$O(\log T)$ Projections for Stochastic Optimization of Smooth and Strongly Convex Functions

Lijun Zhang* ZHANGLIJ@MSU.EDU
Tianbao Yang† TYANG@GE.COM
Rong Jin* RONGJIN@CSE.MSU.EDU
Xiaofei He‡ XIAOFEIHE@CAD.ZJU.EDU.CN

*Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA
†GE Global Research, San Ramon, CA 94583, USA
‡State Key Laboratory of CAD&CG, College of Computer Science, Zhejiang University, Hangzhou 310027, China

Abstract

Traditional algorithms for stochastic optimization require projecting the solution at each iteration into a given domain to ensure its feasibility. When facing complex domains, such as the positive semidefinite cone, the projection operation can be expensive, leading to a high computational cost per iteration. In this paper, we present a novel algorithm that aims to reduce the number of projections for stochastic optimization. The proposed algorithm combines the strength of several recent developments in stochastic optimization, including mini-batches, extra-gradient, and epoch gradient descent, in order to effectively explore the smoothness and strong convexity. We show, both in expectation and with a high probability, that when the objective function is both *smooth* and *strongly convex*, the proposed algorithm achieves the optimal $O(1/T)$ rate of convergence with only $O(\log T)$ projections. Our empirical study verifies the theoretical result.

1. Introduction

The goal of stochastic optimization is to solve the optimization problem

$$\min_{\mathbf{w}\in\mathcal{D}} F(\mathbf{w}),$$

using only the stochastic gradients of $F(\mathbf{w})$. In particular, we assume there exists a gradient oracle, which for any point $\mathbf{w} \in \mathcal{D}$, returns a random vector $\hat{\mathbf{g}}(\mathbf{w})$ that gives an unbiased estimate of the subgradient of $F(\cdot)$ at $\mathbf{w}$. A special case of stochastic optimization is the risk minimization problem, whose objective function is given by

$$F(\mathbf{w}) = \mathrm{E}_{(\mathbf{x},y)}\left[\ell(\mathbf{w};(\mathbf{x},y))\right],$$

where $(\mathbf{x}, y)$ is an instance-label pair, ℓ is a convex loss function that measures the prediction error, and the expectation is taken oven the unknown joint distribution of $(\mathbf{x}, y)$ (Zhang, 2004; Shalev-Shwartz et al., 2009; Hu et al., 2009). The performance of stochastic optimization algorithms is typically characterized by the *excess risk*

$$F(\mathbf{w}_T) - \min_{\mathbf{w}\in\mathcal{D}} F(\mathbf{w}),$$

where T is the number of iterations and $\mathbf{w}_T$ is the solution obtained after making T calls to the gradient oracle.

For general Lipschitz continuous convex functions, stochastic gradient descent exhibits the unimprovable $O(1/\sqrt{T})$ rate of convergence (Nemirovski & Yudin, 1983; Nemirovski et al., 2009). For strongly convex functions, the algorithms proposed in very recent works (Juditsky & Nesterov, 2010; Hazan & Kale, 2011; Rakhlin et al., 2012; Chen et al., 2012) achieve the optimal $O(1/T)$ rate (Agarwal et al., 2012). Although these convergence rates are significantly worse than the results in deterministic optimization, stochastic optimization is appealing due to its low per-iteration complexity. However, this is not the case when the domain $\mathcal{D}$ is complex. This is because most stochastic optimization algorithms require projecting the solution at each iteration into domain $\mathcal{D}$ to ensure its feasibility, an expensive operation when the

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

domain is complex. In this paper, we show that if the objective function is smooth and strongly convex, it is possible to reduce the number of projections dramatically without affecting the convergence rate.

Our work is motivated by the difference in convergence rates between stochastic and deterministic optimization. When the objective function is smooth and convex, under the first-order oracle assumption, Nesterov's accelerated gradient method enjoys the optimal $O(1/T^2)$ rate (Nesterov, 2004; 2005). Thus, *for deterministic optimization of smooth and convex functions, we can achieve an $O(1/\sqrt{T})$ rate by only performing $O(T^{1/4})$ updating.* When the objective function is smooth and strongly convex, the optimal rate for first-order algorithms is $O(1/\alpha^k)$, for some constant $\alpha > 1$ (Nesterov, 2004; 2007). In other words, *for deterministic optimization of smooth and strongly convex functions, we can achieve an $O(1/T)$ rate by only performing $O(\log T)$ updating.* The above observations inspire us to consider the following questions.

1. For Stochastic Optimization of Smooth and Convex functions (SOSC), is it possible to maintain the optimal $O(1/\sqrt{T})$ rate by performing $O(T^{1/4})$ projections?
2. For Stochastic Optimization of Smooth and Strongly Convex functions (SOS^{2}C), is it possible to maintain the optimal $O(1/T)$ rate by performing $O(\log T)$ projections?

For the 1st question, we have found a positive answer from literature. By combining mini-batches (Roux et al., 2008) with the accelerated stochastic approximation (Lan, 2012), we can achieve the optimal $O(1/\sqrt{T})$ rate by performing $O(T^{1/4})$ projections (Cotter et al., 2011). However, a naive application of mini-batches does not lead to the desired $O(\log T)$ complexity for SOS^{2}C. The main contribution of this paper is a novel stochastic optimization algorithm that answers the 2nd question positively. Our theoretical analysis reveals, both in expectation and with a high probability, that the proposed algorithm achieves the optimal $O(1/T)$ rate by only performing $O(\log T)$ projections.

2. Related Work

In this section, we provide a brief review of the existing approaches for avoiding projections.

