrix}\right)\\
&+\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^{t}\},\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t})\\
&\leq \varepsilon_{\text{out}} + \rho\left(a_1 + (N+1)(a_2+a_3) + \textstyle\sum_{i=2}^{3}\sum_{j=1}^{N}||\boldsymbol{x}_{i,j}^{t}||^2 + \sum_{i=1}^{3}||\boldsymbol{z}_i^{t}||^2\right).
\end{aligned} \tag{120}$$

In Eq. (120), $\rho > 0$ is a parameter in ρ-weakly convex function, and $a_i, i = 1, 2, 3$ is the boundness of variable $\boldsymbol{x}_{i,j}, \boldsymbol{z}_i$, as discussed in Jiao et al. (2024). By using the outer layer first order cutting plane, the TLL problems with first and third-level zeroth order constraints can be addressed by the proposed framework.

I.3. TLL with first and second-level zeroth order constraints

In this situation, the first order information at the third-level in TLL problems is accessible. Similarly, we can utilize the first order information to generate the inner layer cutting plane, e.g., ρ-cut. Through combining the inner layer first order cutting plane with the outer layer zeroth order cut, the proposed framework is capable of constructing the cascaded polynomial approximation. The generated inner layer ρ-cut can be expressed as,

$$\begin{aligned}
&\nabla\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t})^{\top}\left(\begin{bmatrix}\{\boldsymbol{x}_{3,j}\}\\ \boldsymbol{z}_1\\ \boldsymbol{z}_2\\ \boldsymbol{z}_3\end{bmatrix} - \begin{bmatrix}\{\boldsymbol{x}_{3,j}^{t}\}\\ \boldsymbol{z}_1^{t}\\ \boldsymbol{z}_2^{t}\\ \boldsymbol{z}_3^{t}\end{bmatrix}\right) + \phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^{t}\},\boldsymbol{z}_1^{t}, \boldsymbol{z}_2^{t}, \boldsymbol{z}_3^{t})\\
&\leq \varepsilon_{\text{in}} + \rho\left((N+1)a_1 + a_2 + a_3 + \textstyle\sum_{j=1}^{N}||\boldsymbol{x}_{3,j}^{t}||^2 + \sum_{i=1}^{3}||\boldsymbol{z}_i^{t}||^2\right).
\end{aligned} \tag{121}$$

By using the inner layer first order cutting plane in Eq. (121), the TLL problems with second and third-level zeroth order constraints can be addressed by the proposed framework.

I.4. Trade-off between gradient-free and gradient-based methods

The proposed framework is highly adaptable, accommodating both fully gradient-unavailable TLL and cases with partial gradient access with minimal modifications. We further discuss and compare the trade-off between gradient-free and gradient-based methods below.

1. **Gradients at first-level are available.** In this case, Eq. (16)-(19) can be replaced with gradient descent steps in DTZO, which introduce less noise per iteration and improve convergence rate (i.e., $\mathcal{O}(1/\epsilon)$). However, Eq. (16)-(19) do not rely on gradients, making them more applicable to scenarios where gradients are unavailable. This represents a trade-off between convergence efficiency and applicability, which can be flexibly adjusted within DTZO.

2. **Gradients at second-level are available.** In this case, the outer layer zeroth order cut can be replaced by first order cut. Since first order cut is generated based on gradients, it introduces less noise in the generation process and can thus result in a superior polynomial relaxation. In contrast, zeroth order cut exhibits broader applicability, as its generation does not depend on gradients. This represents a trade-off between outer layer polynomial relaxation and applicability, which can be effectively controlled within DTZO.

3. **Gradients at third-level are available.** Similar to 2), there exists a trade-off between inner layer polynomial relaxation and applicability. We can flexibly control this trade-off by exchanging the inner layer zeroth order and first order cuts.

