Optimal Stochastic Approximation Algorithms for Strongly Convex Stochastic Composite Optimization, II: Shrinking Procedures and Optimal Algorithms *†

Saeed Ghadimi ‡ Guanghui Lan §

July 1, 2010 (Revised: September 20, 2011)

Abstract

In this paper we study new stochastic approximation (SA) type algorithms, namely, the accelerated SA (AC-SA), for solving strongly convex stochastic composite optimization (SCO) problems. Specifically, by introducing a domain shrinking procedure, we significantly improve the large-deviation results associated with the convergence rate of a nearly optimal AC-SA algorithm presented in [4]. Moreover, we introduce a multi-stage AC-SA algorithm, which possesses an optimal rate of convergence for solving strongly convex SCO problems in terms of the dependence on, not only the target accuracy, but also a number of problem parameters and the selection of initial points. To the best of out knowledge, this is the first time that such an optimal method has been presented in the literature. From our computational results, these AC-SA algorithms can substantially outperform the classic SA and some other SA type algorithms for solving certain classes of strongly convex SCO problems.

Keywords: stochastic approximation, convex optimization, strong convexity, complexity, optimal method, large deviation

1 Introduction

In this paper, we consider a class of strongly convex stochastic composite optimization (SCO) problems given by

$$\Psi^* := \min_{x \in X} \{\Psi(x) := f(x) + \mathcal{X}(x)\}, \tag{1.1}$$

where X is a closed convex set in Euclidean space $\mathcal{E}$, $\mathcal{X}(x)$ is a simple convex function with known structure (e.g. $\mathcal{X}(x) = 0$, $\mathcal{X}(x) = \|x\|_1$), and $f : X \to \mathbb{R}$ is a general convex function such that for some $L \geq 0$, $M \geq 0$ and $\mu \geq 0$,

$$\frac{\mu}{2}\|y - x\|^2 \leq f(y) - f(x) - \langle f'(x), y - x\rangle \leq \frac{L}{2}\|y - x\|^2 + M\|y - x\|, \quad \forall x, y \in X. \tag{1.2}$$

*The manuscript is available on www.optimization-online.org.

†Both authors were partially supported by NSF AWARD CMMI-1000347.

‡Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, (email: sghadimi@ufl.edu).

§Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, (email: glan@ise.ufl.edu).

Here, $f'(x) \in \partial f(x)$ and $\partial f(x)$ denotes the subdifferential of f at x. We assume that the first-order information of f is available via subsequent calls to a stochastic oracle ($\mathcal{SO}$). More specifically, at the t-th call, $x_t \in X$ being the input, the $\mathcal{SO}$ outputs the quantity $F(x_t, \xi_t)$ and a vector $G(x_t, \xi_t)$, where $\{\xi_t\}_{t\geq 1}$ is a sequence of independently and identically distributed (i.i.d) random variables such that $\mathbb{E}[F(x, \xi_t)] = f(x)$ and $\mathbb{E}[G(x, \xi_t)] \equiv g(x) \in \partial f(x)$ for any $x \in X$. We make the following assumption throughout the paper.

A1: For any $x \in X$ and $t \geq 1$, we have $\mathbb{E}\left[\|G(x, \xi_t) - g(x)\|_*^2\right] \leq \sigma^2$.

In some cases, Assumption A1 for the $\mathcal{SO}$ is augmented by the following stronger "light-tail" assumption (see, e.g., [4]).

A2: For any $x \in X$ and $t \geq 1$, we have $\mathbb{E}\left[\exp\{\|G(x, \xi_t) - g(x)\|_*^2/\sigma^2\}\right] \leq \exp\{1\}$.

Since the parameters L, M, μ, σ can be zero, problem (1.1) covers several important classes of convex programming (CP) problems. In particular, if f is a general Lipschitz continuous function with constant M_f, then relation (1.2) holds with $L = 0, \mu = 0$ and $M = 2M_f$. If f is a strongly convex smooth function in $\mathcal{C}^{1,1}_{L/\mu}$ (e.g., [16]), then (1.2) is satisfied with $M = 0$. Clearly, relation (1.2) also holds if f is given as the summation of certain smooth and nonsmooth convex functions. Moreover, problem (1.1) covers different classes of deterministic CP problems if $\sigma = 0$. To subsume all these different possible combinations, we refer to the aforementioned class of CP problems as $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$.

