Data-driven distributionally robust optimization over a network via distributed semi-infinite programming

Ashish Cherukuri, Alireza Zolanvari, Goran Banjac, and Ashish R. Hota

Abstract—This paper focuses on solving a data-driven distributionally robust optimization problem over a network of agents. The agents aim to minimize the worst-case expected cost computed over a Wasserstein ambiguity set that is centered at the empirical distribution. The samples of the uncertainty are distributed across the agents. Our approach consists of reformulating the problem as a semi-infinite program and then designing a distributed algorithm that solves a generic semi-infinite problem that has the same information structure as the reformulated problem. In particular, the decision variables are formed by both local ones that agents are free to optimize over and global ones where they need to agree on. Our distributed algorithm is an iterative procedure that combines the notions of distributed ADMM and the cutting-surface method. We show that the iterates converge asymptotically to a solution of the distributionally robust problem to any pre-specified accuracy. Simulations illustrate our results.

I. INTRODUCTION

Various machine learning problems can be cast as stochastic optimization problems where the decision-maker intends to minimize an expected cost (termed commonly as the loss function). To solve this problem, the decision-maker usually has access to samples of the uncertainty modeled in the problem. In several applications, this sample set can be small, corrupted by noise, and distributed across several processors/agents. To handle the first two challenges, recent works have proposed solving a suitably defined data-driven distributionally robust optimization (DRO) problem [1]. The third challenge is tackled by designing distributed algorithms, where agents iteratively compute decisions using only local communication, see e.g., [2], [3]. In this paper, we wish to address all the three challenges together. We study data-driven DRO problem in a network setting where the data regarding the uncertainty is spread across multiple agents. We then design a distributed algorithm to solve it.

1) Literature review: A recent addition to the class of data-driven methods for solving stochastic optimization problems is data-driven DRO, where the decision-maker minimizes the worst-case expected cost computed over an ambiguity set of distributions that are close to the empirical distribution defined using data. Such an approach ensures that the resulting decision has better out-of-sample performance. Popular choices for metrics that define this closeness are KL-divergence [4], $\phi$-divergence [5], Prohorov metric [6], and the Wasserstein metric [1], [7]. In this paper, we focus on the latter; see [8] for a survey of main ideas.

Much of DRO literature focuses on centralized solution algorithms. In this work, we instead look into the distributed setting, where the data regarding the samples are not stored at a central location, but are instead distributed over multiple agents. The same setup was recently studied in [9], where a distributed primal-dual algorithm was proposed that solves the DRO problem. However, this work considered strong assumptions on the objective function (concavity with respect to the uncertainty over an identified domain) that need not hold for several statistical learning problems. In this work, we consider general objective functions that are locally Lipschitz and convex in the decision variable.

2) Setup and contributions: Our approach builds on [10], where the DRO problem is formulated as a semi-infinite program. Noting the particular form of constraints and the objective of this problem, we form a general semi-infinite program that captures the particular case of DRO. In particular, the objective function of the problem has both global and separable local parts, and the constraints depend on both local and global decision variables. The way samples are distributed over the network, each agent is able to compute a subset of semi-infinite constraints. For such a problem structure, we design the distributed cutting-surface ADMM algorithm that solves the problem to any pre-specified accuracy. The algorithm combines features from the distributed ADMM and the cutting-surface algorithm. We adapt our general algorithm to the DRO problem and establish convergence.

We note that our algorithm solves a general semi-infinite problem in a distributed manner. The work [11] also studies this problem and proposes a cutting-plane based distributed algorithm. However, therein, agents need to share cuts of the feasibility set and each cut is defined by a vector of the size of the primal variable. In our DRO problem, the size of the primal variable is of the order of the data points and is at least of the order of the number of agents. Thus, to implement their algorithm, agents need to communicate in each round a variable having size of the order of agents. This is not scalable. We also note here that to determine the cuts in [11], one needs to solve a maximization problem (defining the semi-infinite constraint) exactly. Since this problem is nonconvex in our setup, we encounter another hurdle. We overcome these roadblocks by focusing on cutting-surface algorithm [10] and developing a distributed version of it.

Our work is related to the broad field of federated learning, where machine learning problems are solved in a decentralized (with the presence of a central computing node)
or distributed manner [12]. A recent work [13] in this area looks into linear distributional shifts in data and designs a decentralized algorithm to solve this problem. In contrast, we consider more general distributional shifts and our algorithm does not require a central computing agent. Due to space reasons, all proofs are omitted and can be found in [14].

**Notation:** Let $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{Z}_{\geq 1}$ denote the set of real, nonnegative real, and positive integer numbers. For $n \in \mathbb{Z}_{\geq 1}$, we denote $[n] := \{1, \ldots, n\}$. The 2-norm in $\mathbb{R}^n$ is represented by $\| \cdot \|$. We let $I_n = (1, \ldots, 1) \in \mathbb{R}^n$ be the vector of ones. The $N$-fold Cartesian product $S \times \cdots \times S$ of a set $S$ is denoted as $S^N$.

