Adaptive smoothing mini-batch stochastic accelerated gradient method for nonsmooth convex stochastic composite optimization

Ruyu Wang\textsuperscript{1} and Chao Zhang\textsuperscript{1*}

\textsuperscript{1}Department of Applied Mathematics, Beijing Jiaotong University, No.3 Shangyuancun, Haidian District, 100044, Beijing, China.

*Corresponding author(s). E-mail(s): zc.njtu@163.com; Contributing authors: wangruyu@bjtu.edu.cn;

Abstract

This paper considers a class of convex constrained nonsmooth convex stochastic composite optimization problems whose objective function is given by the summation of a differentiable convex component, together with a general nonsmooth but convex component. The nonsmooth component is not required to have easily obtainable proximal operator, or have the max structure that the smoothing technique in [1] can be used. In order to solve such type problems, we propose an adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method, which combines the stochastic approximation method, the Nesterov’s accelerated gradient method [1], and the smoothing methods that allow general smoothing approximations. Convergence of the method is established. Moreover, the order of the worst-case iteration complexity is better than that of the state-of-the art stochastic approximation methods. Numerical results are provided to illustrate the efficiency of the proposed AdaSMSAG method for a risk management in portfolio optimization and a family of Wasserstein distributionally robust support vector machine problems with real data.

Keywords: Smoothing method, Stochastic approximation, Constrained convex stochastic programming, Mini-batch of samples, Complexity
1 Introduction

In this paper, we consider the constrained nonsmooth convex stochastic composite minimization problem

$$\psi^* := \min_{x \in X} \{ \psi(x) := f(x) + h(x) \},$$  \hspace{1cm} (1)

where

$$f(x) = E[F(x, \xi)], \quad h(x) = E[H(x, \xi)].$$  \hspace{1cm} (2)

Here \(X\) is a closed convex set in the Euclidean space \(\mathbb{R}^n\), \(\xi\) is a random vector whose probability distribution is supported on \(\Xi \subseteq \mathbb{R}^d\), and \(F(\cdot, \xi) : X \rightarrow \mathbb{R}\) and \(H(\cdot, \xi) : X \rightarrow \mathbb{R}\) are functions such that for every \(\xi \in \Xi\), \(F(\cdot, \xi)\) is convex with \(L\)-Lipschitz continuous gradient \(\nabla F(\cdot, \xi)\), and \(H(\cdot, \xi)\) is nonsmooth convex. We do not assume the proximal operator of \(H(\cdot, \xi) : X \rightarrow \mathbb{R}\) and consequently the proximal operator of \(h\) are easy to compute. Many applications especially in machine learning are in this setting. There are two main difficulties to design an efficient algorithm for (1): the general nonsmoothness of \(h\), and the stochastic setting that leads the difficulty for computing the objective value and the gradient of \(\psi\) for the random vector \(\xi\) in the general support set \(\Xi\).

In the deterministic setting, i.e., \(\Xi\) is a singleton in (2), smoothing algorithms that utilize the structure of the problems to define smoothing functions have been shown to be an effective way to overcome the nonsmoothness in optimization.

Nesterov introduced a smoothing scheme [1] for nonsmooth convex problems with additional assumptions that \(X\) is bounded, and \(h(x)\) is a nonsmooth convex function with explicit max-structure as follows.

$$h(x) = \max_{u \in U} \{ \langle Ax, u \rangle - Q(u) \},$$  \hspace{1cm} (3)

where \(U\) is a bounded closed convex set, \(A\) is a linear operator, and \(Q(\cdot)\) is a continuous convex function. In [1], the max structure in (3) is necessary to construct the smooth approximation \(\tilde{h}_\mu(x)\) of \(h(x)\) with its gradient \(\nabla \tilde{h}_\mu(x)\) being Lipschitz continuous. This method combines the smooth approximation with a fixed predetermined smoothing parameter \(\mu\), and the Nesterov’s accelerated gradient scheme [2, 3] for smooth convex optimization problems, to achieve the worst-case iteration complexity

$$O \left( \frac{1}{\epsilon} \right),$$

where \(\epsilon\) is the desired absolute accuracy of the difference between the original function value at the approximate solution and the the dual function value.
Adaptive smoothing mini-batch stochastic accelerated gradient method

at the corresponding approximate dual solution. It was shown that the order $O \left( \frac{1}{\varepsilon^2} \right)$ of the worst-case iteration complexity cannot be improved for the simplest subgradient method in nonsmooth optimization \[22\], and Nesterov pointed out that the improvement order of $O \left( \frac{1}{\varepsilon} \right)$ is beneficial from the proper use of the structure of the problem \[1\].

Quoc proposed an adaptive smoothing proximal gradient method \[4\] that updates $\mu$ from $\mu_k$ to $\mu_{k+1}$ at each iteration for unconstrained nonsmooth composite convex minimization problems with

$$X = \mathbb{R}^n, \quad f(x) = 0, \quad h(x) = h_1(x) + h_2(x)$$

where $h_1(x)$ is a nonsmooth convex function with explicit max-structure and $h_2(x)$ is a simple nonsmooth convex function such that the proximal operator is easy to compute. This method automatically updates the smoothing parameter at each iteration by

$$\mu_{k+1} = \frac{\tau \mu_1}{k + \tau},$$

where $k \geq 1$, $\tau \geq 1$ is a constant and $\mu_1$ is an initial smoothing parameter. It achieves the $O \left( \frac{1}{\varepsilon} \right)$-worst-case iteration complexity as in \[1\], and the adaptive updating of the smoothing parameter has been demonstrated efficiency in numerical experiments.

Nestrov’s smoothing method \[1\] and Quoc’s adaptive smoothing proximal gradient method \[4\] require the nonsmooth term of the objective function to have the explicit max-structure, or easily obtainable proximal operator. Fortunately, a type of smoothing methods for constrained nonsmooth optimization problems employs the smoothing approximations without such requirements, such as the smoothing projected gradient method \[9\], the smoothing quadratic regularization method \[10\], the smoothing sequential quadratic programming method \[11\], the smoothing sequential quadratic programming framework \[12\], and the smoothing active set method in \[13\]. At each step of a certain smoothing method, the objective function is approximated by a smooth function with a fixed smoothing parameter and a certain smooth algorithm is employed to get the next iterate point. A special updating rule for the smoothing parameter is then checked to determine whether to decrease the smoothing parameter or keep it unchanged. More information about this type of smoothing methods and the various smoothing approximations can be found in the excellent survey paper \[7\].

For the stochastic setting we focus on in this paper, the stochastic approximation (SA) method is one important approach for solving the stochastic convex programming, which can be dated back to the pioneering paper by Robbins and Monro \[14\]. A robust version of the SA method developed by Polyak \[15\], and Polyak and Juditsky \[16\], improves the original version of the SA method. It has clear that for nonsmooth stochastic convex optimization, the order of iteration complexity required to find an $\varepsilon$-approximate solution $\bar{x}$,
Adaptive smoothing mini-batch stochastic accelerated gradient method

i.e. $\psi(\bar{x}) - \psi^* \leq \epsilon$ for a pre-determined accuracy $\epsilon > 0$, cannot be smaller than

$$O\left(\frac{1}{\epsilon^2}\right),$$

as pointed out in [17]. By combining the Nesterov’s accelerated gradient scheme [2, 3] and the SA method, Lan [19] proposed an accelerated SA (AC-SA) method to the nonsmooth stochastic convex problems in a convex compact set, which achieves the order $O\left(\frac{1}{\epsilon^2}\right)$ of iteration complexity. S. Ghadimi et al. [20] extended this method to the nonsmooth stochastic strongly convex problems. In 2016, S. Ghadimi et al. [21] proposed an accelerated gradient (AG) method for the unconstrained nonconvex nonsmooth optimization problems, where the nonsmooth component is required to be a simple convex function such as $\|x\|_1$, for which the proximal operator is easy to compute. Bai et al. In 2021, Wang et al. [28] proposed a mini-batch stochastic Nesterov’s smoothing method for the constrained convex stochastic composite optimization problems. The nonsmooth component has an explicit max structure that may not easy to compute its proximal operator.

In this paper, we propose an adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method for solving stochastic optimization problems (1) with a general nonsmooth convex component $h$. We do not assume that $h$ has the max structure in (3), or its proximal operator is easily obtained. We can adopt any smoothing function $h_\mu(x)$ for $h(x)$ that satisfies Definition 1 below to obtain the smooth problem

$$\min_{x \in X} \left\{ \tilde{\psi}_\mu(x) := f(x) + \tilde{h}_\mu(x) \right\}. \quad (4)$$

We adopt the updating rule of the smoothing parameter similar as that in [4] to fasten the computational speed.