2.1. Mini-batches based algorithms

Instead of updating the solution after each call to the gradient oracle, mini-batches based algorithms use the average gradient over multiple calls to update the solution (Roux et al., 2008; Shalev-Shwartz et al., 2011; Dekel et al., 2011). For a fixed batch size B, the number of updates (and projections) is reduced from $O(T)$ to $O(T/B)$, and the variance of the stochastic gradient is reduced from σ to $\sigma/\sqrt{B}$. By appropriately balancing between the loss cased by a smaller number of updates and the reduction in the variance of stochastic gradients, it is able to maintain the optimal rate of convergence.

The idea of mini-batches can be incorporated into any stochastic optimization algorithm that uses gradient-based updating rules. When the objective function is smooth and convex, combining mini-batches with the accelerated stochastic approximation (Lan, 2012) leads to

$$O\left(\frac{B^2}{T^2} + \frac{1}{\sqrt{T}}\right)$$

rate of convergence (Cotter et al., 2011). By setting $B = T^{3/4}$, we achieve the optimal $O(1/\sqrt{T})$ rate with only $O(T^{1/4})$ projections. When the target function is smooth and strongly convex, we can apply mini-batches to the optimal algorithms for strongly convex functions (Hu et al., 2009; Ghadimi & Lan, 2012), leading to

$$O\left(\frac{B^2}{T^2} + \frac{1}{T}\right)$$

rate of convergence (Dekel et al., 2012). In order to maintain the optimal $O(1/T)$ rate, the value of B cannot be larger than $\sqrt{T}$, implying at least $O(\sqrt{T})$ projections are required. In contrast, the algorithm proposed in this paper achieves an $O(1/T)$ rate with only $O(\log T)$ projections.

2.2. Projection free algorithms

Due to the low iteration cost, Frank-Wolfe algorithm (Frank & Wolfe, 1956) or conditional gradient method (Levitin & Polyak, 1966) has seen a recent surge of interest in machine learning (Hazan, 2008; Clarkson, 2010; Lacoste-Julien et al., 2013). At each iteration of the Frank-Wolfe algorithm, instead of performing a projection that requires solving a constrained quadratic programming problem, it solves a constrained linear programming problem. For many domains of interest, including the positive semidefinite cone and the trace norm ball, the constrained linear problem can be solved more efficiently than a projection problem (Jaggi, 2013), making this kind of methods attractive for large-scale optimization.

In a recent work (Hazan & Kale, 2012), an online variant of the Frank-Wolfe algorithm is proposed. Although the online Frank-Wolfe algorithm exhibits an $O(1/\sqrt{T})$ convergence rate for smooth functions, it is

unable to achieve the optimal $O(1/T)$ rate for strongly convex functions. Besides, the memory complexity of this algorithm is $O(T)$, making it unsuitable for large-scale optimization problems. Another related work is the stochastic gradient descent with only one projection (Mahdavi et al., 2012). This algorithm is built upon the assumption that the solution domain can be characterized by an inequality constraint $g(\mathbf{w}) \leq 0$ and the gradient of $g(\cdot)$ can be evaluated efficiently. Unfortunately, this assumption does not hold for some commonly used domain (e.g., the trace norm ball). Compared to the projection free algorithms, our proposed method is more general because it make no assumption about the solution domain.

3. Stochastic Optimization of Smooth and Strongly Convex Functions

3.1. Preliminaries

We first define smoothness and strongly convexity.

Definition 1. *A function $f : \mathcal{D} \to \mathbb{R}$ is L-smooth w.r.t. a norm $\|\cdot\|$ if f is everywhere differentiable and*

$$\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\|_* \leq L\|\mathbf{w} - \mathbf{w}'\|, \ \forall \mathbf{w}, \mathbf{w}' \in \mathcal{D}.$$

where $\|\cdot\|_*$ is the dual norm.

Definition 2. *A smooth function $f : \mathcal{D} \to \mathbb{R}$ is λ-strongly convex w.r.t. a norm $\|\cdot\|$, if f is everywhere differentiable and*

$$\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\|_* \geq \lambda\|\mathbf{w} - \mathbf{w}'\|, \ \forall \mathbf{w}, \mathbf{w}' \in \mathcal{D}.$$

To simplify our analysis, we assume that both $\|\cdot\|$ and $\|\cdot\|_*$ are the vector ℓ_2 norm in the following discussion.

Following (Hazan & Kale, 2011), we make the following assumptions about the gradient oracle.

- There is a gradient oracle, which, for a given input point $\mathbf{w}$ returns a stochastic gradient $\hat{\mathbf{g}}(\mathbf{w})$ whose expectation is the gradient of $F(\mathbf{w})$ at $\mathbf{w}$, i.e.,

  $$\mathrm{E}[\hat{\mathbf{g}}(\mathbf{w})] = \nabla F(\mathbf{w}).$$

  We further assume the stochastic gradients obtained by calling the oracle are *independent*.
- The gradient oracle is G-bounded, i.e.,

  $$\|\hat{\mathbf{g}}(\mathbf{w})\| \leq G, \ \forall \mathbf{w} \in \mathcal{D}.$$

  We note that this assumption may be relaxed by assuming the orlicz norm of $\hat{\mathbf{g}}(\mathbf{w})$ to be bounded (Lan, 2012), i.e., $\mathrm{E}[\exp(\|\hat{\mathbf{g}}(\mathbf{w})\|^2/G^2)] \leq \exp(1)$. Although our theoretical result holds even under the assumption of bounded orlicz norm, we choose the G-bounded gradient for simplicity.

