Structured Reinforcement Learning for Incentivized Stochastic Covert Optimization

Adit Jain, *Student Member, IEEE*, and Vikram Krishnamurthy, *Fellow, IEEE*

***Abstract*—This letter studies how a stochastic gradient algorithm (SG) can be controlled to hide the estimate of the local stationary point from an eavesdropper. Such problems are of significant interest in distributed optimization settings like federated learning and inventory management. A learner queries a stochastic oracle and incentivizes the oracle to obtain noisy gradient measurements and perform SG. The oracle probabilistically returns either a noisy gradient of the function or a non-informative measurement, depending on the oracle state and incentive. The learner's query and incentive are visible to an eavesdropper who wishes to estimate the stationary point. This letter formulates the problem of the learner performing covert optimization by dynamically incentivizing the stochastic oracle and obfuscating the eavesdropper as a finite-horizon Markov decision process (MDP). Using conditions for interval-dominance on the cost and transition probability structure, we show that the optimal policy for the MDP has a monotone threshold structure. We propose searching for the optimal stationary policy with the threshold structure using a stochastic approximation algorithm and a multi–armed bandit approach. The effectiveness of our methods is numerically demonstrated on a covert federated learning hate-speech classification task.**

***Index Terms*—Stochastic optimal control, machine learning, optimization, stochastic systems.**

I. Introduction

THE LEARNER aims to obtain an estimate $\hat{x}$ for a point $x^* \in \arg\min_{x\in\mathbb{R}^d} f(x)$[1] by querying a stochastic oracle. At each time $k = 1, 2, \ldots$, the learner sends query $q_k \in \mathbb{R}^d$ and incentive i_k to a stochastic oracle in state o_k. The oracle returns a noisy gradient, r_k evaluated at q_k as follows:

$$r_k = \begin{cases} \nabla f(q_k) + \eta_k & \text{with prob. } \Gamma(o_k, i_k) \\ \mathbf{0} \text{ (non-informative)} & \text{with prob. } 1 - \Gamma(o_k, i_k). \end{cases} \quad (1)$$

Here (η_k) are independent, zero-mean finite-variance random variables, and Γ denotes the probability that the learner gets a noisy informative response from the oracle.

Manuscript received 8 March 2024; revised 4 May 2024; accepted 22 May 2024. Date of publication 28 May 2024; date of current version 21 June 2024. This work was supported in part by the Army Research Office under Grant W911NF–24–1–0083, and in part by the National Science Foundation under Grant CCF–2312198 and Grant CCF–2112457. Recommended by Senior Editor V. Ugrinovskii. *(Corresponding author: Adit Jain.)*

The authors are with the School of Electrical and Computer Engineering, Cornell University, Ithaca, NY 14853 USA (e-mail: aj457@cornell.edu).

Digital Object Identifier 10.1109/LCSYS.2024.3406543

[1]By arg min or minimizer we mean a local stationary point of $f \in \mathcal{C}^2$.

An eavesdropper observes query q_k and incentive i_k but not response r_k. The eavesdropper aims to estimate $\hat{x}$, as an approximation to the minimizer of the function the eavesdropper is interested in optimizing. This letter addresses the question: *Suppose the learner uses a stochastic gradient (SG) algorithm to obtain an estimate $\hat{x}$. How can the learner control the SG to hide $\hat{x}$ from an eavesdropper?*

Our proposed approach is to dynamically switch between two SGs. Let $a_k \in \{0 = \texttt{Obfuscate SG}, 1 = \texttt{Learn SG}\}$ denote the chosen SG at time k. The first SG minimizes function f and updates the learner estimate $\hat{x}_k$. The second SG is for obfuscating the eavesdropper with estimates $\hat{z}_k$. The update of both SGs is given by the equation,

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{z}_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{x}_k \\ \hat{z}_k \end{bmatrix} - \mu_k \begin{bmatrix} \mathbb{1}(a_k = 1) & 0 \\ 0 & \mathbb{1}(a_k = 0) \end{bmatrix} \begin{bmatrix} r_k \\ \bar{r}_k \end{bmatrix}, \quad (2)$$

where μ_k is the step size, $\bar{r}_k$ is a synthetic gradient response discussed later and a_k controls the SG to update.

The query q_k by the learner to the oracle is given by,

$$q_k = \hat{x}_k \mathbb{1}(a_k = 1) + \hat{z}_k \mathbb{1}(a_k = 0). \quad (3)$$

and $u_k = (a_k, i_k)$ is the control learner variable (action). The learner needs M informative updates of (2) to achieve the learning objective in N queries. We formulate an MDP whose policy π controls the switching of SGs and incentivization by the learner, to minimize the expected cost balancing obfuscation and learning. The optimal policy π^* solving the MDP is shown to have a threshold structure (Theorem 2) of the form,

$$\pi^*(b, o, n) = \begin{cases} a = 0 \ (\texttt{obfuscate}), & b \le \bar{b}(o, n) \\ a = 1 \ (\texttt{learn}), & b > \bar{b}(o, n), \end{cases}$$

where b is the number of informative learning steps left, n is the number of queries left and $\bar{b}$ is the threshold function dependent on the oracle state o and n. Note that the exact dependence on the incentive is discussed later. We propose a stochastic approximation algorithm to estimate the optimal stationary policy with a threshold structure. We propose a multi-armed bandits based approach with finite-time regret bounds in Theorem 3. The optimal stationary policy with a threshold structure is benchmarked in a numerical study for covert federated hate-speech classification.

