Constrained Stochastic Nonconvex Optimization with State-dependent Markov Data

Abhishek Roy∗ Krishnakumar Balasubramanian† Saeed Ghadimi‡

Abstract

We study stochastic optimization algorithms for constrained nonconvex stochastic optimization problems with Markovian data. In particular, we focus on the case when the transition kernel of the Markov chain is state-dependent. Such stochastic optimization problems arise in various machine learning problems including strategic classification and reinforcement learning. For this problem, we study both projection-based and projection-free algorithms. In both cases, we establish that the number of calls to the stochastic first-order oracle to obtain an appropriately defined $\epsilon$-stationary point is of the order $O(1/\epsilon^{2.5})$. In the projection-free setting we additionally establish that the number of calls to the linear minimization oracle is of order $O(1/\epsilon^{3.5})$. We also empirically demonstrate the performance of our algorithm on the problem of strategic classification with neural networks.

1 Introduction

We consider the following stochastic optimization problem

$$\arg\min_{\theta \in \Theta} f(\theta) = \arg\min_{\theta \in \Theta} \mathbb{E}[F(\theta; x)],$$

(1)

where (i) the expectation is taken over the stationary distribution, $\pi_\theta$, of the random vector $x$, (ii) $F$ (and hence $f$) is a potentially non-convex function in $\theta$, and (iii) $\Theta$ is a compact and convex constraint set. Stochastic approximation algorithms for solving problem (1), given an independent and identically distributed (iid) data stream $\{x_k\}_k$ drawn from $\pi$, are well-studied. Such iid assumptions are commonly made in various machine learning and statistical problems including empirical risk minimization [SSBD14], sparse recovery [BJMO12] and compressed sensing [FR13, Lan20]. We refer to [MB11, ABRW12, RSS12, GL13, SZ13, LZ16, ACD⁺19] for a partial list of non-asymptotic upper and lower bounds on the oracle complexity of widely-used stochastic approximation algorithms like the Stochastic Gradient Descent (SGD) and the Stochastic Conditional Gradient Algorithm.

Our focus in this work is on the case when the data sequence $\{x_k\}_k$ is drawn from a Markov chain with a state-dependent transition kernel $P_\theta$. Such a setting arises in several machine learning applications including but not limited to strategic classification [HMPW16, CDP15, MDPZH20, LW22] and reinforcement learning [Bar92, GSK13, ZJM21, KMMW19, QW20]. Despite their prevalence in practice, a deeper understanding of the non-asymptotic oracle complexity of stochastic approximation for Markovian data is only now starting to emerge. We establish non-asymptotic

∗abroy@ucdavis.edu. Halıcıoğlu Data Science Institute, University of California, San Diego. Work done while being affiliated with the Department of Statistics, UC Davis. Research of this author was supported by National Science Foundation (NSF) grant CCF-1934568.

†kbala@ucdavis.edu. Department of Statistics, University of California, Davis. Research of this author was supported in part by UC Davis CeDAR (Center for Data Science and Artificial Intelligence Research) Innovative Data Science Seed Funding Program and NSF Grant DMS-2053918.

‡sghadimi@uwaterloo.ca. Department of Management Sciences, University of Waterloo. Research of this author was supported by an NSERC Discovery Grant.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).
We use \(x\) where \(\alpha\) where \(c\) respectively in an unconstrained strongly convex setting. Our results take a step towards inducing constraint \(f\) state-dependent Markov chain dynamics in the training data. The objective function, analogous to the gradient is with respect to \(x\). In that case the agents have to possibly learn the best response whose task is to classify a given dataset which is collected from a set of agents. Given the knowledge of the classifier, the agents can distort some of their personal features, in order to get classified in a predetermined target class. This scenario arises in various applications, e.g., spam email filtering, and credit score classification. Optimizing the classifier to classify such strategically modified data where the agents modify the data iteratively can be formulated as problem (1).

Formally, let the classifier be \(h(x; \theta)\) where \(x \in \mathbb{R}^d\) is the feature and \(\theta\) is the parameter to be optimized. \(h(x; \cdot): \Theta \to \mathbb{R}\) is potentially nonconvex. Let the loss function be logistic loss which for a sample \((x, y)\), where \(y \in \{-1, 1\}\) denotes the corresponding class, is given by,

\[
L(\theta; x, y) = \log(1 + \exp(-h(x; \theta))) + (1 - y)h(x; \theta)/2.
\] (2)

We use \(x_S\), and \(x_{-S}\) to denote the subset of feature \(x\) which are respectively strategically modifiable, and non-modifiable by the agents. Then the modified feature (the best response) \(x_S'\) reported by the agent is the solution to the following optimization problem:

\[
x_S' = \arg\max_{x_S} (h(x; \theta) - c(x_S, x_S')),
\] (3)

where \(c(x, x')\) is the cost of modifying \(x_S\) to \(x_S'\). Let the agents iteratively learn \(x_S'\) similar to [LW22]. Note that unlike [LW22], where the authors deploy a logistic regression classifier and the closed form solution of the best response is readily known to the agents, it may not be the case in general. In that case the agents have to possibly learn the best response \(x_S'\) using some iterative optimization algorithm. For example, if the agents use Gradient Ascent then, at every iteration \(k\), a set \(I_k\) of \(n_1 \leq M\) randomly chosen agents out of \(M\) agents modify their features as:

\[
x_{S,i}^k = \begin{cases} 
    x_{S,i}^{k-1} + \alpha \left( \nabla h(x_{S,i}^{k-1}; \theta_k) - \nabla c(x_{S,i}^{k-1}, x_{S,i}') \right) & i \in I_k \\
    x_{S,i}^{k-1} & i \notin I_k
\end{cases}
\] (4)