In this paper, we focus on the strongly convex case where $\mu > 0$. By the classic complexity theory for convex programming (see, e.g., Theorems 5.3.1 and 7.2.6 of [14], Theorem 2.1.13 of [16], [26] and [6]), to find *an ϵ-solution* of (1.1), i.e., a point $\bar{x} \in X$ s.t. $\mathbb{E}[\Psi(\bar{x}) - \Psi^*] \leq \epsilon$, the number of calls (or iterations) to $\mathcal{SO}$ cannot be smaller than

$$\mathcal{O}(1)\left(\sqrt{\frac{L}{\mu}}\log\frac{L\|x_0 - x^*\|^2}{\epsilon} + \frac{(M+\sigma)^2}{\mu\epsilon}\right), \tag{1.3}$$

where x_0 denotes an initial point, x^* is the optimal solution of problem (1.1) and $\mathcal{O}(1)$ represents an absolute constant. However, it is not clear if such a lower complexity bound is achievable or not.

We study stochastic approximation (SA) type algorithms for solving the aforementioned strongly convex SCO problems. The classic SA method mimics the gradient descent method and was first proposed by Robbins and Monro [20]. After that the SA algorithms were widely used in stochastic optimization (see, e.g., [1, 2, 3, 8, 17, 21, 24] and references therein). An important improvement of the SA method was developed by Polyak [18] and Polyak and Juditsky [19], where longer step-sizes were suggested with consequent averaging of the obtained iterates. Recently, Nemirovski et al. [13] demonstrated that a properly modified SA method based on iterate averaging, namely, the mirror-descent SA, can be competitive and even significantly outperform the Sample Average Approximation (see, e.g., [7, 10, 22]) method for a certain class of stochastic optimization problems. They demonstrated that the mirror descent SA exhibits an unimprovable expected rate of convergence for solving general stochastic convex programming problems without assuming strong convexity. They also show that the iteration complexity for the classic SA for solving $\mathcal{F}_{X,0}(L, 0, \sigma, \mu)$ is given by

$$\mathcal{O}(1)\left(\frac{L}{\epsilon}\max\left\{\frac{\bar{G}^2}{\mu^2}, \|x_0 - x^*\|^2\right\}\right), \tag{1.4}$$

where $\bar{G}^2 := \sup_{x\in X} \mathbb{E}[G(x,\xi)]^2$. Note that, in our setting, $M = 0$ and $\bar{G}^2$ is in the order of $\sigma^2 + L^2 \max_{x\in X} \|x - x^*\|^2$ (see Remark 1 of [9]). Clearly, bound (1.4) is substantially worse than (1.3) in terms of the dependence on L, μ and $\|x_0 - x^*\|$, although both of them are of $\mathcal{O}(1/\epsilon)$. As a result, the classic SA method of this type is very sensitive to these problem parameters and the selection of the initial point x_0.

More recently, by properly modifying Nesterov's optimal method for smooth CP [15, 16], Lan [9] presented an optimal method for solving $\mathcal{F}_{X,0}(L, M, \sigma, 0)$, which appears to be the first unified optimal method for smooth, nonsmooth and stochastic convex optimization. Motivated by this work, different variants of Nesterov's method have been studied for solving stochastic CP problems and most of these methods were designed for solving problems with a regularization term added into the objective function, i.e., $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, 0)$ (see, e.g., [4, 5, 11, 12, 25]). In [4], Ghadimi and Lan presented a generic accelerated stochastic approximation (AC-SA) algorithm, which, when employed with proper stepsize policies, can yield optimal or nearly optimal algorithms for solving different classes of problems in $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$. In particular, they show that, if $\mu > 0$, then the iteration complexity for finding an ϵ-solution of (1.1) can be bounded

$$\mathcal{O}(1)\left\{\sqrt{\frac{L\|x_0 - x^*\|^2}{\epsilon}} + \frac{(M+\sigma)^2}{\mu\epsilon}\right\}. \tag{1.5}$$

This bound is significantly better than the bound in (1.4) for the classic SA method applied to $\mathcal{F}_{X,0}(L, 0, \sigma, \mu)$ in terms of the dependence on L, μ, σ and $\|x_0 - x^*\|$ (note that $M = 0$ for this comparison). They also presented an effective validation procedure to assess quality of the solutions generated by the AC-SA algorithm. Note however, that the complexity bound (1.5) is still significantly worse than the lower bound in (1.3) in terms of the dependence on $\|x_0 - x^*\|$. Hence, this algorithm is nearly optimal for solving strongly convex SCO problems with $\mu > 0$.