II. PROBLEM STATEMENT

Consider $n \in \mathbb{Z}_{\geq 1}$ agents communicating over an undirected connected graph $G := (V, E)$. The set of vertices are enumerated as $V := [n]$ and $E \subset V \times V$ represents the set of edges. Each agent $i \in [n]$ can send and receive information from its neighbors $\mathcal{N}_i := \{ j \in V \mid (i, j) \in E \}$. Let $f : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}$: $(x, \xi) \mapsto f(x, \xi)$ be a locally Lipschitz objective function. Assume that for any $\xi \in \mathbb{R}^m$, the map $x \mapsto f(x, \xi)$ is convex. We are interested in solving the following data-driven distributionally robust optimization (DRO) problem in a distributed manner over the network

\[
J_{\text{DRO}} := \inf_{x \in \mathcal{X}} \sup_{Q \in \mathcal{P}_L} \mathbb{E}_Q[f(x, \xi)],
\]

where $\mathcal{X} \subset \mathbb{R}^d$ is a compact convex set, $\xi \sim Q$ is the random variable, $\mathbb{E}_Q[\cdot]$ denotes expectation with respect to the distribution $Q$, and $\mathcal{P}_L$ is the set of distributions supported over a compact set $\Xi \subset \mathbb{R}^m$. Assume that $f$ is bounded on $\mathcal{X} \times \Xi$. We refer to $\mathcal{P}_L$ as the ambiguity set and define it using data and the Wasserstein metric. In particular, assume we have $L \in \mathbb{Z}_{\geq 1}$ samples of the uncertainty $\xi$, collected in the set $\hat{\Xi} := \{\hat{\xi}_1, \hat{\xi}_2, \ldots, \hat{\xi}_L\}$. These samples need not be drawn from any particular distribution, however if they are sampled in an i.i.d manner from an underlying distribution, then we have desirable properties on the optimizer of the DRO problem (1), as illustrated in [1]. Let $P_L := \frac{1}{L} \sum_{l=1}^{L} \delta_{\hat{\xi}_l}$ be the empirical distribution, that is, the distribution defined as the summation of delta functions of equal mass placed on each of the samples. Then, the ambiguity set $\mathcal{P}_L$ is given as

\[
\mathcal{P}_L := \{Q \in \mathcal{M} \mid d_W(Q, P_L) \leq \theta\},
\]

where $\mathcal{M}$ is the set of distributions supported on $\Xi \subset \mathbb{R}^m$, $d_W$ is the 1-Wasserstein metric on the space of distributions, and $\theta > 0$ is the radius. Note that given two distributions $P_1$ and $P_2$ supported on $\Xi$, the 1-Wasserstein distance between them (defined using the Euclidean norm) is given as

\[
d_W(P_1, P_2) := \min_{H \in \mathcal{H}(P_1, P_2)} \left\{ \int_{\Xi \times \Xi} \|\xi - \omega\| \mathrm{d}H(\xi, \omega) \right\},
\]

where $\mathcal{H}(P_1, P_2)$ is the set of all distributions on $\Xi \times \Xi$ with marginals $P_1$ and $P_2$.

The objective of the paper is to design an algorithm for a group of agents to solve the DRO problem (1) in a distributed manner. To this end, we assume that each agent only knows a certain number (at least one) of samples from the set $\hat{\Xi}$.

Denoting the data available to agent $i$ by the set $\hat{\Xi}_i \subset \hat{\Xi}$, we assume that $\hat{\Xi}_i \cap \hat{\Xi}_j = \emptyset$ for all $i, j \in [n]$ and $\hat{\Xi} = \bigcup_{i=1}^{n} \hat{\Xi}_i$. Each agent knows the function $f_i$, the sets $\mathcal{X}_i$ and $\Xi_i$, and the radius of the ambiguity set $\theta > 0$. The challenge comes from the fact that each agent only has limited access to the data.

1) **Motivating example:** We now briefly motivate our problem setup. Consider a statistical learning problem where we aim to minimize the expected value of a loss function

\[
\min_{x \in \mathcal{X}} \mathbb{E}_{P[f(x, \xi)]},
\]

where $x$ is the decision variable and $\xi$ is the uncertainty that stands for the input-output data. Roughly speaking, the value $f(x, \xi)$ encodes the ability of $x$ to predict the relationship between the input-output pair $\xi$. A lower value of $f(x, \xi)$ means higher prediction accuracy. Thus, one seeks to minimize the expected value of this uncertain function.

Usually, the distribution $P$ is unknown and instead a data set $\hat{\Xi} = \{\hat{\xi}_1, \ldots, \hat{\xi}_L\}$ possibly sampled in an i.i.d manner from $P$ is available. Then, the expected value in (2) is replaced with the sample average and one seeks the solution of

\[
\min_{x \in \mathcal{X}} \frac{1}{L} \sum_{l=1}^{L} f(x, \hat{\xi}_l).
\]

Now examine the network setup as explained above where samples $\hat{\Xi}$ are distributed across agents in the network. Then, the minimization problem (3) is equivalently written as the distributed optimization problem

\[
\min_{x \in \mathcal{X}} \sum_{i=1}^{n} \phi_i(x),
\]

where $\phi_i$ is known to agent $i$ and it takes the form $\phi_i(x) = \frac{1}{L} \sum_{l \in \mathcal{U}_i} \delta_{\hat{\xi}_l}(x, \hat{\xi}_l)$. This formulation fits naturally into the traditional distributed optimization setup where agents across the network seek to minimize the sum of cost functions [2], [3]. While the solution of (4) has desirable statistical guarantees, such as out-of-sample performance, when the dataset is large and is sampled from $P$, it often fails to demonstrate such properties when the dataset is small or encounters distributional shifts [1]. The solution of the DRO problem (1) fixes these issues. Thus, one would like to solve (1) in a distributed manner. Unfortunately, the problem (1) cannot be written equivalently in a similar form as (4), and so we need to rethink the design of a distributed algorithm. This motivates our current work.