J. Discussions about Cutting Plane Method and Gradient Estimator

J.1. Cutting Plane Method

Cutting plane method, also called polyhedral approximation (Bertsekas, 2015), is widely used in convex optimization (Franc et al., 2011; Boyd & Vandenberghe, 2007) and distributed optimization (Bürger et al., 2013; Yang et al., 2014). The rationale behind cutting plane method is to use the intersection of a finite number of half-spaces (e.g., $P = \{x|a_l^T x \leq b_l, l = 1, \cdots, L\}$, where $\{x|a_l^T x \leq b_l\}$ represent a half-space (Boyd & Vandenberghe, 2004)) to approximate the feasible region of the original optimization problem (e.g., $x \in \mathcal{X}$) . The approximation can be gradually refined by generating additional half-spaces (Bertsekas, 2015). Recently, cutting plane methods have proven effective in tackling distributed nested optimization problems. By leveraging these methods, such problems can be transformed into decomposable optimization problems, which greatly simplifies the design of distributed algorithms for nested optimization, as discussed in (Jiao et al., 2023; 2024). In (Jiao et al., 2023), cutting plane methods are applied to solve bilevel optimization problems within a distributed framework. Likewise, (Chen et al., 2024d) utilize the cutting plane method to tackle distributed bilevel optimization challenges in downlink multi-cell systems. Building on this, (Jiao et al., 2024) further extend the approach to address distributed trilevel optimization problems. However, existing cutting plane methods for nested optimization rely on the first-order information to generate cutting planes, which are not available in zeroth-order optimization.

In this work, we propose a framework capable of generating zeroth order cuts for nested optimization problems **without** the use of first order information. We theoretically demonstrate that the proposed zeroth order cuts are capable of constructing the cascaded polynomial relaxation without relying on first order information, and this relaxation will be gradually tightened as additional cuts are introduced. Additionally, it is worth mentioning that the proposed zeroth order cuts do not require the convexity of the function and are also the first non-linear cuts in nested optimization. Compared to linear cutting planes, nonlinear cuts usually offer better approximation capabilities for complex functions (Temlyakov, 2003), providing new insights for the further development of cutting plane methods in nested optimization. Please note that simply combining the existing algorithms can not achieve this goal. To further highlight the novelty of the proposed zeroth order cut, we compare it with existing cutting plane methods used in nested optimization in Table 7.

J.2. The Choice of Gradient Estimator

It is worth noting that the proposed framework is versatile, allowing for the integration of various gradient estimators. For instance, the mini-batch sampling-based gradient estimator (Liu et al., 2020; Duchi et al., 2015) can be employed to replace the two-point gradient estimator, reducing variance. Specifically, with mini-batch sampling, Eq. (10), (12) (19), (20), and

Table 7. Comparison of the existing cutting plane methods in nested optimization.

Cutting Plane Method	Convex	Non-convex	Non-linear	Gradient Free
(Jiao et al., 2023)	✓			
(Chen et al., 2024d)	✓			
(Jian et al., 2024)	✓			
(Jiao et al., 2024)	✓	✓		
The Proposed Zeroth Order Cut	✓	✓	✓	✓

(21) can be replaced by the following multi-point gradient estimators.

$$\begin{aligned} &G_\mu^{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t) \\ &= \tfrac{1}{\mu}\sum_{p=1}^{b}[\phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t + \mu\boldsymbol{\mu}_{x_{3,j}}^p\}, \boldsymbol{z}_1^t + \mu\boldsymbol{\mu}_{z_1}^p, \boldsymbol{z}_2^{t\prime} + \mu\boldsymbol{\mu}_{z_2}^p, \boldsymbol{z}_3^t + \mu\boldsymbol{\mu}_{z_3}^p) - \phi_{\text{in}}(\{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^{t\prime}, \boldsymbol{z}_3^t)\boldsymbol{\mu}^{\text{in},p}], \end{aligned} \tag{122}$$