Motivation: The main application of covert (or *learner-private*) optimization is in centralized distributed optimization [9], [10], [12]. One motivating example is in pricing optimization and inventory management, the learner (e.g., e-retailer) queries the distributed oracle (e.g., customers) pricing and product preferences to estimate the optimal

price and quantity of a product to optimize the profit function [1], [10]. A competitor could spoof as a customer and use the optimal price and quantity for their competitive advantage. Our numerical experiment illustrates another application in federated learning, a form of distributed machine learning.

The current literature on covert optimization has been focused on deriving upper and lower bounds on the query complexity for a given obfuscation level [12]. Query complexity for binary and convex covert optimization with a Bayesian eavesdropper has been studied in [10], [12]. These bounds assume a static oracle and a random querying policy can be used to randomly obfuscate and learn. In contrast, the authors have looked at dynamic covert optimization where stochastic control is used to query a stochastic oracle optimally [4]. This is starkly different than the current literature since a stochastic oracle models situations where the quality of gradient responses may vary (e.g., due to Markovian client participation). The success of a response can be determined by the learner (e.g., based on gradient quality [4]) or by the oracle (e.g., based on computational resources availability).

Differences from previous work [4]: To prove that the optimal policy has a monotone threshold structure, [4] requires supermodularity conditions. This letter proves results under more relaxed conditions using interval dominance [8] in Theorem 2, which can incorporate convex cost functions and more general transition probabilities [5]. The action space in this letter includes an incentive the learner provides to the oracle. An incentive that the learner pays is motivated by the learner's cost for obtaining a gradient evaluation of desired quality, it could be a monetary compensation the learner pays or non-monetary, e.g., controlling latency of services to participating clients [11]. We had a generic cost function in [4], but the costs considered in this letter are exact regarding the learner's approximation of the eavesdropper's estimate of $\hat{x}$.

II. Covert Optimization for First-Order Stochastic Gradient Descent

This section describes the two stochastic gradient algorithms, between which the learner dynamically switches to either learn or obfuscate using the MDP formulation of the next section. This section states the assumptions about the oracle, the learner, the eavesdropper, and the obfuscation strategy. We state the result on the number of successful gradient steps the learner needs to achieve the learning objective. The problem formulation for covert optimization is illustrated in Fig. 1.

A. Oracle

The oracle evaluates the gradient of the function f. The following is assumed about the oracle and the function f,

- **O1:** Function $f : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and is lower bounded by f^*. Function f is γ-Lipschitz continuous, $\|\nabla f(x) - \nabla f(z)\| \leq \gamma \|x - z\| \ \forall x, z \in \mathbb{R}^d$.
- **O2:** At time k, the oracle is in state $o_k \in \{1, \ldots, R\}$, where R are the number of oracle states and for the incentive $i_k \in \{i^1, \ldots, i^{n_i}\}$, replies with probability $\Gamma(o_k, i_k)$. $s_k \sim$ Bernoulli$(\Gamma(o_k, i_k))$ denotes success of the reply.
- **O3:** For a reply with success s_k to the query $q_k \in \mathbb{R}^d$, the oracle returns a noisy gradient response r_k according to (1). The noise terms η_k are independent,[2] have zero-mean and finite-variance, $\mathbb{E}[r_k] = \nabla f(q_k)$ and $\mathbb{E}[\|\eta_k\|^2] \leq \sigma^2$.

[2]A slightly weaker assumption based on conditional independence was considered in our paper [4]. We consider independence here for brevity.

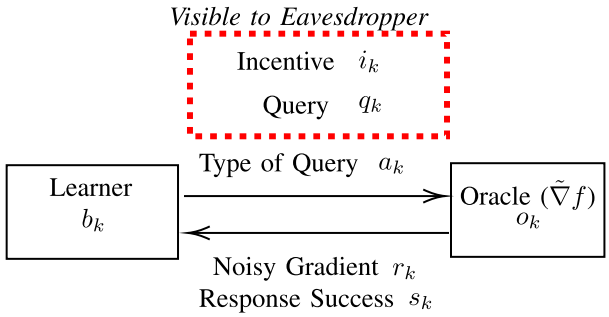

Fig. 1. Dynamic Covert Optimization: Learner sends query q_k and incentive i_k to oracle in state o_k. The oracle evaluates noisy gradient of f at q_k, r_k according to (1). An eavesdropper observes q_k and i_k and aims to approximate the learner's estimate. The learner needs to control the incentive i_k and type of SG (a_k) to query using (3) to achieve the learning objective of (4) and obfuscate the eavesdropper with belief (5).

O1 and O3 are standard assumptions for analyzing oracle-based gradient descent [3]. O2 is motivated by an oracle with a stochastic state (e.g., client participation), and the success is determined by the oracle or by the learner.