2) **Semi-infinite reformulation and cutting-surface method:** Note that (1) consists of an infinite-dimensional inner optimization problem. Following [10], we reformulate this as a semi-infinite program. Then, problem (1) can be solved equivalently by finding a solution of

\[
J^*_{\text{ref}} := \begin{cases} 
\min_{x, \omega, v} & 1_{L}^T v + L \theta s \\
\text{s.t.} & v \in \mathbb{R}^L, \quad s \in \mathbb{R}_{\geq 0}, \quad x \in \mathcal{X}, \\
& f(x, \xi) - \nu_t - s \|\xi - \hat{\xi}_t\| \leq 0, \forall \xi \in \Xi, \quad t \in [L],
\end{cases}
\]

where $\nu_t$ represents the $t$-th component of the vector $v$ and we recall that $1_L$ is the vector of ones. The equivalence here means that $x$ is a solution of (1) if and only if there exists $(v, s)$ such that $(x, v, s)$ is a solution of (5). However, it is
important to note that the optimal value of (5) is $L$ times that of (1). That is, $J_{\text{ref}}^* = LJ_{\text{DRO}}^*$.

The above optimization problem is a particular case of a general convex semi-infinite program that can be written as

$$
\min_{x \in \mathcal{X}} \left\{ \varphi(x) \mid g(x, \xi) \leq 0, \text{ for all } \xi \in \Xi \right\},
$$

where $\mathcal{X} \subset \mathbb{R}^p$ and $\Xi \subset \mathbb{R}^m$ are compact, $\mathcal{X}$ is convex, $\varphi : \mathbb{R}^d \to \mathbb{R}$ is convex, $g$ is continuous and bounded on $\mathcal{X} \times \Xi$, and $g(\cdot, \xi) : \mathbb{R}^d \to \mathbb{R}$ is convex for every $\xi \in \Xi$. Semi-infinite programs are in general computationally challenging to solve, see the survey [15]. We specifically focus our attention on the cutting-surface algorithm given in [10] as it naturally allows a distributed implementation. The cutting-surface algorithm consists of the following steps:

(i) Start with $x^0 \in \mathcal{X}$ and a set $\Xi^0 = \emptyset$ that maintains cuts. A cut is a point in the uncertainty set $\Xi$.

(ii) At iteration $k$, given a finite set of cuts $\Xi^k \subset \Xi$, solve

$$
\min_{x \in \mathcal{X}} \{ \varphi(x) \mid g(x, \xi) \leq 0, \text{ for all } \xi \in \Xi^k \},
$$

and store the solution as $x^{k+1}$.

(iii) Find a new cut $\xi^{k+1}$ corresponding to $x^{k+1}$ that satisfies two properties. First, it is an $\epsilon$-2-optimal solution for the problem $\max_{\xi \in \Xi} g(x^{k+1}, \xi)$. That is,

$$
g(x^{k+1}, \xi^{k+1}) \geq \max_{\xi \in \Xi} g(x^{k+1}, \xi) - \epsilon/2.
$$

Second, $g(x^{k+1}, \xi^{k+1}) > \epsilon/2$. Note that a cut $\xi^{k+1}$ satisfying the above two properties exists as long as $x^{k+1}$ violates the robust constraint by at least $\epsilon$. That is, $\max_{\xi \in \Xi} g(x^{k+1}, \xi) > \epsilon$. Once the cut $\xi^{k+1}$ is found, update the cut set as $\Xi^{k+1} = \Xi^k \cup \{\xi^{k+1}\}$ and move to the next iteration $k + 1$.

(iv) If no such cut is found, then declare $x^{k+1}$ as the desired solution and terminate the procedure.

As established in [10], this algorithm converges in a finite number of steps to an $\epsilon$-feasible solution $x^* \in \mathcal{X}$ satisfying $\varphi(x^*) \leq \varphi^*$, where $\varphi^*$ is the optimal value of (6). Here $\epsilon$-feasibility means that the violation of the robust constraint is no more than $\epsilon$, i.e., $\max_{\xi \in \Xi} g(x^*, \xi) \leq \epsilon$. Our aim is to develop a distributed version of the above algorithm.

In the following section, we design a distributed algorithm that solves a generic semi-infinite program, for which (5) is a special case. Subsequently, in Section IV, we adapt this distributed algorithm to solve (5).

