Finite-sum Composition Optimization via Variance Reduced Gradient Descent

Xiangru Lian
University of Rochester
xiangru@yandex.com

Mengdi Wang
Princeton University
mengdiw@princeton.edu

Ji Liu
University of Rochester
ji.liu.uwisc@gmail.com

Abstract

The stochastic composition optimization proposed recently by Wang et al. [2014] minimizes the objective with the compositional expectation form: $\min_x \ (\mathbb{E}_i F_i \circ \mathbb{E}_j G_j)(x)$. It summarizes many important applications in machine learning, statistics, and finance. In this paper, we consider the finite-sum scenario for composition optimization:

$$\min_x f(x) \coloneqq \frac{1}{n}\sum_{i=1}^{n} F_i\left(\frac{1}{m}\sum_{j=1}^{m} G_j(x)\right).$$

We propose two algorithms to solve this problem by combining the stochastic compositional gradient descent (SCGD) and the stochastic variance reduced gradient (SVRG) technique. A constant linear convergence rate is proved for strongly convex optimization, which substantially improves the sublinear rate $O(K^{-0.8})$ of the best known algorithm.

1 INTRODUCTION

The stochastic composition optimization proposed recently by Wang et al. [2014] minimizes the objective with the compositional expectation form:

$$\min_x \ (\mathbb{E}_i F_i \circ \mathbb{E}_j G_j)(x).$$

It has many emerging applications, ranging from machine and reinforcement learning [Dai et al., 2016, Wang et al., 2016] to risk management [Dentcheva et al., 2015]. It is also related to multi-stage stochastic programming [Shapiro et al., 2014] and adaptive simulation [Hu et al., 2014].

In general the stochastic composition optimization is substantially more difficult than the traditional stochastic optimization: $\min_x \ \mathbb{E}_i F_i(x)$. This is because the composition objective is no longer linear with respect to the joint distribution of data indices (i, j). For example, the best-known algorithms studied in [Wang et al., 2014, 2016] achieve a finite-sample error bound $O(K^{-0.8})$ for strongly convex composition optimization, which deteriorates from the optimal rate $O(K^{-1})$ for generic stochastic optimization.

In this paper, we study the *finite-sum* scenario for composition optimization in the following form

$$\min_{x\in\mathbb{R}^N} f(x) \coloneqq F \circ G(x) = F(G(x)), \tag{1}$$

where the inner function $G : \mathbb{R}^N \to \mathbb{R}^M$ is the empirical mean of m component functions $G_i : \mathbb{R}^N \to \mathbb{R}^M$:

$$G(x) = \frac{1}{m}\sum_{i=1}^{m} G_i(x),$$

and the outer function $F : \mathbb{R}^M \to \mathbb{R}$ is the empirical mean of n component functions $F_j : \mathbb{R}^M \to \mathbb{R}$:

$$F(y) = \frac{1}{n}\sum_{j=1}^{n} F_j(y).$$

The finite-sum composition problem models optimization involving two fixed-size empirical data sets. Randomly selecting component functions G_i, F_j can be viewed as randomized retrieval from each of the two data sets.

In this paper, we propose two efficient algorithms (namely, compositional SVRG-1 and compositional SVRG-2) for the finite-sum composition optimization problem (1). The new algorithms are developed by combining the stochastic compositional gradient descent (SCGD) technique [Wang et al., 2014, 2016] and

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

the stochastic variance reduced gradient (SVRG) technique [Johnson and Zhang, 2013]. The new algorithms are motivated by the fact that SVRG is able to improve the sublinear convergence rate of stochastic gradient descent to linear convergence in the case of classical (strongly convex) finite-sum stochastic optimization. We prove that the two algorithms converge linearly for the finite-sum stochastic composition optimization, with query complexity $O\left((m+n+\kappa^4)\log(1/\epsilon)\right)$ and $O\left((m+n+\kappa^3)\log(1/\epsilon)\right)$ respectively (the κ's are two variants of the condition number, and their definitions will be specified later). To the best of our knowledge, this is the first work on finite-sum stochastic composition optimization and linearly convergent algorithms.

1.1 Related Works

This section reviews algorithms related to composition optimization and SVRG.

Composition Optimization draws much attention recently. Contrary to classical stochastic problems, the objective in composition optimization is no longer a plain summation of component functions. Given an index $i \in \{1, 2, \ldots, n\}$, we cannot use a single query to the oracle to get the gradient of a component function F_i with respect to the optimization variable x. In contrast, we can query the gradient of F_i with respect to an intermediate variable G in a single query, and G is itself a summation of m component functions. Thus to calculate the gradient of a single component function in classical stochastic algorithms, we need at least $O(m)$ queries. When m becomes large, the query complexity for classical stochastic optimization algorithms will significantly increase. This encourages people to search for a more sophisticated way to solve such problems.

Wang et al. [2014] proposed the generic composition optimization for the first time. Two stochastic algorithms - Basic SCGD and accelerating SCGD - are proposed for such optimization, with provable convergence rates. A recent work by Wang et al. [2016] improves the convergence rate of accelerating compositional SGD and finds that the optimal convergence rate can be obtained if $G_j(\cdot)$'s are linear. All convergence rates are listed in Table 1. In addition, some special cases such as risk optimization are studied in [Dentcheva et al., 2015].