B. Learner

Similar to oracle-based first-order gradient descent [3], the learner aims to estimate $\hat{x} \in \mathbb{R}^d$ which is a ϵ-close critical point of the function f,

$$\mathbb{E}\left[\left\|\nabla f(\hat{x})\right\|^2\right] \leq \epsilon. \tag{4}$$

Since f is non-convex and not known in closed-form to the learner, in general, the gradient at z_1 is non-informative about the gradient at z_2 far from z_1. Hence, at time k, the learner can either send a learning or an obfuscating query. We propose controlling the gradient descent of the learner by the query action $a_k \in \{0 = \texttt{obfuscating}, 1 = \texttt{learning}\}$. While learning, the learner updates its estimate, $\hat{x}_k$ by performing the controlled stochastic gradient step of (2). Here, μ_k is the step size chosen to be constant in this letter. In the next section, we will formally state the action space composed of the type of query a_k and the incentive i_k. In order to estimate the number of queries to the oracle that the learner has to spend on learning queries, we first define the successful gradient step. We then state the result on the order of the number of successful gradient steps required for achieving the objective.

Definition 1 (Successful Gradient Step): A gradient step of (2) is successful when the learner queries the oracle with a learning query ($a_k = 1$) and gets a successful reply ($s_k = 1$).

Theorem 1: For an oracle with assumptions (O1-O3), to obtain an estimate $\hat{x}$ which achieves the objective (4), the learner needs to perform M successful gradient steps (Def. 1) with a step size ($\mu = \min(\frac{1}{\gamma}, \frac{\epsilon}{2\sigma^2\gamma})$) where M is of the order, $O(\frac{\sigma^2}{\epsilon^2} + \frac{1}{\epsilon})$. The exact expression is $M = \max(\frac{4F\gamma}{\epsilon}, \frac{8F\gamma\sigma^2}{\epsilon^2})$ where $F = (\mathbb{E}f(x_0) - f^*)$.

Proof for a general setting can be found in [4] and in [3]: Theorem 1 characterizes the number of successful gradient steps, M that the learner needs to perform to achieve the learning objective of (4). Theorem 1 guarantees the existence of a finite queue state space in the next section, which models the number of successful gradient steps left to be taken. M in the MDP formulation of the next section can be chosen

heuristically or be computed exactly if the parameters of the function are known. It also shows that M is inversely dependent on ϵ and incorporates the descent dynamics in the structure of the optimal policy.

C. Obfuscation Strategy

Based on the chosen SG a_k, the learner poses queries using (3) and provides incentives to the oracle. To obfuscate the eavesdropper, the learner runs a parallel stochastic gradient with synthetic responses, $\bar{r}_k$. The synthetic responses can be generated by suitably simulating an oracle, for, e.g., the learner can train a neural network separately with an unbalanced subset of as was done in [4]. If the learner is sure that the eavesdropper has no public dataset to validate, the learner can simply take mirrored gradients with (1). When obfuscating, the learner poses queries from the estimates of the second SG, $\hat{z}_k$. The parallel stochastic gradient ensures that the eavesdropper cannot infer the true learning trajectory from the shape of the trajectory. In summary, the learner obfuscates and learns by dynamically chooses the query q_k, as the current estimate $\hat{x}_k$ from the controlled stochastic gradient step or as the estimate $\hat{z}_k$ of parallel SG. We assume that the learner queries such that the two trajectories are sufficiently separated, and the eavesdropper can cluster the queries and distinguish them uniquely into two trajectories as described next.

D. Eavesdropper

At time k, the eavesdropper observes query q_k and the incentive i_k by the learner. We use the following assumptions to approximate the posterior belief of the eavesdropper,

- **E1:** The eavesdropper is passive, does not affect the oracle, and can not observe the oracle's responses, r_k.
- **E2:** For $k > 0$, the eavesdropper can classify each query of the observed query sequence $(q_1, \ldots, q_k)$ into two unique trajectory sequences, $\mathcal{J}_1^k$ and $\mathcal{J}_2^k$ which can be separated by a hyperplane in $\mathbb{R}^d$.
- **E3:** The eavesdropper knows that either of $\mathcal{J}_1^k$ and $\mathcal{J}_2^k$ is a learning trajectory. Given the information till time k, the eavesdropper computes a posterior belief, δ_k for trajectory $\mathcal{J}_1^k$ being the learning trajectory as,

$$\delta_k = \mathbb{P}\Big(\hat{x} \in \mathcal{J}_1^k | (q_1, i_1, \ldots, q_k, i_k)\Big).$$

Assumption E1 holds if the eavesdropper is considered an insignificant part of the oracle and can not observe the aggregated response. Assumptions E2 and E3 ensure the dynamic obfuscation is for a worst-case eavesdropper.[3] We consider the eavesdropper using an incentive-weighed proportional sampling estimator for the posterior δ_k [4], [12],

$$\delta_k = \frac{\sum_{1\le t\le k} i_t \mathbb{1}\big(q_t \in \mathcal{J}_1^t\big)}{\sum_{1\le t\le k} i_t}. \tag{5}$$

The learner has access to the queries and the incentives and uses (5) as an approximation for the eavesdropper's arg min estimate to compute the cost incurred when learning. The following section formulates an MDP to perform covert optimization using stochastic control. M from Theorem 1 and oracle state are used to model the state space, while the incentives i_k and the type of SG a_k in (2) model the action space.