where \(\alpha\) is the stepsize. With a little abuse of notation, we use \(\nabla h(x_{S,i}^{k-1}; \theta)\) in (4) to denote the fact that the gradient is with respect to \(x_{S,i}^{k-1}\) while \(x_{-S,i}\) remains unchanged. This introduces the state-dependent Markov chain dynamics in the training data. The objective function, analogous to \(f(\theta)\) in (1), is

\[
\min_{\theta \in \Theta} \mathbb{E}_{\pi_\theta} [L(\theta; x, y)],
\]

where \(\pi_\theta\) is the stationary joint distribution of \((x, y)\), and \(\Theta\) is a convex and compact set, e.g., sparsity inducing constraint \(\|\theta\|_1 \leq R\) from some \(R > 0\). The loss evaluated at a single data point \((x, y)\), \(L(\theta; x, y)\), is analogous to \(F(\theta; x)\) in (1). [DX20], and [LW22] study this problem theoretically and empirically respectively in an unconstrained strongly convex setting. Our results take a step towards analyzing this problem in constrained nonconvex setting. We empirically show the performance of the stochastic conditional gradient algorithm on a strategic classification problem in Section 4.1.
1.2 Preliminaries and Main Contributions

Before we present our main contributions, we introduce our convergence criterion. In constrained optimization literature, most commonly used convergence criteria are: (i) Gradient Mapping (GM), and (ii) Frank-Wolfe Gap (FW-gap). The Gradient Mapping at a point \( \bar{\theta} \in \Theta \) is defined as

\[
G_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta}), \beta) := \beta \left( \bar{\theta} - \Pi_{\Theta} \left( \bar{\theta} - \frac{1}{\beta} \nabla f(\bar{\theta}) \right) \right),
\]

where \( \Pi_{\Theta}(x) \) denotes the orthogonal projection of the vector \( x \) onto the set \( \Theta \), i.e.,

\[
\Pi_{\Theta} \left( \bar{\theta} - \frac{1}{\beta} \nabla f(\bar{\theta}) \right) = \arg\min_{y \in \Theta} \left\{ \langle \nabla f(\bar{\theta}), y - \bar{\theta} \rangle + \frac{\beta}{2} \| y - \bar{\theta} \|^2 \right\}.
\]

We will use \( \Pi_{\Theta}(\bar{\theta}, \nabla f(\bar{\theta}), \beta) \) to denote \( \Pi_{\Theta} \left( \bar{\theta} - \nabla f(\bar{\theta}) / \beta \right) \) when there is no confusion. Note that when \( \Theta \equiv \mathbb{R}^d \) we have \( G_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta}), \beta) = \nabla f(\bar{\theta}) \). In other words, for constrained optimization gradient mapping plays an analogous role of the gradient for unconstrained optimization. The gradient mapping is a frequently used measure in the literature as a convergence criterion for nonconvex constrained optimization [Nes18]. We should emphasize here that although the gradient mapping cannot be computed in the stochastic setting, one can still use it as a convergence measure.

[BG22] shows that the above notion of convergence criterion is closely related to the so-called Frank-Wolfe Gap. The FW-gap is defined as

\[
g_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta})) := \max_{y \in \Theta} \langle \nabla f(\bar{\theta}), \bar{\theta} - y \rangle.
\]

The following proposition from [BG22] establishes the relation between the gradient mapping criterion and the Frank-Wolfe gap:

**Proposition 1.1** [BG22] Let \( g_{\bar{\theta}}(\cdot) \) be the Frank-Wolfe gap defined in (6) and \( G_{\bar{\theta}}(\cdot) \) be the gradient mapping defined in (5). Then, we have

\[
\| G_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta}), \beta) \|_2^2 \leq g_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta})), \quad \forall \bar{\theta} \in \Theta.
\]

Moreover, under standard regularity assumption in smooth optimization (specifically, Assumption 2.1, and 2.2), we have

\[
g_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta})) \leq L \| G_{\bar{\theta}}(\bar{\theta}, \nabla f(\bar{\theta}), \beta) \|_2 / \beta.
\]

In this work we use a suboptimality measure, closely related to both GM and the FW-gap. At point \( \theta \in \Theta \), we define the suboptimality measure \( V(\theta, z) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R} \) as [GRW20]

\[
V(\theta, z) := \| \Pi_{\Theta} \left( \theta - z / \beta \right) - \bar{\theta} \|_2^2 + \| z - \nabla f(\bar{\theta}) \|_2^2,
\]

where \( z \), formally defined in Algorithm 1, is the moving-average estimate of \( \nabla f(\bar{\theta}) \). We show the relation among \( V(\theta, z) \), and GM \( G_{\bar{\theta}}(\theta, z, \beta) \) in the following proposition.

**Proposition 1.2** Let \( \{z_k\} \) be the sequence generated in Algorithm 1. Then, for \( k = 1, 2, \cdots, N \), we have

\[
\| G_{\bar{\theta}}(\theta_k, z_k, \beta) \|_2^2 \leq \max(2, 2\beta^2) V(\theta_k, z_k).
\]

The proof is provided in Appendix A. The main objective of this work is to find an \( \epsilon \)-stationary solution to (1), where an \( \epsilon \)-stationary solution is defined as follows:

**Definition** A point \( \bar{\theta} \) is said to be an \( \epsilon \)-stationary solution to (1), if \( \mathbb{E} \left[ V(\bar{\theta}, z) \right] \leq \epsilon \), where the expectation is taken over all the randomness involved in the problem.

For stochastic Frank-Wolfe-type algorithms, the oracle complexity is measured in terms of number of calls to the Stochastic First-order Oracle (SFO) and the Linear Minimization Oracle (LMO) used to solve the sub-problems of the algorithm which involves minimizing a linear function over the convex constraint set. Formally, we have the following definition.