The AdaSMSAG method proposed in this paper is suitable for the stochastic setting. We show the convergence of the AdaSMSAG method by

$$E \left[ \psi(y_N) - \psi^* \right] \leq O\left(\frac{\ln(N + c)}{N}\right),$$

where $c > 0$ is a constant, and $N \geq 1$ is the maximum iteration number. Because $\ln(N + c) < \sqrt{N + c}$, the order of the worst-case iteration complexity of the AdaSMSAG method is better than $O\left(\frac{1}{\epsilon^2}\right)$ that is considered to be the optimal order of the worst-case iteration complexity as in [17, 22]. Similar as for the deterministic setting [1], the optimal order $O\left(\frac{1}{\epsilon^2}\right)$ of the worst-case iteration complexity is valid only for the black-box oracle model of the objective function. In practice, we always know something about the structure of the underlying objective functions. The proper use of the structure of the problem by smoothing approximation adopted in this paper does help in improving the efficiency to find an $\epsilon$-approximate solution.
At the $k$-th iterate, although $f$ and $\tilde{h}_{\mu_k}(x)$ are Lipschitz continuously differentiable, we assume that only the noisy gradients of $f$ and $\tilde{h}_{\mu_k}(x)$ are available via subsequent calls to a stochastic first-order oracle ($\mathcal{SFO}$). That is, when we solve the smooth problem (4) by an iterative method, at the $k$-th iterate, $k \geq 1$, for the input $x_k \in X$, the $\mathcal{SFO}$ would output a stochastic gradient $\nabla \tilde{\psi}_{\mu_k}(x_k, \xi_k)$, where $\xi_k$ is a random vector whose probability distribution is supported on $\Xi \subseteq \mathbb{R}^d$. Throughout the paper, we make the following assumptions for the functions $\tilde{\psi}_{\mu_k}(\cdot, \xi_k)$.

**Assumption 1** For any fixed $\mu_k > 0$ and $x_k \in X$, we have

\begin{align*}
  a) \quad & E \left[ \nabla \tilde{\psi}_{\mu_k}(x_k, \xi_k) \right] = \nabla \tilde{\psi}_{\mu_k}(x_k), \\
  b) \quad & E \left[ \|\nabla \tilde{\psi}_{\mu_k}(x_k, \xi_k) - \nabla \tilde{\psi}_{\mu_k}(x_k)\|^2 \right] \leq \sigma^2,
\end{align*}

where $\sigma > 0$ is a constant, and the expectation $E$ is taken with respect to the random vector $\xi_k \in \Xi$.

The remaining of this paper is organized as follows. In Section 2, we briefly review some basic concepts and results relating to the smoothing function [23] that will be used in our paper. In Section 3, we develop an adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method. We show the convergence, as well as the order of the worst-case iteration complexity of the proposed method that is better than the state-of-the-art stochastic approximation methods for stochastic nonsmooth convex optimization. Numerical experiments on a risk management in portfolio optimization and a family of Wasserstein distributionally robust support vector machine (DRSVM) problems with real data are given in Section 4 to demonstrate the efficiency of our proposed method.

Throughout the paper, we use the following notation. We denote by $\mathbb{Z}_+$ the positive integer set, by $\|x\|$ the Euclidean norm of a vector $x$ in the Euclidean space $\mathbb{R}^n$. Given $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ and $y = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$, the inner product of $x$ and $y$ is defined by $\langle x, y \rangle := \sum_{i=1}^{n} x_i y_i$. The notation $[r]$ stands for the smallest integer greater than or equal to $r \in \mathbb{R}$. For a number $r \in \mathbb{R}$ we denote $[r]_+ := \max\{r, 0\}$.

### 2 Preliminaries

In this section, we review some basic concepts and results that will be used later. We consider a class of smoothing functions with the following definition.

**Definition 1** We call $\tilde{\phi}_\mu$ with $\mu \in (0, \bar{\mu}]$ a smoothing function of the convex function $\phi$ in the feasible region, if $\tilde{\phi}_\mu(\cdot)$ is continuously differentiable in $X$ for any fixed $\mu > 0$ and satisfies the following conditions:

(i) $\lim_{z \to x, \mu \downarrow 0} \tilde{\phi}_\mu(z) = \phi(x), \quad \forall x \in X$;
Adaptive smoothing mini-batch stochastic accelerated gradient method

(i) (convexity) \( \tilde{\phi}_\mu(x) \) is convex with respect to \( x \) in \( X \) for any fixed \( \mu > 0 \);

(ii) (Lipschitz continuity with respect to \( \mu \)) there exists a positive constant \( \kappa \) such that
\[
|\tilde{\phi}_{\mu_2}(x) - \tilde{\phi}_{\mu_1}(x)| \leq \kappa |\mu_1 - \mu_2|, \quad \forall x \in X, \ \mu_1, \mu_2 \in (0, \bar{\mu}];
\]  

(5)

(iii) (Lipschitz continuity with respect to \( x \)) there exists a positive constant \( \kappa \) such that
\[
|\tilde{\phi}_\mu(x) - \phi(x)| \leq \kappa \mu, \quad \forall x \in X, \ \mu \in (0, \bar{\mu}].
\]  

(6)

(iv) (Lipschitz continuity with respect to \( x \)) there exists a constant \( L > 0 \) such that
\[
\nabla \tilde{\phi}_\mu(\cdot) \text{ is Lipschitz continuous on } X \text{ with Lipschitz constant } L \mu.
\]

Note that the smoothing function in Definition 3.1 of [23] requires all the above conditions in Definition 1, besides another gradient consistency property. Definition 1 (iii) implies
\[
|\tilde{\phi}_\mu(x) - \phi(x)| \leq \kappa \mu, \quad \forall x \in X, \ \mu \in (0, \bar{\mu}].
\]  

(6)

Many smoothing functions satisfy Definition 1.

Example 1 Chen and Mangasarian construct a class of smooth approximations of the function \((x^+)\) by convolution [24] as follows. Let \( \rho : \mathbb{R} \rightarrow \mathbb{R}_+ \) be a piecewise continuous density function satisfying \( \rho(s) = \rho(-s) \). It is easy to see that
\[
\tilde{\phi}_\mu(t) := \int_{-\infty}^{\infty} (t - \mu s)^+ \rho(s) ds
\]

is a smoothing function of \((x^+)\). According to the inequality (22) of [7], we have
\[
\kappa = \int_{-\infty}^{\infty} s \rho(s) ds < \infty
\]
in Definition 1 (iv). By choosing different density functions, we can obtain different smoothing function, e.g., the uniform smoothing function, the Neural Networks smoothing function, and the CHKS (Chen-Harker-Kanzow-Smale) smoothing function; see [7] for details.

Using Definition 1, we then find the objective function \( \tilde{\psi}_{\mu_k}(x) \) for the smooth problem in (4) with \( \mu = \mu_k \) is \( L_k \)-smooth on \( X \), i.e., \( \psi_{\mu_k}(x) \) is Lipschitz continuously differentiable with Lipschitz constant \( L_k = L_f + L_{\tilde{h}_{\mu_k}} \) on \( X \), where \( L_f \) is the Lipschitz constant of \( f \). According to Theorem 5.12 of [25], we get \( L_f \geq \|\nabla^2 f(x)\| = \lambda_{\max}(\nabla^2 f(x)) \) for any \( x \in X \), where \( \lambda_{\max}(\nabla^2 f(x)) \) means the maximal eigenvalue of the Hessian matrix of \( f(x) \). Thus
\[
\tilde{\psi}_{\mu_k}(y) \leq \tilde{\psi}_{\mu_k}(x) + \langle \nabla \tilde{\psi}_{\mu_k}(x), y - x \rangle + \frac{L_k}{2} \|y - x\|^2.
\]  

(7)

Given a vector \( g \in \mathbb{R}^n \), we define the generalized projected gradient of \( \psi \) at \( x \in X \) as
\[
P_X(x, g, \gamma) = \frac{1}{\gamma} (x - x'),
\]  

(8)
where $x'$ is given by

$$x' = \arg \min_{y \in X} \left\{ \langle g, y \rangle + \frac{1}{2\gamma} \| y - x \|^2 \right\}. \quad (9)$$

### 3 Adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method

In this section, we develop an adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method for (1). We will show the convergence as well as the order of worst-case iteration complexity, as well as the mini-batch sizes of the AdaSMSAG method.