[3]As mentioned above, it is assumed that the queries are posed such that the two trajectories are sufficiently separated (by a metric known to the eavesdropper). One of the trajectories can be empty for the initial queries.

Algorithm 1 Stochastic Control for Covert Optimization

Input: Policy π, Queries N, Successful Gradient Steps M
Initialize learner queue state $b_N = M$
for k in $1, \ldots, N$ **do**
  Obtain type of SG and incentive, $(a_k, i_k) = \pi(o_k, b_k)$
  Incur cost $c((a_k, i_k), (o_k, b_k))$ from (7)
  Query oracle using query q_k (3) and incentive i_k
  Receive response r_k and success of reply s_k
  Update estimates of the two SGs using (2).
  if s_k=1 **then** $b_{k+1} = b_k - s_k$
  Oracle state evolves, $o_{k+1} \sim \Delta(\cdot|o_k)$
end for
Incur terminal cost $d(b_0)$

III. MDP for Achieving Covert Optimization

We formulate a finite-horizon MDP to solve the learner's decision problem. The learner chooses an incentive, and dynamically either minimizes the function using the estimate $\hat{x}_k$ or obfuscates the eavesdropper using $\hat{z}_k$. The learner wants to perform M successful gradient steps in N total queries. Using interval dominance, we show that the optimal policy of the finite-horizon MDP has a threshold structure. The stochastic control approach for the same is described in Algorithm 1.

A. MDP Formulation for Optimally Switching Between Stochastic Gradient Algorithms

The dynamic programming index, $n = N, \ldots, 0$ denotes the number of queries left and decreases with time k.

State Space: The state space is denoted by $\mathcal{Y} = \mathcal{Y}^B \times \mathcal{Y}^O$ where $\mathcal{Y}^B = \{0, 1, \ldots, M\}$ is the learner queue state space and $\mathcal{Y}^O = \{0, 1, \ldots, R\}$ is the oracle state space. The learner queue state $b_n \in \mathcal{Y}^B$ denotes the number of successful gradient steps (Def. 1) remaining to achieve (4). The oracle state space $\mathcal{Y}^O$ discretizes the stochastic state of the oracle into R levels (e.g., percentages of client participation in FL). y_n denotes the state with n queries remaining.

Action Space: The action space is $\mathcal{U} = \{0 = \texttt{obfuscate}, 1 = \texttt{learn}\} \times \{i^1, \ldots, i^{n_i}\}$. The action when n queries are remaining is given by, $u_n = (a_n, i_n)$ where $a_n \in \{0 = \texttt{obfuscate}, 1 = \texttt{learn}\}$ is the type of the query and $i_n \in \{i^1, \ldots, i^{n_i}\}$ is the incentive. To derive structural results on the optimal policy, we consider the following transformation of the action space, $\mathcal{U} = \{(0, i^1), \ldots, (0, i^{n_i}), (1, i^1), \ldots, (1, i^{n_i})\}$. A deterministic policy for the finite-horizon MDP is denoted by π, a sequence of functions $\pi = (u_n : n = 0, \ldots, N)$. Here, $u_n : \mathcal{Y} \to \mathcal{U}$ maps the state space to the action space. Π denotes the space of all policies.

Transition Probabilities: We assume that the evolution of the oracle state and the learner queue state is Markovian. The oracle state evolves independently of the queue state evolution. In case of a successful gradient step (Def. 1), the queue decreases by one, and the oracle state evolves to a state $o' \in \mathcal{Y}^O$ with probability $\Delta(o'|o) > 0$. Let $\mathcal{P}_{b(}^{o,o'} u) = \mathbb{P}((o', \cdot)|o, b, u)/\Delta(o'|o)$ denote the transition probability vector of the buffer state with future oracle state o' given (o, b, u). The

transition probability from the state $y = (o, b) \in \mathcal{Y}$ to state $y' = (o', b') \in \mathcal{Y}$ with action $u = (a, i)$ can be written as,

$$\mathbb{P}\big(y' = \big(o', b-1\big)|y, u\big) = \Delta(o'|o)\Gamma(o, i)\mathbb{1}(a) \ \forall o',$$
$$\mathbb{P}\big(y' = (o, b)|y, u\big) = (1 - \Gamma(o, i))\mathbb{1}(a) + (1 - \mathbb{1}(a)), \quad (6)$$

and is 0 otherwise. The first equation corresponds to a successful gradient step, and the second to an unsuccessful one. We assume that $\Gamma(o, i)$ (from O2) is increasing in incentive i.

