Distributed Distributionally Robust Optimization with Non-Convex Objectives

Yang Jiao
Tongji University
yangjiao@tongji.edu.cn
Kai Yang
Tongji University
kaiyang@tongji.edu.cn
Dongjin Song
University of Connecticut
donjin.song@uconn.edu

Abstract

Distributionally Robust Optimization (DRO), which aims to find an optimal decision that minimizes the worst case cost over the ambiguity set of probability distribution, has been widely applied in diverse applications, e.g., network behavior analysis, risk management, etc. However, existing DRO techniques face three key challenges: 1) how to deal with the asynchronous updating in a distributed environment; 2) how to leverage the prior distribution effectively; 3) how to properly adjust the degree of robustness according to different scenarios. To this end, we propose an asynchronous distributed algorithm, named Asynchronous Single-loop Gradient Projection (ASPIRE) algorithm with the Iterative Active Set method (EASE) to tackle the distributed distributionally robust optimization (DDRO) problem. Furthermore, a new uncertainty set, i.e., constrained $D$-norm uncertainty set, is developed to effectively leverage the prior distribution and flexibly control the degree of robustness. Finally, our theoretical analysis elucidates that the proposed algorithm is guaranteed to converge and the iteration complexity is also analyzed. Extensive empirical studies on real-world datasets demonstrate that the proposed method can not only achieve fast convergence, and remain robust against data heterogeneity as well as malicious attacks, but also tradeoff robustness with performance.

1 Introduction

The past decade has witnessed the proliferation of smartphones and Internet of Things (IoT) devices, which generate a plethora of data everyday. Centralized machine learning requires gathering the data to a particular server to train models which incurs high communication overhead [46] and suffers privacy risks [43]. As a remedy, distributed machine learning methods have been proposed. Considering a distributed system composed of $N$ workers (devices), we denote the dataset of these workers as $\{D_1, \cdots, D_N\}$. For the $j$th ($1 \leq j \leq N$) worker, the labeled dataset is given as $D_j = \{x_{ij}, y_{ij}\}$, where $x_{ij} \in \mathbb{R}^d$ and $y_{ij} \in \{1, \cdots, c\}$ denote the $i$th data sample and the corresponding label, respectively. The distributed learning tasks can be formulated as the following optimization problem,

$$\min_{w \in \mathcal{W}} F(w) \quad \text{with} \quad F(w) := \sum_j f_j(w),$$

(1)

where $w \in \mathbb{R}^p$ is the model parameter to be learned and $\mathcal{W} \subseteq \mathbb{R}^p$ is a nonempty closed convex set, $f_j(\cdot)$ is the empirical risk over the $j$th worker involving only the local data:

$$f_j(w) = \sum_{i : x_{ij} \in D_j} \frac{1}{|D_j|} L_j(x_{ij}, y_{ij} ; w),$$

(2)

*Corresponding author.

where $\mathcal{L}_j$ is the local objective function over the $j^{th}$ worker. Problem in Eq. (1) arises in numerous areas, such as distributed signal processing \cite{19}, multi-agent optimization \cite{36}, etc. However, such problem does not consider the data heterogeneity \cite{57,40,39,30} among different workers (i.e., data distribution of workers could be substantially different from each other \cite{44}). Indeed, it has been shown that traditional federated approaches, such as FedAvg \cite{33}, built for independent and identically distributed (IID) data may perform poorly when applied to Non-IID data \cite{27}. This issue can be mitigated via learning a robust model that aims to achieve uniformly good performance over all workers by solving the following distributionally robust optimization (DRO) problem in a distributed manner:

$$\min_{w \in \mathcal{W}} \max_{p \in \mathcal{P} \subseteq \Delta_N} F(w, p) := \sum_{j} p_j f_j(w),$$

where $p = [p_1, \cdots, p_N] \subseteq \mathbb{R}^N$ is the adversarial distribution in $N$ workers, the $j^{th}$ entry in this vector, i.e., $p_j$ represents the adversarial distribution value for the $j^{th}$ worker. $\Delta_N = \{p \in \mathbb{R}^N : 1^T p = 1\}$ and $\mathcal{P}$ is a subset of $\Delta_N$. Agnostic federated learning (AFL) \cite{35} firstly introduces the distributionally robust (agnostic) loss in federated learning and provides the convergence rate for (strongly) convex functions. However, AFL does not discuss the setting of $\mathcal{P}$. DRFA-Prox \cite{16} considers $\mathcal{P} = \Delta_N$ and imposes a regularizer on adversarial distribution to leverage the prior distribution. Nevertheless, three key challenges have not yet been addressed by prior works. First, whether it is possible to construct an uncertainty framework that can not only flexibly maintain the trade-off between the model robustness and performance but also effectively leverage the prior distribution? Second, how to design asynchronous algorithms with guaranteed convergence? Compared to synchronous algorithms, the master in asynchronous algorithms can update its parameters after receiving updates from only a small subset of workers \cite{58,10}. Asynchronous algorithms are particularly desirable in practice since they can relax strict data dependencies and ensure convergence even in the presence of device failures \cite{58}. Finally, whether it is possible to flexibly adjust the degree of robustness? Moreover, it is necessary to provide convergence guarantee when the objectives (i.e., $f_j(w_j), \forall j$) are non-convex.

