FULLY PROJECTION-FREE PROXIMAL STOCHASTIC GRADIENT METHOD WITH OPTIMAL CONVERGENCE RATES

YAN LI, XIAOFENG CAO, (Graduate Student Member, IEEE), AND HONGHUI CHEN
Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China
Corresponding author: Yan Li (liyan10@nudt.edu.cn)

ABSTRACT
Proximal stochastic gradient plays an important role in large-scale machine learning and big data analysis. It needs to iteratively update models within a feasible set until convergence. The computational cost is usually high due to the projection over the feasible set. To reduce complexity, many projection-free methods such as Frank-Wolfe methods have been proposed. However, those projection-free methods have to solve a linear programming problem for every update of models which still leads to high computational cost for a complex feasible set, and can be unbearable in practical scenarios. Motivated by this problem, we propose a fully projection-free proximal stochastic gradient method, which has two advantages over previous methods. First, it enjoys high efficiency. The proposed method does not conduct projection directly but finds an approximately correct projection point with a very low computational cost. Second, it achieves tight and optimal convergence rates. Our theoretical analysis shows that the proposed method achieves convergence rates of $O\left(\frac{1}{\sqrt{T}}\right)$ and $O\left(\frac{\log T}{T}\right)$ for convex and strongly convex functions, respectively. These convergence rates successfully match with the known lower bounds. Therefore, in this paper, we provide a valuable insight that some loss of accuracy of projection can improve the efficiency significantly, but does not impair convergence rates. Finally, empirical studies show that the proposed method achieves more than 5× speedup than previous methods.

INDEX TERMS
Proximal stochastic gradient, projection-free optimization, convergence optimization.

I. INTRODUCTION
Proximal stochastic gradient method is widely used to solve large-scale machine learning problems such as support vector machines [1], [2] and logistic regression [3]. Generally, it iteratively finds a descent direction, and then updates the model within a feasible set by following the direction until convergence. Finding such descent direction requires to solve a constrained convex programming problem. Specifically, the standard projected proximal gradient method usually needs to solve a Quadratic Programming (QP) problem with convex constraints to find the descent direction. This translates to apply a projection operation over the feasible set. However, for high dimensional or complex learning problems, solving the QP problem can be very infeasible and not practical since the projection step is always time consuming and can potentially become the computation bottleneck.

To reduce such computational cost, many projection-free methods such as conditional gradient methods [4] or variants of Frank-Wolfe methods [5]–[7] have been proposed. These methods require to solve a Linear Programming (LP) problem instead of a QP problem to find the descent direction. In this regard, they enjoy a higher efficiency than the projected methods. However, solving the LP problem can still lead to a high computational cost when the feasible set is complex. Moreover, since there are usually hundreds or even thousands of iterations, the computational cost can be extremely high, and thus is unbearable in practical scenarios. Therefore, we are motivated by following questions:

- Is there a fully projection-free proximal gradient method, which does not have to solve an LP problem to update models?
- Does such projection-free proximal method achieve the tight and optimal convergence rate?

1 In the paper, the feasible set has the same meaning with the constraints.
In this paper, we give positive answers to these motivating questions. First, we propose a new projection-free proximal method. Inspired by [8], our basic idea is to use the approximate projection point to update models, instead of finding the exactly accurate projection point. The insight is to find an approximately exact direction, and then to search a boundary point by following the direction as the final “projection” point. The accuracy of the approximate direction is essentially important, which is significantly impacted by the curvature of the set. It thus does not have to solve the LP problem, and improves the efficiency significantly.

Second, our theoretical analysis shows that the proposed method successfully achieves tight and optimal convergence rates, which are $O\left(\frac{1}{\sqrt{T}}\right)$ and $O\left(\frac{\log T}{T}\right)$ for convex and strongly convex functions, respectively. We prove that although the approximate projection leads to some loss of accuracy, it does not impair the performance of convergence theoretically. Additionally, extensive empirical studies show that the proposed method has almost the best convergence performance and the highest efficiency over the previous methods. Especially, comparing with previous methods, the proposed method can achieve more than $5 \times$ speedup.

In addition, our major contributions are highlighted as follows:

- We propose a new fully projection-free proximal stochastic gradient method, which enjoys high efficiency.
- The proposed method achieves $O\left(\frac{1}{\sqrt{T}}\right)$ convergence rate for the convex function, which matches with the known lower bound, and is thus tight and optimal.
- The proposed method achieves $O\left(\frac{\log T}{T}\right)$ convergence rates for the strongly convex function, which is thus tight and optimal.

The rest of the paper is organized as follows: Section II discusses the related work. Section III setups the problem formally. Our new projection-free algorithm is presented and its detailed convergence analysis are presented and discussed in Section IV and Section V, followed by experiment details in VI. Finally, Section VII concludes the paper alongside with the future research direction.

II. RELATED WORK

Recent years have witnessed the proliferation of the studies on the projection-free learning and optimization. Here, we briefly review the previous studies and divide the related work into the broad categories as following.

A. PROJECTION-FREE OFFLINE OPTIMIZATION

The conditional gradient descent (a.k.a Frank-Wolfe algorithm) was originally proposed by Frank et al. [9] for offline smooth optimization over polyhedral sets. It was further extended to semi-definite programming by Hazan [10] and general convex optimization by Jaggi [5]. Braun et al. [11] proposed a blended conditional gradient approach by combining the conditional gradient with gradient-based steps. Although these projection-free techniques have been effective in the offline setting, they have yet to be as extensively applied in the online setting.