Learning and Queueing Cost: The learning cost $c_n : \mathcal{Y} \times \mathcal{U} \to \mathbb{R}$, is the cost (with n queries remaining) incurred after every action due to learning at the expense of reduced obfuscation. We consider the following learning cost which is proportional to the logarithm of the improvement in the eavesdropper's estimate ($\propto \log(\delta_n/\delta_{n+1})$) and is given by,

$$c_n(y_n, u_n) = \frac{\psi_1(b_n)}{\psi_2(o_n)} \log\left(\frac{I_n + i_n/\delta_n}{I_n + i_n}\right)\mathbb{1}(a_n) + \frac{\psi_2(o_n)}{\psi_1(b_n)} \log\left(\frac{I_n}{I_n + i_n}\right)(1 - \mathbb{1}(a_n)), \quad (7)$$

where $\psi_1 : \mathcal{Y}^B \to \mathbb{R}^+$ and $\psi_2 : \mathcal{Y}^O \to \mathbb{R}^+$ are positive, convex and increasing cost functions, $I_n = \sum_{k=N-1}^{n+1} i_k$ is the sum of the previous incentives and δ_n is the eavesdropper's estimate of the trajectory $\mathcal{J}_1$ being the true trajectory computed using (5). ψ_1 and ψ_2 are used to incorporate the cost with respect to the oracle and queue state, e.g., the functions ψ_1, and ψ_2 are considered quadratic in the respective states in the experiments. The form of the fractions ensures the structure as discussed next. The first term in (7) denotes the cost incurred in a learning query and is non-negative ($0 \le \delta_k \le 1$). The second term corresponds to an obfuscating query and is non-positive. The cost increases with the queue state and decreases with the oracle state. This incentivizes the learner to drive the system to a smaller queue and learn when the oracle is in a good state. After N queries, the learner pays a terminal queue cost computed using the function $d : \mathcal{Y} \to \mathbb{R}$. The queue cost accounts for learning loss in terms of terminal successful gradient steps left, b_0.

Remark: The incentive improves the response probability Γ, but also allows for improved obfuscation than a non-incentivized setup (a high incentive can be used to misdirect the eavesdropper's belief in (5)).

B. Optimization Problem

The expected total cost for the finite-horizon MDP with the initial state $y_N \in \mathcal{Y}$ and policy π is given by,

$$V^{\pi}(y_N) = \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N} c_n(y_n, u_n) + d(y_0, u_0) \mid y_N, \pi\right]. \quad (8)$$

The optimization problem is to find the optimal policy π^*,

$$V^{\pi^*}(y) = \inf_{\pi \in \Pi} V^{\pi}(y) \ \forall \ y \in \mathcal{Y}. \quad (9)$$

To define the optimal policy using a recursive equation, we first define the value function, V_n with n queries remaining,

$$V_n(y) = \min_{u \in \mathcal{U}}\left(c_n(y, u) + \sum_{y' \in \mathcal{Y}} \mathbb{P}\big(y'|y\big)V_{n-1}\big(y'\big)\right). \quad (10)$$

Let the optimal policy be $\pi^* = (u_n^*)_{n=N}^1$, where $u^*(\cdot)$ is the optimal action with n remaining queries and is the solution of the following stochastic recursion (Bellman's equation),

$$u_n^*(y) = \arg\min_{u \in \mathcal{U}} Q_n(u, y), \quad (11)$$

where the Q-function Q_n is defined as,

$$Q_n(u, y) = c_n(u, y) + \sum_{y' \in \mathcal{Y}} \mathbb{P}\big(y'|y, u\big)V_{n-1}\big(y'\big), \quad (12)$$

with $n = 0, \ldots, N$ and $V_0(y) = d(y)$. If the transition probabilities are unknown, then Q-learning can be used to estimate the optimal policy of (11). However, the following subsection shows that the optimal policy has a threshold structure, which motivates efficient policy search algorithms.

C. Structural Results

The following is assumed to derive the structural results,

R1: The learning cost, c_n is $\uparrow$ (increasing) and convex in the buffer state, d_n for each action $u_n \in \mathcal{U}$.

R2: Transition probability matrix $\mathbb{P}(b'|b, o, u)$ is TP3[4] with $\sum_{b'} b'\mathbb{P}(b'|b, o, u) \uparrow b$ and convex in b.

R3: The terminal cost, d is $\uparrow$ and convex in the queue state, b.

R4: For $\alpha_{b',b,u} > 0$ and $\uparrow u$, $c(b', o, u+1) - c(b', o, u) \le \alpha_{b',b,u}(c(b, o, u+1) - c(b, o, u))$, $b' > b$.

R5: For $\beta_{b',b,u} > 0$ and $\uparrow u$, $\frac{\mathcal{P}_{b'}^{o,o'}(u+1) + \beta_{b',b,u}\mathcal{P}_{b'}^{o,o'}(u)}{1+\beta_{b',b,u}} <_c \frac{\mathcal{P}_{b(}^{o,o'}u) + \beta_{b',b,u}\mathcal{P}_{b(}^{o,o'}u+1)}{1+\beta_{b',b,u}}$, $b' > b, \forall o', o \in \mathcal{Y}^O$; $<_c$ denotes convex dominance.[5]

R6: There exist $\alpha_{b',b,u} = \beta_{b',b,u}$ s.t. (R4) and (R5) hold.