While the aforementioned complexity results evaluate, on average, the performance of the SA-type algorithms over many different runs, the large-deviations associated with these complexity results estimate the performance of a single run of these algorithms (e.g. [4, 9, 13]). In particular, under Assumption A2, Ghadimi and Lan [4] investigated the iteration-complexity of the AC-SA algorithm for finding an (ϵ, Λ)-*solution* of (1.1), i.e., a point $\bar{x} \in X$ s.t. $\text{Prob}\{\Psi(\bar{x}) - \Psi^* > \epsilon\} \leq \Lambda$, for a given confidence level $\Lambda \in (0, 1)$. They showed that, under Assumption A2, the iteration-complexity of the AC-SA algorithm for finding an (ϵ, Λ)-solution for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with $\mu > 0$, after disregarding a few constant factors, is bounded by

$$\mathcal{O}\left\{\left(\frac{1}{\epsilon}\log\frac{1}{\Lambda}\right)^2\right\}.$$

Note, however, that this bound is significantly worse than the one in (1.5) in terms of the dependence on ϵ.

Our contribution in this paper mainly consists of the following aspects. Firstly, observing that a single-run of aforementioned nearly optimal AC-SA algorithm for strongly convex SCO has significantly worse theoretical performance guarantee than the average one over many runs, we develop ways to improve the former iteration-complexity results so that they become comparable to the latter ones. The basic idea we used is to properly "shrink" the feasible set of (1.1) once in a while during the execution of this algorithm. By incorporating such a domain shrinking procedure, we show that the iteration complexity of the above-mentioned nearly optimal AC-SA algorithm for finding

an (ϵ, Λ)-solution for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ can be significantly improved to

$$\mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon\Lambda}\right).$$

Secondly, by properly restarting the AC-SA algorithm for solving SCO problem without assuming strong convexity with certain stepsize policy, we present an optimal algorithm with the iteration complexity bounded by (1.3) for solving strongly convex SCO problems with $\mu > 0$. Hence, this multi-stage AC-SA algorithm possesses an optimal rate of convergence in terms of the dependence on, not only the target accuracy, but also a number of problem parameters and the selection of initial points. To the best of out knowledge, this is the first time that such an optimal method has been presented in the literature. Moreover, we develop ways to improve the large-deviation properties associated with this iteration complexity by using the aforementioned domain shrinking procedure. Finally, we demonstrate through our numerical experiments that the aforementioned optimal or nearly optimal AC-SA algorithms for strongly convex SCO can substantially outperform the classic SA and some other SA type algorithms. We also compare different variants of AC-SA algorithms and point out the situations when one would outperform another.

The paper is organized as follows. In Section 2, we review the nearly optimal AC-SA algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with $\mu > 0$ and develop a domain shrinking procedure to improve its iteration complexity for finding an (ϵ, Λ)-solution of (1.1). Section 3 is devoted to the optimal algorithms for solving $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with $\mu > 0$ and their convergence results. Finally, we present some promising numerical results for the AC-SA algorithms in Section 4.

Notation

- $\mathcal{E}$ is endowed with inner product $\langle\cdot,\cdot\rangle$ and an arbitrary norm $\|\cdot\|$ (not necessarily the one induced by the inner product $\langle\cdot,\cdot\rangle$).
- For the random process $\xi_1, \xi_2, ...$, we set $\xi_{[t]} := (\xi_1, ..., \xi_t)$, and denote by $\mathbb{E}_{|\xi_{[t]}}$ the conditional expectation., $\xi_{[t]}$ being given.

2 Nearly optimal AC-SA algorithms for strongly convex SCO

In this section, we review a nearly optimal AC-SA algorithm for solving strongly convex SCO problems presented in [4]. Moreover, we introduce a domain shrinking procedure that can improve the large-deviation results associated with the convergence rate of this algorithm.

2.1 The AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$

In this subsection, we first review the generic AC-SA algorithm developed in [4, 9] and then show that it can yield a nearly optimal algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with $\mu > 0$. Similarly to how the classic SA evolves from the gradient descent method, the AC-SA algorithm is obtained by replacing the gradients with stochastic (sub)gradients in Nesterov's method for smooth CP [15, 16].