B. PROJECTION-FREE ONLINE OPTIMIZATION

The online variant of conditional gradient descent was proposed by Hazan and Kale [12] which achieves a suboptimal $O\left(\frac{1}{T^2}\right)$ regret. An improved online conditional gradient descent was designed by Garber and Hazan [13] for smooth and strongly convex optimization which achieves the optimal $O\left(\sqrt{T}\right)$ regret. In addition, many other methods have been proposed to handle cumulative constraints [14]–[16]. Recently, Hazan and Minasyan [17] proposed a randomized projection-free algorithm for general online convex optimization with smooth loss functions, and guaranteed a $O\left(\frac{1}{T}\right)$ regret. However, these algorithms can perform arbitrarily poorly if supplied with stochastic gradient estimates.

C. PROJECTION-FREE STOCHASTIC OPTIMIZATION

Johnson and Zhang [18] introduced the variance reduction technique for stochastic gradient descent. It was extended by Xiao and Zhang [19] to the proximal setting. Nitanda [20] applied this technique for accelerating proximal stochastic gradient in the mini-batch setting. However, each of these methods requires an expensive projection step at each iteration. Mahdavi et al. [21] introduced the primal-dual machinery for nearly projection-free algorithm that needs only one projection for the last iteration. This method was later developed in [22] with a complex inequality constraint. Hazan and Luo [23] devised a conditional gradient descent algorithm based on the variance reduction technique. Lan and Zhou [24] proposed a stochastic condition gradient sliding algorithm using Nesterov’s acceleration method. It has been further improved in [4] by introducing the lazy conditional gradient. Xie et al. [25] introduced the recursive variance reduction technique to reduce noise in stochastic gradients prediction. Although these methods have enjoyed success in reducing projection cost, they require either at least one projection or linear optimization both of which can be prohibitive in large scale learning problems.

Our work is closely related to the very recent work of fast approximate projection method proposed in [8]. It introduced the interior point technique in solving online gradient descent which achieves $O\left(\sqrt{T}\right)$ and $\log T$ regret for convex and strongly convex online optimization, respectively. Our proposed method is inspired by [8], but extends it to the proximal stochastic setting.

III. PRELIMINARIES

A. NOTATIONS

Before presenting the work, we illustrate some important notations. $\mathcal{X}$ represents the feasible set, and $\text{bound}(\mathcal{X})$ represents its boundary. $\nabla$ and $\partial$ represent the gradient and the
sub-gradient operators, respectively. \(\|a\|_q\) represents the \(\ell_q\) norm of \(a\) with \(q \in \{1, 2, \infty\}\). \(\|a\|_2\) represents the \(\ell_2\) norm of \(a\) in default. \(\mathbb{E}\) represents the mathematical expectation. \(\precsim\) represents “less than up to a constant”. For example, \(a \precsim b\) means there exists a constant \(c > 0\) such that \(a \leq bc\).

We consider the following optimization problem:

\[
\min_{x \in \mathcal{X}} F(x) := f(x) + r(x). \tag{1}
\]

Here, \(f(x)\) is (strongly) convex, but may be non-smooth. Similarly, \(r(x)\) is convex, but may be non-smooth. The feasible set \(\mathcal{X}\) is denoted by \(\mathcal{X} := \{h(x) \leq 0\}\), where \(h(x)\) is convex and smooth with respect to \(x\). Many machine learning problems can be formulated as this type, we now give a typical example:

**Example 1:** Let us take the risk-averse learning task as an example. Generally, the risk-averse learning aims to learn the optimal model, and simultaneously avoids its performance to change vastly for different instances. It usually has the constraint

\[
\mathcal{X} = \left\{ x : \frac{1}{n} \sum_{t=1}^{n} \left( f_t(x) - \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right)^2 - \iota \leq 0 \right\}, \tag{2}
\]

where \(f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x)\), and \(\iota\) is a given positive constant. For example, the portfolio management is one of those tasks.

Suppose an investor has \(d\) assets in the market. We let \(r_t \in \mathbb{R}_+^d\) denote their rewards observed at the time \(t \in [n]\). The portfolio management is to maximize the mean of total rewards with bounded risk. In the task, \(f_t(x) = -r_t^T x\), and the constraint is to bound the variance. Here, \(r(x) = \lambda \|x\|_2\), and \(\lambda\) is a given constant.

**B. PROXIMAL PROJECTED GRADIENT DESCENT**

The proximal gradient method is often used to solve the optimization problem (1). It performs the \((t+1)\)-th update of model by:

\[
x_{t+1} = P_{\mathcal{X}}(x_t - \eta_t g_t), \tag{3}
\]

where \(\eta_t\) is the learning rate, \(g_t\) is an unbiased estimation of \(f_t\) with \(f_t \in \partial f(x_t)\), that is \(\mathbb{E} g_t = f_t\). \(P_{\mathcal{X}}\) is the projection operator onto a set \(\mathcal{X}\) which is given by:

\[
P_{\mathcal{X}}(v) = \arg\min_{x \in \mathcal{X}} \frac{1}{2} \|v - x\|^2
\]

\[
= \arg\min_{x} \left\{ \frac{1}{2} \|v - x\|^2 + I_{\mathcal{X}}(x) \right\}, \tag{4}
\]

where \(I_{\mathcal{X}}(x) = 0\) if \(x \in \mathcal{X}\), and \(\infty\) otherwise. Thus, at each iteration \(t = 0, 1, 2, \ldots\), proximal gradient method updates its model to the opposite direction of the gradient, and then projects onto \(\mathcal{X}\).