Assumptions (R1) and (R3) are true by the construction of cost in (7) and the terminal cost. (R2) is a standard assumption on bi-diagonal stochastic matrices made when analyzing structural results [5]. (R4), (R5) and (R6) are the generalization of the supermodularity conditions made previously in [4] and are sufficient for interval dominance [5]. Assumption (R4) can be verified for cost of (7) using algebraic manipulation with $\alpha_{b',b,u} \le 1$, and (R5) can be shown with $\beta_{b',b,u} \le 1$ for the bi-diagonal matrix of (6) with $\Gamma(o, i) \uparrow i$. Therefore (R6) can be satisfied for some $\gamma_{b',b,u} = \alpha_{b',b,u} = \beta_{b',b,u} \le 1$. We now state the main structural result,

Theorem 2: Under assumptions (R1-6), the optimal action $u_n^*(y)$ (given by (11)) for the finite-horizon MDP of (9) is increasing in the queue state b.

Proof: **Step 1: Conditions of Interval Dominance:** The following condition with $\gamma_{b',b,u} > 0$,

$$Q_n\big(b', u+1\big) - Q_n\big(b', u\big) \le \gamma_{b',b,u}[Q_n(b, u+1) - Q_n(b, u)], b' > b, \quad (13)$$

is sufficient for $\arg\min Q_n$ to be increasing in b [5], [8],

$$u_n^*(b) = \arg\min_{u \in \mathcal{U}} Q_n(b, u) \uparrow b.$$

We omit the oracle state o from the above expression.

[4]Totally positive of order 3 (TP3) for a matrix P(a) requires that each of 3rd order minor of P(a) is non-negative.

[5]Probability vector p is convex dominated by probability vector q iff $f'p \ge f'q$ for increasing and convex vector f.

By plugging (12) in (13) we need to show the following,

$$\underbrace{c_n(b', u+1) - c_n(b', u) - \gamma_{b,b',u}(c_n(b, u+1) - c_n(b, u))}_{a}$$
$$+ \sum_{o' \in \mathcal{Y}^O} \sum_{b''} [\mathbb{P}((o', b'')|(o, b'), u+1)$$
$$- \mathbb{P}((o', b'')|(o, b'), u) - \gamma_{b,b',u}(\mathbb{P}((o', b'')|(o, b), u+1)$$
$$- \mathbb{P}((o', b'')|(o, b), u))] V(o', b'') \leq 0, b' > b.$$

By (R4) part (a) of the above inequality is satisfied with constant $\alpha_{b,b',u} \leq 1$. The rest of the inequality can be shown using (R5) with a constant $\beta_{b,b',u} \leq 1$ if we assume the value function is increasing and convex (see n.5 and [5]). Finally we apply (R6), with $\alpha_{b,b',u} = \beta_{b,b',u} = \gamma_{b,b',u}$ to show that (13) holds and the optimal action is $\uparrow$ in learner state b. All that remains to be shown is that the value function is increasing and convex, which we now show using (R1, R2, R3) and induction,

Step 2: Value Function is Increasing in b: By (R3), $V_0(y) = d(b)$ is increasing in b. Let $V_n(y) \uparrow b$. TP3 (R2) implies TP2 and hence preserves monotone functions [5]. Therefore by applying preservation of TP2 and linear combination, $\sum_{o' \in \mathcal{Y}^O} \Delta(o'|o) \sum_{b' \in \mathcal{Y}^B} \mathbb{P}(b'|b, o, u) V_n \uparrow b$. By (R1) and (12), $Q_{n+1} \uparrow b$. And therefore by (10), $V_{n+1}(y) \uparrow b$.

Step 3: Value Function is Convex in b: By (R3) $V_0(y) = d(b)$ is convex in b. Let V_n be convex in b. Then by (R2) and applying [5, Lemma 1] along with preservation of convexity under positive weighted sum, $\sum_{o' \in \mathcal{Y}^O} \Delta(o'|o) \sum_{b' \in \mathcal{Y}^B} \mathbb{P}(b'|b, o, u) V_n$ is convex in b. Applying (R1) and (12), Q_{n+1} is convex in b. Since minimization preserves convexity, $V_{n+1} = \min Q_{n+1}$ is convex in b. ∎

Theorem 2 implies that the policy is threshold in the learner queue state; hence, the learner learns more aggressively when the number of successful gradient steps (Def. 1) left is more. This intuitively makes sense from an obfuscation perspective since the learner should ideally spend more time obfuscating when it is closer to the minimizer (the queue state is small).

Using Theorem 2, we can parameterize the optimal policy by the thresholds on the queue state. Although we can construct stochastic approximation for estimating the non-stationary policy, which has a threshold structure and performs computationally better than Q-learning, this approach still requires the number of parameters to be linear in time horizon N. Given this insight, we restrict the search space to stationary policies with a monotone threshold structure, this restriction is common in literature [4], [7].

Let the threshold on queue state for oracle state o and action u be parameterized by $\bar{b} : \mathcal{Y}^O \times \mathcal{U} \to \mathcal{Y}^B$. The optimal stationary policy with a threshold structure can be written as,

$$\pi^*(y) = \sum_{u \in \mathcal{U}} u \mathbb{1}(\bar{b}^*(o, u) \leq b < \bar{b}^*(o, u+1)), \qquad (14)$$

where $\bar{b}^*$ is the optimal threshold function.