Let ω be a continuously differentiable and strongly convex function with modulus ν with respect to $\|\cdot\|$, i.e.,

$$\langle x - z, \nabla\omega(x) - \nabla\omega(z)\rangle \geq \nu\|x - z\|^2, \ \ \forall x, z \in X. \tag{2.1}$$

We define the *prox-function* associated with ω as

$$V(x,z) \equiv V_\omega(x,z) = \omega(z) - [\omega(x) + \langle \nabla\omega(x), z-x\rangle]. \tag{2.2}$$

We assume that $V(x,z)$ is chosen such that the solution of

$$P_\omega(g,x) := \arg\min_{z\in X} \{\langle g,z\rangle + V(x,z) + \mathcal{X}(z)\} \tag{2.3}$$

is easily computable for any $g \in \mathcal{E}^*$ and $x \in X$. Moreover, if there exists a constant $\mathcal{Q}$ such that $V(x,z) \le \mathcal{Q}\|x-z\|^2/2$ for any $x,z \in X$, then we say that $V(\cdot,\cdot)$ is growing quadratically [4]. Without loss of generality, we assume throughout this paper that $\mathcal{Q} = 1$ for the prox-function $V(x,z)$ if it grows quadratically, i.e.,

$$V(x,z) \le \frac{1}{2}\|x-z\|^2, \quad \forall x,z \in X. \tag{2.4}$$

We are now ready to describe the generic AC-SA algorithm.

A generic AC-SA algorithm

Input: $x_0 \in X$, prox-function $V(x,z)$, stepsize parameters $\{\alpha_t\}_{t\ge1}$ and $\{\gamma_t\}_{t\ge1}$ s.t. $\alpha_1 = 1$, $\alpha_t \in (0,1)$ for any $t \ge 2$, and $\gamma_t > 0$ for any $t \ge 1$.

0) Set the initial points $x_0^{ag} = x_0$ and $t = 1$;

1) Set

$$x_t^{md} = \frac{(1-\alpha_t)(\mu+\gamma_t)}{\gamma_t + (1-\alpha_t^2)\mu} x_{t-1}^{ag} + \frac{\alpha_t[(1-\alpha_t)\mu+\gamma_t]}{\gamma_t + (1-\alpha_t^2)\mu} x_{t-1}; \tag{2.5}$$

2) Call the $\mathcal{SO}$ for computing $G_t \equiv G(x_t^{md}, \xi_t)$. Set

$$x_t = \arg\min_{x\in X} \left\{ \alpha_t[\langle G_t, x\rangle + \mathcal{X}(x) + \mu V(x_t^{md}, x)] + [(1-\alpha_t)\mu + \gamma_t]\, V(x_{t-1}, x) \right\}, \tag{2.6}$$

$$x_t^{ag} = \alpha_t x_t + (1-\alpha_t) x_{t-1}^{ag}; \tag{2.7}$$

3) Set $t \leftarrow t+1$ and go to step 1.

The following theorem, whose proof can be found in [4], summarizes the main convergence properties of the generic AC-SA algorithm.

Theorem 1 *Consider the generic AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$ and suppose that condition (2.4) holds whenever $\mu > 0$. Also assume that $\{\alpha_t\}_{t\ge1}$ and $\{\gamma_t\}_{t\ge1}$ are chosen such that*

$$\nu(\mu+\gamma_t) > L\alpha_t^2, \tag{2.8}$$

$$\gamma_1/\Gamma_1 = \gamma_2/\Gamma_2 = \ldots, \tag{2.9}$$

where

$$\Gamma_t := \begin{cases} 1, & t = 1, \\ (1-\alpha_t)\Gamma_{t-1}, & t \ge 2. \end{cases} \tag{2.10}$$

Then,

a) under Assumption A1, we have

$$\mathbb{E}[\Psi(x_t^{ag}) - \Psi^*] \le \mathcal{B}_e(t) := \Gamma_t \gamma_1 V(x_0, x^*) + \Gamma_t \sum_{\tau=1}^{t} \frac{2\alpha_\tau^2(M^2+\sigma^2)}{\Gamma_\tau[\nu(\mu+\gamma_\tau) - L\alpha_\tau^2]}, \tag{2.11}$$

for any $t \ge 1$, where x^* is an arbitrary optimal solution of (1.1);

b) under Assumption A2, we have

$$\text{Prob}\{\Psi(x_t^{ag}) - \Psi^* \ge \mathcal{B}_e(t) + \lambda\, \mathcal{B}_p(t)\} \le \exp\{-\lambda^2/3\} + \exp\{-\lambda\}, \tag{2.12}$$