SVRG is a very powerful technique for large scale optimization. This variance reduced optimization algorithm was originally developed in Johnson and Zhang [2013]. Its main advantage lies on its low storage requirement compared with other variance reduced algorithms [Defazio et al., 2014a,b, Schmidt et al., 2013]. The SVRG technique has been extended by many works [Allen-Zhu and Yuan, 2015, Harikandeh et al., 2015, Kolte et al., 2015, Konečnỳ et al., 2016, Nitanda, 2014, 2015, Xiao and Zhang, 2014] for solving stochastic optimization. Similar algorithms include Konecnỳ and Richtárik [2013], Shalev-Shwartz and Zhang [2013]. For SVRG applied on classical stochastic problems, Johnson and Zhang [2013] proved a $O(\rho^s)$ convergence rate on strongly convex objectives, where ρ is a constant smaller than 1 and s is the epoch number. The query complexity per epoch is $n+\kappa$ where n is the number of component functions and κ is the condition number of the objective. For classical gradient descent, to obtain the same convergence rate, the query complexity per iteration has to be $n\kappa$ (see Nesterov [2013]), which is generally larger than $n+\kappa$. We list the results for SVRG on classical stochastic optimization problems with various types of objectives in Table 2. Gong and Ye [2014] extends the analysis of SVRG for strongly convex optimization to the more general *optimally* strongly convex optimization [Liu and Wright, 2015, Wang et al., 2014] and proves similar convergence rate. Recently, the asynchronous parallel version of SVRG is also studied [Reddi et al., 2015].

1.2 Notation

Throughout this paper, we use the following simple notations.

- $[z]_i$ denotes the ith component of vector (or vector function) z;
- We use the following notations for derivatives of functions. Given any smooth function

$$\begin{aligned} H:\ &\mathbb{R}^N \to \mathbb{R}^M \\ &x \mapsto H(x), \end{aligned}$$

∂H is the Jacobian of H defined by

$$\partial H := \frac{\partial H}{\partial x} = \begin{pmatrix} \frac{\partial [H]_1}{\partial [x]_1} & \cdots & \frac{\partial [H]_1}{\partial [x]_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial [H]_M}{\partial [x]_1} & \cdots & \frac{\partial [H]_M}{\partial [x]_N} \end{pmatrix}.$$

The value of the Jacobian at some point a is denoted by $\partial H(a)$.

For a scalar function

$$\begin{aligned} h:\ &\mathbb{R}^N \to \mathbb{R} \\ &x \mapsto h(x), \end{aligned}$$

the gradient of h is defined by

$$\nabla h(x) = \left(\frac{\partial h(x)}{\partial x}\right)^\top = \left(\frac{\partial h}{\partial [x]_1}, \ldots, \frac{\partial h}{\partial [x]_N}\right)^\top \in \mathbb{R}^N.$$

Table 1: Convergence rates of stochastic composition gradient descent. K is the number of iterations. For fair comparison, the convergence rates for the proposed algorithms (Compositional SVRG-1 and Compositional SVRG-2) have taken the query complexity per iteration (epoch) into consideration.

	Nonconvex	Convex	Strongly Convex
Basic SCGD [Wang et al., 2014]	$O(K^{-1/4})$	$O(K^{-1/4})$	$O(K^{-2/3})$
Acclerating SCGD [Wang et al., 2014]	$O(K^{-2/7})$	$O(K^{-2/7})$	$O(K^{-4/5})$
Acclerating SCGD [Wang et al., 2016] $*$ means if $G_j(\cdot)$'s are linear	$O(K^{-4/9})$ or $O(K^{-1/2})^*$	$O(K^{-2/7})$ or $O(K^{-1/2})^*$	$O(K^{-4/5})$ or $O(K^{-1})^*$
Compositional SVRG-1 ($0<\rho<1$)	-	-	$O\left(\rho^{\frac{K}{m+n+\kappa^4}}\right)$
Compositional SVRG-2 ($0<\rho<1$)	-	-	$O\left(\rho^{\frac{K}{m+n+\kappa^3}}\right)$

Table 2: Query complexity and convergence rate for SVRG on different objectives. QCPE stands for query complexity per epoch. ρ is a constant smaller than 1. s is the epoch number. n is the number of component functions.

SVRG	Nonconvex [Reddi et al., 2016] [Allen-Zhu and Hazan, 2016]	Convex [Reddi et al., 2016]	Strongly Convex [Johnson and Zhang, 2013]
Rate	$O(1/(sn^{1/3}))$	$O(1/(s\sqrt{n}))$	$O(\rho^s)$
QCPE	$O(n)$	$O(n)$	$O(n+\kappa)$

Under this set of notations, with the chain rule we can easily find the gradient of a composition function $f=F(G(x))$ to be:

$$\nabla f(x)=(\partial G(x))^\top \nabla F(G(x)).^1$$

- x^* denotes the optimal solution of (1);
- Given a multiset $\mathcal{A}$[2], we use $\text{len}(\mathcal{A})$ to denote the number of elements in $\mathcal{A}$. For example, if $\mathcal{A}=\{1,2,3,1\}$, then $\text{len}(\mathcal{A})=4$. We use $\mathcal{A}[i]$ to represent the ith element in $\mathcal{A}$.
- $\mathbb{E}_i$ denotes taking expectation w.r.t. the random variable i.
- $\mathbb{E}$ denotes taking expectation w.r.t. all random variables.