A stochastic variant of proximal projected gradient is stochastic proximal gradient descent. In this setting, we randomly pick \(i_t\) from \([1, 2, \ldots, n]\) at each iteration \(t = 1, 2, \ldots\), and take the following update:

\[
x_{t+1} = P_{\mathcal{X}}(x_t - \eta_t g_t, i_t).
\]

**Algorithm 1 FAP: Fully Projection-Free Proximal Gradient Method**

**Require:** Initialize the sequence of learning rate \(\{\eta_t\}_{t=0}^{T-1}\), number of iterations \(T\), and \(x_0 \in \mathcal{X}\).

1: for \(t = 0, 1, 2, \ldots, T-1\) do
2: Play \(x_t\), and query a stochastic gradient \(g_t\) of \(f_t\) with \(f_t \in \partial f(x_t)\) such that \(\mathbb{E} g_t = f_t\).
3: Update model by

\[
x_{t+\frac{1}{2}} = \arg\min_{x \in \mathbb{R}^d} (\langle g_t, x - x_t \rangle + \frac{1}{2\eta_t} \|x - x_t\|^2 + r(x))\tag{6}
\]

4: if \(x_{t+\frac{1}{2}} \in \mathcal{X}\) then \(x_{t+1} = x_{t+\frac{1}{2}}\).
5: else
6: \(x_{t+1} = \text{AProj}_{\mathcal{X}} \left( x_t, x_{t+\frac{1}{2}} \right)\) according to Algorithm 2.
7: return \(\frac{1}{T} \sum_{t=1}^{T} x_t\).

For stochastic proximal gradient descent, it only needs to compute a single stochastic gradient \(g_{t,i_t}\) at each iteration\(^2\); in contrast, proximal projected gradient requires the computation of \(n\) gradients. Although the computational cost of stochastic proximal gradient is \(1/n\) that of the proximal projected gradient per iteration, the projection step is still computationally expensive, making stochastic proximal gradient impractical for large-scale optimization problem.

Combining 3 and 4, we rewrite the model update rule as following:

\[
x_{t+1} = \arg\min_{x \in \mathcal{X}} (\langle g_t, x - x_t \rangle + \frac{1}{2\eta_t} \|x - x_t\|^2 + r(x)), \tag{5}
\]

where \(r(x)\) is a regularizer, e.g., \(r(x) = \|x\|_1\). When \(r(x)\) is convex with respect to \(x\), (5) is a constrainedQP problem.

Solving (5) usually leads to a very high computational cost. Notice that, there are always more than hundreds of updates of models, and every update has to conduct one projection over \(\mathcal{X}\). Therefore, the total computational cost is still extremely high for large-scale optimization problem.

**IV. FULLY PROJECTION-FREE PROXIMAL STOCHASTIC GRADIENT**

In this section, to efficiently solve the large-scale optimization problem, we propose a fully projection-free proximal stochastic gradient (FAP) method to largely reduce the computational cost. We first discuss the algorithm design and later establish its employment analysis.

\[
b = \arg\min_{x \in \mathbb{R}^d} (\langle g_t, x - a \rangle + \frac{1}{2\eta_t} \|x - a\|^2 + r(x)) \tag{7}
\]

**A. ALGORITHM DESIGN**

The detailed procedure of our proposed FAP algorithm is shown in Algorithm 1. At each iteration \(t = 0, 1, 2, \ldots\),

\(^2\)For the notation simplicity, we do not clearly distinguish \(g_t\) and \(g_{t,i_t}\) in the following paper.
Specifically, consider $a \in X$ and $b \notin X$, we use Figure 1 to illustrate how to find the approximate projection of $b$ over $X := \{x : h(x) \leq 0\}$ based on $a$. We first find $z$ which is the intersection of $b - a$ and the boundary of $X$ (i.e., $\partial X$). The approximate direction is the normal vector $n$ at $z$, where the normal vector is defined as following.

**Definition 1 (Normal vector):** Given a closed convex set $X$ and $z \in \partial X$, $n$ is a normal vector to $X$ at $z$, if

$$n \cdot (y - z) \leq 0, \forall y \in X$$  \hspace{1cm} (8)

Specifically, if $\|n\| = 1$, $n$ is a unit normal vector. Hence, the normal vector $n$ at $z$ can be calculated as $n = \frac{\nabla h(z)}{\|\nabla h(z)\|}$.

We then denote $s = b - z$. The projection of $s$ on the linear subspace orthogonal to $n$ can be calculated as $s - \langle n, s \rangle n$. Therefore, the approximated projection is the intersection of $\{b - \langle n, s \rangle n - \theta n : \theta \geq 0\}$ and the boundary of $X$, i.e., the point $\text{AProj}_X(a, b)$ in Figure 1. Its algorithmic details are presented in Algorithm 2.