III. DISTRIBUTED CUTTING-SURFACE ADMM FOR CONVEX SEMI-INFINITE PROGRAM

In this section, we provide a distributed algorithm to solve the following semi-infinite optimization problem

$$
\min_{\{y_i, z_i\}_{i=1}^n} \sum_{i=1}^n (\varphi_i(z_i) + h_i(y_i))
\quad \text{s.t.} \quad y_i \in \mathcal{Y},
\quad z_i \in \mathcal{Z}_i, \quad \forall i \in [n],
\quad g_i(y_i, z_i, \xi) \leq 0, \quad \forall \xi \in \Xi, \quad i \in [n].
$$

(7)

Here, functions $\{\varphi_i, h_i\}_{i=1}^n$ are convex, sets $\mathcal{Y} \subset \mathbb{R}^d$, $\mathcal{Z}_i \subset \mathbb{R}^{p_i}, i \in [n]$ are convex compact, and $\Xi \subset \mathbb{R}^m$ is compact. Further, $\varphi_i$ and $h_i$ are bounded on $\mathcal{Z}_i$ and $\mathcal{Y}$, respectively, and $g_i$ is locally Lipschitz and convex in $(y_i, z_i)$ for any fixed $\xi \in \Xi$. The problem (7) is convex under these conditions. Before we move on to the distributed algorithm, we comment about the information structure and the connection to the reformulated DRO problem (5). We assume that each agent $i$ only knows $\varphi_i, h_i, g_i, \mathcal{Y}, \mathcal{Z}_i, \text{ and } \Xi$. The variable $z_i \in \mathbb{R}^{p_i}$ is a local decision variable for agent $i$ while $y \in \mathbb{R}^d$ is a global variable that all agents need to agree on. Note that the semi-infinite constraint for each agent $i$, $g_i(y, z_i, \xi) \leq 0$ for all $\xi \in \Xi$, depends on both the local and the global variable. Comparing the two, one observes that problem (5) is a special case of (7) with decision variables $z_i = (\nu)_{\xi, \xi \in \Xi} \in \mathbb{R}^{||\Xi||}$, $y = (x, s) \in \mathbb{R}^{d+1}$, and objective functions $\varphi_i(z_i) = \mathbb{I}_{\Xi} z_i$, $h_i(y) = |\Xi|/d$. The parallelism between constraints is more involved and we will address this point in the subsequent section.

Our distributed algorithm builds on a class of distributed ADMM algorithms where auxiliary variables are used for each edge of the network to impose consensus, see e.g., [16]–[18]. Next, we overview the derivation of such an algorithm. Let $\{u_{ij}\}_{(i,j) \in \mathcal{E}}$ be local estimates of $y$ maintained by agents and $\{\tilde{u}_{ij}\}_{(i,j) \in \mathcal{E}}$ be the auxiliary variables that will ensure consensus over the global decision variable $y$. Using these variables, one can write (7) equivalently as

$$
\min_{\{y_{i,j}, z_{i,j}\}_{i,j=1}^n} \sum_{i=1}^n \left( \varphi_i(z_i) + h_i(y_i) \right)
\quad \text{s.t.} \quad y_i \in \mathcal{Y}, \quad \forall i \in [n], \quad z_i \in \mathcal{Z}_i, \quad \forall i \in [n],
\quad g_i(y_i, z_i, \xi) \leq 0, \quad \forall \xi \in \Xi, \quad i \in [n],
\quad y_{i,j} = u_{ij}, \quad \forall i \in [n], \quad (i,j) \in \mathcal{E},
\quad y_{j,i} = u_{ij}, \quad \forall j \in [n], \quad (i,j) \in \mathcal{E}.
$$

Note that constraints $y_{i,j} = u_{ij}$ is the above formulation impose consensus on $y$. Next, we derive a distributed ADMM algorithm using the reformulation (8). The derivation is given for completeness and can be found both in [16] and [17]. In contrast with the optimization problems considered in these works, the problem (8) has semi-infinite constraints for each agent and we only require consensus over a subset of primal variables. For now, we do not deal with the computational issues related to semi-infinite constraints. We will revisit this point once we have derived the distributed ADMM. The augmented Lagrangian corresponding to the problem (8) is

$$
L_{\rho}(Y, Z, U, \Lambda, \Gamma)
:= \sum_{i=1}^n \left[ \varphi_i(z_i) + h_i(y_i) + \sum_{j \in N_i} \left( \lambda_{ij}^T (y_j - u_{ij}) - \frac{\rho}{2} ||y_j - u_{ij}||^2 \right) \right],
$$

where $\rho > 0$ is a parameter, $\mathcal{Y} := \{y_i\}_{i \in [n]}$, $\mathcal{Z} := \{z_i\}_{i \in [n]}$, and $U := \{u_{ij}\}_{(i,j) \in \mathcal{E}}$ are the primal variables and $\Lambda := \{\lambda_{ij}\}_{(i,j) \in \mathcal{E}}$ and $\Gamma := \{\gamma_{ij}\}_{(i,j) \in \mathcal{E}}$ are the dual variables corresponding to the consensus constraints. Note that when forming the Lagrangian we have only dualized the consensus constraints. Denoting the feasibility set for each agent $i$ as $\mathcal{F}_i := \{(y_i, z_i) \mid y_i \in \mathcal{Y}_i, z_i \in \mathcal{Z}_i, g_i(y_i, z_i, \xi) \leq 0, \forall \xi \in \Xi \}$, the distributed ADMM consists of the following updates

$$
(y_{i,j}^{k+1}, z_{i,j}^{k+1}) \leftarrow \arg \min_{(y_i, z_i) \in \mathcal{F}_i} \varphi_i(z_i) + h_i(y_i)
$$