2 Preliminary: SVRG

We review the standard SVRG algorithm in this section for completion. Consider to solve the following finite sum optimization problem

$$\min_x f(x)=\frac{1}{n}\sum_{i=1}^n F_i(x).$$

The SVRG algorithm basically stores the gradient of f at a reference point $\tilde{x}$ (the reference point will be updated for every a few iterations): $\tilde{f}':=\frac{1}{n}\sum_{i=1}^n \nabla F_i(\tilde{x})$. Based on such a reference gradient, SVRG estimates the gradient at each iteration by

$$\hat{f}'_k:=\tilde{f}'-\nabla F_i(\tilde{x})+\nabla F_i(x_k)$$

where i is uniformly randomly sampled from $\{1,2,\ldots,n\}$. The next iterate is updated by

$$x_{k+1}=x_k-\gamma_k \hat{f}'_k.$$

The computation complexity per iteration is comparable to SGD. The estimated gradient is also an *unbiased* estimate for the true gradient

$$\mathbb{E}(\hat{f}'_k)=f'_k:=\frac{1}{n}\sum_{i=1}^n \nabla F_i(x_k).$$

The key improvement lies on that the variance $\mathbb{E}(\|\hat{f}'_k-f'_k\|^2)$ decreases to zero when x_k converges to the optimal point for SVRG while it is a constant for SGD. Therefore, SVRG admits a much better convergence rate (linear convergence for strongly convex optimization and sublinear for convex optimization) than SGD. For completeness, the complete SVRG algorithm is shown in Algorithm 1.

3 COMPOSITIONAL-SVRG ALGORITHMS

This section introduces two proposed compositional-SVRG algorithms for solving the finite sum composition optimization in (1). In the spirit of SVRG,

[1]Note that the gradient operator always calculates the gradient with respect to the first level variable. That is to say, by $\nabla F(G(x))$ we mean the gradient of $F(y)$ at $y=G(x)$, *not* the gradient of $F(G(x))$ with respect to x.

[2]A multiset is a generalization of the concept of a set, allowing duplicated elements.

Algorithm 1 SVRG [Johnson and Zhang, 2013]

Require: K (update frequency), γ (step length), S (total number of epochs), $\tilde{x}_0$ (initial point)
Ensure: $\tilde{x}_S$.
for $s = 1, 2, \ldots, S$ **do**
  Update the reference point $\tilde{x} \leftarrow \tilde{x}_{s-1}$
  $\tilde{f}' \leftarrow \nabla f(\tilde{x})$ ▷ n queries (non-composition optimization)
  $x_0 \leftarrow \tilde{x}$
  for $k = 0, 1, 2, \ldots, K-1$ **do**
    Uniformly sample pick i_k from $\{1, 2, \ldots, n\}$
    Estimate $\nabla f(x_k)$ using

$$\hat{f}'_k = \tilde{f}' - \nabla F_i(\tilde{x}) + \nabla F_i(x_k)$$

    ▷ 2 queries (non-composition optimization)
    Update x_{k+1} by

$$x_{k+1} \leftarrow x_k - \gamma \hat{f}'_k$$

  end for
  $\tilde{x}_s \leftarrow x_r$ for randomly chosen $r \in \{0, \cdots, K-1\}$
end for

the two compositional-SVRG algorithms need a reference point $\tilde{x}$ to estimate the gradients (but in different ways). However, unlike SVRG, the estimated gradients are biased due to the "composition" structure in the objective.

3.1 Compositional-SVRG-1

In the first proposed algorithm, given a reference point $\tilde{x}$, we first store the gradient $\tilde{f}' = \nabla f(\tilde{x})$ and the value of the inner function $\tilde{G} := G(\tilde{x})$. To estimate the gradient at the current iterate x_k, one needs to estimate $G(x_k)$ first by sampling a mini-batch multiset $\mathcal{A}_k$ with size A:

$$\hat{G}_k = \tilde{G} - \frac{1}{A} \sum_{1 \le j \le A} (G_{\mathcal{A}_k[j]}(\tilde{x}) - G_{\mathcal{A}_k[j]}(x_k)). \quad (2)$$

Based on the estimate of $G(x_k)$, the gradient $\nabla f(x_k)$ is estimated by

$$\hat{f}'_k = (\partial G_{j_k}(x_k))^\top \nabla F_{i_k}(\hat{G}_k) - (\partial G_{j_k}(\tilde{x}))^\top \nabla F_{i_k}(\tilde{G}) + \tilde{f}' \quad (3)$$

where i_k is uniformly sampled from $\{1, 2, \ldots, n\}$ and j_k is uniformly sampled from $\{1, 2, \ldots, m\}$.

Note that unlike SVRG, this estimated $\hat{f}$ is usually *biased*. More specifically,

$$\mathbb{E}_{i_k, j_k, \mathcal{A}_k}(\hat{f}'_k) \neq \nabla f(x_k).$$

This is also the key challenge to prove the linear convergence in the analysis. The Compositional-SVRG-1 algorithm is summarized in Algorithm 2. The query complexity in each step is provided in Algorithm 2 for convenience.