Recall that the full projection operation $P_X(b)$ is to find the closest point to $b$ in $X$. Since $n$ is the normal vector at $z$ to $X$, $b - \langle n, s \rangle n$ is intuitively closer to $b$ than any other point in $X$. Notice that $b - \langle n, s \rangle n$ may potentially not in $X$. In this case, we set $b - \langle n, s \rangle n$ as the starting point, then search following the direction of $n$ and find the closest point of $b - \langle n, s \rangle n$ in $X$.

**Algorithm 2 AProj$_X(a, b)$ [8]: Approximate Projection of $b$ Over $X := \{x : h(x) \leq 0\}$ Based on $a$**

1. Find $z \in (a, b) \cap \text{bound}(X)$. \hspace{1cm} $\triangleright$ Need $O\left(\log \frac{1}{\epsilon}\right)$ queries of $h(.)$.
2. Denote auxiliary variables $s$ and $n$ by $s = b - z$ and $n = \frac{\nabla h(z)}{\|\nabla h(z)\|}$.
3. Find the minimal $\theta_s$ such that $\theta_s \in \{\theta : \theta \geq 0, \text{and } z_\theta \in X\}$, where \(z_\theta := b - \langle n, s \rangle n - \theta n\). \hspace{1cm} $\triangleright$ Need $O\left(\log \frac{1}{\epsilon}\right)$ queries of $h(.)$.
4. return $\hat{z}_{\theta_s}$.

Therefore, we can efficiently find the approximate projection point.

**B. IMPLEMENTATION ANALYSIS**

The implementation of Algorithm 1 is highly efficient. The most expensive step is to find the approximate projection point, that is Algorithm 2. Specifically, Algorithm 2 needs to calculate the $b$ and $\theta_s$. Since $h(a) \leq 0$ and $h(b) > 0$, we can use bisection search to find $z$, which requires $O\left(\log \frac{1}{\epsilon}\right)$ queries of $h(.)$ to achieve at most $\epsilon$ error, e.g., $|h(z) - h(z_{\text{exact}})| \leq \epsilon$. Now suppose there is a point $y \in X$ and $y$ is also on the $\{b - \langle n, s \rangle n - \theta n : \theta \geq 0\}$. In this setting, we have $h(y) \leq 0$, and $h(b - \langle n, s \rangle n) > 0$. Thus, we can again use bisection search to find the minimum $\theta_s$, which still requires $O\left(\log \frac{1}{\epsilon}\right)$ queries of $h(.)$ for $\epsilon$ error.

**V. CONVERGENCE ANALYSIS**

In this section, we present our analysis of convergence rate of the proposed FAP algorithm. Before presenting convergence rates, we present some assumptions, which will be used in theoretical analysis.

**Assumption 1:** We make some basic assumptions as follows.

- We assume both $f(x)$ and $r(x)$ are convex but non-smooth with respect to $x$.
- We assume $\|f\| \leq G$ and $\|e\| \leq H$, where $f \in \partial f(x)$ and $e \in \partial r(x)$ for any $x \in X$.
- We assume $g$ is an unbiased estimation of $f$ with $f \in \partial f(x)$, that is, $E g = f$, and $E\|g - f\| \leq \sigma$ for any $x \in X$.
- We assume $\|x - y\| \leq R$ for any $x \in X$ and $y \in X$.

**Assumption 2:** We assume $f(x)$ is $\alpha$-strongly convex with respect to $x$, that is,

$$f(x) - f(y) \geq \langle f', x - y \rangle + \frac{\alpha}{2}\|x - y\|^2$$

holds for any $x \in X$ and $y \in X$.

It is trivial to verify that a general convex function is also a 0-strongly convex function. In the theoretical analysis, we use

$$\mathcal{R}_T^{\text{ave}} := \mathbb{E} F \left( \frac{1}{T} \sum_{t=1}^{T} x_t \right) - F(x_s)$$

VOLUME 8, 2020

165907
to measure the distance between the model yielded by Algorithm 1 and the optimal model. Here, $x_*$ represents the optimal model, that is, $x_* = \text{argmin}_{x \in \mathcal{X}} F(x)$.

**Definition 2 (Smoothness):** A function $\phi(x) : \mathcal{X} \mapsto \mathbb{R}$ is $L$-smooth with $L > 0$ when we have

$$\phi(x) - \phi(y) \leq \langle \nabla \phi(y), x - y \rangle + \frac{L}{2} \|x - y\|^2$$

for any $x \in \mathcal{X}$ and $y \in \mathcal{X}$.

**Definition 3 (Global curvature):** The global curvature of $\mathcal{X}$ is defined by

$$\mu = \max_{x \in \text{bound}(\mathcal{X})} \frac{L_h}{\|\nabla h(x)\|},$$

where $\text{bound}(\mathcal{X})$ represents the boundary of $\mathcal{X}$.

**Lemma 1 (Theorem 3.1 in [8]):** Suppose $h(\cdot)$ is convex and smooth, and the feasible set $\mathcal{X}$ is defined by $\mathcal{X} := \{u : h(u) \leq 0\}$ with the global curvature $\mu$. Given any $a \in \mathcal{X}$, $b \notin \mathcal{X}$, and its approximate projection $\text{AP}x(a, b)$ yielded by Algorithm 2, the following inequality

$$\|\text{AP}x(a, b) - y\| \leq \|b - y\| + \mu \|a - b\|^2$$

holds for any $y \in \mathcal{X}$.