4772
One could implement the above steps using only local communication. Nonetheless, with a particular initialization, some of the variables can be eliminated and the required communication and computation at each iteration can be reduced. Specifically, we do the following (see [16] and [17] for details): (a) write the explicit expression for $u_{ij}^{k+1}$ using the first-order condition of optimality, (b) use this expression in the summation of $\lambda_{ij}^{k+1}$ and $\gamma_{ij}^{k+1}$ over the entire network to get $\lambda_{ij}^{k+1} + \gamma_{ij}^{k+1} = 0$, (c) initialize $\lambda_{ij}^{0} = \gamma_{ij}^{0} = 0$ and use the equality derived in (b) to define an update scheme for a new variable $p_{ij}^{k+1} := 2 \sum_{j \in N_i} \lambda_{ij}^{k+1}$, and (d) rewrite the primal update $\left(\xi_{i}^{k+1}, y_{i}^{k+1}\right)$ using the variable $p_{ij}^{k+1}$. This simplification reduces the ADMM into the update scheme

$$
y_{i}^{k+1} \leftarrow y_{i}^{k} + \rho \sum_{j \in N_i} \left(\lambda_{ij}^{k} - y_{j}^{k}\right),$$
$$
\left(\xi_{i}^{k+1}, z_{i}^{k+1}\right) \leftarrow \arg \min_{(y_{i}, z_{j} \in F_{i}^{k+1}}} \left\{ \sum_{j \in N_i} \left(\frac{1}{2} \left\|y_{i}^{k} - y_{j}^{k}\right\|^2 + \rho \left\|z_{i}^{k} - y_{i}^{k}\right\|^2 \right) \right\}.
$$

The above scheme can be implemented in a distributed manner where each agent $i$ at iteration $k$, broadcasts to its neighbors the variable $y_{i}^{k}$ and updates ($p_{ij}^{k}, y_{ij}^{k}, z_{ij}^{k}$). However, this scheme requires solving a semi-infinite program in (9b) at each iteration due to the definition of $F_{i}$. To overcome this limitation, we present the distributed cutting-surface ADMM method, formally given in Algorithm 1, which combines the features of distributed ADMM and the cutting-surface algorithm explained in Section II.

[Informal description of Algorithm 1]: In each iteration $k$ of this procedure, each agent $i$ executes an ADMM step (Lines 3 and 4). Different from (9), the feasibility set $F_{i}^{k}$ only imposes a finite number of constraints on the primal variables $(y_{i}, z_{i})$, see Line 11. Each such constraint is defined by a point in the set $\Xi_{i}^{k}$ and is referred to as a cut. Line 5 finds an $\epsilon$-2-approximate maximizer $\xi_{i}^{k+1}$ of the constraint at the primal optimizer $(y_{i}^{k+1}, z_{i}^{k+1})$ that was obtained in the ADMM step. If the constraint value at $(y_{i}^{k+1}, z_{i}^{k+1}, \xi_{i}^{k+1})$ is higher than $\epsilon/2$, then $\xi_{i}^{k+1}$ is added as a valid cut to the cut set $\Xi_{i}^{k}$. Line 11 updates the feasibility set.

In our distributed algorithm, agents form better outer approximations of the semi-infinite constraint set as the iterations progress. Note that as explained in the cutting-surface algorithm, we seek only an $\epsilon/2$-approximate maximizer for determining the cut as this maximization problem is nonconvex in general. The following result provides the convergence guarantees of Algorithm 1.

**Algorithm 1:** Distributed cutting-surface ADMM

**Executed by:** agents $i \in [n]$

**Data:** $\varphi_{i}$, $h_{i}$, $g_{i}$, sets $\mathcal{Y}$, $\Xi$, regularization parameter $\rho > 0$, and error tolerance $\epsilon > 0$

**Initialize:** agent $i$ sets iteration count $k = 0$,

variables $p_{ij}^{0} = 0$, $z_{ij}^{0} \in \mathcal{Z}_{i}$, $y_{ij}^{0} \in \mathcal{Y}$, cut set $\mathcal{Z}_{i}^{0} = \emptyset$, feasibility set $\mathcal{F}_{i}^{0} = \mathcal{Z}_{i} \times \mathcal{Y}$

1. **repeat**

   1.1. /* Each agent $i$: */

   1.2. Exchange $y_{ij}^{k}$ with neighbors $N_{i}$

   1.3. Set $p_{ij}^{k+1} \leftarrow p_{ij}^{k} + \rho \sum_{j \in N_{i}} \left( y_{ij}^{k} - y_{ij}^{k} \right)$

   1.4. Set

   $$
   \left( y_{ij}^{k+1}, z_{ij}^{k+1} \right) \leftarrow \arg \min_{(y_{i}, z_{j} \in F_{i}^{k+1})} \left\{ \sum_{j \in N_{i}} \left( \frac{1}{2} \left\|y_{i}^{k} - y_{j}^{k}\right\|^2 + \rho \left\|z_{i}^{k} - y_{i}^{k}\right\|^2 \right) \right\}.
   $$