To this end, we propose ASPIRE-EASE to effectively address the aforementioned challenges. Firstly, different from existing works, the prior distribution is incorporated within the constraint in our formulation, which can not only leverage the prior distribution more effectively but also achieve guaranteed feasibility for any adversarial distribution within the uncertainty set. The prior distribution can be obtained from side information or uniform distribution \cite{41}, which is necessary to construct the uncertainty (ambiguity) set and obtain a more robust model \cite{16}. Specifically, we formulate the prior distribution informed distributionally robust optimization (PD-DRO) problem as:

$$\min_{z \in \mathcal{Z}, \{w_j \in \mathcal{W}\}} \max_{p \in \mathcal{P}} \sum_{j} p_j f_j(w_j)$$

$$\text{s.t.} \quad z = w_j, \hspace{0.5cm} j = 1, \cdots, N,$$

$$\text{var.} \quad z, w_1, w_2, \cdots, w_N,$$

where $z \in \mathbb{R}^p$ is the global consensus variable, $w_j \in \mathbb{R}^p$ is the local variable (local model parameter) of $j^{th}$ worker and $\mathcal{Z} \subseteq \mathbb{R}^p$ is a nonempty closed convex set. $\mathcal{P} \subseteq \mathbb{R}_+^N$ is the uncertainty (ambiguity) set of adversarial distribution $p$, which is set based on the prior distribution. To solve the PD-DRO problem in an asynchronous distributed manner, we first propose Asynchronous Single-look Active Gradient projecTion (ASPIRE), which employs simple gradient projection steps for the update of primal and dual variables at every iteration, thus is computationally efficient. Next, the itErative Active SEt method (EASE) is employed to replace the traditional cutting plane method to improve the computational efficiency and speed up the convergence. We further provide the convergence guarantee for the proposed algorithm. Furthermore, a new uncertainty set, i.e., constrained $D$-norm ($CD$-norm), is proposed in this paper and its advantages include: 1) it can flexibly control the degree of robustness; 2) the resulting subproblem is computationally simple; 3) it can effectively leverage the prior distribution and flexibly set the bounds for every $p_j$.

**Contributions.** Our contributions can be summarized as follows:

1. We formulate a PD-DRO problem with $CD$-norm uncertainty set. PD-DRO incorporates the prior distribution as constraints which can leverage prior distribution more effectively and guarantee robustness. In addition, $CD$-norm is developed to model the ambiguity set around the prior distribution and it provides a flexible way to control the trade-off between model robustness and performance.

2. We develop a single-loop asynchronous algorithm, namely ASPIRE-EASE, to optimize PD-DRO in an asynchronous distributed manner. ASPIRE employs simple gradient projection steps to
update the variables at every iteration, which is computationally efficient. And EASE is proposed to replace cutting plane method to enhance the computational efficiency and speed up the convergence. We demonstrate that even if the objectives $f_j(w_j), \forall j$ are non-convex, the proposed algorithm is guaranteed to converge. We also theoretically derive the iteration complexity of ASPIRE-EASE.

3. Extensive empirical studies on four different real world datasets demonstrate the superior performance of the proposed algorithm. It is seen that ASPIRE-EASE can not only ensure the model’s robustness against data heterogeneity but also mitigate malicious attacks.

2 Preliminaries

2.1 Distributionally Robust Optimization

Optimization problems often contain uncertain parameters. A small perturbation of the parameters could render the optimal solution of the original optimization problem infeasible or completely meaningless [5]. Distributionally robust optimization (DRO) [28, 17, 7] assumes that the probability distributions of uncertain parameters are unknown but remain in an ambiguity (uncertainty) set and aims to find a decision that minimizes the worst case expected cost over the ambiguity set, whose general form can be expressed as,

$$\min_{x \in X} \max_{P \in \mathcal{P}} \mathbb{E}_P[r(x, \xi)],$$

(5)

where $x \in X$ represents the decision variable, $\mathcal{P}$ is the ambiguity set of probability distributions $P$ of uncertain parameters $\xi$. Existing methods for solving DRO can be broadly grouped into two widely-used categories [42]: 1) Dual methods [15, 50, 18] reformulate the primal DRO problems as deterministic optimization problems through duality theory. Ben-Tal et al. [2] reformulate the robust linear optimization (RLO) problem with an ellipsoidal uncertainty set as a second-order cone optimization problem (SOCP). 2) Cutting plane methods [34, 6] (also called adversarial approaches [21]) continuously solve an approximate problem with a finite number of constraints of the primal optimization problem. Recently, several new methods [41, 29, 23] have been developed to solve DRO, which need to solve robust linear optimization (RLO) problem with an ellipsoidal uncertainty set as a second-order cone optimization problem (SOCP). 2) Cutting plane methods [34, 6] (also called adversarial approaches [21]) continuously solve an approximate problem with a finite number of constraints of the primal DRO problem, and subsequently check whether new constraints are needed to refine the feasible set. In this section, we introduce the cutting plane method for PD-DRO in Eq. (4). We first reformulate the inner maximization problem at every iteration.