3.2 Compositional-SVRG-2

In the second proposed algorithm, given a reference point $\tilde{x}$, we still first store the gradient $\tilde{f}' = \nabla f(\tilde{x})$ and the value of the inner function $\tilde{G} := G(\tilde{x})$. However, here we further store the value of the Jacobian $\tilde{G}' := \partial G(\tilde{x})$. To estimate the gradient at the current iterate x_k, one still estimates $G(x_k)$ first by sampling a mini-batch multiset $\mathcal{A}_k$ with size A:

$$\hat{G}_k = \tilde{G} - \frac{1}{A} \sum_{1 \le j \le A} (G_{\mathcal{A}_k[j]}(\tilde{x}) - G_{\mathcal{A}_k[j]}(x_k)). \quad (4)$$

Here comes the difference from Algorithm 2. We also estimates $\partial G(x_k)$ by sampling a mini-batch multiset $\mathcal{B}_k$ with size B:

$$\hat{G}'_k := \tilde{G}' - \frac{1}{B} \sum_{0 \le j \le B} \left(\partial G_{\mathcal{B}_k[j]}(\tilde{x}) - \partial G_{\mathcal{B}_k[j]}(x_k)\right). \quad (5)$$

Based on the estimation of $G(x_k)$ and $\partial G(x_k)$, the gradient $\nabla f(x_k)$ is estimated by

$$\hat{f}'_k = (\hat{G}'_k)^\top \nabla F_{i_k}(\hat{G}_k) - (\tilde{G}')^\top \nabla F_{i_k}(\tilde{G}) + \tilde{f}'. \quad (6)$$

where i_k is uniformly sampled from $\{1, 2, \ldots, n\}$. Thus Algorithm 3 features one more estimation in each iteration. This extra computation pays off by an improved convergence rate.

Even though we have an extra estimation here, this estimated $\hat{f}$ is still *biased*. More specifically,

$$\mathbb{E}_{i_k, \mathcal{B}_k, \mathcal{A}_k}(\hat{f}'_k) \neq \nabla f(x_k).$$

The Compositional-SVRG-2 algorithm is summarized in Algorithm 3. The query complexity in each step is provided in Algorithm 3 for convenience.

4 THEORETICAL ANALYSIS

In this section we will show the convergence results for Algorithms 2 and 3. Due to the page limitation, all proofs are provided in the supplement. Before we show the main results, let us make some global assumptions below, which are commonly used for the analysis of stochastic composition optimization algorithms.

Strongly Convex Objective $f(x)$ in (1) is strongly convex with parameter μ_f:

$$f(y) \ge f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu_f}{2} \|y - x\|^2, \quad \forall x, y.$$

Algorithm 2 Compositional-SVRG-1

Require: K (the total number of iterations in the inner loop), S (the total number of iterations in the outer loop), A (the size of the minibatch multiset), γ (steplength), and $\tilde{x}_0$ (initial point).

Ensure: $\tilde{x}_S$.

for $s = 1, 2, \ldots, S$ **do**
  Update the reference point: $\tilde{x} \leftarrow \tilde{x}_{s-1}$
  $\tilde{G} \leftarrow G(\tilde{x})$ ▷ m queries
  $\tilde{f}' \leftarrow \nabla f(\tilde{x})$ ▷ $m+n$ queries
  $x_0 \leftarrow \tilde{x}$
  for $k = 0, 1, 2, \ldots, K-1$ **do**
    Uniformly sample from $\{1, 2, \ldots, m\}$ for A times *with replacement* to form a mini-batch multiset $\mathcal{A}_k$
    Estimate $G(x_k)$ by $\hat{G}_k$ using (2) ▷ $2A$ queries
    Uniformly sample i_k from $\{1, 2, \ldots, n\}$ and j_k from $\{1, 2, \ldots, m\}$
    Estimate $\nabla f(x_k)$ by $\hat{f}'_k$ using (3) ▷ 4 queries
    Update x_{k+1} by

$$x_{k+1} = x_k - \gamma \hat{f}'_k$$

  end for
  $\tilde{x}_s \leftarrow x_r$ for randomly chosen $r \in \{0, \cdots, K-1\}$
end for

Bounded Jacobian of Inner Functions We assume that the following upper bounds on all inner component functions:

$$\|\partial G_j(x)\| \leqslant B_G, \quad \forall x, \forall j \in \{1, \cdots, m\}. \tag{7}$$

Lipschitzian Gradients We assume there exist constants L_F, L_G and L_f satisfying $\forall x, \forall y, \forall i \in \{1, \cdots, n\}$, $\forall j \in \{1, \cdots, m\}$:

$$\|\nabla F_i(x) - \nabla F_i(y)\| \leqslant L_F \|x - y\|, \tag{8}$$

$$\|\partial G_j(x) - \partial G_j(y)\| \leqslant L_G \|x - y\|, \tag{9}$$

$$\|(\partial G_j(x))^\top \nabla F_i(G(x)) - (\partial G_j(y))^\top \nabla F_i(G(y))\| \leqslant L_f \|x - y\|, \tag{10}$$

Note that we immediately have

$$\begin{aligned}
&\|\nabla f(x) - \nabla f(y)\| \\
=&\frac{1}{mn}\left\|\sum_{i,j}\left(\partial G_j(x))^\top \nabla F_i(G(x)) - (\partial G_j(y))^\top \nabla F_i(G(y)\right)\right\| \\
\leqslant&\frac{1}{mn}\sum_{i,j}\left\|\partial G_j(x))^\top \nabla F_i(G(x)) - (\partial G_j(y))^\top \nabla F_i(G(y)\right\| \\
\leqslant& L_f \|x - y\|, \forall x, y.
\end{aligned} \tag{11}$$

Thus the Lipschitz constant for the gradient of the whole objective f is also L_f.

Algorithm 3 Compositional-SVRG-2

Require: $K, S, A, B, \gamma, \tilde{x}_0$ ▷ The meaning of the variables are the same as in Algorithm 2. B is the size of another minibatch multiset.