Define $\mathbf{w}_*$ as the optimal solution that minimizes $F(\mathbf{w})$, i.e., $\mathbf{w}_* = \operatorname{argmin}_{\mathbf{w}\in\mathcal{D}} F(\mathbf{w})$. Using the strongly convexity of $F(\mathbf{w})$, we have (Hazan & Kale, 2011)

$$\frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}_*\|^2 \leq F(\mathbf{w}) - F(\mathbf{w}_*) \leq \frac{2G^2}{\lambda}, \forall \ \mathbf{w} \in \mathcal{D}. \quad (1)$$

3.2. The Algorithm

Algorithm 1 shows the proposed method for Stochastic Optimization of Smooth and Strongly Convex functions (SOS^{2}C), that achieves the optimal $O(1/T)$ rate of convergence by performing $O(\log T)$ projections. The inputs of the algorithm are: (1) η, the step size, (2) M, the fixed number of updates per epoch/stage, (3) B^1, the initial batch size, and (4) T, the total number of calls to the gradient oracle. With a slight abuse of notation, we use $\hat{\mathbf{g}}(\mathbf{w}, i)$ to denote the stochastic gradient at $\mathbf{w}$ obtained after making the i-th call to the oracle. We denote the projection of $\mathbf{w}$ onto the domain $\mathcal{D}$ by $\Pi_{\mathcal{D}}(\mathbf{w})$.

Similar to the epoch gradient descent algorithm (Hazan & Kale, 2011), the proposed algorithm consists of two layers of loops. It uses the outer (**while**) loop to divide the learning process into a sequence of epochs (Step 5 to Step 12). Similar to (Hazan & Kale, 2011), the number of calls to the gradient oracle made by Algorithm 1 increases exponentially over the epoches, a key that allows us to achieve the optimal $O(1/T)$ convergence rate for strongly convex functions. We note that other techniques, such as the α-suffix averaging (Rakhlin et al., 2012), can also be used as an alternative.

In the inner (**for**) loop of each epoch, we combine the idea of mini-batches (Dekel et al., 2011) with extra-gradient descent (Nemirovski, 2005; Juditsky et al., 2011). We choose extra-gradient descent because it allows us to replace in the excess risk bound $\mathrm{E}[\|\hat{\mathbf{g}}(\mathbf{w})\|^2]$ with $\mathrm{E}[\|\hat{\mathbf{g}}(\mathbf{w}) - \mathrm{E}[\hat{\mathbf{g}}(\mathbf{w})]\|^2]$, the variance of the stochastic gradient $\hat{\mathbf{g}}(\mathbf{w})$, thus opening the door to fully exploring the capacity of mini-batches in variance reduction.

To be more specific, in the k-th epoch, we maintain two sequences of solutions $\{\mathbf{w}_t^k\}_{t=1}^M$ and $\{\mathbf{z}_t^k\}_{t=1}^M$, where $\mathbf{z}_t^k$ is an auxiliary solution that allows us to effectively explore the smoothness of the loss function. At each iteration t of the k-th epoch, we calculate the average gradients $\bar{\mathbf{g}}_t^k$ and $\bar{\mathbf{f}}_t^k$ by calling the gradient oracle B^k times (Steps 6 and 8), and update the solutions $\mathbf{w}_t^k$ and $\mathbf{z}_t^k$ using the average gradients (Steps 7 and 9). The batch size B^k is fixed inside each epoch but doubles from epoch to epoch (Step 11). This is in contrast to most mini-batches based algorithms that have a fixed batch size. This difference is critical for

Algorithm 1 $\log T$ Projections for SOS^{2}C

1: **Input:** parameters η, M, B^1 and T
2: Initialize $\mathbf{w}_1^1 \in \mathcal{D}$ arbitrarily
3: Set $k = 1$
4: **while** $2M \sum_{i=1}^{k} B^i \leq T$ **do**
5: **for** $t = 1$ to M **do**
6: Compute the average gradient at $\mathbf{w}_t^k$ over B^k calls to the gradient oracle

$$\bar{\mathbf{g}}_t^k = \frac{1}{B^k} \sum_{i=1}^{B^k} \hat{\mathbf{g}}(\mathbf{w}_t^k, i)$$

7: Update

$$\mathbf{z}_t^k = \Pi_{\mathcal{D}} \left(\mathbf{w}_t^k - \eta \bar{\mathbf{g}}_t^k \right)$$

8: Compute the average gradient at $\mathbf{z}_t^k$ over B^k calls to the gradient oracle

$$\bar{\mathbf{f}}_t^k = \frac{1}{B^k} \sum_{i=1}^{B^k} \hat{\mathbf{g}}(\mathbf{z}_t^k, i)$$

9: Update

$$\mathbf{w}_{t+1}^k = \Pi_{\mathcal{D}} \left(\mathbf{w}_t^k - \eta \bar{\mathbf{f}}_t^k \right)$$

10: **end for**
11: $\mathbf{w}_1^{k+1} = \frac{1}{M} \sum_{t=1}^{M} \mathbf{z}_t^k$, and $B^{k+1} = 2B^k$
12: $k = k + 1$
13: **end while**
14: **Return:** $\mathbf{w}_1^k$

achieving $O(1/T)$ convergence rate with only $O(\log T)$ updates.

Finally, it is worth mentioning that the Euclidean projection in Steps 7 and 9 can be replaced by the more general "prox-mapping" defined by a Bregman distance function to better capture the geometry of $\mathcal{D}$ (Nemirovski, 2005).

3.3. The main results

The following theorem bounds the expected excess risk of the solution return by Algorithm 1 and the number of projections.

Theorem 1. *Set the parameters in Algorithm 1 as*

$$\eta = \frac{1}{\sqrt{6}L}, \ M = \frac{4}{\eta\lambda} \ \text{and} \ B^1 = 12\eta\lambda.$$

The final point $\mathbf{w}_1^k$ returned by Algorithm 1 makes at most T calls to the gradient oracle, and has its excess risk bounded by

$$\mathrm{E}[F(\mathbf{w}_1^k) - F(\mathbf{w}_*)] \leq \frac{384G^2}{\lambda T} = O\left(\frac{1}{T}\right),$$

and the total number of projections bounded by

$$\frac{8\sqrt{6}L}{\lambda} \left\lfloor \log_2 \left(\frac{T}{96} + 1 \right) \right\rfloor = O\left(\log T\right).$$

Theorem 1 shows that in expectation, Algorithm 1 achieve an $O(1/T)$ convergence with $O(\log T)$ updates. The following theorem gives a high probability bound of the excess risk for Algorithm 1.