In this section, we introduce the cutting plane method for PD-DRO in Eq. (4). We first reformulate PD-DRO by introducing an additional variable $h \in \mathcal{H}$ ($\mathcal{H} \subseteq \mathbb{R}^1$ is a nonempty closed convex set) and protection function $g(\{w_j\})$ [55]. Introducing additional variable $h$ is an epigraph reformulation [3, 56]. In this case, Eq. (4) can be reformulated as the form with uncertainty in the constraints:

$$\min_{z \in \mathcal{Z}, \{w_j\} \in \mathcal{W}, h \in \mathcal{H}} h$$

s.t. $\sum_j p f_j(w_j) + g(\{w_j\}) - h \leq 0,$

(6)

$$z = w_j, \ j = 1, \cdots, N,$$

$$\text{var.} \ z, w_1, w_2, \cdots, w_N, h,$$

where $p$ is the nominal value of the adversarial distribution for every worker and $g(\{w_j\}) = \max_{p \in \mathcal{P}} \sum_j (p_j - p) f_j(w_j)$ is the protection function. Eq. (6) is a semi-infinite program (SIP) which contains infinite constraints and cannot be solved directly [42]. Denoting the set of cutting plane parameters in $(t+1)$th iteration as $A^t \subseteq \mathbb{R}^N$, the following function is used to approximate $g(\{w_j\})$:

$$\overline{g}(\{w_j\}) = \max_{a_t \in A^t} a_t^T f(w) = \max_{a_t \in A^t} \sum_j a_{t,j} f_j(w_j),$$

(7)

$$\text{where} \ a_t = [a_{t,1}, \cdots, a_{t,N}] \in \mathbb{R}^N \text{ denotes the parameters of } l^{th} \ \text{cutting plane in } A^t \text{ and } f(w) = [f_1(w_1), \cdots, f_N(w_N)] \in \mathbb{R}^N. \ \text{Substituting the protection function } g(\{w_j\}) \ \text{with } \overline{g}(\{w_j\}), \ \text{we can obtain the following approximate problem:}$$

$$\min_{z \in \mathcal{Z}, \{w_j\} \in \mathcal{W}, h \in \mathcal{H}} h$$

s.t. $\sum_j (\overline{p} + a_{t,j}) f_j(w_j) - h \leq 0, \forall a_t \in A^t,$

(8)

$$z = w_j, \ j = 1, \cdots, N,$$

$$\text{var.} \ z, w_1, w_2, \cdots, w_N, h.$$
3 ASPIRE

Distributed optimization is an attractive approach for large-scale learning tasks since it does not require data aggregation, which protects data privacy while also reducing bandwidth requirements. When the neural network models (i.e., $f_j(w_j)$, $\forall j$ are non-convex functions) are used, solving problem in Eq. (8) in a distributed manner facing two challenges: 1) Computing the optimal solution to a non-convex subproblem requires a large number of iterations and therefore is highly computationally intensive if not impossible. Thus, the traditional Alternating Direction Method of Multipliers (ADMM) is ineffective. 2) The communication delays of workers may differ significantly, thus, asynchronous algorithms are strongly preferred.

To this end, we propose the A\textbf{s}ynchronous S\textbf{ingle}-looP alternat\textbf{I}ve g\textbf{R}adiant p\textbf{r}oj\textbf{e}ction (ASPIRE). The advantages of the proposed algorithm include: 1) ASPIRE uses simple gradient projection steps to update variables in each iteration and therefore it is computationally more efficient than the traditional ADMM method, which seeks to find the optimal solution in non-convex (for $w_j$, $\forall j$) and convex (for $z$ and $h$) optimization subproblems every iteration, 2) the proposed asynchronous algorithm does not need strict synchronization among different workers. Therefore, ASPIRE remains resilient against communication delays and potential hardware failures from workers. Details of the algorithm are given below. Firstly, we define the node as master which is responsible for updating the global variable $z$, and we define the node which is responsible for updating the local variable $w_j$ as worker $j$. In each iteration, the master updates its variables once it receives updates from at least $S$ workers, i.e., active workers, satisfying $1 \leq S \leq N$. $Q^{t+1}$ denotes the index subset of workers from which the master receives updates during $(t+1)^{th}$ iteration. We also assume the master will receive updated variables from every worker at least once for each $\tau$ iterations. The augmented Lagrangian function of Eq. (8) can be written as:

$$L_p = h + \sum_i \lambda_i \left( \sum_j (\bar{\|} + w_{ij} f_j(w_j) - h \right) + \sum_j \phi_j^t(z - w_j) + \sum_j \frac{\kappa_1}{2} ||w_j||^2,$$

where $L_p = L_p(\{w_j\}, z, h, \{\lambda_i\}, \{\phi_j\})$, $\lambda_i \in \Lambda$, $\forall l$ and $\phi_j \in \Phi$, $\forall j$ represent the dual variables of inequality and equality constraints in Eq. (8), respectively. $\Lambda \subseteq \mathbb{R}^l$ and $\Phi \subseteq \mathbb{R}^p$ are nonempty closed convex sets, constant $\kappa_1 > 0$ is a penalty parameter. Note that Eq. (9) does not consider the second-order penalty term for inequality constraint since it will invalidate the distributed optimization. Following [52], the regularized version of Eq. (9) is employed to update all variables as follows,

$$\tilde{L}_p(\{w_j\}, z, h, \{\lambda_i\}, \{\phi_j\}) = L_p - \sum_j \frac{c_1^t}{2} ||\lambda_i||^2 - \sum_j \frac{c_2^t}{2} ||\phi_j||^2,$$

where $c_1^t$ and $c_2^t$ denote the regularization terms in $(t + 1)^{th}$ iteration. To avoid enumerating the whole dataset, the mini-batch loss could be used. A batch of instances with size $m$ can be randomly sampled from each worker during each iteration. The loss function of these instances from $j^{th}$ worker is given by $f_j(w_j) = \sum_{i=1}^m \frac{1}{m} C_j(x_i^j, y_i^j; w_j)$. It is evident that $\mathbb{E}[f_j(w_j)] = f_j(w_j)$ and $\mathbb{E}[\nabla f_j(w_j)] = \nabla f_j(w_j)$. In $(t + 1)^{th}$ master iteration, the proposed algorithm proceeds as follows.