**Definition 2** For a given point \( \theta \in \Theta \), SFO returns the stochastic gradient \( \nabla F(\theta, x) \). Given a vector \( z \), LMO returns a vector \( v := \arg\min_{y \in \Theta} \langle z, y \rangle \).
works are invariably focused on specific objective functions arising in the reinforcement learning setup, while our focus is on obtaining results for a general class of functions.

There has been significant recent advancements in the conditional gradient algorithm literature although it was developed long back [FW56, LP66]; see [Mig94, Jag13, LJJ15, LJJ15, HJN15, GKS21, BS17], for a non-exhaustive list of recent works. Sample-average approximation algorithms for constrained convex optimization with \( \phi \)-mixing data was considered in [WPT+21], [SSY18], and [AL22] analyze projected SGD for constrained non-convex optimization with time-homogeneous Markov chain. None of these works consider state-dependent data distribution except [DX20]. But unlike [DX20], we consider constrained nonconvex optimization. There also exists work in the reinforcement learning literature on understanding stochastic optimization with Markovian data; see, for example [XXLZ20, BRS18, DNPR20]. However, such works are invariably focused on specific objective functions arising in the reinforcement learning setup, while our focus is on obtaining results for a general class of functions.

### Conditional Gradient-Type Method

There has been significant recent advancements in the conditional gradient algorithm literature although it was developed long back [FW56, LP66]; see [Mig94, Jag13, LJJ15, LJJ15, HJN15, GKS21, BS17], for a non-exhaustive list of recent works. [HK12, HL16] provided expected oracle complexity results for stochastic conditional gradient algorithm in the stochastic convex setup. Better rates were provided by a sliding procedure in [LZ16]. In the non-convex setting, [RSPS16, YSC19, HL16] considered variance reduced stochastic conditional gradient algorithms, and provided expected oracle complexities. [QLX18] analyzed the sliding algorithm in the non-convex setting and provided results for the reinforcement mapping criterion. All of the above works use increasing orders of mini-batch based gradient-estimate.

To avoid mini-batches, a moving-average gradient estimator based on only one-sample in each iteration for a stochastic conditional gradient-type algorithm was proposed in [MHK20] and [ZSM+20] for the convex and non-convex setting. However, several restrictive assumptions have been made in
We now introduce the precise assumptions we make in this work. Let $F$ be the constraint set

**Proposition 2.1 (Lipschitz continuous gradient [AMP05])** Let Assumption 2.4 be true. Then, for every $\theta \in \Theta$ and $\theta' \in \Theta'$, there exists a constant $L > 0$ such that for any $\theta_1, \theta_2 \in \Theta$, we have $|f(\theta_1) - f(\theta_2)| \leq L ||\theta_1 - \theta_2||_2$. Assumption 2.3 is common in stochastic optimization literature. Assumption 2.4(a) is a frequently used assumption in Markov chain literature. It implies that for every $\theta \in \Theta$, there exists a stationary distribution $\pi_\theta(x)$, and the chain is $\lambda$-uniformly ergodic [AMP05]. Assumption 2.4(c) provides smoothness guarantee on the function $f(\cdot)$. More formally, we have the following proposition.

**Proposition 2.1 (Lipschitz continuous gradient [AMP05])** Let Assumption 2.4 be true. Then $f(\cdot)$ has Lipschitz continuous gradient, i.e., there is a constant $L_G > 0$ such that for any $\theta_1, \theta_2 \in \Theta$:

$$||\nabla f(\theta_1) - \nabla f(\theta_2)||_2 \leq L_G ||\theta_1 - \theta_2||_2.$$  (16)
Finally, the most important implication of Assumption 2.4 is that it ensures the existence and regularity of a solution $u(\theta, x)$ to Poisson equation of the transition kernel $P_0$ given by $u(\theta, x) - P_0 u(\theta, x) = \nabla F(\theta, x) - \nabla f(\theta)$. Solution of Poisson equation has been crucial in analyzing additive functionals of Markov chain (see [AMP05] for details). In this work, the Poisson equation solution facilitates a decomposition of the noise as presented in Lemma 3.1 which is a key component of our analysis.

3 Main Result

In this section we present our main result on the oracle complexity to establish a bound on $\mathbb{E}[V(\theta_k, z_k)]$. In order to do so we use Algorithm 1, and 2 similar to [XBG22]. If an exact method but are valid for any method which can solve (17) within an error of the order of $\|y - \theta_k\|^2_2$. We should emphasize that our results are not limited to ICG but are valid for any method which can solve (17) within an error of the order of $\|y - \theta_k\|^2_2$.

Algorithm 1 Inexact Averaged Stochastic Approximation (I-ASA)

**Input:** $z_0, \theta_0 \in \mathbb{R}^d, \eta_k = (N + k)^{-a}, 1/2 < a < 1, \beta,

for $k = 1, 2, \ldots, N$ do

$y_k = \begin{cases} \min_{y \in \Theta} \left\{ \langle z_k, y - \theta_k \rangle + \frac{\beta}{2} \|y - \theta_k\|^2_2 \right\} & \text{(Projection)} \\
ICG(z_k, \theta_k, \beta, t_k, \omega) & \text{(No Projection)} \end{cases}$

$\theta_{k+1} = \theta_k + \eta_{k+1}(y_k - \theta_k)$

$z_{k+1} = (1 - \eta_{k+1})z_k + \eta_{k+1}\nabla F(\theta_k, x_{k+1})$

end for

**Output:** $\theta_R$ where $P(R = i) = \frac{\eta_i}{\sum_{j=1}^{N} \eta_j}$ for $i = 1, 2, \ldots, N$.

Algorithm 2 Inexact Conditional Gradient (ICG)

**Input:** $z, \theta, \beta, t, \omega$.

Set $w_0 = \theta$

for $i = 1, 2, \ldots, t - 1$ do

Find $v_i$ such that

$$\langle v_i, z + \beta(w_i - \theta) \rangle \leq \arg\min_{v \in \Theta} \langle v, z + \beta(w_i - \theta) \rangle + \beta \omega D^2_{\Theta} / (i + 2)$$

$w_{i+1} = (1 - \mu_i)w_i + \mu_i v_i$ where $\mu_i = \frac{2}{i+2}$

end for

**Output:** $w_t$

respectively.