Theorem 2. *Set the parameters in Algorithm 1 as*

$$\eta = \frac{1}{\sqrt{6}L}, \ M = \frac{4}{\eta\lambda} \ \text{and} \ B^1 = \alpha\eta\lambda,$$

where α is defined below. For any $0 < \delta < 1$, let

$$\tilde{\delta} = \frac{\delta}{k^\dagger},$$

$$k^\dagger = \left\lfloor \log_2 \left(\frac{T}{8\alpha} + 1 \right) \right\rfloor = O(\log T), \tag{2}$$

$$\begin{aligned} \alpha = & \max \left\{ 400 \log^2 \frac{8M}{\tilde{\delta}}, \right. \\ & \left. 1 + 64 \log^2 \frac{8M}{\tilde{\delta}} \left(\log \frac{4N}{\tilde{\delta}} + \frac{4}{9} \log^2 \frac{4N}{\tilde{\delta}} \right) \right\} \end{aligned} \tag{3}$$

$$= O\left[\left(\log\log T + \log \frac{1}{\delta} \right)^4 \right],$$

$$N = \left\lceil \log_2 \frac{4MT}{\eta\lambda} \right\rceil = O(\log T). \tag{4}$$

The final point $\mathbf{w}_1^k$ returned by Algorithm 1 makes at most T calls to the gradient oracles, performs

$$\frac{8\sqrt{6}L}{\lambda} \left\lfloor \log_2 \left(\frac{T}{8\alpha} + 1 \right) \right\rfloor = O\left(\log T\right)$$

projections, and with a probability at least $1 - \delta$, has its excess risk bounded by

$$F(\mathbf{w}_1^k) - F(\mathbf{w}_*) \leq \frac{32\alpha G^2}{\lambda T} = O\left(\frac{(\log\log T + \log 1/\delta)^4}{T} \right).$$

Remark: It is worth noting that we achieve the high probability bound without making any modifications to Algorithm 1. This is in contrast to the epoch gradient descent algorithm (Hazan & Kale, 2011) that needs to shrink the domain size in order to obtain the desirable high probability bound, which could potentially lead to an additional computational cost in performing projection. We remove the

shrinking step by effectively exploring the peeling technique (Bartlett et al., 2005).

The number of projections required by Algorithm 1, according to Theorem 2, exhibits a linear dependence on the conditional number L/λ, which can be very large when dealing with ill-conditioned optimization problems. In the deterministic setting, the convergence rate only depends on the square root of the conditional number (Nesterov, 2004; 2007). Thus, we conjecture that it may be possible to improve the dependence on the conditional number to its square root in the stochastic setting, a problem that will be examined in the future.

4. Analysis

We here present the proofs of main theorems. The omitted proofs are provided in the supplementary material.

4.1. Proof of Theorem 1

Since we make use of the the multi-stage learning strategy, the proof provided below is similar to the proof in (Hazan & Kale, 2011). We begin by analyzing the property of the inner loop in Algorithm 1, which is a combination of mini-batches and the extra-gradient descent. To this end, we have the following lemma.

Lemma 1. *Let $\eta = 1/[\sqrt{6}L]$ in Algorithm 1. Then, we have*

$$\begin{aligned} F\left(\frac{1}{M}\sum_{t=1}^{M}\mathbf{z}_t^k\right) - F(\mathbf{w}_*) \leq & \frac{\|\mathbf{w}_1^k - \mathbf{w}_*\|^2}{2M\eta} \\ & + \frac{3\eta}{M}\left(\sum_{t=1}^{M}\|\bar{\mathbf{g}}_t^k - \mathbf{g}_t^k\|^2 + \sum_{t=1}^{M}\|\bar{\mathbf{f}}_t^k - \mathbf{f}_t^k\|^2\right) \quad (5) \\ & + \frac{1}{M}\sum_{t=1}^{M}\langle \mathbf{f}_t^k - \bar{\mathbf{f}}_t^k, \mathbf{z}_t^k - \mathbf{w}_*\rangle \quad (6) \\ & - \frac{\lambda}{2M}\sum_{t=1}^{M}\|\mathbf{z}_t^k - \mathbf{w}_*\|^2, \end{aligned}$$

where

$$\mathbf{g}_t^k = \nabla F(\mathbf{w}_t^k) \text{ and } \mathbf{f}_t^k = \nabla F(\mathbf{z}_t^k).$$

Taking the conditional expectation of the inequality, we have

$$\mathrm{E}_{k-1}\left[F\left(\frac{1}{M}\sum_{t=1}^{M}\mathbf{z}_t^k\right)\right] - F(\mathbf{w}_*) \leq \frac{\|\mathbf{w}_1^k - \mathbf{w}_*\|^2}{2M\eta} + \frac{6\eta G^2}{B^k}.$$

where $\mathrm{E}_{k-1}[\cdot]$ denotes the expectation conditioned on all the randomness up to epoch $k-1$.

The quantity in (5) illustrates the advantage of the extra-gradient descent, i.e., it is able to produce variance-dependent upper bound when applied to stochastic optimization. Because of mini-batches, the expectations of $\|\bar{\mathbf{g}}_t^k - \mathbf{g}_t^k\|^2$ and $\|\bar{\mathbf{f}}_t^k - \mathbf{f}_t^k\|^2$ are smaller than G^2/B^k, which leads to the tight upper bound in the second inequality.