At the $k$-th iterate of the AdaSMSAG method, we randomly choose a mini-batch samples $\xi^{[m_k]} = \{\xi_{k,1}, \ldots, \xi_{k,m_k}\}$ of the random vector $\xi \in \Xi$, where $m_k$ is the batch size. We denote by $\xi^{[k]} = \{\xi^{[m_1]}, \ldots, \xi^{[m_k]}\}$ the history of mini-batch samples from the first iterate up to the $k$-th iterate.

For any $k \geq 1$, we denote the mini-batch stochastic gradient $\nabla \tilde{\psi}^{k}_{\mu_k}$ by

$$\nabla \tilde{\psi}^{k}_{\mu_k} = \frac{1}{m_k} \sum_{j=1}^{m_k} \nabla \tilde{\psi}^{k}_{\mu_k}(x_k, \xi_{k,j}). \quad (10)$$

For ease of notation, we denote

$$\delta^{i}_{\mu_k}(x) := \nabla \tilde{\psi}^{k}_{\mu_k} - \tilde{\psi}^{k}_{\mu_k}(x), \quad \delta^{i}_{\mu_k}(x_k) := \nabla \tilde{\psi}^{k}_{\mu_k}(x_k, \xi_{k,i}) - \tilde{\psi}^{k}_{\mu_k}(x_k), \quad i = 1, \ldots, m_k, \quad (11)$$

$$S^{j}_{\mu_k}(x) := \sum_{i=1}^{j} \delta^{i}_{\mu_k}(x), \quad j = 1, \ldots, m_k, \quad (12)$$

where $\delta^{i}_{\mu_k}(x_k) = \frac{1}{m_k} \sum_{i=1}^{m_k} \delta^{i}_{\mu_k}(x_k)$.

The following lemma addresses the relations of $\tilde{\psi}^{k}_{\mu_k}$ and $\nabla \tilde{\psi}^{k}_{\mu_k}$ to that of $\tilde{\psi}^{k}_{\mu_k}(x_k)$ and $\nabla \tilde{\psi}^{k}_{\mu_k}(x_k)$, which can be shown without difficulty by using the arguments similar as in [26].

**Lemma 1** Under Assumption 1, we have for any $k \geq 1$ and $\mu_k > 0$,

(a) $E \left[ \nabla \tilde{\psi}^{k}_{\mu_k} \right] = E \left[ \nabla \tilde{\psi}^{k}_{\mu_k}(x_k) \right]$,  

(b) $E \left[ \left\| \nabla \tilde{\psi}^{k}_{\mu_k} - \tilde{\psi}^{k}_{\mu_k}(x_k) \right\|^2 \right] \leq \frac{\sigma^2}{m_k}$,

where the expectation is taken with respect to the history of mini-batch samples $\xi^{[k]}$.

Now we give the AdaSMSAG method as follows.
Algorithm 1 Adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method

Require: the iteration limit \( N > 0 \), the batch sizes \( \{m_k\}_{k \geq 1} \) with \( m_k \geq 1 \), \( \{\alpha_k\}_{k \geq 1} \) with \( 0 < \alpha_k < 1 \), \( c > 0 \), \( \mu_0 > 0 \), \( \{\beta_k\}_{k \geq 1} \) with \( \beta_k > 0 \), and \( \{\theta_k\}_{k \geq 1} \) with \( \theta_k > 0 \), \( z_0 = y_0 \).

1: for \( k = 1, \ldots, N \) do
2: \( x_k = \alpha_k z_{k-1} + (1 - \alpha_k)y_{k-1} \)
3: \( \mu_k = \frac{\mu_0}{c + r} \)
4: Call the \( \mathcal{SFOM} \) \( m_k \) times to obtain \( \nabla \tilde{\psi}_{\mu_k}^k(x_k, \xi_{k,i}), i = 1, \ldots, m_k \).
5: Set \( \nabla \tilde{\psi}^k_{\mu_k} = \frac{1}{m_k} \sum_{i=1}^{m_k} \nabla \tilde{\psi}_{\mu_k}^k(x_k, \xi_{k,i}) \), and compute
6: \( y_k = \arg\min_{y \in X} \{ \nabla \tilde{\psi}^k_{\mu_k}, y - x_k \} + \frac{\beta_k}{2} \| y - x_k \|^2 \}
7: \( z_k = \arg\min_{x \in X} \{ \nabla \tilde{\psi}^k_{\mu_k}, x - x_k \} + \frac{\theta_k}{2} \| x - z_{k-1} \|^2 \}
8: end for
9: output: \( y_N \).

We have the following simple observations in Lemmas 2 and 3.

Lemma 2 If \( \mu_k \) is monotonically decreasing with \( k \), we have for any \( k \geq 1 \) and \( x, y_k \in X \),

\[
\psi(y_k) - \psi(x) \leq \tilde{\psi}_{\mu_k}(y_k) - \tilde{\psi}_{\mu_k}(x) + 2\kappa \mu_k, \quad (14) \\
\tilde{\psi}_{\mu_{k+1}}(y_k) - \tilde{\psi}_{\mu_{k+1}}(x) \leq \tilde{\psi}_{\mu_k}(y_k) - \tilde{\psi}_{\mu_k}(x) + 2\kappa (\mu_k - \mu_{k+1}). \quad (15)
\]

Proof For any fixed \( x, y_k \in X \) and \( \mu_k > 0 \), we have

\[
\psi(y_k) - \psi(x) \overset{(1)}{=} [f(y_k) + h(y_k)] - [f(x) + h(x)] \\
\overset{(6)}{=} [f(y_k) + \bar{h}_{\mu_k}(y_k)] - [f(x) + \bar{h}_{\mu_k}(x)] + 2\kappa \mu_k \\
\overset{(4)}{=} \tilde{\psi}_{\mu_k}(y_k) - \tilde{\psi}_{\mu_k}(x) + 2\kappa \mu_k.
\]

For the inequality (15), in view of \( \mu_{k+1} \leq \mu_k \) we find that

\[
\tilde{\psi}_{\mu_{k+1}}(y_k) - \tilde{\psi}_{\mu_{k+1}}(x) \overset{(4)}{=} f(y_k) + \bar{h}_{\mu_{k+1}}(y_k) - [f(x) + \bar{h}_{\mu_{k+1}}(x)] \\
\overset{(5)}{=} f(y_k) + \bar{h}_{\mu_k}(y_k) + \kappa (\mu_k - \mu_{k+1}) \\
- f(x) + \bar{h}_{\mu_k}(x) - \kappa (\mu_k - \mu_{k+1}) \\
\overset{(4)}{=} \tilde{\psi}_{\mu_k}(y_k) - \tilde{\psi}_{\mu_k}(x) + 2\kappa (\mu_k - \mu_{k+1}).
\]

\[\square\]
**Lemma 3** Let $y_k$ and $z_k$ be obtained by Algorithm 1. By the generalized projected gradient in (8)-(9), we have

$$x_k - y_k = \frac{P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k)}{\beta_k},$$  

$$z_{k-1} - z_k = \frac{P_X(z_{k-1}, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma z_k)}{\theta_k},$$  

$$\langle \nabla \tilde{\Psi}_{\mu_k}^k, P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k) \rangle \geq \|P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k)\|^2.$$

**Proof** In view of the definitions of $y_k$, $z_k$ and the generalized projected gradient, we have

$$P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k) = \beta_k(x_k - y_k)$$

$$P_X(z_{k-1}, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma z_k) = \theta_k(z_{k-1} - z_k).$$

Letting $g = \nabla \tilde{\Psi}_{\mu_k}^k$, $V(y, x) = 1/2\|y - x\|^2$ with convexity parameter $\alpha = 1$, and $h(x) = 0$ in (2.3) of [20], we have by Lemma 1 of [20] that

$$\langle \nabla \tilde{\Psi}_{\mu_k}^k, P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k) \rangle \geq \|P_X(x_k, \nabla \tilde{\Psi}_{\mu_k}^k, \gamma y_k)\|^2.$$

\[ \Box \]

Below we develop Lemmas 5 and 6 which are important for developing the convergence results.