IV. Estimating the Optimal Stationary Policy With a Threshold Structure

In this section, we propose two methods to approximate the optimal stationary policy.[6] for the finite-horizon MDP of (9) which has the monotone threshold structure of (14). The first method uses a stochastic approximation to update the parameters over the learning episodes iteratively. The second method uses a multi-armed bandit formulation to perform discrete optimization over the space of thresholds. The proposed methods can be extended to a non-stationary policy space with an increased time and memory complexity.

[6]In this section, the optimal stationary policy is referred to as optimal policy.

A. Simultaneous Perturbation Stochastic Approximation

Taking the thresholds of the stationary policy of (14) as the parameters, a simultaneous perturbation stochastic approximation (SPSA) based algorithm can be used to find the parameters for the optimal policy. We update the policy parameters using approximate gradients of the costs computed using perturbed parameters. We use the following sigmoidal approximation for the threshold policy of (14),

$$\hat{\pi}(y, \bar{b}) = \sum_{u \in \mathcal{U}} \frac{1}{1 + \exp(-(b - \bar{b}(o,u))/\tau)}, \qquad (15)$$

where τ is an approximation parameter. The parameters are the $F = |\mathcal{U}||\mathcal{Y}_O|$ threshold values and are represented by Θ. For the optimal parameters, the approximate policy converges to the optimal policy as $\tau \to 0$ [7]. For the learning episode i and current parameter set Θ_i, the actions are computed using the current approximate policy (15). The policy parameters are perturbed independently with probability ½ by $\pm\delta$. Two learning episodes are performed with each set of perturbed policy parameters (Θ_i^+, Θ_i^-). The costs from the two episodes are used to obtain the approximate gradient $\tilde{\nabla} C_i$ by the method of finite differences. The policy parameters are updated using a stepsize ϕ_i,

$$\Theta_{i+1} = \Theta_i - \phi_i \tilde{\nabla} C_i.$$

Under regularity conditions on the noise in the approximate gradients, the approximate policy parameters asymptotically converge in distribution to the set of parameters of the optimal stationary policies with a threshold structure [6]. The SPSA algorithm can also be used with a constant step size to track changes in the system [4], [6]. The computational complexity for each learning episode is $O(F + N)$.

B. Multi-Armed Bandit Approach

The problem of searching the thresholds for (14) is solved by considering the values each threshold can take and then taking the product space of the thresholds as bandit arms. Each threshold can take values over learner state space $\mathcal{Y}_B$, which is of the cardinality $M + 1$. Consider each permutation of the $F = |\mathcal{U}||\mathcal{Y}_O|$ thresholds as an arm, making the total number of arms $(M+1)^F$. The selection of an arm gives a corresponding stationary policy of the form (11), and a reward (negative of the cumulative cost of the episode) is obtained by interacting with oracle for time horizon N. The noisy reward is sampled from a distribution centered at the expected value (8). **(B1)** The reward is assumed to be sampled independently for a given policy, and the noise is assumed to be sub-Gaussian [2]. For brevity, we omit the definition of regret and the exact upper bound, both of which can be found in [2, Ch. 2]. We now state the result on the regret for searching the thresholds.

Theorem 3: Consider the finite-horizon MDP of (9) for covert optimization with an oracle (O1-O3) to achieve (4). The optimal stationary policy with a threshold structure (14)

TABLE I
THE OPTIMAL STATIONARY POLICY WITH A THRESHOLD STRUCTURE OUTPERFORMS GREEDY POLICY BY 35% ON EAVESDROPPER ACCURACY AND RANDOM POLICY BY 38% ON LEARNER ACCURACY

Type of Policy	Learner Acc.	Eaves. Acc.	Incentive
Optimal Policy	90%	54%	254
Optimal Policy from [4]	89%	53%	290
Greedy Policy	91%	89%	300
Random Policy	52%	53%	190

can be searched using the upper confidence bound algorithm under (B1) with an expected regret after T episodes bounded by $O(M^F \log T)$, where, M is of the order $O(1/\epsilon + \sigma^2/\epsilon^2)$ and $F = |\mathcal{Y}_O||\mathcal{U}|$ is the number of thresholds.

The proof follows from Theorem 1 and plugging the number of arms in the standard regret bound for UCB [2]. Although the regret for this approach is bounded, the significant limitations are that the bound is exponential in the state and action space and, compared to SPSA, it cannot track changes in the system.

V. EXAMPLE: COVERT FEDERATED LEARNING FOR HATE-SPEECH CLASSIFICATION

We demonstrate the proposed covert optimization framework on a numerical experiment for hate-speech classification using federated learning in the presence of an eavesdropper. An eavesdropper spoofs as a client and misuses the optimal weights to generate hate speech, which goes undetected by the classifier. The detailed motivation and experimental setup can be found in [4]. A balanced subset of the civil comments toxicity dataset by Jigsaw AI is used, which has comments along with annotations for whether the comment is toxic or not The federated learning setup consists of $N_c = 35$ clients, each having $N_d = 689$ data points. A fully connected neural network attached to a pre-trained transformer is trained with a cross-entropy loss to classify the comments as toxic or not. The accuracy is reported on a balanced validation dataset.[7]