Ensure: $\tilde{x}_S$.

for $s = 1, 2, \ldots, S$ **do**
  Update the reference point $\tilde{x} \leftarrow \tilde{x}_{s-1}$
  $\tilde{G} \leftarrow G(\tilde{x})$ ▷ m queries
  $\tilde{G}' \leftarrow \partial G(\tilde{x})$ ▷ m queries
  $\tilde{f}' \leftarrow \nabla f(\tilde{x})$ ▷ n queries
  $x_0 \leftarrow \tilde{x}$
  for $k = 0, 1, 2, \ldots, K-1$ **do**
    Uniformly sample from $\{1, 2, \ldots, m\}$ for A and B times *with replacement* to form two mini-batch multiset $\mathcal{A}_k$ and $\mathcal{B}_k$ respectively
    Estimate $G(x_k)$ by $\hat{G}_k$ using (4) ▷ $2A$ queries
    Estimate $\partial G(x_k)$ by $\hat{G}'_k$ using (5) ▷ $2B$ queries
    Uniformly sample pick i_k from $\{1, 2, \ldots, n\}$
    Estimate $\nabla f(x_k)$ by $\hat{f}'_k$ using (6) ▷ 2 queries
    Update x_{k+1} by

$$x_{k+1} \leftarrow x_k - \gamma \hat{f}'_k$$

  end for
  $\tilde{x}_s \leftarrow x_r$ for randomly chosen $r \in \{0, \cdots, K-1\}$
end for

Solution Existence The problem (1) has at least one solution x^*.

Next we use two theorems to show our main results on the compositional-SVRG algorithms. All theorems and corollaries hold under the assumptions stated above.

Theorem 1 (Convergence for Algorithm 2)**.** *For Algorithm 2 we have*

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\|x_k - x^*\|^2 \leqslant \frac{\beta_1}{\beta_2}\mathbb{E}\|\tilde{x} - x^*\|^2,$$

where

$$\beta_1 = \frac{1}{K} + \left(\frac{16\gamma B_G^2 L_F^2}{\mu_f} + 4\gamma^2 B_G^2 L_F^2\right)\frac{8B_G^2}{A} + 10\gamma^2 L_f^2;$$

$$\beta_2 = \frac{7\mu_f \gamma}{4} - \left(\frac{16\gamma B_G^2 L_F^2}{\mu_f} + 4\gamma^2 B_G^2 L_F^2\right)\frac{8B_G^2}{A} - 8\gamma^2 L_f^2.$$

To ensure the convergence, one should appropriately choose parameters γ, A, and K to make the ratio $\beta_1/\beta_2 < 1$. The following provides one specification for those parameters.

Corollary 1 (Linear Rate for Algorithm 2)**.** *Choose parameters in Algorithm 2 as follows:*

$$\begin{aligned}\gamma &= \frac{\mu_f}{32L_f^2},\\ A &= \frac{512B_G^4L_F^2}{\mu_f^2},\\ K &= \frac{512L_f^2}{\mu_f^2},\end{aligned}$$

The following convergence rate for Algorithm 2 holds:

$$\mathbb{E}\|\tilde{x}_{s+1}-x^*\|^2 = \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\|x_k-x^*\|^2 \leqslant \frac{7}{8}\mathbb{E}\|\tilde{x}_s-x^*\|^2.$$

Corollary 1 essentially suggests a linear convergence rate. To achieve a fixed solution accuracy ϵ, that is, $\mathbb{E}(\|\tilde{x}_s - x^*\|^2) \leq \epsilon$, the required number of queries is $O\left((m+n+KA+K)\log(1/\epsilon)\right) = O\left(\left(m+n+\frac{L_F^2L_f^2}{\mu_f^4}+\frac{L_f^2}{\mu_f^2}\right)\log(1/\epsilon)\right)$ based on the query complexity of each step in Algorithm 2. Let $\kappa = \max\{\frac{L_F}{\mu_f}, \frac{L_f}{\mu_f}\}$.[3] we see the query complexity of Algorithm 2 is $O((m+n+\kappa^4)\log(1/\epsilon))$. Note that this κ^4 is much smaller in some special cases, because L_F/μ_f (and also L_G/L_f in Corollary 2 we will discuss later) could be much smaller than L_f/μ_f.

To analyze the convergence of Algorithm 3, we need one more assumption on the gradients of the outer functions:

Bounded Gradients of Outer Functions

$$\|\nabla F_i(x)\| \leqslant B_F, \forall x, \forall i \in \{1,\cdots,n\}. \quad (12)$$

Then together with the new assumption in (12), we have the following convergence result for Algorithm 3.

Theorem 2 (Convergence for Algorithm 3)**.** *For algorithm 3 we have*

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}(f(x_k)-f^*) \leqslant \frac{\beta_3}{\beta_4}\mathbb{E}(f(\tilde{x})-f^*),$$

where

$$\begin{aligned}\beta_3 =& \frac{2}{K\mu_f}+\frac{256\gamma B_G^4L_F^2}{\mu_f A}\\ &+\gamma^2\left(\frac{128}{\mu_f}\left(\frac{B_F^2L_G^2}{B}+\frac{B_G^4L_F^2}{A}\right)+20L_f\right);\\ \beta_4 =& \frac{3\gamma}{2}-\frac{256\gamma B_G^4L_F^2}{\mu_f A}\\ &-\gamma^2\left(\frac{128}{\mu_f}\left(\frac{B_F^2L_G^2}{B}+\frac{B_G^4L_F^2}{A}\right)+16L_f\right).\end{aligned}$$

[3]Note that in classical SVRG, G is an identity function, so $L_F = L_f$. The κ reduces to the conventional condition number.