**Lemma 4** (Lemma 2 of [20]) Let the convex function $p : X \to \mathbb{R}$, the points $\bar{x}, \bar{y} \in X$ and the scalars $\mu_1, \mu_2 \geq 0$ be given. Let $\omega : X \to \mathbb{R}$ be a differentiable convex function and $V(x, z)$ be the Bregman’s distance. If

$$u^* \in \arg \min_{u \in X} \{p(u) + \mu_1 V(\bar{x}, u) + \mu_2 V(\bar{y}, u)\},$$

then for any $u \in X$, we have

$$p(u^*) + \mu_1 V(\bar{x}, u^*) + \mu_2 V(\bar{y}, u^*) \leq p(u) + \mu_1 V(\bar{x}, u) + \mu_2 V(\bar{y}, u) - (\mu_1 + \mu_2) V(u^*, u).$$

**Lemma 5** Let $x^*$ be an arbitrary optimal solution of (1). Denote $D_{k-1}^2 = \|x^* - x_{k-1}\|^2$. Assume that

$$\alpha_k \beta_k - \theta_k \leq 0.$$  

We have for any $k \geq 1$,

$$\langle \nabla \tilde{\Psi}_{\mu_k}^k, y_k - \alpha_k x^* - (1 - \alpha_k)y_{k-1} \rangle \leq -\beta_k/2 \|y_k - x_k\|^2 + \alpha_k \theta_k \left(D_{k-1}^2 - D_k^2\right).$$

**Proof** Letting $p(u) = \langle \nabla \tilde{\Psi}_{\mu_k}^k, u - x_k \rangle$, $u = x^*$, $\bar{x} = z_{k-1}$, $u^* = z_k$, $\mu_1 = \theta_k$, and $\mu_2 = 0$. In Lemma 4, we have by the definition of $z_k$ that

$$\langle \nabla \tilde{\Psi}_{\mu_k}^k, z_k - x^* \rangle \leq \theta_k \left(D_{k-1}^2 - D_k^2\right) - \theta_k \|z_k - z_{k-1}\|^2/2.$$
10 Adaptive smoothing mini-batch stochastic accelerated gradient method

Similarly, letting \( p(u) = \langle \nabla \tilde{\Psi}^k_{\mu_k}, u - x_k \rangle \), \( u = x \in X, \tilde{x} = x_k, u^* = y_k, \mu_1 = \beta_k \), and \( \mu_2 = 0 \) in Lemma 4, we have by the definition of \( y_k \) that

\[
\langle \nabla \tilde{\Psi}^k_{\mu_k}, y_k - x \rangle \leq \frac{\beta_k}{2} \left( \|x_k - x\|^2 - \|y_k - x\|^2 - \|y_k - x_k\|^2 \right).
\]

By choosing \( x = \alpha_k z_k + (1 - \alpha_k)y_{k-1} \in X \) in the above inequality, we obtain

\[
\langle \nabla \tilde{\Psi}^k_{\mu_k}, y_k - \alpha_k z_k - (1 - \alpha_k)y_{k-1} \rangle \leq \frac{\beta_k}{2} \left( \|x_k - \alpha_k z_k - (1 - \alpha_k)y_{k-1}\|^2 - \|y_k - x\|^2 \right) = \frac{\beta_k}{2} \left( \alpha_k \|z_{k-1} - z_k\|^2 - \|y_{k-1} - z_{k-1}\|^2 - \|y_k - x\|^2 \right),
\]

where the last equality follows from the definition of \( x_k \) in Algorithm 1. Multiplying (21) by \( \alpha_k \) and then summing it to the above inequality, we have

\[
\langle \nabla \tilde{\Psi}^k_{\mu_k}, y_k - \alpha_k x^* - (1 - \alpha_k)y_{k-1} \rangle \leq \frac{\alpha_k (\alpha_k \beta_k - \theta_k)}{2} \|z_{k-1} - z_k\|^2 - \frac{\beta_k}{2} \|y_k - x\|^2 + \alpha_k \theta_k \left( D_{k-1}^2 - D_k^2 \right) \leq \frac{\beta_k}{2} \|y_k - x\|^2 + \alpha_k \theta_k \left( D_{k-1}^2 - D_k^2 \right),
\]

since \( \alpha_k \beta_k - \theta_k \leq 0 \) according to (19). \( \square \)

For ease of notation, we define

\[
\Delta_{\mu_{k-1}} = \tilde{\psi}_{\mu_{k-1}}(y_{k-1}) - \tilde{\psi}_{\mu_{k-1}}(x^*), \quad (22)
\]

\[
\Gamma_k = \Delta_{\mu_k} - (1 - \alpha_k) \Delta_{\mu_{k-1}} - 2\kappa (1 - \alpha_k) (\mu_{k-1} - \mu_k). \quad (23)
\]

Lemma 6 Let \( x^* \) be an arbitrary optimal solution of (1). Assume that

\[
\alpha_k \beta_k - \theta_k \leq 0 \quad \text{and} \quad \beta_k - L_k > 0. \quad (24)
\]

For any \( k \geq 1 \) we have

\[
\Gamma_k \leq \alpha_k \theta_k \left( D_{k-1}^2 - D_k^2 \right) + \frac{\|\delta_{\mu_k}(x_k)\|^2}{2(\beta_k - L_k)} + \alpha_k (\delta_{\mu_k}(x_k), x^* - z_{k-1}). \quad (25)
\]

Proof Based on (15), we have

\[
(1 - \alpha_k) \tilde{\psi}_{\mu_k}(y_{k-1}) - \tilde{\psi}_{\mu_k}(x^*) = (1 - \alpha_k) \left[ \tilde{\psi}_{\mu_k}(y_{k-1}) - \tilde{\psi}_{\mu_k}(x^*) \right] - \alpha_k \tilde{\psi}_{\mu_k}(x^*) \leq (1 - \alpha_k) \left[ \tilde{\psi}_{\mu_{k-1}}(y_{k-1}) - \tilde{\psi}_{\mu_{k-1}}(x^*) \right] + 2\kappa (1 - \alpha_k) (\mu_{k-1} - \mu_k) - \alpha_k \tilde{\psi}_{\mu_k}(x^*) \leq (1 - \alpha_k) \Delta_{\mu_{k-1}} + 2\kappa (1 - \alpha_k) (\mu_{k-1} - \mu_k) - \alpha_k \tilde{\psi}_{\mu_k}(x^*). \quad (22)
\]

By the convexity of \( \tilde{\psi}_{\mu_k} \), we have

\[
\tilde{\psi}_{\mu_k}(x_k) \leq \tilde{\psi}_{\mu_k}(x) - \langle \nabla \tilde{\psi}_{\mu_k}(x_k), x - x_k \rangle, \forall x \in X. \quad (27)
\]
Adaptive smoothing mini-batch stochastic accelerated gradient method

In view of $\tilde{\psi}_{\mu_k}$ is an $L_k$-smooth function over a given convex set $X$, we have

$$
\tilde{\psi}_{\mu_k}(y_k) \leq \tilde{\psi}_{\mu_k}(x_k) + \langle \nabla \tilde{\psi}_{\mu_k}(x_k), y_k - x_k \rangle + \frac{L_k}{2} \|y_k - x_k\|^2
$$

(7)

$$
= (1 - \alpha_k)\tilde{\psi}_{\mu_k}(x_k) + \alpha_k\tilde{\psi}_{\mu_k}(x_k) + \langle \nabla \tilde{\psi}_{\mu_k}(x_k), y_k - x_k \rangle + \frac{L_k}{2} \|y_k - x_k\|^2
$$

(27)

Subtracting $\tilde{\psi}_{\mu_k}(x^*) + (1 - \alpha_k)\Delta_{\mu_{k-1}} + 2\kappa(1 - \alpha_k)(\mu_{k-1} - \mu_k)$ from both sides of the above inequality, and according to (26), we get

$$
\Gamma_k \leq (1 - \alpha_k)\tilde{\psi}_{\mu_k}(y_{k-1}) - (1 - \alpha_k)\tilde{\psi}_{\mu_k}(x^*) - (1 - \alpha_k)\Delta_{\mu_{k-1}} - 2\kappa(1 - \alpha_k)(\mu_{k-1} - \mu_k) - (1 - \alpha_k)\langle \nabla \tilde{\psi}_{\mu_k}(x_k), y_{k-1} - x_k \rangle + \alpha_k\tilde{\psi}_{\mu_k}(x^*) - \alpha_k\langle \nabla \tilde{\psi}_{\mu_k}(x_k), x^* - x_k \rangle + \langle \nabla \tilde{\psi}_{\mu_k}(x_k), y_k - x_k \rangle + \frac{L_k}{2} \|y_k - x_k\|^2
$$