for any $\lambda > 0$ and $t \ge 1$, where

$$\mathcal{B}_p(t) := \sigma\Gamma_t R_X(x^*) \left(\sum_{\tau=1}^{t} \frac{\alpha_\tau^2}{\Gamma_\tau^2}\right)^{\frac{1}{2}} + \Gamma_t \sum_{\tau=1}^{t} \frac{2\alpha_\tau^2\sigma^2}{\Gamma_\tau[\nu(\mu+\gamma_\tau) - L\alpha_\tau^2]}, \tag{2.13}$$

$$R_X(x^*) := \max_{x\in X} \|x - x^*\|. \tag{2.14}$$

By properly specifying the selection of stepsize parameters α_t and β_t, $t \ge 1$, we present in Proposition 2 a nearly optimal algorithm for solving strongly convex SCO problems. The proof of this result can be found in Proposition 9 of [4].

Proposition 2 *Let $\{x_t^{ag}\}_{t\ge1}$ be computed by the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with*

$$\alpha_t = \frac{2}{t+1} \quad \textit{and} \quad \gamma_t = \frac{4L}{\nu t(t+1)}, \quad \forall\, t \ge 1. \tag{2.15}$$

If $\mu > 0$ and condition (2.4) holds, then under Assumption A1, we have

$$\mathbb{E}[\Psi(x_t^{ag}) - \Psi^*] \le \mathcal{D}_e(t) := \frac{4LV(x_0, x^*)}{\nu t(t+1)} + \frac{8(M^2+\sigma^2)}{\nu\mu(t+1)}, \quad \forall\, t \ge 1. \tag{2.16}$$

If Assumption A2 holds, then, $\forall \lambda > 0, \forall t \ge 1$,

$$\text{Prob}\{\Psi(x_t^{ag}) - \Psi^* \ge \mathcal{D}_e(t) + \lambda \mathcal{D}_p(t)\} \le \exp\{-\lambda^2/3\} + \exp\{-\lambda\}, \tag{2.17}$$

where

$$\mathcal{D}_p(t) := \frac{2\sigma R_X(x^*)}{\sqrt{3t}} + \frac{8\sigma^2}{\nu\mu(t+1)}. \tag{2.18}$$

For the sake of future reference, we call the above AC-SA algorithm with stepsize policy (2.15) the single-stage AC-SA algorithm.

We now make a few remarks about the single-stage AC-SA algorithm and its convergence properties in Proposition 2. First, in view of the lower complexity bound (1.3), the AC-SA algorithm with the stepsize policy (2.15) achieves the optimal rate of convergence for solving $\mathcal{F}_{X,\mathcal{X}}(0, M, \sigma, \mu)$, i.e., for those problems without a smooth component. It is also nearly optimal for solving $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$, in the sense that the second term $8(M^2+\sigma^2)/[\nu\mu(t+1)]$ of $\mathcal{D}_e(t)$ in (2.16) is unimprovable. The first

term of $\mathcal{D}_e(t)$ (for abbreviation, L-component) depends on the product of L and $V(x_0, x^*)$, which can be as big as $LV(x_0, x^*) \leq 2(t+1)(M^2+\sigma^2)/\mu$ without affecting the rate of convergence (up to a constant factor 2). Note however that the L-component of $\mathcal{D}_e(t)$ is significantly worse than the corresponding one, i.e., the first term in the lower complexity bound (1.3).

Second, observe that the bounds $\mathcal{D}_e(t)$ and $\mathcal{D}_p(t)$, defined in (2.16) and (2.18) respectively, are not in the same order of magnitude, that is, $\mathcal{D}_e(t) = \mathcal{O}(1/t)$ and $\mathcal{D}_p(t) = \mathcal{O}(1/\sqrt{t}\,)$. As discussed in Section 4.2 of [4], under Assumption A1, the number of iterations for finding an (ϵ, Λ)-solution $\bar{x} \in X$ such that $\text{Prob}\{\Psi(\bar{x}) - \Psi^* < \epsilon\} \geq 1 - \Lambda$ for a given confidence level $\Lambda \in (0,1)$ can be bounded by

$$\mathcal{O}(1)\left\{\frac{1}{\Lambda}\left(\sqrt{\frac{LV(x_0,x^*)}{\nu\epsilon}} + \frac{M^2+\sigma^2}{\nu\mu\epsilon}\right)\right\}. \tag{2.19}$$