$$\begin{aligned} &G_\mu^{\text{out}}(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ &= \tfrac{1}{\mu}\sum_{p=1}^{b}[\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t + \mu\boldsymbol{\mu}_{x_{2,j}}^p\}, \{\boldsymbol{x}_{3,j}^t + \mu\boldsymbol{\mu}_{x_{3,j}}^p\}, \boldsymbol{z}_1^t + \mu\boldsymbol{\mu}_{z_1}^p, \boldsymbol{z}_2^t + \mu\boldsymbol{\mu}_{z_2}^p, \boldsymbol{z}_3^t + \mu\boldsymbol{\mu}_{z_3}^p) \\ &\qquad -\phi_{\text{out}}(\{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t)\boldsymbol{\mu}^{\text{out},p}], \end{aligned} \tag{123}$$

$$\begin{aligned} &G_{\boldsymbol{x}_{1,j}}(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ &= \tfrac{1}{\mu}\sum_{p=1}^{b}[f_{1,j}(\boldsymbol{x}_{1,j}^t + \mu\boldsymbol{u}_{k,1}^p, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)\boldsymbol{u}_{k,1}^p] + 2\phi_j(\boldsymbol{x}_{1,j}^t - \boldsymbol{z}_1^t), \end{aligned} \tag{124}$$

$$\begin{aligned} G_{\boldsymbol{x}_{2,j}}(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) &= \nabla_{\boldsymbol{x}_{2,j}} o(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ &+\tfrac{1}{\mu}\sum_{p=1}^{b}[f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t + \mu\boldsymbol{u}_{k,2}^p, \boldsymbol{x}_{3,j}^t) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)\boldsymbol{u}_{k,2}^p], \end{aligned} \tag{125}$$

$$\begin{aligned} G_{\boldsymbol{x}_{3,j}}(\{\boldsymbol{x}_{1,j}^t\}, \{\boldsymbol{x}_{2,j}^t\}, \{\boldsymbol{x}_{3,j}^t\}, \boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) &= \nabla_{\boldsymbol{x}_{3,j}} o(\{\boldsymbol{x}_{2,j}^t\},\{\boldsymbol{x}_{3,j}^t\},\boldsymbol{z}_1^t, \boldsymbol{z}_2^t, \boldsymbol{z}_3^t) \\ &+\tfrac{1}{\mu}\sum_{p=1}^{b}[f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t + \mu\boldsymbol{u}_{k,3}^p) - f_{1,j}(\boldsymbol{x}_{1,j}^t, \boldsymbol{x}_{2,j}^t, \boldsymbol{x}_{3,j}^t)\boldsymbol{u}_{k,3}^p], \end{aligned} \tag{126}$$

where $\boldsymbol{\mu}^{\text{in},p} = [\{\boldsymbol{\mu}_{x_{3,j}}^p\}, \boldsymbol{\mu}_{z_1}^p, \boldsymbol{\mu}_{z_2}^p, \boldsymbol{\mu}_{z_3}^p]$, $\boldsymbol{\mu}^{\text{out},p} = [\{\boldsymbol{\mu}_{x_{2,j}}^p\}, \{\boldsymbol{\mu}_{x_{3,j}}^p\}, \boldsymbol{\mu}_{z_1}^p, \boldsymbol{\mu}_{z_2}^p, \boldsymbol{\mu}_{z_3}^p]$, $\boldsymbol{u}_{k,1}^p$, $\boldsymbol{u}_{k,2}^p$, $\boldsymbol{u}_{k,3}^p, p = 1, \cdots b$ are drawn from $\mathcal{N}(0, \mathbf{I})$, and b represents the number of samples used in the multi-point gradient estimator.

K. Future Work

This study is the first work that considers how to address the trilevel zeroth order optimization problems. The proposed framework is not only capable of addressing the single-level and bilevel zeroth order learning problems but can also be applied to a broad class of TLL problems, e.g., TLL with partial zeroth order constraints. However, higher-level nested learning problems, specifically those with more than three levels, are not considered in this work and will be addressed in future research.