This theorem admits a similar structure to Theorem 1. We essentially need to appropriately choose γ, K, A, and B to make $\beta_3/\beta_4 < 1$. The following corollary provides a specification for these parameters.

Corollary 2 (Linear Rate for Algorithm 3)**.** *Choose parameters in Algorithm 3 as follows:*

$$\begin{aligned}\gamma &= \frac{1}{320L_f};\\ K &\geqslant \frac{5120L_f}{\mu_f};\\ A &\geqslant \max\left\{\frac{1024B_G^4L_F^2}{\mu_f^2}, \frac{32B_G^4L_F^2}{5\mu_fL_f}\right\};\\ B &\geqslant \frac{32B_F^2L_G^2}{5\mu_fL_f},\end{aligned}$$

we have the following linear convergence rate for Algorithm 3:

$$\mathbb{E}(f(\tilde{x}_{s+1})-f^*) \leqslant \frac{9}{17}\mathbb{E}(f(\tilde{x}_s)-f^*).$$

Let $\kappa = \max\{\frac{L_F}{\mu_f}, \frac{L_G}{L_f}, \frac{L_f}{\mu_f}\}$. Corollary 2 suggests a total query complexity of $O((m+n+K(A+B))\log(1/\epsilon)) = O((m+n+\kappa^3)\log(1/\epsilon))$. Here the query complexity is slightly better[4] than that in Corollary 1. However we need a new assumption that the gradient of F_i's to be bounded and we need an extra estimation for the Jacobian of G at the beginning of each epoch.

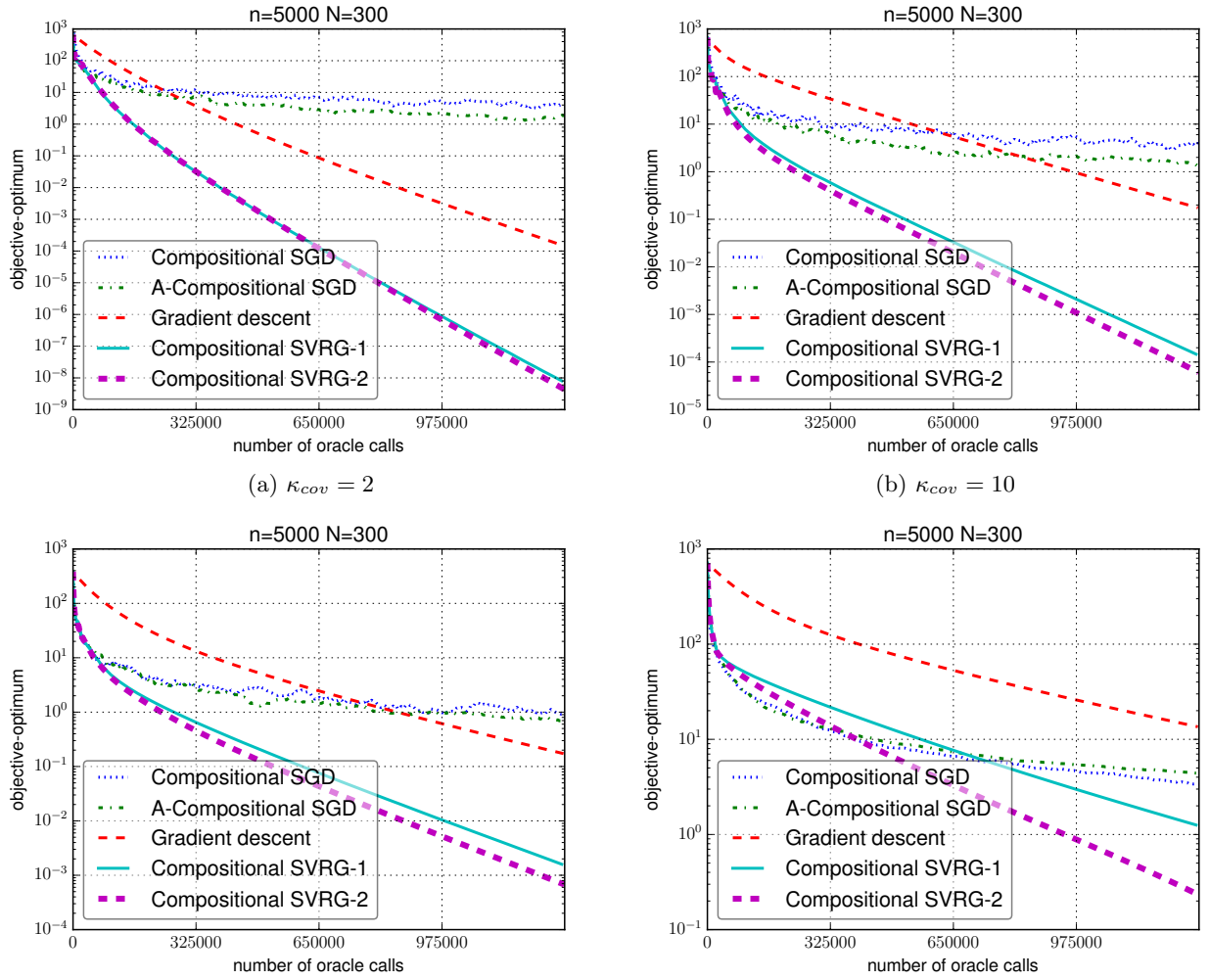
5 EXPERIMENT

We conduct empirical studies for the proposed two algorithms by comparing them to three state-of-the-art algorithms:

- Gradient descent,
- Compositional SGD [Wang et al., 2014, Algorithm 1],
- Accelerating Compositional SGD [Wang et al., 2016, Algorithm 1].