Lemma 1 is a guarantee of the approximate projection. We recommend [8] for proof details.

**Lemma 2:** If $\eta_t > 0$, we have

$$\|x_{t+1} - x_t\| \leq \eta_t (\sigma + G + H),$$

and

$$\|x_{t+1} - x_t\| \leq \eta_t (\sigma + G + H) + \eta_t^2 \mu (\sigma + G + H)^2.$$

**Proof:** According to the first-order optimization condition of (6), that is, there is $e \in \partial h(x_{t+1/2})$ satisfying

$$x_{t+1/2} - x_t = \eta_t (-g_t - e).$$

Thus, we have

$$\|x_{t+1/2} - x_t\| \leq \eta_t \|g_t + e\| \leq \eta_t \|g_t - f_t\| + \eta_t \|f_t\| + \eta_t \|e\| \leq \eta_t (\sigma + G + H).$$

It completes the first part of proof. According to Lemma 1, we have

$$\|x_{t+1} - x_t\| \leq \eta_t (\sigma + G + H) + \eta_t^2 \mu (\sigma + G + H)^2.$$

It finally completes the proof.

**Lemma 3:** Given any vectors $g_t, u_t \in \mathbb{R}^d$, $u_* \in \mathbb{R}^d$, and a constant scalar $\eta > 0$, if

$$u_{t+1} = \text{argmin}_{u \in \mathbb{R}^d} \langle g, u - u_t \rangle + \frac{1}{2\eta} \|u - u_t\|^2 + r(u),$$

we have

$$\langle g, u_{t+1} - u_* \rangle \leq \frac{1}{2\eta} \left(\|u_* - u_t\|^2 - \|u_* - u_{t+1}\|^2\right).$$

**Proof:** It is trivial to be obtained by using Lemma 25 in [26].

**Lemma 4:** Choosing $\eta_t > 0$ and $\eta_{t+1} \leq \eta_t$ for any $t \in [T]$, we have

$$\frac{1}{T} \sum_{t=1}^{T} \langle g_t, x_{t+1} - x_* \rangle \leq \frac{\mu^2}{T} \sum_{t=1}^{T} \eta_t^3 + \frac{\mu}{T} \sum_{t=1}^{T} \eta_t + \frac{1}{T} \sum_{t=2}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) + 1 + \eta_1 T.$$


**Theorem 1:** Choosing a non-increasing sequence \(\{\eta_t\}_{t=1}^T\) in Algorithm 1, under Assumption 1, we have

\[
\mathcal{R}_T^{ave} \lesssim \frac{1 + \eta_1}{T \eta_1} + \frac{\mu + 1}{T} \sum_{t=1}^T \eta_t^3 + \frac{1}{T} \sum_{t=1}^T \left( \frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \frac{\alpha}{2} \right).
\]

**Proof:**

\[
\mathcal{R}_T^{ave} \leq \mathbb{E} \left[ \frac{1}{T} \sum_{t=1}^T (g_t(x_t, x_{t+1} - x_a) - f(x_t) - r(x_a)) \right] + \mathcal{I}_1 + \mathcal{I}_2 + \mathcal{I}_3,
\]

where \(f_t \in \partial f(x_t)\) and \(\mathbb{E} g_t = f_t\).

According to Lemma 4, we have

\[
\mathcal{I}_1 \lesssim \frac{\mu^2}{T} \sum_{t=1}^T \eta_t^3 + \frac{\mu + 1}{T} \sum_{t=1}^T \eta_t + \frac{1}{T} \sum_{t=2}^T \left( \frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + \frac{1 + \eta_1}{T \eta_1}.
\]

Let us begin to bound \(I_2\). We have

\[
I_2 = -\frac{1}{T} \sum_{t=1}^T (f(x_a) - f(x_t) - \langle f_t, x_t - x_a \rangle)
\]

\[
\leq -\frac{1}{T} \sum_{t=2}^T \alpha \|x_t - x_a\|^2.
\]

The last inequality holds due to the \(\alpha\) strong convexity of \(f(x)\).

Since \(\mathbb{E} g_t = f_t\), we have \(\mathbb{E} \frac{1}{T} \sum_{t=1}^T (f_t - g_t, x_t - x_a) = 0\).

Besides, we have

\[
\frac{1}{T} \sum_{t=1}^T (g_t, x_t - x_{t+1}) \leq \frac{1}{T} \sum_{t=1}^T \|g_t\| \|x_t - x_{t+1}\| \lesssim \frac{1}{T} \sum_{t=1}^T \eta_t.
\]

The last inequality holds due to Lemma 2. Substituting those bounds, we finally complete the proof. \(\square\)

According to Theorem 1, a larger \(\mu\) leads to a slower convergence. Recall that \(\mu\) is the global curvature of the feasible set \(\mathcal{X}\), and a large \(\mu\) means that \(\mathcal{X}\) looks more like a ‘circle’. Theorem 1 implies the following insight.

**Insight 1:** Theorem 1 shows that the convergence of Algorithm 1 is highly impacted by the geometry of the feasible set \(\mathcal{X}\).