Based on Lemma 1, we get the following lemma that bounds the expected excess risk in each epoch.

Lemma 2. *Define*

$$\Delta_k = F(\mathbf{w}_1^k) - F(\mathbf{w}_*).$$

Set the parameters $\eta = 1/[\sqrt{6}L]$, $M = 4/[\eta\lambda]$ and $B^1 = 12\eta\lambda$ in Algorithm 1. For any k, we have

$$\mathrm{E}[\Delta_k] \leq V_k = \frac{G^2}{\lambda 2^{k-2}}.$$

Proof. It is straightforward to check that

$$B^k = 12\eta\lambda 2^{k-1} = \frac{24\eta G^2}{V_k}. \quad (7)$$

We prove this lemma by induction on k. When $k = 1$, we know that

$$\Delta_1 = F(\mathbf{w}_1^1) - F(\mathbf{w}_*) \overset{(1)}{\leq} \frac{2G^2}{\lambda} = \frac{G^2}{\lambda 2^{1-2}} = V_1.$$

Assume that $\mathrm{E}[\Delta_k] \leq V_k$ for some $k \geq 1$, and we prove the inequality for $k+1$. From Lemma 1, we have

$$\mathrm{E}_{k-1}\left[F\left(\mathbf{w}_1^{k+1}\right)\right] - F(\mathbf{w}_*) \leq \frac{\|\mathbf{w}_1^k - \mathbf{w}_*\|^2}{2M\eta} + \frac{6\eta G^2}{B^k}.$$

Thus

$$\begin{aligned} & \mathrm{E}\left[F\left(\mathbf{w}_1^{k+1}\right)\right] - F(\mathbf{w}_*) \\ \leq & \frac{\mathrm{E}[\|\mathbf{w}_1^k - \mathbf{w}_*\|^2]}{2M\eta} + \frac{6\eta G^2}{B^k} \\ \overset{(1)}{\leq} & \frac{\mathrm{E}[2(F(\mathbf{w}_1^k) - F(\mathbf{w}_*))/\lambda]}{2M\eta} + \frac{6\eta G^2}{B^k} \\ \overset{(7)}{=} & \frac{\mathrm{E}[\Delta_k]}{M\eta\lambda} + \frac{V_k}{4} \leq \frac{V_k}{4} + \frac{V_k}{4} = V_{k+1}. \end{aligned}$$

□

We are now at the position to prove Theorem 1.

Proof of Theorem 1. From the stopping criterion of the outer loop in Algorithm 1, we know that the number of the epochs is given by the largest value of k such that

$$2M\sum_{i=1}^{k} B^i \leq T.$$

Since

$$2M\sum_{i=1}^{k} B^i = 24M\eta\lambda\sum_{i=1}^{k} 2^{i-1} = 96(2^k - 1),$$

the final epoch is given by

$$k^\dagger = \left\lfloor \log_2\left(\frac{T}{96} + 1\right)\right\rfloor,$$

and the final output is $\mathbf{w}_1^{k^\dagger+1}$. From Lemma 2, we have

$$\mathrm{E}[F(\mathbf{w}_1^{k^\dagger+1})] - F(\mathbf{w}_*) \leq V_{k^\dagger+1} = \frac{G^2}{2^{k^\dagger-1}\lambda} \leq \frac{384G^2}{\lambda T},$$

where we use the fact

$$2^{k^\dagger} \geq \frac{1}{2}\left(\frac{T}{96} + 1\right) \geq \frac{T}{192}.$$

The total number of projections is

$$2Mk^\dagger = \frac{8\sqrt{6}L}{\lambda}\left\lfloor \log_2\left(\frac{T}{96} + 1\right)\right\rfloor.$$

□

4.2. Proof of Theorem 2

Compared to the proof of Theorem 1, the main difference here is that we need a high probability version of Lemma 1. Specifically, we need to provide high probability bounds for the quantities in (5) and (6).

To bound the variances given in (5), we need the following norm concentration inequality in Hilbert Space (Smale & Zhou, 2009).

Lemma 3. *Let $\mathcal{H}$ be a Hilbert Space and let ξ be a random variable on $(\mathcal{Z}, \rho)$ with values in $\mathcal{H}$. Assume $\|\xi\| \leq B < \infty$ almost surely. Let $\{\xi_i\}_{i=1}^m$ be independent random draws of ρ. For any $0 < \delta < 1$, with a probability at least $1 - \delta$,*

$$\left\|\frac{1}{m}\sum_{i=1}^{m}(\xi_i - \mathrm{E}[\xi_i])\right\| \leq \frac{4B}{\sqrt{m}}\log\frac{2}{\delta}.$$

Based on Lemma 3, it is straightforward to prove the following lemma.

Lemma 4. *With a probability at least $1-\tilde{\delta}/2$, we have*

$$\|\bar{\mathbf{g}}_t^k - \mathbf{g}_t^k\| \leq \frac{4G}{\sqrt{B^k}}\log\frac{4M}{\tilde{\delta}},\ \forall\ t = 1, \ldots, M. \tag{8}$$

Similarly, with a probability at least $1 - \tilde{\delta}/4$, we have

$$\|\bar{\mathbf{f}}_t^k - \mathbf{f}_t^k\| \leq \frac{4G}{\sqrt{B^k}}\log\frac{8M}{\tilde{\delta}},\ \forall\ t = 1, \ldots, M. \tag{9}$$

We define the Martingale difference sequence:

$$Z_t^k = \langle \mathbf{f}_t^k - \bar{\mathbf{f}}_t^k, \mathbf{z}_t^k - \mathbf{w}_*\rangle.$$

In order to bound the summation of Z_t^k in (6), we make use of the Bernstein's inequality for martingales (Cesa-Bianchi & Lugosi, 2006) and the peeling technique described in (Bartlett et al., 2005), leading to the following Lemma.