We consider $M = 45$ successful gradient steps and $N = 100$ queries, and the oracle levels are based on client participation. Each client participates in a Markovian fashion with a probability of staying connected or not connected as 0.8. $R = 3$ oracle states correspond to the minimum number of participating clients $\mathcal{Y}^O = [1 = 1, 2 = 12, 3 = 24]$. We consider $n_i = 3$ incentive levels as $\{1, 2, 3\}$. The number of samples each client contributes in each round depends on the incentive, as [10%, 40%, 80%] of N_d for the respective incentives. We consider a round successful if the number of samples exceeds 4000. The empirical success probabilities are $\Gamma(o, i) = [[0, 0.1, 0.2], [0.1, 0.2, 0.6], [0.3, 0.6, 0.9]]$. The functions ψ_1 and ψ_2 in (7) are quadratic in b and o, respectively. This satisfies assumptions R1, R2, R3. The empirical success probabilities along with the resulting cost function of (7) ensure that R4, R5, R6 are satisfied for $\alpha_{b,b',u} = \beta_{b,b',u} \leq 1$. The queue cost is $d(b) \propto b^4$. The optimal stationary policy with the threshold structure is obtained using SPSA with $\phi_k = 0.01$, $\delta = 0.1$, and $H = 3000$ episodes.

The results are averaged for $N_{mc} = 100$ runs and reported in Table I. The greedy policy learns first with a maximum incentive, and random policy uniformly samples from the action space. The optimal policy is better than the greedy policy in terms of the eavesdropper accuracy corresponding to the maximum a posteriori trajectory of (5). The optimal policy outperforms the random policy on learner accuracy. The learner saves 14% incentive spent compared to the greedy policy. We also benchmark against the optimal policy from [4] with constant incentivization ($i_k = 3$) and similar to the greedy policy, the accuracies are comparable, but the optimal policy of this letter improves incentive expenditure by 12%.

[7]The results are reproducible and can be found on the Github repository: github.com/aditj/CovertOptimization. The repository also contains links to the dataset, the complete set of experimental parameters, and a supplementary document with additional benchmarks and illustrations.

VI. CONCLUSION

The proposed MDP framework solves the learner's problem of dynamically optimizing a function by querying and incentivizing a stochastic oracle and obfuscating an eavesdropper by switching between two stochastic gradients. Using interval dominance, we prove structural results on the monotone threshold nature of the optimal policy. In our numerical experiments, the optimal stationary policy with the threshold structure outperformed the greedy policy on the eavesdropper accuracy and the incentive spent. In future work, the problem of obfuscating sequential eavesdroppers can be formulated as a Bayesian social learning problem, where initially the eavesdropper is obfuscated maximally to make it stop participating and its departure provides an indication to the subsequent eavesdroppers that the learner is obfuscating. Hence, the eavesdroppers can eventually be made to herd, forming an information cascade so that they don't eavesdrop anymore, regardless of whether the learner is learning or not.

REFERENCES

[1] C. Bersani, H. Dagdougui, C. Roncoli, and R. Sacile, "Distributed product flow control in a network of inventories with stochastic production and demand," *IEEE Access*, vol. 7, pp. 22486–22494, 2019.
[2] S. Bubeck, N. Cesa-Bianchi, and S. Bubeck. *Regret Analysis of Stochastic and Nonstochastic Multi-Armed Bandit Problems*. Norwell, MA, USA: Now Publ., 2012.
[3] S. Ghadimi and G. Lan, "Stochastic first- and zeroth-order methods for nonconvex stochastic programming," *SIAM J. Optim.*, vol. 23, no. 4, pp. 2341–2368, Jan. 2013.
[4] A. Jain and V. Krishnamurthy, "Controlling federated learning for covertness," *Trans. Mach. Learn. Res.*, Jan. 2024.
[5] V. Krishnamurthy, "Interval dominance based structural results for Markov decision process," *Automatica*, vol. 153, Jul. 2023, Art. no. 111024.
[6] H. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*. New York, NY, USA: Springer, Jul. 2003.
[7] M. H. Ngo and V. Krishnamurthy, "Monotonicity of constrained optimal transmission policies in correlated fading channels with ARQ," *IEEE Trans. Signal Process.*, vol. 58, no. 1, pp. 438–451, Jan. 2010.
[8] J. K.-H. Quah and B. Strulovici, "Comparative statics, informativeness, and the interval dominance order," *Econometrica*, vol. 77, no. 6, pp. 1949–1992, 2009.
[9] X. Shi, G. Wen, and X. Yu, "Finite-time convergent algorithms for time-varying distributed optimization," *IEEE Control Syst. Lett.*, vol. 7, pp. 3223–3228, 2023.
[10] J. N. Tsitsiklis, K. Xu, and Z. Xu. "Private sequential learning," *Oper. Res.*, vol. 69, no. 5, pp. 1575–1590, Sep. 2021.
[11] L. Witt, M. Heyer, K. Toyoda, W. Samek, and D. Li, "Decentral and incentivized federated learning frameworks: A systematic literature review," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3642–3663, Feb. 2023.
[12] J. Xu, K. Xu, and D. Yang, "Learner-private convex optimization," *IEEE Trans. Inf. Theory*, vol. 69, no. 1, pp. 528–547, Jan. 2023.