1) Active workers update the local variables $w_j$ as follows,

$$w_j^{t+1} = \left\{ \mathcal{P}_W(w_j^t - \alpha_w \nabla w_j \tilde{L}_p(\{w_j^t\}, z^t, h^t, \{\lambda_i^t\}, \{\phi_j^t\}) \right\}, \forall j \in Q^{t+1},$$

where $\tilde{t}_j$ is the last iteration during which worker $j$ was active. It is seen that $\forall j \in Q^{t+1}$, $w_j^t = w_j^{\tilde{t}_j}$ and $\phi_j^t = \phi_j^{\tilde{t}_j}$. $\alpha_w$ represents the step-size and let $\alpha_w^t = \eta_w^t$ when $t < T_1$ and $\alpha_w^t = \eta_w^t$ when $t \geq T_1$, where $\eta_w$ and constant $\eta_w$ will be introduced below. $\mathcal{P}_W$ represents the projection onto the closed convex set $W$ and we set $\mathcal{W} = \{w_j ||w_j||_\infty \leq \alpha_1\}$, $\alpha_1$ is a positive constant. And then, the active workers $(j \in Q^{t+1})$ transmit their local model parameters $w_j^{t+1}$ and loss $f_j(w_j)$ to the master.

2) After receiving the updates from active workers, the master updates the global consensus variable $z$, additional variable $h$ and dual variables $\lambda_i$ as follows,

$$z^{t+1} = \mathcal{P}_Z(z^t - \eta_z^t \nabla_z \tilde{L}_p(\{w_j^{t+1}\}, z^t, h^t, \{\lambda_i^t\}, \{\phi_j^t\})),$$

$$h^{t+1} = \mathcal{P}_H(h^t - \eta_h^t \nabla_h \tilde{L}_p(\{w_j^{t+1}\}, z^{t+1}, h^t, \{\lambda_i^t\}, \{\phi_j^t\})),$$
\[ \lambda_i^{t+1} = \mathcal{P}_\Lambda (\lambda_i^t + \rho_1 \nabla \lambda_i \bar{L}_p (\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_j^t\}, \{\phi_j^t\})) \quad l = 1, \ldots, |\mathbf{A}|, \]  

where \( \eta^t_1, \eta^t_2 \) and \( \rho_1 \) represent the step-sizes, \( \mathcal{P}_\mathcal{Z}, \mathcal{P}_\mathcal{H} \) and \( \mathcal{P}_\Lambda \) respectively represent the projection onto the closed convex sets \( \mathcal{Z}, \mathcal{H} \) and \( \Lambda \). We set \( \mathcal{Z} = \{z| ||z||_\infty \leq \alpha_1\} \), \( \mathcal{H} = \{h| 0 \leq h \leq \alpha_2\} \) and \( \Lambda = \{\lambda| 0 \leq \lambda \leq \lambda_3\} \), where \( \alpha_2 \) and \( \alpha_3 \) are positive constants. \(|\mathbf{A}|\) denotes the number of cutting planes. Then, master broadcasts \( z^{t+1}, h^{t+1}, \{\lambda_j^{t+1}\} \) to the active workers.

3) Active workers update the local dual variables \( \phi_j \) as follows,

\[ \phi_j^{t+1} = \begin{cases} \mathcal{P}_\Phi (\phi_j^t + \rho_2 \nabla \phi_j \bar{L}_p (\{w_j^{t+1}\}, z^{t+1}, h^{t+1}, \{\lambda_j^t\}, \{\phi_j^t\})) \quad \forall j \in Q^{t+1}, \\ \phi_j^t \quad \text{else} \end{cases} \]

where \( \rho_2 \) represents the step-size and \( \mathcal{P}_\Phi \) represents the projection onto the closed convex set \( \Phi \) and we set \( \Phi = \{\phi_j| ||\phi_j||_\infty \leq \alpha_4\} \), \( \alpha_4 \) is a positive constant. And master can also obtain \( \{\phi_j^{t+1}\} \) according to Eq. (15). It is seen that the projection operation in each step is computationally simple since the closed convex sets have simple structures [4].

4 Iterative Active Set Method

Cutting plane methods may give rise to numerous linear constraints and lots of extra message passing [5]. Moreover, more iterations are required to obtain the \( \varepsilon \)-stationary point when the size of a set containing cutting planes increases which corresponds to a larger \( M \), which can be seen in Theorem 1. To improve the computational efficiency and speed up the convergence, we consider removing the inactive cutting planes. The proposed iIterative Active SEt method (EASE) can be divided into the two steps: during \( T_1 \) iterations, 1) solving the cutting plane generation subproblem to generate cutting plane, and 2) removing the inactive cutting plane every \( k \) iterations, where \( k > 0 \) is a pre-set constant and can be controlled flexibly.