(26)

$$
\leq \langle \nabla \tilde{\psi}_{\mu_k}(x_k), y_k - \alpha_k x^* - (1 - \alpha_k) y_{k-1} \rangle + \frac{L_k}{2} \|y_k - x_k\|^2
$$

(11)

$$
\leq \langle \nabla \tilde{\psi}_{\mu_k}^k - \delta_{\mu_k}(x_k), y_k - \alpha_k x^* - (1 - \alpha_k) y_{k-1} \rangle + \frac{L_k}{2} \|y_k - x_k\|^2
$$

(20)

$$
\leq \alpha_k\theta_k \left( D_{k-1}^2 - D_k^2 \right) - \frac{L_k}{2} \|y_k - x_k\|^2 + \langle \delta_{\mu_k}(x_k), x_k - y_k \rangle - \langle \delta_{\mu_k}(x_k), x^* - (1 - \alpha_k) y_{k-1} \rangle
$$

$$
\leq \alpha_k\theta_k \left( D_{k-1}^2 - D_k^2 \right) + \frac{\|\delta_{\mu_k}(x_k)\|^2}{2(\beta_k - L_k)} + \alpha_k\langle \delta_{\mu_k}(x_k), x^* - z_{k-1} \rangle.
$$

The last inequality follows from the fact that $at - \frac{bt^2}{2} \leq \frac{a^2}{2b}$ for any $a, t, b > 0$, and we choose special $a = \|\delta_{\mu_k}(x_k)\|$, $t = \|x_k - y_k\|$, $b = \beta_k - L_k > 0$ here. □

Proposition 1 Under Assumption 1, we have for any $k \geq 1$

$$
E[\Gamma_k] \leq \alpha_k\theta_k E\left[ D_{k-1}^2 - D_k^2 \right] + \frac{\sigma^2}{2m_k(\beta_k - L_k)}, \tag{28}
$$

where $\Gamma_k$ is defined in (23) and the expectation $E$ is taken with respect to $\xi[k]$.

Proof Based on Lemma 1 (a), we have for any $k \geq 1$

$$
E \left[ \langle \delta_{\mu_k}(x_k), x^* - z_{k-1} \rangle \right]
$$

$$
= E \left\{ E \left[ \langle \delta_{\mu_k}(x_k), x^* - z_{k-1} \rangle | \xi[k-1] \right] \right\}
$$

$$
= E \left\{ E \left[ \langle \delta_{\mu_k}(x_k)|\xi[k-1] \rangle, x^* - z_{k-1} \right] \right\}
$$

(11) 

$$
= E \left\{ E \left[ \nabla \tilde{\psi}_{\mu_k}^k|\xi[k-1]\right] - \nabla \tilde{\psi}_{\mu_k}(x_k), x^* - z_{k-1} \right\}
$$
Adaptive smoothing mini-batch stochastic accelerated gradient method

\[ = E \left\{ \left( \nabla \tilde{\psi}_{\mu_k} (x_k) - \nabla \tilde{\psi}_{\mu_k} (x_k), x^* - z_{k-1} \right) \right\} \]

\[ = 0, \quad (29) \]

where the first equality is based on Theorem 3.24 of [27] and the expectation \( E \) is taken with respect to \( \xi_{[k]} \). Taking the expectation on both sides of (25) with respect to \( \xi_{[k]} \), and using the observations (29), and Lemma 1 (b), we get (28) as we desired. \( \square \)

We can choose special sequences \( \alpha_k, \beta_k \) and \( \theta_k \) such that the requirements (24) in Lemma 6 are satisfied as in the following theorem.

**Theorem 2** We choose in the AdaSMSAG method (Algorithm 1) the parameters

\[ \alpha_k = \frac{1}{k}, \quad \mu_k = \frac{\alpha_0}{k + c}, \quad c \in \mathbb{Z}_+, \quad \mu_0 = \frac{cL}{L_f} > 0, \quad \beta_k = 2L_k, \quad \theta_k \equiv \theta = 6L_f, \]

and the batch sizes \( m_k \equiv m \geq 1 \) for \( k = 1, \ldots, N \). Under Assumption 1, we have for any \( k \geq 1 \)

\[ E [\Delta_{\mu_k}] \leq \frac{6L_f D_0^2}{k} + \frac{2ck^2L}{L_f k} \ln (k + c) + \frac{\sigma^2 c^2}{2mL_f}. \quad (30) \]

**Proof** In view of \( \mu_k = \frac{\alpha_0}{k + c}, c \in \mathbb{Z}_+, \mu_0 = \frac{cL}{L_f} > 0 \), we find for any \( k \geq 1 \),

\[ \alpha_k \beta_k = \frac{2L_k}{k} = \frac{2}{k} \left( \frac{\mu_k}{L_f} + \frac{L}{\mu_k} \right) = 2L \left( \frac{1}{k} + \frac{1}{c^2} + \frac{1}{ck} \right) \leq 6L_f = \theta_k, \]

and

\[ \beta_k - L_k = L_k > 0. \]

Thus the assumptions in (24) hold. Let us denote for simplicity

\[ P_k := 2\kappa (1 - \alpha_k) (\mu_{k-1} - \mu_k), \quad \text{and} \quad Q_k := \frac{\sigma^2}{2m(\beta_k - L_k)}. \quad (31) \]

By (28) and (31), we have for any \( k \geq 1 \)

\[ E [\Delta_{\mu_k}] + \alpha_k \theta E \left[ D_{k-1}^2 \right] \leq (1 - \alpha_k) E [\Delta_{\mu_{k-1}}] + \alpha_k \theta E \left[ D_{k-1}^2 \right] + P_k + Q_k. \]

Multiplying both sides of this inequality by \( \frac{1}{\alpha_k \theta} \), we obtain

\[ \frac{1}{\alpha_k \theta} E [\Delta_{\mu_k}] + E \left[ D_k^2 \right] \leq \frac{1 - \alpha_k}{\alpha_k \theta} E [\Delta_{\mu_{k-1}}] + E \left[ D_{k-1}^2 \right] + \frac{P_k + Q_k}{\alpha_k \theta}. \quad (32) \]

Noticing that \( \alpha_k = \frac{1}{k} \) for \( k = 1, \ldots, N \), we have for any \( k \geq 2 \),

\[ \frac{1}{\alpha_k} = \frac{1}{\alpha_{k-1}}. \]

According to the above equality and (32), we have

\[ \frac{1}{\alpha_k \theta} E [\Delta_{\mu_k}] + E \left[ D_k^2 \right] \leq \frac{1}{\alpha_{k-1} \theta} E [\Delta_{\mu_{k-1}}] + E \left[ D_{k-1}^2 \right] + \frac{P_k + Q_k}{\alpha_k \theta} \leq \ldots \]
Based on the definitions of $P_k$, $Q_k$, and the elementary inequalities $\sum_{i=1}^{k+c} \frac{1}{i} < 1 + \ln(k + c)$ and $\sum_{i=1}^{k+c} \frac{1}{i} \geq \frac{3}{2}$, we get

$$\sum_{i=1}^{k} \frac{P_i}{\alpha_i \theta} = \frac{2\kappa \mu_0}{\theta} \sum_{i=1}^{k} (i-1) \left( \frac{1}{i+c} - \frac{1}{i+c+1} \right)$$

$$\leq \frac{2\kappa \mu_0}{\theta} \sum_{i=2}^{k} \frac{1}{i+c}$$

$$\leq 2\kappa c^2 L \theta L_f \ln(k + c), \tag{34}$$

and

$$\sum_{i=1}^{k} \frac{Q_i}{\alpha_i \theta} = \frac{\sigma^2}{2m} \sum_{i=1}^{k} \frac{1}{L_i + \mu_i}$$

$$\leq \frac{\sigma^2 c^2 k}{2m \theta L_f}. \tag{35}$$

With the above observations (34), and (35), and multiplying (33) by $\alpha_k \theta = \theta = \frac{6 L_f}{k}$, we obtain

$$E[\Delta_{\mu_k}] \leq \frac{6 L_f D_0^2}{k} + 2 \kappa c^2 L \ln(k + c) + \frac{\sigma^2 c^2}{2m L_f}.$$  