   1.5. Determine $\xi_{i}^{k+1} \in \Xi$ such that

   $$
   g_{i}(y_{i}^{k+1}, z_{i}^{k+1}, \xi_{i}^{k+1}) \geq \max_{\xi} g_{i}(y_{i}^{k+1}, z_{i}^{k+1}, \xi) - \frac{\epsilon}{2}
   $$

   1.6. if $g_{i}(y_{i}^{k+1}, z_{i}^{k+1}, \xi_{i}^{k+1}) > \epsilon$ then

   1.7. \hspace{1em} Set $\mathcal{Z}_{i}^{k+1} \leftarrow \mathcal{Z}_{i} \cup \{ \xi_{i}^{k+1} \}$

   1.8. else

   1.9. \hspace{1em} Set $\mathcal{Z}_{i}^{k+1} \leftarrow \mathcal{Z}_{i}^{k}$

   1.10. Set $\mathcal{F}_{i}^{k+1} \leftarrow \{ (z, y) | z \in \mathcal{Z}_{i}, y \in \mathcal{Y}, g_{i}(y, z, \xi) \leq 0, \forall \xi \in \mathcal{Z}_{i}^{k+1} \}$

   1.11. Set $k \leftarrow k + 1$

   1.12. **until** termination condition is met

**Proposition III.1.** (Convergence of distributed cutting-surface ADMM): For Algorithm 1, there exists $K \in \mathbb{Z}_{\geq 1}$ such that $\mathcal{Z}_{i}^{k} = \mathcal{Z}_{i}^{k}$ for all $i \in [n]$ and all $k_{1}, k_{2} \geq K$. Moreover, any limit point $(\bar{Y}, \bar{Z}) := \{ (y_{i}, z_{i}) | i \in [n] \}$ of the sequence $\{ (Y_{k}, Z_{k}) := \{ (y_{i}, z_{i}) | i \in [n] \} \}_{k=0}^{\infty}$ satisfies (a) $\bar{y}_{i} = \bar{y}_{j}$ for all $i, j \in [n]$,

(b) $\max_{i \in [n]} \max_{\xi \in \Xi} g_{i}(y_{i}, z_{i}, \xi) \leq \epsilon$,

and (c) $\sum_{i=1}^{n} \left( \varphi_{i}(z_{i}) + h_{i}(y_{i}) \right) \leq J^{*}$, where $J^{*}$ is the optimum value of (7).

The above result shows that any accumulation point of the iterates maintained by the agents satisfies three properties. First, they achieve consensus over the global variable. Second, the semi-infinite constraint is satisfied with $\epsilon$ accuracy, and third, the objective value at the network-wide converged value is no more than the optimal value of the semi-infinite problem (7). Thus, one can claim that (7) is solved up to an $\epsilon$ accuracy. Note that such an optimizer could be found in finite number of steps in the centralized cutting-surface method explained in Section II-2. The convergence is asymptotic here due to the distributed nature of our algorithm.

**Remark III.2.** (Generalizations of Algorithm 1): In Al...
algorithm 1, the cut addition step can be skipped in some iterations by every agent. As long as one cut is added in every few iterations, the convergence still holds. On the same token, the guarantees are not affected if more than one cut is added at each iteration. Furthermore, one need not start adding the cuts with e tolerances. Initially, the tolerance can be high so that agents quickly reach consensus. Subsequently, agents can set the tolerance at a lower value to improve the accuracy of constraint set. Such modifications can possibly be used to improve the rate of convergence.

IV. DISTRIBUTED CUTTING-SURFACE ADMM FOR DRO

Here, we adapt the distributed algorithm presented in Algorithm 1 to the DRO problem (5). To this end, we recall the comparison between (5) and (7). In the former, \((x, s)\) are global variables on which all agents need to agree on and \(v^g := (v^g_k)_{k \in [n]}\) is the local variable. In words, \(v^g\) consists of decision variables \(v^g_k\) of problem (5) for which the data point \(\xi^k\) is held by agent \(i\). In problem (7), \(y\) is the global variable and \(z^i\)’s are the local ones. Functions \(\varphi^i\) and \(h^i\) in (7) can then be analogously written for (5) as:

\[
v^g_i \mapsto \frac{1}{|\mathcal{E}_i|} v^g_i \quad \text{and} \quad (x, s) \mapsto \left|\mathcal{E}_i\right| \theta s,
\]

respectively. The constraint function \(g_i\) for (5) reads as:

\[
(x, s, v^g, \xi) \mapsto \max_{\xi' \in \mathcal{E}_i} f(x, \xi') - (v^g_i)\xi' - s\|\xi - \xi'\|, \tag{11}
\]

where \((v^g_i)\xi'\) refers to the component of the vector \(v^g_i\) corresponding to the sample \(\xi'\). In problem (7), we assume that the global and local variables, for each agent \(i\), are restricted to convex compact sets \(\mathcal{Y}\) and \(\mathcal{Z}_i\), respectively. However, while \(x\) belongs to the convex compact set \(\mathcal{X}\) in (5), we do not have such a constraint for variables \(s\) and \(v\). Thus, to use the results of the previous section, we find next a compact convex domain for (5) that does not disturb the optimizers.