Lemma 5. *We use E_1 to denote the event that all the inequalities in (9) hold. On event E_1, with a probability at least $1 - \tilde{\delta}/4$, we have*

$$\begin{aligned}\sum_{t=1}^{M} Z_t^k \leq{}& \frac{4G^2\eta M}{B^k}\log^2\frac{8M}{\tilde{\delta}} \\ &+ \frac{G^2}{\lambda B^k}\left[1 + 64\log^2\frac{8M}{\tilde{\delta}}\left(\log\frac{4n}{\tilde{\delta}} + \frac{4}{9}\log^2\frac{4n}{\tilde{\delta}}\right)\right] \\ &+ \frac{\lambda}{2}\sum_{t=1}^{M}\|\mathbf{z}_t^k - \mathbf{w}_*\|^2,\end{aligned}$$

where

$$n = \left\lceil \log_2\frac{4MB^k}{\eta\lambda}\right\rceil. \tag{10}$$

Substituting the results in Lemmas 4 and 5 into Lemma 1, we obtain the lemma below.

Lemma 6. *For any $0 < \tilde{\delta} < 1$, with a probability at least $1 - \tilde{\delta}$, we have*

$$\begin{aligned}F\left(\frac{1}{M}\sum_{t=1}^{M}\mathbf{z}_t^k\right) - F(\mathbf{w}_*) \leq{}& \frac{\|\mathbf{w}_1^k - \mathbf{w}_*\|^2}{2M\eta} \\ &+ \frac{100G^2\eta}{B^k}\log^2\frac{8M}{\tilde{\delta}} \\ &+ \frac{G^2}{\lambda B^k M}\left[1 + 64\log^2\frac{8M}{\tilde{\delta}}\left(\log\frac{4n}{\tilde{\delta}} + \frac{4}{9}\log^2\frac{4n}{\tilde{\delta}}\right)\right],\end{aligned}$$

where n is given in (10).

Based on Lemma 6, we provide a high probability version of Lemma 2, that bounds the excess risk in each epoch with a high probability.

Lemma 7. *Set the parameters $\eta = 1/[\sqrt{6}L]$, $M = 4/[\eta\lambda]$ and $B^1 = \alpha\eta\lambda$ in Algorithm 1, where α is defined in (3). For any k, with a probability at least $(1 - \tilde{\delta})^{k-1}$, we have*

$$\Delta_k = F(\mathbf{w}_1^k) - F(\mathbf{w}_*) \leq V_k = \frac{G^2}{\lambda 2^{k-2}}.$$

Now, we provide the proof of Theorem 2.

Proof of Theorem 2. The number of epochs made is given by the largest value of k satisfying $2M\sum_{i=1}^{k} B^i \leq T$. Since

$$2M\sum_{i=1}^{k} B^i = 2M\alpha\lambda\eta\sum_{i=1}^{k} 2^{i-1} = 8\alpha(2^k - 1),$$

$k^\dagger$ defined in (2) is the value of the final epoch, and the final output is $\mathbf{w}_1^{k^\dagger+1}$. From Lemma 7, we have with a probability at least $(1-\tilde{\delta})^{k^\dagger}$

$$\begin{aligned} &F(\mathbf{w}_1^{k^\dagger+1}) - F(\mathbf{w}_*) = \Delta_{k^\dagger+1} \\ \leq &V_{k^\dagger+1} = \frac{G^2}{2^{k^\dagger-1}\lambda} = \frac{2G^2}{2^{k^\dagger}\lambda} \leq \frac{32\alpha G^2}{\lambda T}, \end{aligned}$$

where we use the fact

$$2^{k^\dagger} \geq \frac{1}{2}\left(\frac{T}{8\alpha} + 1\right) \geq \frac{T}{16\alpha}.$$

We complete the proof by using the property that $(1-\frac{1}{x})^x$ is an increasing function when $x > 1$, which implies

$$\begin{aligned} (1-\tilde{\delta})^{k^\dagger} &= \left(1 - \frac{\delta}{k^\dagger}\right)^{k^\dagger} = \left(\left(1 - \frac{1}{k^\dagger/\delta}\right)^{k^\dagger/\delta}\right)^{\delta} \\ &\geq \left(\left(1 - \frac{1}{1/\delta}\right)^{1/\delta}\right)^{\delta} = 1 - \delta. \end{aligned}$$

□

5. Experiments

In this section, we present numerical experiments to support our theoretical analysis. We studied the following algorithms:

1. $\log T$: the proposed algorithm that is optimal for SOS^{2}C but only needs $\log(T)$ projections;
2. EP_GD: the epoch gradient descent developed in (Hazan & Kale, 2011), which is also optimal for SOS^{2}C but needs $O(T)$ projections;
3. SGD: the stochastic gradient descent with step size $\eta_t = 1/(\lambda t)$ (Shalev-Shwartz et al., 2011), which achieves $O(\log T/T)$ rate of convergence for general SOS^{2}C and needs $O(T)$ projections.

We first consider the a simple stochastic optimization problem adapted from (Rakhlin et al., 2012), which is both smooth and strongly convex. The objective function is $F(W) = \frac{1}{2}\|W\|_F^2$ and the domain is the 5×5 dimensional positive semidefinite (PSD) cone. The stochastic gradient oracle, given a point W, returns the stochastic gradient $W + Z$ where Z is uniformly distributed in $[-1,1]^{5\times 5}$. Because of the noise matrix Z, all the immediate solutions are not PDS and we need to project them back to the PSD cone. To ensure the eigendecomposition only involving real numbers, we further require Z to be symmetric. Notice that for this problem we know $W_* = \operatorname{argmin}_{W\succeq o} F(W) = 0^{5\times 5}$. Since the gradient of W_* is $0^{5\times 5}$, it can be shown that SGD also achieves the optimal $O(1/T)$ rate of convergence on this problem (Rakhlin et al., 2012).