□

**Corollary 1** Let the parameters $\alpha_k$, $\beta_k$, and $\theta_k$ for $k \geq 1$ in the AdaSMSAG method be set as in Theorem 2. Let the number of calls to the $\mathcal{SFO}$ at each iteration of the AdaSMSAG method is

$$m = \left\lceil \frac{\sigma^2 N}{4 \kappa L \ln(N + c)} \right\rceil. \tag{36}$$

Under Assumption 1, we have for any $N \geq 1$,

$$E[\psi(y_N) - \psi^*] \leq \frac{6 L_f^2 D_0^2}{L_f N} + 2 \kappa c^2 L \ln(N + c). \tag{37}$$
Adaptive smoothing mini-batch stochastic accelerated gradient method

Proof In view of (14), (22) and (30), we have
\[
E \left[ \psi(y_N) - \psi^* \right] \leq E \left[ \tilde{\psi}_{\mu_N}(y_N) - \tilde{\psi}_{\mu_N}(x^*) \right] + 2\kappa \mu_N
\]
\[
(22) \quad E |\Delta_{\mu_N}| + 2\kappa \mu_N \leq \frac{6L_f D_0^2}{N} + 2\kappa c^2 L \ln (N + c) + \frac{\sigma^2 c^2}{2mL_f} + \frac{2\kappa c^2 L}{L_f (N + c)}
\]
\[
(30) \quad \leq \frac{6L_f D_0^2}{L_f N} + 2\kappa c^2 L \ln (N + c) + \frac{\sigma^2 c^2}{2mL_f}.
\]
By setting in (38)
\[
m = \left\lceil \frac{\sigma^2 N}{4\kappa L \ln (N + c)} \right\rceil,
\]
we obtain (37) as we desired. \hfill \Box

Remark 1 Since \(\ln(N + c) < \sqrt{N + c}\), we get
\[
\frac{\ln(N + c)}{N} < \frac{\sqrt{N + c}}{N} = \frac{1}{\sqrt{N + c}} N + c.
\]
From (37), in order to get an \(\epsilon\)-approximate solution, we need
\[
E[\psi(y_N) - \psi^*] \leq O\left(\frac{\ln(N + c)}{N}\right) \leq \epsilon.
\]
Thus the order of worst-case iteration complexity for finding an \(\epsilon\)-approximate solution is better than \(O\left(\frac{1}{\epsilon^2}\right)\) obtained by the state-of-the-art stochastic approximation methods [17, 19–21].

4 Numerical results

In this section, we do several numerical experiments to illustrate the efficiency of the AdaSMSAG method by comparing it with several state-of-the-art stochastic approximation methods. We consider a risk management portfolio optimization, and a family of Wasserstein distributionally robust support vector machine (DRSVM) problems using real data.

We compare our AdaSMSAG method with the following stochastic approximation methods.
- The mini-batch stochastic Nesterov’s smoothing (MSNS) method [28].
- The mini-batch mirror descent stochastic approximation (M-MDSA) method [17].
- The randomized stochastic projected gradient (RSPG) method [26].
- The two-phase RSPG (2-RSPG) method [26].
- The two-phase RSPG variant (2-RSPG-V) method [26].

The MSNS method, the M-MDSA method, the RSPG method, the 2-RSPG method, and the 2-RSPG-V method are state-of-the-art stochastic approximation methods. The MSNS method can solve the first application, i.e., the risk
management on portfolio optimization since its nonsmooth component has an explicit max structure. The other four methods can not solve the nonsmooth model defined in (1) with guaranteed convergence, because of the relatively complex nonsmooth term that leads the difficulty of obtaining its proximal operator. For comparison, we apply these four methods to solve the smooth counterpart in (4) with fixed smoothing parameter $\mu_k \equiv 0.001$. Under this value, the solution obtained by solving the smooth counterpart (4) is closer to the optimal solution of the original problem (1). It’s worth noting that the 2-RSPG method and the 2-RSPG-V method include two phases, i.e., the optimization phase and the post-optimization phase. In either the 2-RSPG method, or the 2-RSPG-V method, it generates several candidate outputs in the optimization phase, and the final output is selected from these candidate outputs according to some rules in the post-optimization phase. Since the MDSA method calls one random vector per iteration (the batch size is equal to 1), the total running time is relatively long. We therefore use the mini-batch MDSA method (M-MDSA) to replace the MDSA method in comparison as done in [26]. The batch sizes of the M-MDSA method are set to be the same as that for our AdaSMSAG method.

We denote by $N_{total}$ the total number of calls to the stochastic first-order oracle. For each problem, $N_{total}$ are set to be the same for different methods. Since the RSPG, 2-RSPG and 2-RSPG-V methods are randomized SA methods, they stop randomly before up to the maximum number of calls. We then set $N_{total}$ for these three methods to be the maximum number of calls to the stochastic oracle performed in the optimization phase.

Denote by $NS$ the number of the training samples. We use the $NS$ training samples to estimate the parameter, $\sigma^2$. We follow the way of estimating the parameter $\sigma^2$ as in [26]. Using the training samples, we compute the stochastic gradients of the objective function $\lceil NS/100 \rceil$ times at 100 randomly selected points and then take the average of the variances of the stochastic gradients for each point as an estimation of $\sigma^2$.

Our experiments performed in MATLAB R2018b on a laptop with Windows 10, 1 dualcore 2.4 GHz CPU and 8 GB of RAM. All reported CPU times are in seconds.

4.1 Risk management in portfolio optimization

The first application is risk management on portfolio optimization [29]. Suppose that there are $B$ assets and their random return rates are denoted as $R = (R_1, R_2, \ldots, R_B)^T$. Assume that for each asset $b = 1, \ldots, B$, there are many realizations for its random return rate, with each denoted as $r_b$. Denote the random vector $r = (r_1, r_2, \ldots, r_B)^T$. Let

$$Z = \left\{ z \in \mathbb{R}^B : \sum_{b=1}^{B} z_b = 1, z \geq 0 \right\}$$
be the set of all feasible portfolios, where \( z_b \) is the proportion of wealth allocated to asset \( b \).

In risk-aware portfolio optimization, the decision maker optimizes an expectation risk measure objective. Conditional Value-at-Risk (CVaR) is one of the most widely investigated risk measures (see [30, 31]). Mathematically, the \( \alpha \) level CVaR objective of portfolio optimization is [29].

\[
\min_{z \in \mathcal{Z}, \eta \in \mathbb{R}} E_r \left[ \eta + \frac{1}{1 - \alpha} \left( - \sum_{b=1}^{B} r_b z_b - \eta \right) \right]. 
\] (39)

The objective function now is convex but nonsmooth function. In particular, the problem will be large scale when the number \( N \) of realizations is large. We can then apply our method to tackle this large scale risk-aware portfolio optimization problem. According to [7], we have the smoothing problem of (39)

\[
\min_{z \in \mathcal{Z}, \eta \in \mathbb{R}} E_r \left[ \eta + \frac{\mu}{1 - \alpha} \ln \left( 1 + e^{-r^T z - \eta} \right) \right],
\] (40)

where \( \mu \) is the smoothing parameter.

Here, we choose the risk level \( \alpha = 0.05 \). Our data pool consists of the daily returns of exchange-traded funds (ETFs) for \( B = 20 \) assets\(^1\) from October 2007 to October 2017 obtained from the Yahoo Finance web site. So there are 2540 outcomes in total. We impose uniform distribution on those outcomes as the empirical distribution since the true probability distributions on the outcomes of these assets are unknown. The training set contains the daily returns from October 2007 to September 2017. And the out-of-sample performance bases on the return in October 2017.

We compare the out-of-sample performance of the portfolios using two standard criteria in finance [32, 33]: (i) Sharpe ratio; (ii) cumulative wealth. These two evaluation metrics represent different focuses on measuring portfolio performance. The Sharpe ratio (SR) measures the reward-to-risk ratio of a portfolio strategy, which is computed as the portfolio return normalized by its standard deviation:

\[
SR = \frac{\hat{\mu}}{\hat{\sigma}}, \quad \hat{\mu} = \frac{1}{T} \sum_{t=0}^{T-1} \mu_t, \quad \hat{\sigma} = \sqrt{\frac{1}{T} \sum_{t=0}^{T-1} (\mu_t - \hat{\mu})^2},
\] (41)

where \( \hat{\mu} \) is the mean of portfolio returns and \( \hat{\sigma} \) represents the standard deviation of portfolio returns.