Lemma IV.1. (Compact domain DRO problem (5)): Any optimizer \((x^*, s^*, v^*)\) of the DRO problem given in (5) belongs to the convex compact set \(\mathcal{X} \times \mathcal{B}_s \times \mathcal{B}_v\), where \(\mathcal{B}_s := [0, (\mathcal{J} - f) / \theta]\) and \(\mathcal{B}_v = \mathcal{I}^L\), with interval \(\mathcal{I} := [f, f + L(\mathcal{J} - f)]\) and \(f\) and \(\mathcal{J}\) are bounds such that \(f \leq f(x, \xi) \leq \mathcal{J}\) for all \((x, \xi) \in \mathcal{X} \times \mathcal{E}\).

Having described the parallelism between (5) and (7) and having identified the compact domains, we present Algorithm 2 that solves the DRO problem.

\begin{algorithm}
\caption{(Distributed cutting-surface ADMM for DRO)}
\textbf{Algorithm 2. (Distributed cutting-surface ADMM for DRO):} The procedure involves executing Algorithm 1 with the following elements:
\textbf{Parameters:} regularization \(\rho > 0\) and tolerance \(\epsilon > 0\)
\textbf{Variables:} \((x^g, s^g) \in \mathbb{R}^{d+1}\) as \(y_i\); \(v^g_i \in \mathbb{R}^{|\mathcal{E}_i|}\) as \(z^i\); and the auxiliary variable \(p_i \in \mathbb{R}^{d+1}\) remains the same
\textbf{Sets:} \(\mathcal{X} \times \mathcal{B}_s\) as \(\mathcal{Y}\); \(\mathcal{I}^{\mathcal{E}_i}\) as \(\mathcal{Z}_i\) (see Lemma IV.1 for definitions); and the uncertainty set \(\mathcal{E}\) remains the same
\textbf{Functions:} maps in (10) as \(\varphi_i\) and \(h_i\) and map (11) as \(g_i\)
\end{algorithm}

In the above algorithm, agents need to agree beforehand upon the error tolerance \(\epsilon\) and the regularization parameter \(\rho\). The former of these can be selected independently by each agent and the algorithm still converges. The guarantee on the obtained optimizer depends on the largest of these tolerance values. The following result summarizes the convergence guarantees of the algorithm. It uses the convergence result of the previous section and the fact that if one satisfies the semi-infinite constraints in (5) with an \(\epsilon\) error and optimizes the objective function under these constraints, then one obtains an \(\epsilon\)-optimal solution of the DRO problem (1).

Proposition IV.2. (Convergence of distributed cutting-surface ADMM for DRO): For Algorithm 2, there exists a finite time \(K \in \mathbb{Z}_{\geq 1}\) such that \(\mathcal{E}_i^k = \mathcal{E}_i^{k_2}\) for all \(i \in [n]\) and all \(k_1, k_2 \geq K\). Furthermore, for any limit point \((X, S, V) = (x^g, s^g, v^g)\) of the sequence \(\{(X^k, S^k, V^k) := ((x^g)^k, (s^g)^k, (v^g)^k)\}_{i \in [n]}\) the following is true:

(i) \((x^{g_1}, s^{g_2}) = (x^{g_1}, s^{g_2})\) for all \(i, j \in [n]\),

(ii) denoting \(v^*\) as the collection \((v^g_i)^*\) and \(s^* = s^{g_2}\), the inequality \(L\theta s^* \leq L\theta^*\) holds, where \(J^*\) is the optimal value of (1), and

(iii) denoting \(x^* = x^{g_2}\), we have \(\sup_{y \in \mathcal{Y}} \mathbb{E}[f(x^*, \xi)] \leq J^{*}_{DRO} + \epsilon\).

The above result implies that asymptotically, each agent arrives at a point \(x\) that is \(\epsilon\)-optimal for the DRO problem (1). That is, the cost incurred at \(x\) is no more than \(\epsilon\) higher than the optimal value of (1).

Remark IV.3. (Comparison with literature): The work [9] gives a distributed saddle-point algorithm to solve the DRO problem (1). While the network structure and the availability of samples is same as our case, our work considerably generalizes the problem setup. Firstly, the method in [9] relies on identifying a subset of \(\mathcal{X} \times \Xi\) where the function defining the semi-infinite constraint is concave in the uncertainty \(\xi\). Our algorithm does not require such identification. Secondly, the objective function is assumed to be differentiable there while we can handle nonsmooth convex functions. Finally, we do not assume that agents have access to the number of other agents in the network or the total number samples.