It matches with intuitive understanding. Consider a special case: \(h(x)\) is linear with respect to \(x\). First, in the case, \(L_h = 0\), and thus \(\mu = 0\). Theorem 1 is exactly same with the known result of the proximal projected gradient method. Second, note that the approximate projection is the exact projection in the case of linear \(h(x)\). Therefore, Algorithm 1 becomes traditional proximal projected gradient method, and their convergence rates are exactly same for linear \(h(x)\).

After that, by choosing some specific assumptions about \(f(x)\), we obtain the following convergence rates.

**Corollary 1:** Choose \(\eta_t = \frac{1}{\sqrt{T}}\) for any \(t \in [T]\) in Algorithm 1. If \(f(x)\) is general convex (that is, 0-strongly convex) with respect to \(x\), we have

\[
\mathcal{R}_T^{ave} \lesssim \frac{1 + \log T}{T} + \frac{1}{T}.
\]

**Corollary 2:** Choose \(\eta_t = \frac{2}{a_t}\) with \(\alpha > 0\) for any \(t \in [T]\) in Algorithm 1. If \(f(x)\) is \(\alpha\)-strongly convex with respect to \(x\), we have

\[
\mathcal{R}_T^{ave} \lesssim \log T + \frac{1}{T}.
\]

**Tightness.** Algorithm 1 achieves \(\mathcal{O} \left( \frac{1}{\sqrt{T}} \right)\) and \(\mathcal{O} \left( \log T \right)\) convergence rates for the general convex \(f(x)\) and the strongly convex \(f(x)\), respectively. Previous researches shows that the lower bounds of convergence rate for convex and strongly convex are \(\Omega \left( \frac{1}{\sqrt{T}} \right)\) and \(\Omega \left( \log T \right)\), respectively. Therefore, our convergence rates are tight and optimal. It implies the following insight.

**Insight 2:** Some loss of accuracy due to the approximate projection helps to improve the efficiency, and is not harmful to the convergence rates.

**VI. EXPERIMENTS**

In this section, we show the learning tasks, experimental settings and conduct the comprehensive evaluation.

**A. LEARNING TASKS**

As shown in Section IV, we conduct empirical studies on two risk-averse learning tasks: Portfolio Management (PM) and Logistic Regression (LR). The feasible set is denoted by (2).

- **Portfolio Management.** Suppose an investor has \(d\) assets in the market. \(r_t \in \mathbb{R}^d_+\) denotes their rewards observed at the time \(t \in [n]\). In the task, \(f(x) = -r^\top x\), \(r(x) = \lambda \|x\|_1\), and \(\lambda\) is a given constant.

- **Logistic regression.** Suppose the data matrix \(A \in \mathbb{R}^{n \times d}\) consisting of \(n\) instances, and every instance has \(d\) features. The label matrix is \(y \in \{1, -1\}^n\), and \(y_i \in \{1, -1\}\) is the label of the \(i\)-th instance \(A_i \in \mathbb{R}^{1 \times d}\). In the task, we let \(f(x) = \log(1 + \exp(-y_i A_i x))\), and \(r(x) = \lambda \|x\|_1\). Similarly, \(\lambda\) is also a given constant.

**B. EXPERIMENTAL SETTINGS**

We collect eight wide-use datasets from the internet. For the PM task, we conduct the experiment on datasets of FF49Industries, DowJones, FTSE100, and NASDAQComp. \(^3\) For the LR task, we conduct the experiment on datasets of

\(^3\)These datasets can be available at https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.
TABLE 1. Statistics of datasets. ‘# inst.’ means ‘number of instances’, and similarly ‘# feat.’ means ‘number of features’.

| datasets       | # inst. | # feat. | task |
|----------------|---------|---------|------|
| FF49Industries | 2,325   | 49      | PM   |
| DowJones       | 1,363   | 28      | PM   |
| FTSE100        | 717     | 83      | PM   |
| NASDAQComp     | 685     | 1,203   | PM   |
| ijcnn1         | 49,990  | 22      | LR   |
| a9a            | 32,561  | 123     | LR   |
| w8a            | 49,749  | 300     | LR   |
| covtype        | 581,012 | 54      | LR   |

ijcnn1, a9a, w8a, and covtype. Details of those datasets are presented in Table 1.

We have two benchmarks to compare with our proposed method, i.e., FAP. 1) FWM, a strong projection-free optimization method in [11], and 2) PGD, an efficient optimization method with projection proposed in [27]. Besides, the optimal model is same for all those methods, and we use the empirical loss $F(x)$ as the metric to measure the performance of those methods. The learning rate $\eta_t$ for all those methods is set to be $\eta_t = \frac{1}{\sqrt{t}}$, and we set $\lambda = 10^{-3}$ in all empirical studies. For every iteration, we use a mini-batch of instances to update models, and the size of a mini-batch is $\frac{n}{10}$. Here, $n$ represents the number of instances in a dataset. Finally, we implement those methods by using Matlab 2016b. All codes are run in a server with an Intel I7 CPU, 16 GB memory, and 1TB SSD storage.