When a projection operator is unavailable or computationally costly, we use Algorithm 2 instead to solve (17). At iteration $k$, Algorithm 2 finds an approximate solution to (17) based on the conditional gradient algorithm. Algorithm 2 needs access to LMO which is often much cheaper and simpler to compute than projection operator. We should emphasize that our results are not limited to ICG method but are valid for any method which can solve (17) within an error of the order of $\{\eta_k\}$.

Theorem 3.1 Let Assumption 2.1-2.4 be true. Then, for Algorithm 1,

(a) when a projection operator is available, choosing

$$\eta_k = (N + k)^{-3/5}, \quad \beta = 1$$

for $k = 1, 2, \ldots, N$ we have

$$\mathbb{E}[V(\theta_R, z_R)] = O \left( N^{-\frac{2}{5}} \right)$$

(b) when Algorithm 2 is used to solve (17), choosing

$$\eta_k = (N + k)^{-3/5}, \quad t_k = \eta_k^{-2}, \quad \beta = 1, \quad \omega = 1, \quad \mu_i = 2/(i + 2)$$
for $k = 1, 2, \ldots, N$ we have
\[ \mathbb{E} [V(\theta_R, z_R)] = O \left( N^{-\frac{3}{2}} \right), \]
where the expectations are taken with respect to all the randomness of the algorithm, and an independent integer random variable $R \in \{1, 2, \ldots, N\}$ with probability mass function,
\[ P(R = k) = \eta_k / \sum_{k=1}^{N} \eta_k, \quad k \in \{1, 2, \ldots, N\}. \]

Remark 1 Note that total number of LMO calls are $\sum_{k=1}^{N} t_k = \sum_{k=1}^{N} t_k = \sum_{k=1}^{N} (N + k)^{2a} = O(N^{11/5})$. In other words, to achieve $\|G_0(\theta_R, \nabla f(\theta_R), \beta)\|_2^2 \leq \mathbb{E} [V(\theta_R, z_R)] \leq \epsilon$, SFO and LMO complexities are respectively $\epsilon^{-2.5}$, and $\epsilon^{-5.5}$. Note that the SFO complexity will be $\epsilon^{-2.5}$ as long as one has an approximation of the projection operator with approximation error $O(\eta_k)$.

Remark 2 In Theorem 3.1, one obtains sublinear rate $\max(N^{-1}, N^{2-4a})$ with $\eta_k = (N + k)^{-a}$ for $1/2 < a < 1$. Choosing $a = 3/5$ provides the fastest rate of convergence.

Before sketching the outline of the proof, we present the following lemma which provides a decomposition of the noise $\xi_k(\theta_{k-1}, \theta_k)$ — one of the key result used in the proof of the main theorem. The lemma and its proof are almost same as Lemma A.5 in [Lia10] with the only difference that unlike [Lia10], the iterates are of SGD, we need to prove it for the iterates of Algorithm 1. We provide the proof in Appendix A.

Lemma 3.1 Let Assumption 2.1-2.4 be true. Then the following decomposition takes place:
\[ \xi_k(\theta_{k-1}, \theta_k) = e_k + \nu_k + \zeta_k, \]
where, $\{e_k\}$ is martingale difference sequence, $\mathbb{E} \|\nu_k\|_2 \leq \eta_k$, and $\zeta_k = (\tilde{\zeta}_k - \tilde{\zeta}_{k+1})/\eta_k$, where $\mathbb{E} \|\tilde{\zeta}_k\|_2 \leq \eta_k$.

Outline of the proof of Theorem 3.1: A key step in the analysis of Algorithm 1 involves controlling the expectation of interaction with noise of the form $(\nabla f(\theta_k) - \nabla f(\theta_{k-1}), \xi_k(\theta_{k-1}, \theta_k))$. For iid or martingale difference data it is easy to control because $\mathbb{E} [\nabla f(\theta_k) - \nabla f(\theta_{k-1}), \xi_{k+1}(\theta_{k-1}, \theta_{k+1})] | \mathcal{F}_k] = 0$. But this is no longer true for Markov chain data. To resolve the issue, first notice that under our assumptions, the noise sequence $\xi_k$ can be decomposed into the sum of a martingale difference sequence $\{e_k\}$ and some residual terms $\{\nu_k\}$, and $\{\zeta_k\}$ as shown in Lemma 3.1. Then the key step is to introduce a different sequence of hypothetical iterates $(\hat{\theta}_k, \hat{y}_k, \hat{z}_k)$ for which the noise is small enough so that we can bound $\mathbb{E} \left[ V(\hat{\theta}_k, \hat{z}_k) \right]$, and then show that these hypothetical iterates and the original sequence generated by Algorithm 1 are close enough so that $\mathbb{E} [V(\theta_k, z_k)]$ is of the same order as $\mathbb{E} \left[ V(\hat{\theta}_k, \hat{z}_k) \right]$. This step is the main novelty of the proof.