The cutting planes are generated according to the uncertainty set. For example, if we employ ellipsoid uncertainty set, the cutting plane is generated via solving a SOCP. In this paper, we propose \( CD \)-norm uncertainty set, which can be expressed as follows,

\[ \mathcal{P} = \{p| -\bar{p}_j \leq p_j - q_j \leq \bar{p}_j, \sum_j \left| \frac{p_j - q_j}{p_j} \right| \leq \Gamma, 1^T p = 1\} \]  

where \( \Gamma \in \mathbb{R}^1 \) can flexibly control the level of robustness, \( q = [q_1, \ldots, q_N] \in \mathbb{R}^N \) represents the prior distribution, \( -\bar{p}_j \) and \( \bar{p}_j \) represent the lower and upper bounds for \( p_j - q_j \) respectively. The setting of \( q \) and \( \bar{p}_j, \forall j \) are based on the prior knowledge. \( D \)-norm is a classical uncertainty set (which is also called as budget uncertainty set) [3]. We call Eq. (16) \( CD \)-norm uncertainty set since \( p \) is a probability vector so all the entries of this vector are non-negative and add up to exactly one, i.e., \( 1^T p = 1 \). Due to the special structure of \( CD \)-norm, the cutting plane generation subproblem is easy to solve and the level of robustness in terms of the outage probability, i.e., probabilistic bounds of the violations of constraints can be flexibly adjusted via a single parameter \( \Gamma \). We claim that \( \ell_1 \)-norm (or twice total variation distance) uncertainty set is closely related to \( CD \)-norm uncertainty set. Nevertheless, there are two differences: 1) \( CD \)-norm uncertainty set could be regarded as a weighted \( \ell_1 \)-norm with additional constraints. 2) \( CD \)-norm uncertainty set can flexibly set the lower and upper bounds for every \( p_j \) (i.e., \( q_j - \bar{p}_j \leq p_j \leq \bar{p}_j + \bar{p}_j \)), while \( 0 \leq p_j \leq 1, \forall j \) in \( \ell_1 \)-norm uncertainty set. Based on the \( CD \)-norm uncertainty set, the cutting plane can be derived as follows.

1) Solve the following problem,

\[ \begin{align*} 
\mathbf{p}^{t+1} &= \text{arg} \max_{p_1, \ldots, p_N} \sum_j (p_j - p) f_j(w_j) \\
\text{s.t.} & \sum_j \left| \frac{p_j - q_j}{p_j} \right| \leq \Gamma, \quad -\bar{p}_j \leq p_j - q_j \leq \bar{p}_j, \forall j, \sum_j p_j = 1 \\
\text{var.} & p_1, \ldots, p_N 
\end{align*} \]  

where \( \mathbf{p}^{t+1} = [p_1^{t+1}, \ldots, p_N^{t+1}] \in \mathbb{R}^N \). Let \( \bar{\mathbf{a}}^{t+1} = \mathbf{p}^{t+1} - \mathbf{p} \), where \( \mathbf{p} = [p, \ldots, p] \in \mathbb{R}^N \). This first step aims to obtain the distribution \( \bar{\mathbf{a}}^{t+1} \) by solving problem in Eq. (17). This problem can be effectively solved through combining merge sort [13] (for sorting \( \bar{p}_j f_j(w_j), j = 1, \ldots, N \)) with few basic arithmetic operations (for obtaining \( p_j^{t+1}, j = 1, \ldots, N \)). Since \( N \) is relatively large in
Algorithm 1 ASPIRE-EASE

**Initialization:** iteration $t = 0$, variables $\{w^0_j\}$, $z^0$, $h^0$, $\{\lambda^0_j\}$, $\{\phi^0_j\}$ and set $A^0$.

repeat
  for active worker do
    updates local $w^t_j$ according to Eq. (11);
  end for

active workers transmit local model parameters and loss to master;

master receives updates from active workers do
  updates $z^{t+1}$, $h^{t+1}$, $\{\lambda^{t+1}_j\}$, $\{\phi^{t+1}_j\}$ in master according to Eq. (12), (13), (14), (15);

master broadcasts $z^{t+1}$, $h^{t+1}$, $\{\lambda^{t+1}_j\}$ to active workers;

for active worker do
  updates local $\phi^{t+1}_j$ according to Eq. (15);
end for

if $(t + 1) \mod k = 0$ and $t < T$ then
  master updates $A^{t+1}$ according to Eq. (19) and (20), and broadcast parameters to all workers;
end if

$t = t + 1$;
until convergence

5 Convergence Analysis

**Definition 1** (Stationarity gap) Following [52, 52, 53], the stationarity gap of our problem at $t$th iteration is defined as:

$$
\nabla G^t = \begin{bmatrix}
\frac{1}{\sigma_w}(w_j - \mathcal{P}_w(w_j - \alpha_w \nabla w, L_p((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\}))) \\
\frac{1}{\eta_z}(z - \mathcal{P}_z(z - \eta_z \nabla z, L_p((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\}))) \\
\frac{1}{\eta_h}(h - \mathcal{P}_h(h - \eta_h \nabla h, L_p((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\}))) \\
\frac{1}{\rho_1}(\lambda^t_j - \mathcal{P}_\lambda(\lambda^t_j + \rho_1 \nabla \lambda, L_p((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\}))) \\
\frac{1}{\rho_2}(\phi^t_j - \mathcal{P}_\phi(\phi^t_j + \rho_2 \nabla \phi, L_p((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\})))
\end{bmatrix}, \quad (21)
$$

where $\nabla G^t$ is the simplified form of $\nabla G((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\})$.