The cumulative wealth (CW) of a portfolio measures the total profit yield from the portfolio strategy across investment periods without considering any

\(^1\)The abbreviations of these 20 assets are: DIA, EEM, EFA, EWJ, EWZ, FXI, IWM, IYR, OIH, QQQ, SMH, SPY, VWO, XLB, XLE, XLF, XLI, XLK, XLP, XLU.
risks and costs, which is computed by

$$CW = \prod_{t=0}^{T-1} \omega_t^T R_t,$$  (42)

where $R_t$ is the return vector from $t$ to $t + 1$.

In each problem, we consider the $\epsilon$-approximate solution with $\epsilon = 0.05$ and 0.01, respectively. For each problem, we run 20 times and record the average results. Over the 20 runs, the average values of the total number $N$ of iterations and the batch sizes $m$, as well as the parameters $L_f, L, \sigma^2$ of the AdaSMSAG method are listed in Table 1.

**Table 1** The average parameters over 20 runs

| $\epsilon$ | $m$  | $N$   | $L_f$ | $L$  | $\sigma^2$ |
|-----------|------|-------|-------|------|------------|
| 0.05      | 110  | 458   | 1     | 0.25 | 1.0081     |
| 0.01      | 520  | 2813  | 1     | 0.25 | 1.0089     |

We draw the curves of the average objective values corresponding to the training data v.s. the CPU time for the AdaSMSAG, MSNS, M-MDSA and RSPG methods. We do not include the curves of the 2-RSPG and 2-RSPG-V methods in Fig. 1, because they have several candidate outputs in optimization phase. Due to scalability, the curves of the AdaSMSAG and the MSNS methods are close at the end. We also provide small graph for each subfigure to see clear the differences of the two methods. We can see that the AdaSMSAG method provides the computed solution with the smallest objective values corresponding to the training data.

![Fig. 1](image-url) Average objective values corresponding to the training data v.s. CPU time of 20 runs.
To see stability, we show in Table 2 the average CPU time, SR and CW, over 20 runs, corresponding to the testing data at the computed solution $\hat{x}$ by a certain method. From Table 2, the average CPU time, Sharpe ratio and cumulative wealth of the AdaSMSAG method are better than that of the other methods in almost all cases.

**Table 2** The mean SR, CW and CPU time of 20 runs

| $\epsilon$ | ALG.  | Time (Avg.) | SR (Avg.) | CW (Avg.) |
|------------|-------|-------------|-----------|-----------|
| 0.05       | AdaSMSAG | 0.0751      | 187.1667  | 5.0649    |
|            | MSNS   | 0.7778      | 175.0285  | 4.8300    |
|            | M-MDSA | 0.0776      | 170.6928  | 2.1473    |
|            | RSPG   | 3.3500      | 166.0503  | 2.4966    |
|            | 2-RSPG | 0.6081      | 166.3938  | 2.6216    |
|            | 2-RSPG-V | 0.6237     | 169.8957  | 2.6340    |
| 0.01       | AdaSMSAG | 0.5872      | 214.9344  | 5.8409    |
|            | MSNS   | 27.0953     | 213.1842  | 5.0811    |
|            | M-MDSA | 0.6264      | 187.1180  | 2.9408    |
|            | RSPG   | 30.1006     | 187.1667  | 2.6519    |
|            | 2-RSPG | 7.6664      | 196.4444  | 2.6889    |
|            | 2-RSPG-V | 6.6914     | 196.7941  | 2.6741    |

### 4.2 Wasserstein distributionally robust support vector machine (DRSVM) problem

The application of our method is on a family of Wasserstein distributionally robust support vector machine (DRSVM) problems. The SVM is one of the most frequently used classification methods and has enjoyed notable empirical successes in machine learning and data analysis [34–36]. However, even for this seemingly simple learning model, there are very few works addressing the development of fast algorithms for its Wasserstein DRO formulation, which takes the form

$$\inf_w \left\{ \frac{T}{2} \|w\|^2 + \sup_{Q \in B^\epsilon(\hat{P}_n)} E_{(x,y) \sim Q} [\ell_w(x,y)] \right\}$$

and can be reformulated as [37]

$$\min_{w,\lambda} \lambda \epsilon + \frac{T}{2} \|w\|^2 + E_i \left[ \max \left\{ 1 - w^T z_i, 1 + w^T z_i - \lambda \kappa, 0 \right\} \right],$$

s.t. $\|w\| \leq \lambda$.  \hfill (43)
According to [7], we have the smoothing problem of (43)

$$\min_{w, \lambda} \lambda \varepsilon + \frac{\tau}{2} \|w\|^2_2 + E_i \left[ \mu \ln \left( e^{\frac{1-w^T z_i}{\mu}} + e^{\frac{1+w^T z_i - \lambda}{\mu}} + 1 \right) \right],$$

s.t. \( \|w\|^2_2 \leq \lambda \),

(44)

where \( \mu \) is the smoothing parameter.

It is well known that the \( l_2 \) norm epigraphic projection has a closed form formula, which is given as follows:

**Lemma 7** (Theorem 3.3.6 of [38]) Let \( C \) be the icecream cone \( \{(x, r) \in X \times \mathbb{R} : \|x\| \leq r\} \). Then for every \( (x, r) \in X \times \mathbb{R} \):

$$P_C(x, r) = \begin{cases} (x, r), & \|x\| \leq r, \\ (0, 0), & r < \|x\| < -r, \\ \left( \frac{x}{\|x\|_2} + \frac{r}{2}, \frac{\|x\|_2 + r}{2} \right), & \|x\| \geq r. \end{cases}$$

We do numerical experiments on four real datasets described below for our experiments.

- Pima Indians Diabetes dataset comes from the UCI repository, downloaded from the web\(^2\). Pima Indians Diabetes dataset is a dataset with 768 observations and 8 variables. A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases.

- Wisconsin breast cancer dataset from the UCI repository\(^3\) can be downloaded from the web. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

- Image Segmentation dataset also comes from the UCI repository\(^4\). The instances were drawn randomly from a database of 7 outdoor images. The images were handsegmented to create a classification for every pixel. We analyzed Image Segmentation dataset in its 2-class versions, i.e., Image Segmentation(B). Image Segmentation(B) from the first 3 images and the remaining ones.

- Mnist dataset of handwritten digits, available from the web\(^5\). MnistData-10 dataset sampled 10% from Mnist dataset. The digits have been size-normalized and centered in a fixed-size image. We analyzed MnistData-10 dataset in its 2-class versions, i.e., MnistData-10(B). MnistData-10(B) from

\(^2\)https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes
\(^3\)https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
\(^4\)https://archive.ics.uci.edu/ml/datasets/Image+Segmentation
\(^5\)http://yann.lecun.com/exdb/mnist
the first 5 digits and the remaining ones. The details of the described datasets are resumed in Table 3.

| Dataset             | Classes | Sample size | Dimension |
|---------------------|---------|-------------|-----------|
| Pima                | 2       | 768         | 8         |
| Wisconsin breast cancer | 2         | 699         | 10        |
| Image Segmentation(B) | 2       | 2310        | 19        |
| MnistData-10(B)      | 2       | 6996        | 784       |

We choose the optimal values of $\tau, \varepsilon$ via 3-fold cross-validation (CV) using 20 random runs, which are determined by varying them on the grid $\{2, 1.5, 1, 0.5, 0.1\}$ and the values with the best average accuracy are chosen for each of the AdaSMSAG, M-MDSA, RSPG, 2-RSPG, and 2-RSPG-V methods. In each problem, we consider the $\varepsilon$-approximate solution with $\varepsilon = 0.05$, $L_f = \frac{\tau^2}{2}$, $L = \frac{1}{4}$.

We record in Table 4 the average optimal values of parameters $\tau, \varepsilon$, together with the corresponding average accuracy and the average CPU time in seconds. We can see that our AdaSMSAG method has the best average accuracy in all the datasets. The CPU time of the AdaSMSAG method is not the shortest, but it is only a little bit longer than the shortest one and hence is acceptable.

5 Conclusion

In this paper, we propose an adaptive smoothing mini-batch stochastic accelerated gradient (AdaSMSAG) method for solving a class of nonsmooth convex stochastic composite minimization problems with noisy first-order information. We show the convergence of the AdaSMSAG method, together with its optimal iteration complexity. Numerical experiments on a risk management in portfolio optimization and a family of Wasserstein distributionally robust support vector machine (DRSVM) problems using real datasets, demonstrate the effectiveness and efficiency of the proposed AdaSMSAG method, compared with several state-of-the-art methods.