Specifically, consider the following sequence:
\[ \hat{\theta}_0 = \theta_0 \quad \hat{z}_0 = z_0 \]
\[ \hat{y}_k = \argmin_{y \in \Theta} \left\{ \hat{z}_k, y - \hat{\theta}_k \right\} + \frac{\beta}{2} \|y - \hat{\theta}_k\|_2^2 \]
\[ \hat{\theta}_{k+1} = \hat{\theta}_k + \eta_{k+1} (\hat{y}_k - \hat{\theta}_k) \]
\[ \hat{z}_{k+1} = z_{k+1} + \tilde{\zeta}_{k+2} \]
This also means,
\[ \hat{z}_{k+1} = (1 - \alpha \eta_{k+1}) \hat{z}_k + \alpha \eta_{k+1} (\nabla f(\theta_k) + \tilde{\zeta}_{k+1}), \]
where, $\tilde{\zeta}_k = e_k + \nu_k + \zeta_k$. Note that by Lemma 3.1, $\mathbb{E} [e_k] = 0$, and $\mathbb{E} \|\nu_k + \tilde{\zeta}_k\|_2 \leq \eta_k$. First we show that by choosing $\eta_k = (N + k)^{-a}$, $1/2 < a < 1$, and $t_k = 1/\eta_k^2$ one has $\mathbb{E} \|\hat{\theta}_k - \theta_k\|_2^2 = O \left( N^{2-4a} \right)$, and $\mathbb{E} [V(\theta_k, z_k)] \leq 2 \mathbb{E} \left[ V(\hat{\theta}_k, \hat{z}_k) \right] + O \left( N^{2-4a} \right)$. Then we establish the bound on $V(\theta_k, z_k)$. Combining the above two facts proves Theorem 3.1. We defer the detailed proof to Appendix A.1.
3.1 State-independent Markov Chain

While our main goal in this work is to analyze Algorithm 1 for constrained nonconvex optimization with state-dependent Markov chain data, we provide the following result on the complexity of Algorithm 1 for Markov chain data with state-independent transition kernel for the sake of completion. Here we use $P$ to denote the transition kernel (as opposed to $P_\theta$ for state-dependent kernel). Note that under Assumption 2.4(a), for each $\theta$, the chain is $V$-uniformly ergodic, and hence, exponentially mixing [MT12] in the following sense:

**Definition 3** A Markov chain is said to be exponentially mixing, if there exists $C, r > 0$ such that,

\[ \| P^n(x, \cdot) - \pi \|_V \leq C \exp(-rn), \]  

(25)

where $P^n(x, \cdot)$ is the distribution of $X_n$ with initial state $X_0 = x$.

Now we present our result on the complexity of Algorithm 1 to find an $\epsilon$-stationary solution to (1) for exponentially-mixing Markov chain data with state-independent transition kernel.

**Theorem 3.2** Let Assumption 2.1-2.3 be true. Let Assumption 2.4(a)-(b) be true with $P_\theta$ replaced by $P$. Then, for Algorithm 1,

(a) when the projection operator is available, choosing

\[ \eta_k = 1/\sqrt{N}, \quad \beta = 1 \]  

for $k = 1, 2, \cdots, N$ we have

\[ \mathbb{E} [V(\theta_R, z_R)] = O \left( \log N/\sqrt{N} \right), \]

(b) when Algorithm 2 is used, choosing

\[ \eta_k = 1/\sqrt{N}, \quad t_k = \lceil \sqrt{k} \rceil, \quad \beta = 1, \quad \omega = 1, \quad \mu_i = 2/(i+2) \]  

(27)

for $k = 1, 2, \cdots, N$ we have

\[ \mathbb{E} [V(\theta_R, z_R)] = O \left( \log N/\sqrt{N} \right), \]

where the expectation is taken with respect to all the randomness of the algorithm, and an independent integer random variable $R \in \{1, 2, \cdots, N\}$ whose probability mass function is given by,

\[ P(R = k) = \eta_k / \sum_{k=1}^{N} \eta_k \quad k \in \{1, 2, \cdots, N\}. \]

We defer the proof to the Appendix.

**Remark 3** To find an $\epsilon$-stationary point, the total number of calls to SFO and LMO are $\tilde{O}(\epsilon^{-2})$, and $\tilde{O}(\epsilon^{-3})$, where $\tilde{O}()$ denotes the order ignoring logarithmic factors.

**Remark 4** The authors of [AL22] obtain the same rate as in Theorem 3.2 for constrained (but projection-based) nonconvex optimization with state-independent exponentially mixing data. In the state-dependent case, since the transition kernel of the Markov chain is controlled by $\theta_k$, and the transition kernel is assumed to be only Lipschitz smooth in $\theta$ (15), the chain does not necessarily exponentially mix. In the state-independent case, since the chain mixes exponentially we obtain the same rate as well. While their results are for projection-based algorithms, we analyze a projection-free LMO-based algorithm since LMO is often computationally cheaper than projection.

4 Experimental Evaluation

4.1 Strategic Classification

In this section we illustrate our algorithm on the strategic classification problem as described in Section 1.1 with the GiveMeSomeCredit\(^4\) dataset. The main task is a credit score classification

\(^4\)Available at [https://www.kaggle.com/c/GiveMeSomeCredit/data](https://www.kaggle.com/c/GiveMeSomeCredit/data)
problem where the bank (learner) has to decide whether a loan should be granted to a client. Given the knowledge of the classifier the clients (agents) can distort some of their personal traits in order to get approved for a loan. Here we use a 2-layer neural network with width $m$ as the classifier, given by