**Definition 2** ($\varepsilon$-stationary point) $((w^t_j), z^t, h^t, \{\lambda^t_j\}, \{\phi^t_j\})$ is an $\varepsilon$-stationary point ($\varepsilon \geq 0$) of a differentiable function $L_p$, if $||\nabla G^t|| \leq \varepsilon$. $T(\varepsilon)$ is the first iteration index such that $||\nabla G^t|| \leq \varepsilon$, i.e.,

$T(\varepsilon) = \min\{ t \mid ||\nabla G^t|| \leq \varepsilon \}$.

**Assumption 1** (Smoothness/Gradient Lipschitz) $L_p$ has Lipschitz continuous gradients. We assume that there exists $L > 0$ satisfying
\[ \|\nabla_{\theta} L_{p}(\{w_j\}, z, h, \{\lambda_t\}, \{\phi_j\}) - \nabla_{\theta} L_{p}(\{\hat{w}_j\}, z, \hat{h}, \{\hat{\lambda}_t\}, \{\hat{\phi}_j\})\| \leq L\|\hat{w}_{\text{cat}} - w_{\text{cat}}; z - \hat{z}; h - \hat{h}; \lambda_{\text{cat}} - \hat{\lambda}_{\text{cat}}; \phi_{\text{cat}} - \hat{\phi}_{\text{cat}}\|, \]

where \(\theta \in \{\{w_j\}, z, h, \{\lambda_t\}, \{\phi_j\}\}\) and \(\cdot|\cdot\) represents the concatenation. \(w_{\text{cat}} - \hat{w}_{\text{cat}} = [w_1 - \hat{w}_1; \cdots; w_N - \hat{w}_N] \in \mathbb{R}^{P^N}, \lambda_{\text{cat}} - \hat{\lambda}_{\text{cat}} = [\lambda_1 - \hat{\lambda}_1; \cdots; \lambda_{|A^t|} - \hat{\lambda}_{|A^t|}] \in \mathbb{R}^{|A^t|}, \phi_{\text{cat}} - \hat{\phi}_{\text{cat}} = [\phi_1 - \hat{\phi}_1; \cdots; \phi_N - \hat{\phi}_N] \in \mathbb{R}^{P^N}.

**Assumption 2 (Boundedness)** Before obtaining the \(\varepsilon\)-stationary point (i.e., \(t \leq T(\varepsilon) - 1\)), we assume variables in master satisfy that \(\|z^{t+1} - z^t\|^2 + \|h^{t+1} - h^t\|^2 + \sum_{l} |\lambda_{t+1}^l - \lambda_{t}^l|^2 \geq \vartheta\), where \(\vartheta > 0\) is a relative small constant. The change of the variables in master is upper bounded within \(t\) iterations:

\[ \|z^{t} - z^{t-k}\|^2 \leq \tau k_1 \vartheta, \quad \|h^{t} - h^{t-k}\|^2 \leq \tau k_1 \vartheta, \quad \sum_{l} |\lambda_{t}^l - \lambda_{t-k}^l|^2 \leq \tau k_1 \vartheta, \forall 1 \leq k \leq \tau, \]

where \(k_1 > 0\) is a constant.

**Setting 1 (Bounded \(|A^t|\) \& \(|A^t| \leq M, \forall t\), i.e., an upper bound is set for the number of cutting planes.**

**Setting 2 (Setting of \(c_1, c_2\))** \(c_1^t = \frac{1}{\rho_1(t+1)^\frac{3}{2}} \geq \varrho_1\) and \(c_2^t = \frac{1}{\rho_2(t+1)^\frac{3}{2}} \geq \varrho_2\) are nonnegative non-increasing sequences, where \(\varrho_1\) and \(\varrho_2\) are positive constants and meet \(M\varrho_1^2 + N\varrho_2^2 \leq \frac{\varepsilon^2}{4}T\).

**Theorem 1 (Iteration complexity)** Suppose Assumption 1 and 2 hold. We set \(\eta_{\text{mw}} = \eta_{\text{hw}} = \frac{1}{\rho_1^2},\) and \(\eta_{\text{bw}} = \frac{1}{\rho_2^2}\). And we set constants \(\rho_1 < \min\left\{\frac{2}{L + 2\varepsilon + \frac{2}{\varrho_1^2}}, \frac{1}{15k_1N}\right\}\) and \(\rho_2 \leq \frac{2}{L + 2\varepsilon + \frac{2}{\varrho_2^2}}\), respectively. For a given \(\varepsilon\), we have:

\[ T(\varepsilon) \sim O\left(\max\left\{\frac{4M\rho_1^2}{\rho_1^2 + 4N\rho_2^2} + \frac{4N\rho_2^2}{\rho_1^2 + 4N\rho_2^2} 1 \cdot \frac{1}{\varepsilon^2} \cdot \frac{1}{\varepsilon^2} \cdot \frac{1}{\varepsilon^2} \right\}\right), \]

where \(\sigma, \gamma, \tau, k, d, d_0, d_0, T_1\) are constants. The detailed proof is given in Appendix A.

There exists a wide array of works regarding the convergence analysis of various algorithms for nonconvex/convex optimization problems involved in machine learning [25, 53]. Our analysis, however, differs from existing works in two aspects. First, we solve the non-convex PD-DRO in an asynchronous distributed manner. Compared to solving the non-convex PD-DRO in a centralized manner, solving it in an asynchronous distributed manner poses significant challenges in algorithm design and convergence analysis. Secondly, we do not assume the inner problem can be solved nearly optimally for each outer iteration, which is numerically difficult to achieve in practice [4]. Instead, ASPIRE-EASE is single loop and involves simple gradient projection operation at each step.