Moreover, if Assumption A2 holds, then the number of iterations for finding an (ϵ, Λ)-solution of (1.1) can be bounded by

$$\mathcal{O}(1)\left\{\sqrt{\frac{LV(x_0,x^*)}{\nu\epsilon}} + \frac{M^2+\sigma^2}{\nu\mu\epsilon} + \frac{\sigma^2}{\nu\mu\epsilon}\log\frac{1}{\Lambda} + \left(\frac{\sigma R_X(x^*)}{\epsilon}\log\frac{1}{\Lambda}\right)^2\right\}. \tag{2.20}$$

Note, however, that the above iteration-complexity bound has a significant worse dependence on ϵ than the one in (2.19), although it only logarithmically depends on $1/\Lambda$.

2.2 A domain shrinking procedure for the single-stage AC-SA algorithm

Our goal in this subsection is to introduce a "domain shrinking" procedure for the single-stage AC-SA algorithm presented in Subsection 2.1 in order to significantly improve the theoretical iteration-complexity bound (2.20).

More specifically, for a given a positive constant $\hat{S} > 0$ and an initial bound $\mathcal{V}_0$ such that $\Psi(x_0) - \Psi^* \leq \mathcal{V}_0$, we replace identity (2.6) in the generic AC-SA algorithm by

$$x_t = \arg\min_{x \in X_t}\left\{\alpha_t[\langle G_t, x\rangle + \mathcal{X}(x) + \mu V(x_t^{md}, x)] + [(1-\alpha_t)\mu + \gamma_t]\, V(x_{t-1}, x)\right\}. \tag{2.21}$$

Here,

$$X_t := \left\{x \in X : \|x - x^+_{\mathcal{S}(\mathcal{K}_t)}\|^2 \leq [\mathcal{R}(\mathcal{K}_t)]^2\right\}, \tag{2.22}$$

$$\mathcal{K}_t := \max\left\{0, \left\lceil \log\frac{t}{\hat{S}}\right\rceil\right\}, \quad \mathcal{S}(k) = \left\{\begin{array}{ll} 0, & k=0 \\ \hat{S}\,2^{k-1}, & k \geq 1 \end{array}\right., \quad \text{and} \quad \mathcal{R}(k) := \sqrt{\frac{2\mathcal{V}_0}{\mu 2^k}}. \tag{2.23}$$

Note that the notation log has a base of 2 throughout this paper. Observe that in the above variant of the AC-SA algorithm, the feasible set X_t of (2.21) will be reduced once in a while as the algorithm advances. For the sake of future references, we call the algorithm described above *the shrinking AC-SA algorithm for* $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$. Clearly, to efficiently implement this algorithm, it is necessary to assume that subproblem (2.21) can be easily solved. This assumption does hold for many practical problems, e.g., for the unconstrained problems where $X = \mathbb{R}^n$ in Section 4. In fact, if the original subproblem (2.5) is easy to solve, one can always conveniently solve (2.21), for example, by using the Lagrangian relaxation methods, since the latter subproblem has only one more additional simple constraint than (2.5).

The remaining part of this subsection will be dedicated to the convergence analysis of the above shrinking AC-SA algorithm. Lemma 3 below describes some convergence properties for the shrinking AC-SA algorithm using general stepsize parameters.

Lemma 3 *Consider the shrinking AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$ with $\{\alpha_t\}_{t\geq 1}$ and $\{\gamma_t\}_{t\geq 0}$ satisfying (2.8) and (2.9). Suppose that $\mu > 0$ and condition (2.4) holds. Also assume that the algorithm is run for t iterations and that*

$$\|x^* - x^+_{\mathcal{S}(i)}\|^2 \leq [\mathcal{R}(i)]^2, \quad \forall\, i = 1,\ldots,\mathcal{K}_t, \tag{2.24}$$

where $\mathcal{S}(i), \mathcal{R}(i)$ and $\mathcal{K}_t$ are defined in (2.23) and

$$x^+_{t-1} := \frac{\alpha_t\mu}{\mu+\gamma_t}x^{md}_t + \frac{(1-\alpha_t)\mu+\gamma_t}{\mu+\gamma_t}x_{t-1}. \tag{2.25}$$