We use the mean-variance optimization in portfolio management as the objective. Given N assets, let $r_t \in \mathbb{R}^N (t = 1,\ldots,n)$ be the reward vectors observed at different time points. The goal is to maximize the return of the investment as well as controlling the investment risk. Let $x \in \mathbb{R}^N$ be the quantities invested

[4]Here we mean "roughly better", because the definitions of κ are different in Corollary 1 and 2, though in most cases they are of the same order.



(a) $\kappa_{cov} = 2$ (b) $\kappa_{cov} = 10$

(c) $\kappa_{cov} = 20$ (d) $\kappa_{cov} = 50$

Figure 1: Mean-variance portfolio optimization on synthetic data ($n = 2000, N = 200$). The y-axis is the objective value minus the optimal value of the objective. The x-axis is the number of oracle calls. The "Compositional SVRG-1" is the Algorithm 2, the "Compositional SVRG-2" is the Algorithm 3. The "Compositional SGD" is the Algorithm 1 in Wang et al. [2014] and The "A-Compositional SGD" is the Algorithm 1 in Wang et al. [2016]. Both SVRG version algorithms use "Compositional-SGD" algorithm to initialize first several steps. The κ_{cov} is the conditional number of the covariance matrix of the corresponding Gaussian distribution used to generate reward vectors in each figure. Subfigures (a), (b), (c), and (d) draw the convergence curves for all algorithms with each figure having a different κ_{cov}.

to each portfolio. The problem can be formulated into the following mean-variance optimization[5]:

$$\max_x \quad \frac{1}{n}\sum_{i=1}^{n}\langle r_i, x\rangle - \frac{1}{n}\sum_{i=1}^{n}\left(\langle r_i, x\rangle - \frac{1}{n}\sum_{j=1}^{n}\langle r_j, x\rangle\right)^2.$$

It can be viewed as an instance of the composition optimization (1) with the specification for $G_j(\cdot)$ and $F_i(\cdot)$ as the following:

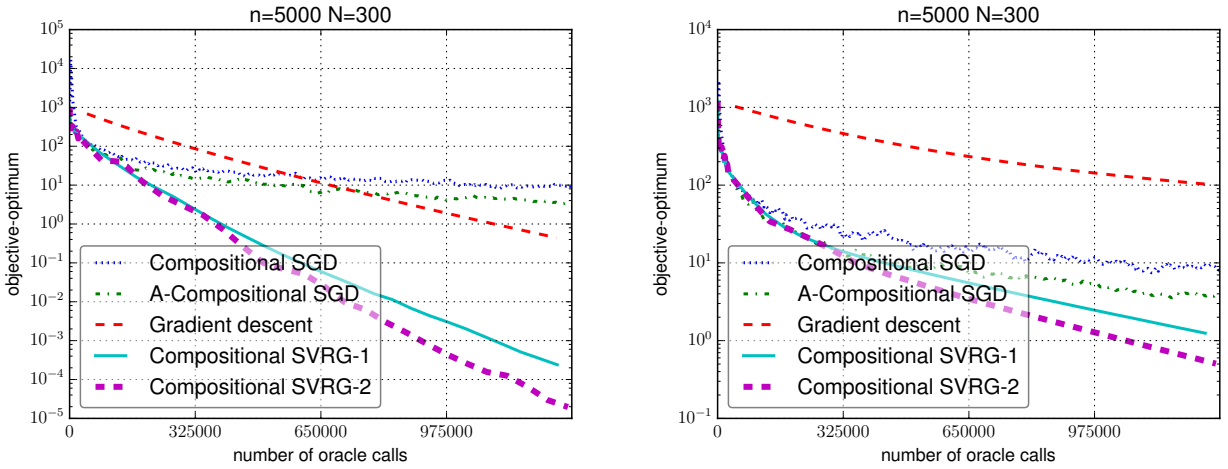
$$G_j(x) = \begin{pmatrix} x \\ \langle r_j, x\rangle \end{pmatrix}, j = 1, \ldots, n;$$

$$F_i(y) = -y_{N+1} + (\langle r_i, y_{1:N}\rangle - y_{N+1})^2, i = 1, \ldots, n,$$

where $y_{1:N}$ denotes the sub-vector consisting of the first N components of the vector $y \in \mathbb{R}^{N+1}$ and y_{N+1} denotes the last component of y.

In the experiment we choose $n = 2000$ and $N = 200$. The reward vectors are generated with the procedure below:

[5] This formulation is just used for proof of concept. A more efficient way would be simply calculating $\sum_j r_j$. Then the problem reduces to a standard stochastic optimization.



(a) $\kappa_{cov} = 3$ (b) $\kappa_{cov} = 40$

Figure 2: Mean-variance portfolio optimization on synthetic data with $n = 5000, N = 300$. Other settings are the same as Figure 1.