### 6 Experiment

In this section, we conduct experiments on four real-world datasets to assess the performance of the proposed method. Specifically, we evaluate the robustness against data heterogeneity, robustness against malicious attacks and efficiency of the proposed method. Ablation study is also carried out to demonstrate the excellent performance of ASPIRE-EASE.

#### 6.1 Datasets and Baseline Methods

We compare the proposed ASPIRE-EASE with baseline methods based on SHL [20], Person Activity [26], Single Chest-Mounted Accelerometer (SM-AC) [9] and Fashion MNIST [51] datasets. The baseline methods include Indj (learning the model from an individual worker \(j\)), MixXeven (the model from all workers with even weights using ASPIRE), FedAvg [33], AFL [35] and DRFA-Prox [16]. The detailed descriptions of datasets and baselines are given in Appendix C.

In our empirical studies, since the downstream tasks are multi-class classification, the cross entropy loss is used on each worker (i.e., \(L_j(\cdot), \forall j\)). For SHL, Person Activity, and SM-AC datasets, we adopt the deep multilayer perceptron [49] as the base model. And we use the same logistic regression model as in [35, 16] for Fashion MNIST dataset. The base models are trained with SGD. More details are given in Appendix C. Following related works in this direction [41, 35, 16], worst case performance are reported for the comparison of robustness. Specifically, we use \(\text{Acc}_w\) and \(\text{Loss}_w\) to represent the worst case test accuracy and training loss (i.e., the test accuracy and training loss on the worker with worst performance), respectively. We also report the standard deviation \(\text{Stdev}\) of
Table 1: Performance comparisons based on $\text{Acc}_w$ (%) $\uparrow$, $\text{Loss}_w$ $\downarrow$ and $\text{Std}$ $\downarrow$ (↑ and ↓ respectively denote higher scores represent better performance and lower scores represent better performance). The boldfaced digits represent the best results, “−” represents not available.

<table>
<thead>
<tr>
<th>Model</th>
<th>SHL</th>
<th>Person Activity</th>
<th>SC-MA</th>
<th>Fashion MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\text{Acc}_w$ $\uparrow$</td>
<td>$\text{Loss}_w$ $\downarrow$</td>
<td>$\text{Std}$ $\downarrow$</td>
<td>$\text{Acc}_w$ $\uparrow$</td>
</tr>
<tr>
<td>max{Ind$_j$}</td>
<td>99.28±1.65</td>
<td>29.4</td>
<td>49.38±1.08</td>
<td>17.5</td>
</tr>
<tr>
<td>Mix$_{Even}$</td>
<td>69.87±3.10</td>
<td>0.00±0.016</td>
<td>1.65±0.017</td>
<td>3.00</td>
</tr>
<tr>
<td>FedAvg [33]</td>
<td>69.96±3.07</td>
<td>0.80±0.023</td>
<td>1.54±0.019</td>
<td>3.13</td>
</tr>
<tr>
<td>AFL [35]</td>
<td>78.11±0.99</td>
<td>58.36±0.37</td>
<td>0.81±0.014</td>
<td>0.99</td>
</tr>
<tr>
<td>DRFA-Prox [20]</td>
<td>78.34±1.66</td>
<td>0.52±0.034</td>
<td>1.09±0.037</td>
<td>1.26</td>
</tr>
<tr>
<td>ASPIRE-EASE</td>
<td>98.51±0.13</td>
<td>8.51±0.19</td>
<td>1.05±0.010</td>
<td>0.82</td>
</tr>
<tr>
<td>ASPIRE-EASE$_{r}$</td>
<td>98.94±0.27</td>
<td>59.54±0.21</td>
<td>1.05±0.016</td>
<td>0.79</td>
</tr>
</tbody>
</table>

[Acc$_1$, · · ·, Acc$_N$] (the test accuracy on every worker). In the experiment, $S$ is set as 1, that means the master will make an update once it receives a message. Each experiment is repeated 10 times, both mean and standard deviations are reported. We implement our model with PyTorch and conduct all the experiments on a server with two TITAN V GPUs.

6.2 Results

Robustness against Data Heterogeneity. We first assess the robustness of the proposed ASPIRE-EASE by comparing it with baseline methods when data are heterogeneously distributed across different workers. Specifically, we compare the $\text{Acc}_w$, $\text{Loss}_w$ and $\text{Std}$ of different methods on all datasets. The performance comparison results are shown in Table 1. In this table, we can observe that max{Ind$_j$}, which represents the best performance of individual training over all workers, exhibits the worst robustness on SHL, Person Activity, and SC-MA. This is because individual training (max{Ind$_j$}) only learns from the data in its local worker and cannot generalize to other workers due to different data distributions. Note that max{Ind$_j$} is unavailable for Fashion MNIST since each worker only contains one class of data and cross entropy loss cannot be used in this case. max{Ind$_j$} also does not have $\text{Loss}_w$, since Ind$_j$ is trained only on individual worker $j$. The FedAvg and Mix$_{Even}$ exhibit better performance than max{Ind$_j$} since they consider the data from all workers. Nevertheless, FedAvg and Mix$_{Even}$ only assign the fixed weight for each worker. AFL is more robust than FedAvg and Mix$_{Even}$ since it not only utilizes the data from all workers but also considers optimizing the weight of each worker. DRFA-Prox outperforms AFL since it also considers the prior distribution and regards it as a regularizer in the objective function. Finally, we can observe that the proposed ASPIRE-EASE shows excellent robustness, which can be attributed to two factors: 1) ASPIRE-EASE considers data from all workers and can optimize the weight of each worker; 2) compared with DRFA-Prox which uses prior distribution as a regularizer, the prior distribution is incorporated within the constraint in our formulation (Eq. 4), which can be leveraged more effectively. And it is seen that ASPIRE-EASE can perform periodic communication since ASPIRE-EASE$_{per}$, which represents ASPIRE-EASE with periodic communication, also has excellent performance.