Then, under Assumption A2,

$$\text{Prob}\left\{\Psi(x^{ag}_t) - \Psi^* + \mu V(x_t,x^*) \geq \mathcal{B}_e(t) + \lambda\,\mathcal{B}^s_p(t)\right\} \leq \exp(-\lambda^2/3) + \exp(-\lambda),\ \forall\lambda > 0, \tag{2.26}$$

where $\mathcal{B}_e(t)$ is defined in (2.11),

$$\mathcal{B}^s_p(t) := \Gamma_t\left[\sigma\Theta^{\frac{1}{2}}_t + \sigma^2\sum_{\tau=1}^t \frac{2\alpha^2_\tau}{\Gamma_\tau[\nu(\mu+\gamma_\tau) - L\alpha^2_\tau]}\right] \quad \textit{and} \quad \Theta_t := \sum_{\tau=1}^t \frac{\alpha^2_\tau[\mathcal{R}(\mathcal{K}_\tau)]^2}{\Gamma^2_\tau}. \tag{2.27}$$

Proof. It can be easily seen from the definitions in (2.23) that $\mathcal{K}_\tau \leq \mathcal{K}_t$ for any $1 \leq \tau \leq t$. We then conclude from this observation and assumption (2.24) that

$$\|x^* - x^+_{\mathcal{S}(\mathcal{K}_\tau)}\|^2 \leq [\mathcal{R}(\mathcal{K}_\tau)]^2, \quad \tau = 1,\ldots,t, \tag{2.28}$$

which, in view of (2.22), implies that $x^* \in \bigcap_{\tau=1}^t X_\tau$. Moreover, by using relations (3.16) and (3.27) in [4], we have, for any $x \in \bigcap_{\tau=1}^t X_\tau$,

$$\Psi(x^{ag}_t) - \Psi(x) + \mu V(x_t,x) \leq \Gamma_t\gamma_1 V(x_0,x) + \Gamma_t\sum_{\tau=1}^t\left[\frac{\alpha_\tau}{\Gamma_\tau}\langle\delta_\tau, x - x^+_{\tau-1}\rangle + \frac{\alpha^2_\tau(M+\|\delta_\tau\|_*)^2}{\Gamma_\tau\left[\nu(\mu+\gamma_\tau) - L\alpha^2_\tau\right]}\right], \tag{2.29}$$

Let $\zeta_\tau := \Gamma^{-1}_\tau\alpha_\tau\langle\delta_\tau, x^* - x^+_{\tau-1}\rangle$. Clearly, we have

$$\mathbb{E}_{|\xi_{[\tau-1]}}[\langle\delta_\tau, x^* - x^+_{\tau-1}\rangle] = 0, \tag{2.30}$$

and hence that $\{\zeta_\tau\}_{\tau\geq 1}$ is a martingale-difference. Moreover, by (2.21), (2.22) and (2.28), we have

$$\|x^* - x^+_{\tau-1}\| \leq \|x^* - x^+_{\mathcal{S}(\mathcal{K}_\tau)}\| + \|x^+_{\tau-1} - x^+_{\mathcal{S}(\mathcal{K}_\tau)}\| \leq 2\mathcal{R}(\mathcal{K}_\tau), \quad \tau = 1,\ldots,t, \tag{2.31}$$

which, in view of Assumption A2, implies that

$$\begin{aligned}\mathbb{E}_{|\xi_{[\tau-1]}}\left\{\exp\left[\zeta^2_\tau/\left(2\Gamma^{-1}_\tau\alpha_\tau\sigma\mathcal{R}(\mathcal{K}_\tau)\right)^2\right]\right\} &\leq \mathbb{E}_{|\xi_{[\tau-1]}}\left[\exp\{(\|\delta_\tau\|_*\|x^* - x^+_{\tau-1}\|)^2/\left(2\sigma\mathcal{R}(\mathcal{K}_\tau)\right)^2\}\right] \\ &\leq \mathbb{E}_{|\xi_{[\tau-1]}}\left[\exp\{(\|\delta_\tau\|_*)^2/\sigma^2\}\right] \leq \exp(1).\end{aligned}$$

By (2.30), (2.31) and Lemma 6 of [4], we have

$$\forall \lambda \geq 0 : \text{Prob}\left\{\sum_{\tau=1}^{t} \zeta_\tau > 2\lambda\sigma \left[\sum_{\tau=1}^{t} \Gamma_\tau^{-2}\alpha_\tau^2[\mathcal{R}(\mathcal{K}_\tau)]^2\right]^{\frac{1}{2}}\right\} \leq \exp\{-\lambda^2/3\}.$$

Combining (2.29), the previous conclusion and relation (3.30) of [4], we obtain (2.26). ∎

We specialize in Lemma 4 the convergence properties in Lemma 3 for the the shrinking AC-SA algorithm with stepsize policy (2.15).