1. Generate a random Gaussian distribution on $\mathbb{R}^N$ with the condition number of its covariance matrix denoted by κ_{cov}.
2. Each r_i is sampled from this Gaussian distribution with all elements set to its absolute value to make sure the problem has a solution.

We can then control κ_{cov} to roughly control the κ of our composition optimization problem, because κ is proportional to κ_{cov}. We report the comparison results in Figure 1. The initial points are chosen to be the same for all algorithms and the x-axis in Figure 1 is the computational cost measured by the number of queries to the oracle. That is, whenever the algorithm queries $\nabla F_i(y)$ or $\partial G_i(x)$ or $G_i(x)$ for some i at some point, the x-axis value is incremented by 1. Like the SVRG algorithm [Johnson and Zhang, 2013], both compositional-SVRG algorithms run the compositional-SGD algorithm (which is the Algorithm 1 in Wang et al. [2014]) for the first 10000 iterations and then run the proposed SVRG algorithms. We observe that

- The proposed two algorithms (Compositional SVRG-1 and Compositional SVRG-2) converge at a linear rate and outperform other algorithms overall;
- Compositional SVRG-2 becomes faster than Compositional SVRG-1 when κ_{cov} becomes larger, while they are comparable when κ_{cov} are small. This observation is consistent with our theoretical analysis, since Compositional SVRG-2 has a better dependency on the condition number than Compositional SVRG-1.

We also test our algorithms on problem with a larger size ($n = 5000, N = 300$), and show the results in Figure 2.

6 Conclusion and Future Work

This paper considers the finite-sum composition optimization and proposes two efficient algorithm by using the SVRG technique to reduce variance of compositional gradient. The proposed two algorithms admit the linear convergence rate for strongly convex objectives with query complexity $O((m+n+\kappa^4)\log(1/\epsilon))$ and $O((m+n+\kappa^3)\log(1/\epsilon))$ respectively. To the best of our knowledge, this is the first work to study the general finite-sum composition optimization. The future work will be 1) the convergence rate and query complexity for weakly convex problem; 2) the convergence rate and query complexity for nonconvex optimization; 3) how (or is it possible) to improve the query complexity to $O((m+n+\kappa)\log(1/\epsilon))$ to make it consistent with SVRG for the classical stochastic optimization?

Acknowledgements

Mengdi Wang was supported by NSF CMMI 1653435 and NSF DMS 1619818. We thank the reviewers for their constructive comments and suggestions, especially for pointing out the potentially more efficient implementation in the experiment.

References

Z. Allen-Zhu and E. Hazan. Variance reduction for faster non-convex optimization. *arXiv preprint arXiv:1603.05643*, 2016.

Z. Allen-Zhu and Y. Yuan. Improved SVRG for non-strongly-convex or sum-of-non-convex objectives. *arXiv preprint arXiv:1506.01972*, 2015.

B. Dai, N. He, Y. Pan, B. Boots, and L. Song. Learning from conditional distributions via dual kernel embeddings. *arXiv preprint arXiv:1607.04579*, 2016.

A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014a.

A. Defazio, J. Domke, and T. S. Caetano. Finito: A faster, permutable incremental gradient method for big data problems. In *ICML*, pages 1125–1133, 2014b.

D. Dentcheva, S. Penev, and A. Ruszczyński. Statistical estimation of composite risk functionals and risk optimization problems. *Annals of the Institute of Statistical Mathematics*, pages 1–24, 2015.

P. Gong and J. Ye. Linear convergence of variance-reduced stochastic gradient without strong convexity. *arXiv preprint arXiv:1406.1102*, 2014.

R. Harikandeh, M. O. Ahmed, A. Virani, M. Schmidt, J. Konečnỳ, and S. Sallinen. Stopwasting my gradients: Practical SVRG. In *Advances in Neural Information Processing Systems*, pages 2251–2259, 2015.

J. Hu, E. Zhou, and Q. Fan. Model-based annealing random search with stochastic averaging. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 24(4):21, 2014.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

R. Kolte, M. Erdogdu, and A. Ozgür. Accelerating SVRG via second-order information. In *NIPS Workshop on Optimization for Machine Learning*, 2015.

J. Konecnỳ and P. Richtárik. Semi-stochastic gradient descent methods. 2013.

J. Konečnỳ, J. Liu, P. Richtárik, and M. Takáč. Mini-batch semi-stochastic gradient descent in the proximal setting. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):242–255, 2016.

J. Liu and S. J. Wright. Asynchronous stochastic coordinate descent: Parallelism and convergence properties. *SIAM Journal on Optimization*, 25(1):351–376, 2015.

Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems*, pages 1574–1582, 2014.

A. Nitanda. Accelerated stochastic gradient descent for minimizing finite sums. *arXiv preprint arXiv:1506.03016*, 2015.

S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. J. Smola. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems*, pages 2647–2655, 2015.

S. J. Reddi, A. Hefny, S. Sra, B. Póczós, and A. Smola. Stochastic variance reduction for nonconvex optimization. *arXiv preprint arXiv:1603.06160*, 2016.

M. Schmidt, N. L. Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388*, 2013.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

A. Shapiro, D. Dentcheva, et al. *Lectures on stochastic programming: modeling and theory*, volume 16. SIAM, 2014.

M. Wang, E. X. Fang, and H. Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, pages 1–31, 2014.

M. Wang, J. Liu, and E. X. Fang. Accelerating stochastic composition optimization. *arXiv preprint arXiv:1607.07329*, 2016.

L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.