C. RESULTS
1) THE PROPOSED METHOD CAN ACHIEVE THE BETTER CONVERGENCE PERFORMANCE THAN BENCHMARKS

Figures 2 and 3 show the evaluation of the convergence performance. Figure 2 plots the results with the PM task and Figure 3 plots the results with the LR task. Figure 2 shows that FAP significantly outperform both the FWM and PGD with a large performance gap (or we can say with a xx% average performance gap) across all datasets. As the number of rounds increases, the performance gap remains stable in Figure 2. Meanwhile, the empirical loss of FWM and PGD is always at the same level as the round moves on. Figure 3 also shows that FAP always achieves the lowest empirical loss as compared with other methods. In particular, FAP is slightly better than PGD, but still hit a large performance gap compared with FWM. The empirical loss of FAP is much lower during the initial training rounds compared with the other two methods. In addition, the performance gap is more obvious on the large dataset, that is, covtype having 581,012 instances and 54 features. In summary, the proposed FAP method always achieves the optimal convergence rate as compared with other methods which validates our theoretical analysis.
2) THE PROPOSED METHOD CAN ACHIEVE THE HIGHER EFFICIENCY THAN BENCHMARKS

Table 2 shows the computation cost in terms of CPU seconds for each method. Values in brackets represents the variance. As a result, FAP significantly outperforms the other two benchmarks. Compared with the FWM and PGD, FAP can respectively achieve $149.25 \times$ and $190.02 \times$ speedup on dataset \textit{FF49Industries}, $156.31 \times$ and $193.95 \times$ speedup on dataset \textit{DowJones}, $153.07 \times$ and $202.40 \times$ speedup on dataset \textit{FTSE100}, $94.16 \times$ and $142.44 \times$ speedup on dataset \textit{NASDAQComp}, $337.74 \times$ and $377.17 \times$ speedup on dataset \textit{ijcnn1}, $42.09 \times$ and $47.36 \times$ speedup on dataset \textit{a9a}, $9.74 \times$ and $13.01 \times$ speedup on dataset \textit{w8a}, $4.74 \times$ and $5.24 \times$ speedup on dataset \textit{covtype}. The remarkable superiority of efficiency is due to the differences of model updating approach, where PGD has to solve a quadratic programming problem to update models, and FWM has to solve a linear programming problem to complete the update of models. Compared with those two updating approaches, FAP updates model with a fully projection-free approach by using proximal gradients, which can greatly enhance the efficacy of the computation.

TABLE 2. Comparison of computational cost (CPU seconds). Values in brackets represents the variance.

| time   | FWM          | PGD          | FAP          |
|--------|--------------|--------------|--------------|
| \textit{FF49.} | 65.67(±2.15) | 83.61(±2.19) | 0.44(±0.01)  |
| \textit{DowJ.} | 65.65(±2.15) | 81.46(±2.17) | 0.42(±0.01)  |
| \textit{FTSE.} | 65.82(±2.15) | 87.03(±2.17) | 0.43(±0.01)  |
| \textit{NASDAQ.} | 68.74(±2.16) | 103.98(±2.67) | 0.73(±0.01)  |
| \textit{ijcnn.} | 77.68(±1.23) | 86.75(±1.25) | 0.23(±0.01)  |
| \textit{a9a}  | 82.50(±3.35) | 92.83(±3.52) | 1.96(±0.09)  |
| \textit{w8a}  | 90.00(±3.21) | 120.26(±4.07) | 9.24(±1.01)  |
| \textit{covt.} | 101.11(±4.34) | 111.62(±4.31) | 21.32(±1.24) |

VII. CONCLUSION AND FUTURE WORK

We propose a new fully projection-free method, which updates models by using proximal gradients. Our theoretical analysis shows that the proposed method achieves the tight and optimal convergence rate. The superiority of the proposed method has been validated by extensive empirical studies.
In the future, the proposed method can be extended to handle the non-convex loss function, or to work in the decentralized- or asynchronous settings.

REFERENCES

[1] Y. Xu, I. Akrotirianakis, and A. Chakraborty, “Proximal gradient method for huberized support vector machine,” Pattern Anal. Appl., vol. 19, no. 4, pp. 989–1005, May 2015.

[2] N. Ito, A. Takeda, and K.-C. Toh, “A unified formulation and fast accelerated proximal gradient method for classification,” J. Mach. Learn. Res., vol. 18, no. 1, pp. 510–558, 2017.

[3] X. Shen and Y. Gu, “Nonconvex sparse logistic regression via proximal gradient descent,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Apr. 2018, pp. 1–5.

[4] G. Lan, S. Pokutta, Y. Zhou, and D. Zink, “Conditional accelerated lazy stochastic gradient descent,” in Proc. 34th Int. Conf. Mach. Learn. (ICML), 2017, pp. 1965–1974.

[5] M. Jaggi, “Revisiting Frank-Wolfe: Projection-free sparse convex optimization,” in Proc. 30th Int. Conf. Mach. Learn. (ICML), 2013, pp. 427–435.

[6] Z. Allen-Zhu, E. Hazan, W. Hu, and Y. Li, “Linear convergence of a Frank-Wolfe type algorithm over trace-norm balls,” in Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS), 2017, pp. 6192–6201.

[7] S. Lacoste-Julien and M. Jaggi, “On the global linear convergence of Frank-Wolfe optimization variants,” in Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS), 2015, pp. 496–504.

[8] K. Levy and A. Krause, “Projection free online learning over smooth sets,” in Proc. Int. Conf. Artif. Intell. Statist. (AISTATS), 2019, pp. 1458–1466.