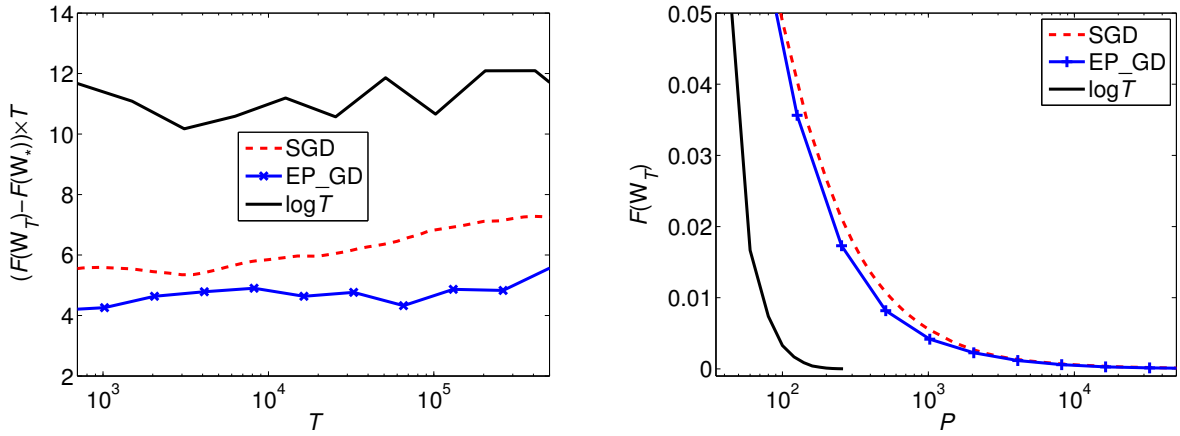
Let W_T be the solution returned after making T calls to the gradient oracle. To verify if the proposed algorithm achieves an $O(1/T)$ convergence, we measure $(F(W_T) - F(W_*)) \times T$ versus T, which is given in Fig. 1(a). We observe that when T is sufficiently large, quantity $(F(W_T) - F(W_*)) \times T$ essentially becomes a constant for all three algorithms, implying $O(1/T)$ convergence rates for all the algorithms. We also observe that the constant achieved by the proposed algorithm is slightly larger than the two competitors, which can be attributed to the term $(\log\log T)^4$ in our bound in Theorem 2. To demonstrate the advantage of our algorithm, we plot the value of the objective function versus the number of projections P in Fig. 1(b). We observe that using our algorithm, the objective function is reduced significantly faster than other algorithms w.r.t. the number of projections.

In the second experiment, we apply our algorithm to the regularized distance metric learning (Jin et al., 2009). The goal is to solve the following problem

$$\min_{W\succeq 0} \mathrm{E}_{(\mathbf{x}_i,y_i),(\mathbf{x}_j,y_j)}[\ell(y_{ij}(1 - \|\mathbf{x}_i - \mathbf{x}_j\|_M^2))] + \frac{\lambda}{2}\|W\|_F^2,$$

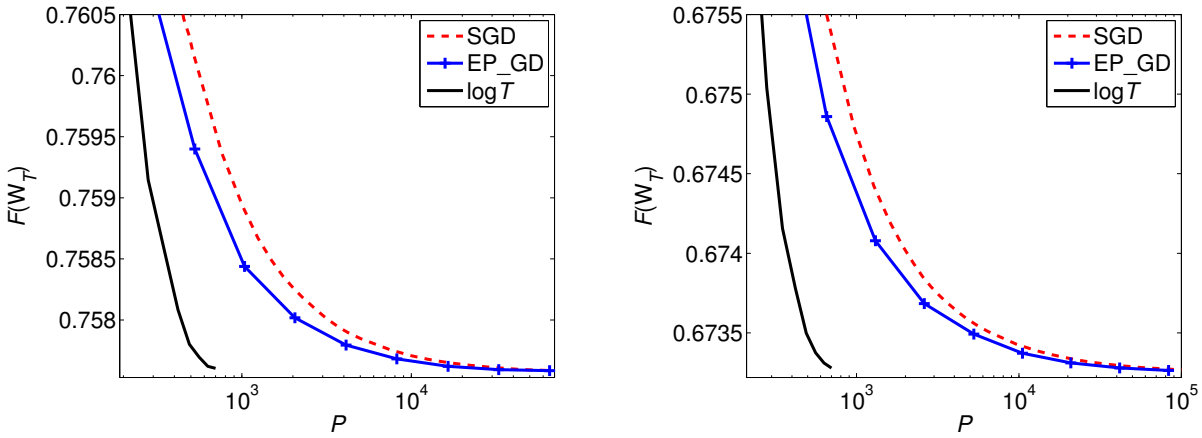
where $\mathbf{x}_i$ is the instance, and y_i is $\mathbf{x}_i$'s label, y_{ij} is derived from labels y_i and y_j (i.e., $y_{ij} = 1$ if $y_i = y_j$ and -1 otherwise), $\|\mathbf{x}\|_M^2 = \mathbf{x}^\top M\mathbf{x}$, and $\ell(z) = \log(1+\exp(-z))$ is the logit loss. We set $\lambda = 0.1$ and test our algorithm on the Mushrooms and Adult data sets (Chang & Lin, 2011).

During the optimization process, the call to the gradient oracle corresponds to generate a training pair $\{(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)\}$ randomly. When the oracle gives us two training examples belonging to the same class (i.e., a must-link constraint), the stochastic gradient is a PSD matrix, which could result in non-PSD intermediate solutions and makes the projection step necessary. To estimate the value of objective function, we evaluate the average empirical loss on 10^4 testing pairs, which are also generated randomly. Fig. 2 shows the value of the objective function versus the number of projections P. Again, this result validates that the proposed algorithm $\log T$ is able to reduce the number of projections dramatically without hurting the performance. More results can be found in the supplementary material.