V. SIMULATIONS

Here, we demonstrate the application of Algorithm 2 to a distributionally robust linear regression problem. Consider \(n = 6\) agents communicating over a graph defined by an undirected ring with additional edges \(\{1, 4\}, \{2, 3\}\). Each data point \(\xi^k = (\tilde{u}^k, \tilde{p}^k)\) consists of an input vector \(\tilde{u}^k \in \mathbb{R}^d\) and a scalar output \(\tilde{p}^k\). For constructing the dataset, we assume that \(\tilde{u}^k\) is sampled from multivariate normal distribution with mean zero and covariance as the identity matrix. The output takes value \(\tilde{u}^k = (1, 2, 3, 1)^T \tilde{u} + \tilde{p}^k\), where \(\tilde{p}^k\) is uniformly sampled from the interval \([-1, 1]\). We assume that each agent has access to 10 samples of this form. Note that the input-output pairs have an affine relationship defined by the vector \(w = (1, 2, 3, 1, 0)^T\). That is, \(\tilde{p}^k = w^T (\tilde{u}^k, 1)^T + \tilde{p}^k\). The aim for the agents is to
identify a vector $x^*$ that explains the affine relationship with desirable out-of-sample guarantees. To this end, agents wish to solve the DRO problem $\inf_{x \in \mathcal{X}} \sup_{\xi \in \hat{\mathcal{X}}} \mathbb{E}_Q[f(x, \xi)]$ where $x \in \mathbb{R}^p$, $\mathcal{X} := [0, 5]^p$, and $f(x, \xi) := (u - x^\top (u; 1))^2$ is the quadratic loss that determines the prediction accuracy for input-output pair $\xi = (u, v)$. The radius of the ambiguity set is $\theta = 0.01$. Recall that $L = 60$ as we have six agents and each agent has 10 samples. For our distributed algorithm, we set the error tolerance and regularizer as $\epsilon = 0.01$ and $\rho = 0.05$, respectively. Note that the maximization problems solved in each iteration by each agent (line 5 in Algorithm 1) is nonconvex in our case as $f$ is convex in $\xi$. Thus, we make use of the branch-and-bound algorithm proposed in [10, Algorithm 2] to find an $\epsilon/2$ approximate solution to the problem. Figure 1 depicts the execution of our algorithm. As shown, the agents achieve consensus over the global variable and find the solution of the DRO problem as $x^* = (0.89, 1.84, 2.89, 0.83, 0.06)$.

VI. CONCLUSIONS

We have designed and analyzed a distributed algorithm that solves the data-driven DRO problem to a pre-specified arbitrary accuracy. Our algorithm combines the features of cutting-surface algorithm and distributed ADMM. There are several open future research directions. First, we aim to analyze the convergence of our algorithm to the optimum when the accuracy is improved along the iterations. Next, we plan to design finite-memory algorithms that can solve the DRO problem to a certain approximation even when the sample size is large. Finally, we intend to explore primal-dual distributed algorithms to handle semi-infinite constraints.

REFERENCES

[1] P. Mohajerin Esfahani and D. Kuhn, “Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations,” Mathematical Programming, vol. 171, no. 1-2, pp. 115–166, 2018.

[2] G. Notarstefano, I. Notarnicola, and A. Camisa, “Distributed optimization for smart cyber-physical networks,” Foundations and Trends in Systems and Control, vol. 7, no. 3, pp. 253–383, 2019.

[3] A. Nedić and J. Liu, “Distributed optimization for control,” Annual Review of Control, Robotics, and Autonomous Systems, vol. 1, no. 1, pp. 77–103, 2018.

[4] Z. Hu and L. J. Hong, “Kullback-Leibler divergence constrained distributionally robust optimization,” 2013. Available at Optimization Online.

[5] R. Jiang and Y. Guan, “Data-driven chance constrained stochastic program,” Mathematical Programming, Series A, vol. 158, pp. 291–327, 2016.

[6] E. Erdoğan and G. Iyengar, “Ambiguous chance constrained problems and robust optimization,” Mathematical Programming, Series B, vol. 107, pp. 37–61, 2006.

[7] R. Gao and A. J. Kreuzeg, “Distributionally robust stochastic optimization with Wasserstein distance,” 2016. Available at arxiv.org/abs/1604.02199.

[8] D. Kuhn, P. Mohajerin Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh, “Wasserstein distributionally robust optimization: Theory and applications in machine learning,” in INFORMS Tutorials in Operations Research, pp. 130–166, 2019.

[9] A. Cherukuri and J. Cortés, “Consensus over the variable $x$,” Foundations and Trends in Systems and Control, vol. 7, no. 3, pp. 253–383, 2019.

[10] R. Hettich and K. O. Kortanek, “Semi-infinite programming: Theory, methods, and applications,” SIAM Review, vol. 35, no. 3, pp. 380–429, 1993.

[11] G. Banjac, F. Rey, P. Goulart, and J. Lygeros, “Decentralized resource allocation via dual consensus ADMM,” in American Control Conference, pp. 2789–2794, 2019.

[12] A. Cherukuri, A. Zolanvari, G. Banjac, and A. R. Hota, “Data-driven distributionally robust optimization over a network via distributed semi-infinite programming,” 2022. Available at arxiv.org/abs/2103.10321.

[13] R. Hettich and K. O. Kortanek, “Semi-infinite programming: Theory, methods, and applications,” SIAM Review, vol. 35, no. 3, pp. 380–429, 1993.

[14] G. Banjac, F. Rey, P. Goulart, and J. Lygeros, “Decentralized resource allocation via dual consensus ADMM,” in American Control Conference, pp. 2789–2794, 2019.

[15] A. Cherukuri, A. Zolanvari, G. Banjac, and A. R. Hota, “Data-driven distributionally robust optimization over a network via distributed semi-infinite programming,” 2022. Available at arxiv.org/abs/2103.10321.

[16] R. Hettich and K. O. Kortanek, “Semi-infinite programming: Theory, methods, and applications,” SIAM Review, vol. 35, no. 3, pp. 380–429, 1993.