$$h(x; W, A, B) = \sum_{i=1}^{m} A_i v(W_i^\top x + B_i),$$

where $v(\cdot)$ is the activation function, $W_i \in \mathbb{R}^{d_i \times d}$, $W = [W_1, W_2, \ldots, W_d]^\top \in \mathbb{R}^{m \times d}$, $A = (A_1, A_2, \ldots, A_m) \in \mathbb{R}^m$, $B = (B_1, B_2, \ldots, B_m) \in \mathbb{R}^m$. We will use $\theta$ to collectively denote $(W, A, B)$. We impose the constraint of sparsity on the classifier given by $\|\theta\|_1 \leq R$ for some $R > 0$. As loss function we consider logistic loss as shown in (2). We consider a quadratic cost given by $c(x, x') = \|x_S - x_S'\|_2^2/(2\lambda)$ where $\lambda$ is the sensitivity of the underlying distribution on $\theta$. We assume that the agents iteratively learn $x^*_S$ similar to [LW22]. Note that unlike [LW22], the closed form of best response is not known here. So we assume that the agents use Gradient Ascent (GA) to learn the best response. For $\|\theta\|_1 \leq R$ constraint, the LMO in Algorithm 2 at iteration $k$ is given by $- R \max_i q_i$, where $i = \arg \max \{q_i\} \in \{0, \ldots, d\}$, and $q_i$ is the $i$-th coordinate of $q$. We select a subset of randomly chosen $M = 2000$ samples (agents) such that the dataset is balanced. Each agent has 10 features. Note that since Algorithm 1 computes the gradient on one sample at every iterate, the computation time is independent of the total number of agents. We assume that the agents can modify Revolving Utilization, Number of Open Credit Lines, and Number of Real Estate Loans or Lines. In this experiment we set $n_1 = 200$. Similar to [LW22], we set $\alpha = 0.5\lambda$, and $\lambda = 0.01$. For the classifier, the activation function is chosen as sigmoidal, and $m = 400$. We set $N = 20000$, and $R = 4000$. All the parameters of Algorithm 1 are chosen as described in (19). Figure 4.1 shows that Algorithm 1 finds an $\epsilon$-stationary point of the strategic classification problem. We show that Algorithm 1 performs comparably with Averaged Stochastic Approximation with the projection operator. Each curve in Figure 4.1 is an average of 50 repetitions.

### 4.2 Single Index Model with Trace-norm Ball Constraint

In this section we illustrate our algorithm on a synthetic example of single-index model regression with a nuclear-norm constraint on the model parameter. Let $\|\cdot\|_s$ denote the nuclear norm. The features $\{x_k\}_k \in \mathbb{R}^{d_1 \times d_2}$ are a matrix-valued time-series given by,

$$x_k = Ax_{k-1} + E_k + W_k v\theta_k,$$

where $A \in \mathbb{R}^{d_1 \times d_1}$ is a matrix with spectral radius less than 1, $E_k \in \mathbb{R}^{d_1 \times d_2}$ is the noise matrix with each entry of $E_k$ is $\text{i.i.d.} \ N(0, 1)$ random variable, $W_k$ is a Bernoulli(0.5) random variable, and $v \in \mathbb{R}$. For a fixed $\theta_k = \theta$, $\{x_k\}_k$ has a stationary distribution as shown in Proposition 1 of [CXY21]. $\{E_k\}_k$, and $\{W_k\}_k$ are $\text{i.i.d}$ sequence. This Markov chain follows conditions (b) and (c) of Assumption 2.4 since the evolution of $x_k$ only involves linear terms in $\theta_k$. The responses $\{y_k\}_k$ are generated according to the following single index model,

$$y_k = g(x_k^\top \theta^*) + \tilde{E}_k,$$

where $\{\tilde{E}_k\}_k$ is an $\text{i.i.d}$ sequence of standard normal random variables, $\theta^* \in \mathbb{R}^{d_1 \times d_2}$ is a matrix with $\|\theta^*\|_s \leq 1$, and $g(\cdot): \mathbb{R} \to \mathbb{R}$ is the link function. For this experiment we choose $g(x) = 3x + \log V(\theta_k, z_k).$
Figure 2: Single-index model with nuclear-norm constraint: (Left): Performance of Algorithm 1 with and without the projection operator. (Middle): Test Mean Squared Error (MSE) with Algorithm 1 with and without the projection operator. (Right): $\|\theta - \theta^*\|_F$ with Algorithm 1 with and without the projection operator.

The advantages of conditional-gradient based method for nuclear-norm ball constrained problems have been studied extensively [JS10, Jag13, HJN15]. The main advantage of ICG-based method is that calculating the LMO in this case requires computation of the leading singular vector of gradient matrix whereas to calculate the projection on the trace-norm ball one needs to compute the complete singular value decomposition. Let $u_1, v_1$ are the leading left and right singular vectors of the noisy gradient matrix evaluated at $(\theta, x, y)$, $-2(y - g(x^\top \theta))g'(x^\top \theta)x$. Then the LMO is given by $-u_1v_1^\top$.

For this experiment we choose $d_1 = 10$, $d_2 = 20$, $\nu = 0.1$, and $N = 2000$. Rest of the parameters of Algorithm 1 are chosen according to Theorem 3.1. In Figure 2, we compare the projection-based and ICG based version of Algorithm 1 with respect to $V(\theta_k, z_k)$, test Mean Squared Error (MSE), and $\|\theta_k - \theta^*\|_F$ where $\|\cdot\|_F$ is the Fröbenius norm. Figure 2 shows that the performance of projection-based and the ICG-based versions of Algorithm 1 are almost same. Each plot in Figure 2 is the average of 50 repetitions.

5 Discussion

In this work we provide oracle complexity results for the stochastic conditional gradient algorithm to find an $\epsilon$-stationary point of a constrained nonconvex optimization problem with state-dependent Markovian data. In Theorem 3.1, we show that the number of calls to the SFO and LMO required by the stochastic conditional gradient-type method in Algorithm 1, with state-dependent Markovian data, is $O(\epsilon^{-2.5})$ and $O(\epsilon^{-3.5})$ respectively. To the best of our knowledge, these are the first oracle complexity results in this setting. In Theorem 3.2, we show that SFO and LMO complexity in the case of state-independent Markovian data is $O(\epsilon^{-2})$ and $O(\epsilon^{-3})$ respectively, which matches the corresponding results in the iid setting.