[9] M. Frank and P. Wolfe, “An algorithm for quadratic optimization,” Naval Res. Logistics Quart., vol. 3, nos. 1–2, pp. 95–110, 1956.

[10] E. Hazan, “Sparse approximate solutions to semidefinite programs,” in LATIN 2008: Theoretical Informatics, E. S. Laber, C. Bornstein, L. T. Nogueira, and L. Faria, Eds. Berlin, Germany: Springer, 2008, pp. 306–316.

[11] G. Braun, S. Pokutta, D. Tu, and S. Wright, “Blended conditional gradients,” in Proc. Int. Conf. Mach. Learn. (ICML), vol. 97, Jun. 2019, pp. 735–743.

[12] E. Hazan and S. Kale, “Projection-free online learning,” in Proc. Int. Conf. Mach. Learn. (ICML), 2012, pp. 1843–1850.

[13] D. Garber and E. Hazan, “A linearly convergent conditional gradient algorithm with applications to online and stochastic optimization,” SIAM J. Optim., vol. 26, no. 3, pp. 1–35, Jan. 2013.

[14] J. Yuan and A. Lamperski, “Online convex optimization for cumulative constraints,” in Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS), 2018, pp. 6140–6149.

[15] M. Mahdavi, R. Jin, and T. Yang, “Trading regret for efficiency: Online convex optimization with long term constraints,” J. Mach. Learn. Res., vol. 13, no. 1, pp. 2503–2528, 2012.

[16] N. Liakopoulos, A. Destounis, G. Paschos, T. Spyropoulos, and P. Mertikopoulos, “Cautious regret minimization: Online optimization with long-term budget constraints,” in Proc. Int. Conf. Mach. Learn. (ICML), 2019, pp. 3944–3952.

[17] E. Hazan and E. Minasyan, “Faster projection-free online learning,” 2020, arXiv:2001.11568. [Online]. Available: https://arxiv.org/abs/2001.11568

[18] R. Johnson and T. Zhang, “Accelerating stochastic gradient descent using predictive variance reduction,” in Proc. Adv. Neural Inf. Process. Syst., vol. 1, 2013, pp. 315–323.

[19] L. Xiao and T. Zhang, “A proximal stochastic gradient method with progressive variance reduction,” SIAM J. Optim., vol. 24, no. 4, pp. 2057–2075, Jan. 2014.

[20] A. Nitanda, “Stochastic proximal gradient descent with acceleration techniques,” in Proc. Adv. Neural Inf. Process. Syst., Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., 2014, pp. 1574–1582.

[21] M. Mahdavi, T. Yang, R. Jin, S. Zhu, and J. Yi, “Stochastic gradient descent with only one projection,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), 2012, pp. 494–502.

[22] J. Chen, T. Yang, Q. Lin, L. Zhang, and Y. Chang, “Optimal stochastic strongly convex optimization with a logarithmic number of projections,” in Proc. 32nd Conf. Uncertainty Artif. Intell. (UAI), Arlington, VA, USA, 2016, pp. 122–131.

[23] E. Hazan and H. Luo, “Variance-reduced and projection-free stochastic optimization,” in Proc. Int. Conf. Mach. Learn. (ICML), 2016, pp. 1263–1271.

[24] G. Lan and Y. Zhou, “Conditional gradient sliding for convex optimization,” SIAM J. Optim., vol. 26, no. 2, pp. 1379–1409, Jan. 2016.

[25] J. Xie, Z. Shen, C. Zhang, B. Wang, and H. Qian, “Efficient projection-free online methods with stochastic recursive gradient,” in Proc. AAAI, 2019, pp. 6446–6453.

[26] P. Joulin, A. Gygörgyi, and C. Szepesvári, “A modular analysis of adaptive (non-)convex optimization: Optimism, composite objectives, and variational bounds,” in Proc. 28th Int. Conf. Algorithmic Learn. Theory (ALT), vol. 76, 2017, pp. 681–720.

[27] R. Dixit, A. S. Bedi, R. Tripathi, and K. Rajawat, “Online learning with inexact proximal online gradient descent algorithms,” IEEE Trans. Signal Process., vol. 67, no. 5, pp. 1338–1352, Mar. 2019.

YAN LI received the bachelor’s degree from the National University of Defense Technology, Changsha, China, in 2014, where she is currently pursuing the Ph.D. degree with the College of Systems Engineering. Her research interests include networking, pattern recognition, and machine learning.

XIAOFENG CAO (Graduate Student Member, IEEE) received the bachelor’s and master’s degrees from the National University of Defense Technology, Changsha, China, in 2014 and 2016, respectively, where he is currently pursuing the Ph.D. degree with the College of Systems Engineering. He is also a Visiting Student with the Broadband Communications Research (BBCR) Group, University of Waterloo, Waterloo, ON, Canada. His research interests include cloud/edge computing and data-driven application design. He is a member of the IEEE Computer Society, the Communication Society, and the Vehicular Technology Society.

HONGHUI CHEN received the M.S. degree in operational research and the Ph.D. degree in management science and engineering from the National University of Defense Technology, Changsha, China, in 1994 and 2007, respectively. He is currently a Professor with the College of System Engineering, National University of Defense Technology. His research interests include information systems, cloud computing, and information retrieval.