Within ASPIRE-EASE, the level of robustness can be controlled by adjusting $\Gamma$. Specially, when $\Gamma = 0$, we obtain a nominal optimization problem in which no adversarial distribution is considered. The size of the uncertainty set will increase with $\Gamma$ (when $\Gamma \leq N$), which enhances the adversarial robustness of the model. As shown in Figure 1, the robustness of ASPIRE-EASE can be gradually enhanced when $\Gamma$ increases. More results are available in Figure C of Appendix C.

Robustness against Malicious Attacks. To assess the model robustness against malicious attacks, malicious workers with backdoor attacks [14, 48], which attempt to mislead the model training process, are added to the distributed system. Following [14], we report the success attack rate of backdoor attacks for comparison. It can be calculated by checking how many instances in the backdoor dataset can be misled and categorized into the target labels. Lower success attack rates indicate more robustness against backdoor attacks. The comparison results are summarized in Table 2 and more detailed settings of backdoor attacks are available in Appendix C. In Table 2 we observe that AFL can be attacked easily since it could assign higher weights to malicious workers. Compared to AFL, FedAvg and Mix$_{Even}$ achieve relatively lower success attack rates since they assign equal weights to the malicious workers and other workers. DRFA-Prox can achieve even lower success attack rates since it can leverage the prior distribution to assign lower weights for malicious workers. The proposed ASPIRE-EASE achieves the lowest success attack rates since it can leverage the prior distribution more effectively. Specifically, it will assign lower weights to malicious workers with tight theoretical guarantees.
Figure 1: $\Gamma$ control the degree of robustness (worst case performance in the problem) on (a) Person Activity, (b) SC-MA datasets.

Figure 2: Comparison of the convergence time on worst case worker on (a) Person Activity, (b) SC-MA datasets.

Figure 3: Comparison of ASPIRE-CP and ASPIRE-EASE regarding the number of cutting planes on (a) Person Activity, (b) SC-MA datasets.

**Efficiency.** In Figure 2, we compare the convergence speed of the proposed ASPIRE-EASE with AFL and DRFA-Prox by considering different communication and computation delays for each worker. The proposed ASPIRE-EASE has two variants, ASPIRE-CP (ASPIRE with cutting plane method), ASPIRE-EASE(-) (ASPIRE-EASE without asynchronous setting). More results are available in Figure C3 of Appendix C. Based on the comparison, we can observe that the proposed ASPIRE-EASE generally converges faster than baseline methods and its two variants. This is because 1) compared with AFL, DRFA-Prox, and ASPIRE-EASE(-), ASPIRE-EASE is an asynchronous algorithm in which the master updates its parameters only after receiving the updates from active workers instead of all workers; 2) unlike DRFA-Prox, the master in ASPIRE-EASE only needs to communicate with active workers once per iteration; 3) compared with ASPIRE-CP, ASPIRE-EASE utilizes active set method instead of cutting plane method, which is more efficient. It is seen from Figure 2 that, the convergence speed of ASPIRE-EASE mainly benefits from the asynchronous setting.

**Ablation Study.** For ASPIRE, compared with cutting plane method, EASE is more efficient since it considers removing the inactive cutting planes. To demonstrate the efficiency of EASE, we firstly compare ASPIRE-EASE with ASPIRE-CP concerning the number of cutting planes used during the training. In Figure 3, we can observe that ASPIRE-EASE uses fewer cutting planes than ASPIRE-CP, thus is more efficient. The convergence speed of ASPIRE-EASE and ASPIRE-CP in Figure 2 also suggests that ASPIRE-EASE converges much faster than ASPIRE-CP. More results are available in Figure C3 and C4, Appendix C.

### 7 Conclusion

In this paper, we present ASPIRE-EASE method to effectively solve the distributed distributionally robust optimization problem with non-convex objectives. In addition, $CD$-norm uncertainty set has been proposed to effectively incorporate the prior distribution into the problem formulation, which allows for flexible adjustment of the degree of robustness of DRO. Theoretical analysis has also been conducted to analyze the convergence properties and the iteration complexity of ASPIRE-EASE. ASPIRE-EASE exhibits strong empirical performance on multiple real-world datasets and is effective in tackling DRO problems in a fully distributed and asynchronous manner. In the future work, more uncertainty sets could be designed for our framework and more update rule for variables in ASPIRE could be considered.
Acknowledgments and Disclosure of Funding

The work of Yang Jiao and Kai Yang was supported in part by the Fundamental Research Funds for the Central Universities of China, in part by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), in part by the National Natural Science Foundation of China under Grant 61771013, and in part by the Fundamental Research Funds of Shanghai Jiading District.

References


Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes] See Section 1
   (b) Did you describe the limitations of your work? [Yes] See Section 7
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] There is no potential negative societal impact of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 5
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A and B

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The references of the data used in this paper are added in Section 6.1
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Section C.2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 6
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 6.1

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]