There are various avenues for further extensions. Establishing lower bounds on the oracle complexity of projection-free algorithms in the Markovian data setting is extremely interesting. It is also intriguing to establish upper and lower bounds on the oracle complexity for more general types of dependent data sequences arising in applications, including $\phi$ and $\alpha$ mixing sequences. Yet another exciting direction is that of designing algorithms adaptive to the dependency in the data that achieve potentially better oracle complexity bounds.
References

[ABRW12] Alekh Agarwal, Peter Bartlett, Pradeep Ravikumar, and Martin Wainwright. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Transactions on Information Theory*, 58(5):3235–3249, 2012. (Cited on page 1.)

[ACD+19] Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*, 2019. (Cited on page 1.)

[AL22] Ahmet Alacaoglu and Hanbaek Lyu. Convergence and complexity of stochastic subgradient methods with dependent data for nonconvex optimization. *arXiv preprint arXiv:2203.15797*, 2022. (Cited on pages 4, 5, and 8.)

[AMP05] Christophe Andrieu, Éric Moulines, and Pierre Priouret. Stability of stochastic approximation under verifiable conditions. *SIAM Journal on control and optimization*, 44(1):283–312, 2005. (Cited on pages 2, 4, 5, 6, 16, and 25.)

[Bar92] Peter L Bartlett. Learning with a slowly changing distribution. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 243–252, 1992. (Cited on page 1.)

[BJMO12] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Structured sparsity through convex optimization. *Statistical Science*, 27(4):450–468, 2012. (Cited on page 1.)

[BMP12] Albert Benveniste, Michel Méritier, and Pierre Priouret. *Adaptive algorithms and stochastic approximations*, volume 22. Springer Science & Business Media, 2012. (Cited on page 4.)

[Bor09] Vivek S Borkar. *Stochastic approximation: a dynamical systems viewpoint*, volume 48. Springer, 2009. (Cited on page 4.)

[BRS18] Jalaj Bhandari, Daniel Russo, and Raghav Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Conference on learning theory*, pages 1691–1692. PMLR, 2018. (Cited on page 4.)

[BS17] Amir Beck and Shimrit Shtern. Linearly convergent away-step conditional gradient for non-strongly convex functions. *Mathematical Programming*, 164(1-2):1–27, 2017. (Cited on page 4.)

[CDP15] Yang Cai, Constantinos Daskalakis, and Christos Papadimitriou. Optimum statistical estimation with strategic data sources. In *Conference on Learning Theory*, pages 280–296. PMLR, 2015. (Cited on page 1.)

[CXY21] Rong Chen, Han Xiao, and Dan Yang. Autoregressive models for matrix-valued time series. *Journal of Econometrics*, 222(1):539–560, 2021. (Cited on page 9.)

[DAJJ12] John C Duchi, Alekh Agarwal, Mikael Johansson, and Michael I Jordan. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012. (Cited on page 4.)

[DL22] Ron Dorfman and Kfir Y Levy. Adapting to mixing time in stochastic optimization with markovian data. *arXiv preprint arXiv:2202.04428*, 2022. (Cited on page 4.)

[DMPS18] Randal Douc, Eric Moulines, Pierre Priouret, and Philippe Soulier. *Markov chains*. Springer, 2018. (Cited on page 2.)
[DNPR20] Thinh T Doan, Lam M Nguyen, Nhan H Pham, and Justin Romberg. Convergence rates of accelerated markov gradient descent with applications in reinforcement learning. arXiv preprint arXiv:2002.02873, 2020. (Cited on page 4.)

[DX20] Dmitriy Drusvyatskiy and Lin Xiao. Stochastic optimization with decision-dependent distributions. arXiv preprint arXiv:2011.11173, 2020. (Cited on pages 2 and 4.)

[FR13] Simon Foucart and Holger Rauhut. An invitation to compressive sensing. In A mathematical introduction to compressive sensing, pages 1–39. Springer, 2013. (Cited on page 1.)

[FW56] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. Naval research logistics quarterly, 3(1-2):95–110, 1956. (Cited on page 4.)

[GKS21] Dan Garber, Atara Kaplan, and Shoham Sabach. Improved complexities of conditional gradient-type methods with applications to robust matrix recovery problems. Mathematical Programming, 186(1):185–208, 2021. (Cited on page 4.)

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization, 23(4):2341–2368, 2013. (Cited on page 1.)

[GRW20] Saeed Ghadimi, Andrzej Ruszczynski, and Mengdi Wang. A single timescale stochastic approximation method for nested stochastic optimization. SIAM Journal on Optimization, 30(1):960–979, 2020. (Cited on pages 2, 3, 5, 6, 22, 23, and 24.)

[GSK13] Yair Goldberg, Rui Song, and Michael R Kosorok. Adaptive q-learning. In From Probability to Statistics and Back: High-Dimensional Models and Processes–A Festschrift in Honor of Jon A. Wellner, pages 150–162. Institute of Mathematical Statistics, 2013. (Cited on page 1.)

[HJN15] Zaid Harchaoui, Anatoli Juditsky, and Arkadi Nemirovski. Conditional gradient algorithms for norm-regularized smooth convex optimization. Mathematical Programming, 152(1-2):75–112, 2015. (Cited on pages 4 and 10.)

[HK12] Elad Hazan and Satyen Kale. Projection-free online learning. In 29th International Conference on Machine Learning, ICML 2012, pages 521–528, 2012. (Cited on page 4.)

[HL16] Elad Hazan and Haipeng Luo. Variance-reduced and projection-free stochastic optimization. In International Conference on Machine Learning, pages 1263–1271, 2016. (Cited on page 4.)

[HMPW16] Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. Strategic classification. In Proceedings of the 2016 ACM conference on innovations in theoretical computer science, pages 111–122, 2016. (Cited on page 1.)

[Jag13] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In International Conference on Machine Learning, pages 427–435. PMLR, 2013. (Cited on pages 4, 10, 17, and 25.)