References

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

[2] Nesterov, Y.: A method for unconstrained convex minimization problem with the rate of convergence $O\left(\frac{1}{k^2}\right)$. Soviet Math. Dokl. **27**(2), 372-376 (1983).
Adaptive smoothing mini-batch stochastic accelerated gradient method

Table 4 The optimal values of $\tau, \varepsilon$, accuracy and CPU time determined by 3-fold CV

| Dataset                  | ALG.    | $\tau$ | $\varepsilon$ | Acc  | CPU   |
|--------------------------|---------|--------|---------------|------|-------|
| Pima Indians Diabetes    | AdaSMSAG| 0.1    | 0.1           | 0.7560| 2.6130|
|                          | M-MDSA  | 0.1    | 0.1           | 0.5970| 2.7961|
|                          | RSPG    | 0.1    | 1.5           | 0.6074| 16.3706|
|                          | 2-RSPG  | 0.1    | 0.5           | 0.6322| 11.8550|
|                          | 2-RSPG-V| 0.1    | 0.1           | 0.6251| 12.3136|
| Wisconsin breast cancer  | AdaSMSAG| 0.1    | 0.1           | 0.9888| 1.1035|
|                          | M-MDSA  | 0.1    | 0.1           | 0.9470| 1.1356|
|                          | RSPG    | 1.5    | 0.5           | 0.9269| 0.8306|
|                          | 2-RSPG  | 0.1    | 0.5           | 0.9428| 12.8805|
|                          | 2-RSPG-V| 1      | 2             | 0.9381| 0.1680|
| Image Segmentation(B)    | AdaSMSAG| 0.5    | 1.5           | 0.6705| 4.0589|
|                          | M-MDSA  | 0.1    | 0.5           | 0.4261| 76.7589|
|                          | RSPG    | 2      | 1.5           | 0.4859| 12.3613|
|                          | 2-RSPG  | 2      | 2             | 0.5579| 4.1024|
|                          | 2-RSPG-V| 2      | 0.1           | 0.5620| 5.9542|
| MnistData-10(B)          | AdaSMSAG| 2      | 2             | 0.7256| 26.1302|
|                          | M-MDSA  | 0.1    | 1.5           | 0.5829| 2020.9000|
|                          | RSPG    | 1.5    | 1.5           | 0.5569| 240.9487|
|                          | 2-RSPG  | 1      | 0.1           | 0.6140| 50.7536|
|                          | 2-RSPG-V| 1      | 1.5           | 0.6343| 45.7028|

[3] Nesterov, Y.: Introductory Lectures on Convex Optimization: a basic course. Springer, New York, 87 (2003).

[4] Quoc, T.: Adaptive smoothing algorithms for nonsmooth composite convex minimization. Comput. Optim. Appl. 66(3), 425-451 (2016).

[5] Kreimer, J., Rubinstein, R. Y.: Nondifferentiable optimization via smooth approximation: General analytical approach. Ann. Oper. Res. 39(1), 97-119 (1992).

[6] Necoara, I., Suykens, J. A. K.: Application of a smoothing technique to decomposition in convex optimization. IEEE Trans. Autom. Control. 53(11), 2674-2679 (2008).

[7] Chen, X.: Smoothing methods for nonsmooth, nonconvex minimization. Math. Program. 134(1), 71-99 (2012).

[8] Chen, X.: Smoothing methods for complementarity problems and their applications: A survey. J. Oper. Res. Soc. Jpn. 43(1), 32-47 (2000).

[9] Zhang, C., Chen, X.: Smoothing projected gradient method and its application to stochastic linear complementarity problems. SIAM J. Optim. 20, 627-649 (2009).
Adaptive smoothing mini-batch stochastic accelerated gradient method

[10] Bian, W., Chen, X.: Linearly constrained non-Lipschitz optimization for image restoration, SIAM J. Imaging Sci., 8 2294-2322, 2015.

[11] Xu, M., Ye, J. J., Zhang, L.: Smoothing SQP methods for solving degenerate nonsmooth constrained optimization problems with applications to bilevel programs, SIAM J. Optim., 25 1388-1410, 2015.

[12] Liu, Y.-F., Ma, S., Dai, Y.-H., Zhang, S.: A smoothing SQP framework for a class of composite Lq minimization over polyhedron, Math. Program., Ser. A, 158 467-500, 2016.

[13] Zhang, C., Chen, X.: A smoothing active set method for linearly constrained non-Lipschitz nonconvex optimization. SIAM J. Optim. 30, 1-30 (2020).

[14] Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Statist. 22(3), 400-407 (1951).

[15] Polyak, B.: New stochastic approximation type procedures. Automat. i Telemekh. 7(2), 98-107 (1990). (English translation: Automation and Remote Control).

[16] Polyak, B., Juditsky, A.: Acceleration of stochastic approximation by averaging. SIAM J. Control Optim. 30(4), 838-855 (2006).

[17] Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM J. Optim. 19(4), 1574-1609 (2009).

[18] Ouyang, H., Gray, A.: Stochastic smoothing for nonsmooth minimizations: accelerating SGD by exploiting structure. ICML (2012).

[19] Lan, G.: An optimal method for stochastic composite optimization. Math. Program. 133(1), 365-397 (2012).

[20] Ghadimi, S., 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).

[21] Ghadimi, S., Lan, G.: Accelerated gradient methods for nonconvex nonlinear and stochastic programming. Math. Program. 156(1-2), 59-99 (2016).

[22] Nemirovski, A., Yudin, D.: Problem complexity and method efficiency in optimization. Wiley-Interscience Series in Discrete Mathematics. John Wiley, Chichester, New York (1983).
Adaptive smoothing mini-batch stochastic accelerated gradient method

[23] Bian, W., Chen, X.: A smoothing proximal gradient algorithm for non-smooth convex regression with cardinality penalty. SIAM J. Numerical Analysis. 58, 858-883 (2020).

[24] Chen, C., Mangasarian, O. L.: A class of smoothing functions for nonlinear and mixed complementarity problems. Comput. Optim. Appl. 5(2), 97-138 (1996).

[25] Beck, A.: First-order methods in optimization. SIAM, Philadelphia (2017).

[26] Ghadimi, S., Lan, G., Zhang, H.: Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Math. Program. 155(1-2), 267-305 (2016).

[27] Wasserman, L.: All of statistics: a concise course in statistical inference. Springer, New York, 26 (2004).

[28] Wang, R., Zhang, C., Wang, L., Shao Y.: Mini-batch stochastic Nesterov’s smoothing method for constrained convex stochastic composite optimization. arXiv preprint cs/2108027 (2021).

[29] Huang, W., Zhang, X.: Randomized smoothing variance reduction method for large-scale non-smooth convex optimization. SN. OR. Forum. 2(2), 1-28 (2021).

[30] Rockafellar, R. T., Uryasev, S.: Optimization of Conditional Value-at-Risk. J. Risk. 2, 21-42 (2000).

[31] Krokhmal, P., Palmquist, J., Uryasev, S.: Portfolio optimization with Conditional Value-at-Risk objective and constraints. J. Risk. 4, 43-68 (2002).

[32] DeMiguel, V., Garlappi, L., Nogales, F. J., Uppal, R.: A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. Manage. Sci. 55(5), 798-812 (2009).

[33] Shen, W., Wang, J., Ma, S.: Doubly regularized portfolio with risk minimization. AAAI. 28(1), (2014).

[34] Noble, W. S.: What is a support vector machine? Nat. Biotechnol. 24(12), 1565-1567 (2006).

[35] Huang, S., Cai, N., Pacheco, P. P., Narrandes, S., Wang, Y., Xu, W.: Applications of support vector machine (SVM) learning in cancer genomics. Cancer Genom. Proteom. 15(1), 41-51 (2018).
Adaptive smoothing mini-batch stochastic accelerated gradient method

[36] Rodriguez, R., Vogt, M., Bajorath, J.: Support vector machine classification and regression prioritize different structural features for binary compound activity and potency value prediction. ACS Omega. 2(10), 6371-6379 (2017).

[37] Li J., Chen C., So A. M.-C.: Fast epigraphical projection-based incremental algorithms for Wasserstein distributionally robust support vector machine. NIPS. 33, (2020).

[38] Bauschke H. H.: Projection algorithms and monotone operators. Ph.D. thesis, Simon Fraser University, Canada (1996).