(a) $(F(\mathbf{w}_T) - F(\mathbf{w}_*)) \times T$ versus T (b) $F(\mathbf{w}_T)$ versus the number of projections P

Figure 1. Results for stochastic optimization of $F(W) = \frac{1}{2}\|W\|_F^2$ over the PSD cone. The experiments are repeated 10 times and the averages are reported.



(a) Mushrooms (b) Adult

Figure 2. Results for the regularized distance metric learning on the Mushrooms and Adult data sets. $F(W_T)$ is measured on 10^4 testing pairs and the horizontal axis P measures the number of projections performed by each algorithm. The experiments are repeated 10 times and the averages are reported.

6. Conclusion

In this paper, we study the problem of reducing the number of projections in stochastic optimization by exploring the property of smoothness. When the target function is smooth and strongly convex, we propose a novel algorithm that achieves the optimal $O(1/T)$ rate of convergence by only performing $O(\log T)$ projections.

An open question is how to extend our results to stochastic composite optimization (Lan, 2012), where the objective function is a combination of non-smooth and smooth stochastic components. We plan to explore the composite gradient mapping technique (Nesterov, 2007), to see if we can achieve an $O(1/T)$ rate of convergence with only $O(\log T)$ updates.

Acknowledgments

This work is partially supported by Office of Navy Research (ONR Award N00014-09-1-0663 and N000141210431), National Basic Research Program of China (973 Program) under Grant 2012CB316400, and National Natural Science Foundation of China (Grant No: 61125203).

References

Agarwal, A., Bartlett, P.L., P.Ravikumar, and Wainwright, M.J. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Trans. Inf. Theory*, 58(5):3235–3249, 2012.

Bartlett, P.L., Bousquet, O., and Mendelson, S. Local rademacher complexities. *Ann. Stat.*, 33(4):1497–1537, 2005.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Chang, C. and Lin, C. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2 (3):27:1–27:27, 2011.

Chen, X., Lin, Q., and Pena, J. Optimal regularized dual averaging methods for stochastic optimization. In *NIPS 25*, pp. 404–412, 2012.

Clarkson, K.L. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Trans. Algorithms*, 6 (4):63:1–63:30, 2010.

Cotter, A., Shamir, O., Srebro, N., and Sridharan, K. Better mini-batch algorithms via accelerated gradient methods. In *NIPS 24*, pp. 1647–1655, 2011.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. Optimal distributed online prediction. In Getoor, Lise and Scheffer, Tobias (eds.), *ICML*, pp. 713–720, 2011.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. Optimal distributed online prediction using mini-batches. *J. Mach. Learn. Res.*, 13:165–202, 2012.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2): 95–110, 1956.

Ghadimi, S. and Lan, G. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework. *SIAM J. Optim.*, 22(4):1469–1492, 2012.

Hazan, E. Sparse approximate solutions to semidefinite programs. In *LATIN*, pp. 306–316, 2008.

Hazan, E. and Kale, S. Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization. In *COLT*, pp. 421–436, 2011.

Hazan, E. and Kale, S. Projection-free online learning. In *ICML*, pp. 521–528, 2012.

Hu, C., Kwok, J., and Pan, W. Accelerated gradient methods for stochastic optimization and online learning. In *NIPS 22*, pp. 781–789, 2009.

Jaggi, M. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML*, 2013.

Jin, R., Wang, S., and Zhou, Y. Regularized distance metric learning: Theory and algorithm. In *NIPS 22*, pp. 862–870, 2009.

Juditsky, A. and Nesterov, Y. Primal-dual subgradient methods for minimizing uniformly convex functions. Technical report, 2010.

Juditsky, A., Nemirovski, A., and Tauvel, C. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.

Lacoste-Julien, S., Jaggi, M., Schmidt, M., and Pletscher, P. Block-coordinate frank-wolfe optimization for structural svm. In *ICML*, 2013.

Lan, G. An optimal method for stochastic composite optimization. *Math. Program.*, 133:365–397, 2012.

Levitin, E.S. and Polyak, B.T. Constrained minimization methods. *USSR Computational Mathematics and Mathematical Physics*, 6(5):1–50, 1966.

Mahdavi, M., Yang, T., Jin, R., Zhu, S., and Yi, J. Stochastic gradient descent with only one projection. In *NIPS 25*, pp. 503–511, 2012.

Nemirovski, A. Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. Optim.*, 15(1):229–251, 2005.

Nemirovski, A. and Yudin, D.B. *Problem complexity and method efficiency in optimization*. John Wiley & Sons Ltd, 1983.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. Robust stochastic approximation approach to stochastic programming. *SIAM J. Optim.*, 19(4):1574–1609, 2009.

Nesterov, Y. *Introductory lectures on convex optimization: a basic course*, volume 87 of *Applied optimization*. Kluwer Academic Publishers, 2004.

Nesterov, Y. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.

Nesterov, Y. Gradient methods for minimizing composite objective function. Core discussion papers, 2007.

Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*, pp. 449–456, 2012.

Roux, N.L., Manzagol, P., and Bengio, Y. Topmoumoute online natural gradient algorithm. In *NIPS 20*, pp. 849–856, 2008.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. Stochastic convex optimization. In *COLT*, 2009.

Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. Pegasos: primal estimated sub-gradient solver for svm. *Math. Program.*, 127(1):3–30, 2011.

Smale, S. and Zhou, D. Geometry on probability spaces. *Constr. Approx.*, 30:311–323, 2009.

Zhang, T. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML*, pp. 919–926, 2004.