[JS10] Martin Jaggi and Marek Sulovský. A simple algorithm for nuclear norm regularized problems. In ICML, 2010. (Cited on page 10.)

[KMMW19] Belhal Karimi, Blażej Miasojedow, Eric Moulines, and Hoi-To Wai. Non-asymptotic analysis of biased stochastic approximation scheme. In Conference on Learning Theory, pages 1944–1974. PMLR, 2019. (Cited on page 1.)

[KY03] Harold Kushner and George Yin. Stochastic approximation and recursive algorithms and applications, volume 35. Springer Science & Business Media, 2003. (Cited on page 4.)

[Lan20] Guanghui Lan. First-order and stochastic optimization methods for machine learning. Springer, 2020. (Cited on page 1.)
[Lia10] Faming Liang. Trajectory averaging for stochastic approximation mcmc algorithms. *The Annals of Statistics*, 38(5):2823–2856, 2010. (Cited on pages 7 and 16.)

[LJJ15] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pages 496–504, 2015. (Cited on page 4.)

[LP66] Evgeny Levitin and Boris Polyak. Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5):1–50, 1966. (Cited on page 4.)

[LW22] Qiang Li and Hoi-To Wai. State dependent performative prediction with stochastic approximation. In *International Conference on Artificial Intelligence and Statistics*, pages 3164–3186. PMLR, 2022. (Cited on pages 1, 2, and 9.)

[LZ16] Guanghui Lan and Yi Zhou. Conditional gradient sliding for convex optimization. *SIAM Journal on Optimization*, 26(2):1379–1409, 2016. (Cited on pages 1 and 4.)

[MB11] Eric Moulines and Francis Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. *Advances in neural information processing systems*, 24, 2011. (Cited on page 1.)

[MDPZH20] Celestine Mendler-Dünner, Juan Perdomo, Tijana Zrnic, and Moritz Hardt. Stochastic optimization for performative prediction. *Advances in Neural Information Processing Systems*, 33:4929–4939, 2020. (Cited on page 1.)

[MHK20] Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *Journal of machine learning research*, 2020. (Cited on pages 4 and 5.)

[Mig94] Athanasios Migdalas. A regularization of the frank—wolfe method and unification of certain nonlinear programming methods. *Mathematical Programming*, 65(1):331–345, 1994. (Cited on page 4.)

[MT12] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012. (Cited on page 8.)

[Nes18] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018. (Cited on page 3.)

[QLX18] Chao Qu, Yan Li, and Huan Xu. Non-convex conditional gradient sliding. In *International Conference on Machine Learning*, pages 4208–4217. PMLR, 2018. (Cited on page 4.)

[QW20] Guannan Qu and Adam Wierman. Finite-time analysis of asynchronous stochastic approximation and q-learning. In *Conference on Learning Theory*, pages 3185–3205. PMLR, 2020. (Cited on page 1.)

[RSPS16] Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. Stochastic frank-wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1244–1251. IEEE, 2016. (Cited on page 4.)

[RSS12] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on International Conference on Machine Learning*, pages 1571–1578, 2012. (Cited on page 1.)

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. (Cited on page 1.)

[SSXY20] Tao Sun, Yuejiao Sun, Yangyang Xu, and Wotao Yin. Markov chain block coordinate descent. *Computational Optimization and Applications*, 75(1):35–61, 2020. (Cited on page 4.)
[SSY18] Tao Sun, Yuejiao Sun, and Wotao Yin. On markov chain gradient descent. *Advances in neural information processing systems*, 31, 2018. (Cited on page 4.)

[SZ13] Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International conference on machine learning*, pages 71–79. PMLR, 2013. (Cited on page 1.)

[TD17] Vladislav B Tadić and Arnaud Doucet. Asymptotic bias of stochastic gradient search. *The Annals of Applied Probability*, 27(6):3255–3304, 2017. (Cited on page 4.)

[WPT+21] Yafei Wang, Bo Pan, Wei Tu, Peng Liu, Bei Jiang, Chao Gao, Wei Lu, Shangling Jui, and Linglong Kong. Sample average approximation for stochastic optimization with dependent data: Performance guarantees and tractability. *arXiv preprint arXiv:2112.05368*, 2021. (Cited on page 4.)

[XBG22] Tesi Xiao, Krishnakumar Balasubramanian, and Saeed Ghadimi. A projection-free algorithm for constrained stochastic multi-level composition optimization. *arXiv preprint arXiv:2202.04296*, 2022. (Cited on pages 2, 4, 5, and 6.)

[XXLZ20] Huaqing Xiong, Tengyu Xu, Yingbin Liang, and Wei Zhang. Non-asymptotic convergence of adam-type reinforcement learning algorithms under markovian sampling. *arXiv preprint arXiv:2002.06286*, 2020. (Cited on page 4.)

[YSC19] Alp Yurtsever, Suvrit Sra, and Volkan Cevher. Conditional gradient methods via stochastic path-integrated differential estimator. In *International Conference on Machine Learning*, pages 7282–7291. PMLR, 2019. (Cited on page 4.)

[ZJM21] Kelly Zhang, Lucas Janson, and Susan Murphy. Statistical inference with m-estimators on adaptively collected data. *Advances in Neural Information Processing Systems*, 34, 2021. (Cited on page 1.)

[ZSM+20] Mingrui Zhang, Zebang Shen, Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. One-sample Stochastic Frank-Wolfe. In *International Conference on Artificial Intelligence and Statistics*, pages 4012–4023. PMLR, 2020. (Cited on pages 2, 4, and 5.)

**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]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (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]
   (b) Did you include complete proofs of all theoretical results? [Yes]

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]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report the average of 50 trails
   (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)? [No] We did not calculate the exact timings. However, our experiments are fairly small-scale ones run on a personal laptop computer, and our main contributions are theoretical.
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? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (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]