Lemma 4 *Consider the shrinking AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with stepsize policy (2.15). Suppose that $\mu > 0$ and condition (2.4) holds. Also assume that the algorithm is run for $N > \hat{S}$ iterations. Let E_k, $0 \leq k \leq \mathcal{K}_N$, denote the event of*

$$E_k := \left\{\|x^* - x^+_{\mathcal{S}(i)}\|^2 \leq [\mathcal{R}(i)]^2, \forall\, 0 \leq i \leq k\right\}. \tag{2.32}$$

Then, under Assumption A2, we have, $\forall \lambda > 0$,

$$\text{Prob}\left\{\Psi(x_N^{ag}) - \Psi^* \geq \mathcal{D}_e(N) + \lambda\mathcal{D}_p^s(N)|E_{\mathcal{K}_N}\right\} \leq \exp(-\lambda^2/3) + \exp(-\lambda), \tag{2.33}$$

where $\mathcal{D}_e(N)$ is given by (2.16) and

$$\mathcal{D}_p^s(N) := \frac{32\sqrt{14}\,\sigma\mathcal{V}_0^{\frac{1}{2}}\hat{S}^{\frac{1}{2}}}{3\mu^{\frac{1}{2}}N} + \frac{8\sigma^2}{\nu\mu N}. \tag{2.34}$$

Moreover, for any $1 \leq k \leq \mathcal{K}_N$ and $\lambda > 0$, we have

$$\text{Prob}\left\{\|x^* - x^+_{\mathcal{S}(k)}\|^2 \geq \frac{2}{\mu\min\{1,\nu\}}\left[\hat{\mathcal{D}}_e^s(k) + \lambda\hat{\mathcal{D}}_p^s(k)\right]|E_{k-1}\right\} \leq \exp(-\lambda^2/3) + \exp(-\lambda), \tag{2.35}$$

where

$$\hat{\mathcal{D}}_e^s(k) := \frac{16L\mathcal{V}_0}{\nu\mu\hat{S}^2\,2^{2k}} + \frac{16(M^2+\sigma^2)}{\nu\mu\hat{S}\,2^k}, \quad \textit{and} \quad \hat{\mathcal{D}}_p^s(k) := \frac{8\sqrt{14}\sigma\mathcal{V}_0^{\frac{1}{2}}}{3\mu^{\frac{1}{2}}\hat{S}^{\frac{1}{2}}2^k} + \frac{16\sigma^2}{\nu\mu\hat{S}\,2^k}. \tag{2.36}$$

Proof. We first show (2.33). Let $\mathcal{B}_e(t)$ and $\mathcal{D}_e(t)$ be defined in (2.11) and (2.16), respectively. It is shown in the proof of Proposition 9 of [4] that $\mathcal{B}_e(N) \leq \mathcal{D}_e(N)$. Hence, in view of Lemma 3, it suffices to show that $\mathcal{B}_p^s(N) \leq \mathcal{D}_p^s(N)$. Clearly, by (2.10) and (2.15) we have

$$\Gamma_t = \frac{2}{t(t+1)}, \quad \frac{\gamma_t}{\Gamma_t} = \frac{2L}{\nu}, \quad \nu(\mu+\gamma_t) - L\alpha_t^2 \geq \nu\mu, \tag{2.37}$$

which implies

$$\sum_{\tau=1}^{t} \frac{\alpha_\tau^2}{\Gamma_\tau\left[\nu(\mu+\gamma_\tau) - L\alpha_\tau^2\right]} \leq \sum_{\tau=1}^{t} \frac{\alpha_\tau^2}{\nu\mu\Gamma_\tau} \leq \frac{2t}{\nu\mu}.$$

By (2.15), (2.27) and the above inequality, we have

$$\mathcal{B}_p^s(t) \leq \sigma\Gamma_t[\Theta(t)]^{\frac{1}{2}} + \frac{8\sigma^2}{\nu\mu(t+1)} \leq \frac{2\sigma[\Theta(t)]^{\frac{1}{2}}}{t^2} + \frac{8\sigma^2}{\nu\mu t}. \tag{2.38}$$

To bound $\Theta(t)$ in the above relation, let us denote $S_k \equiv \mathcal{S}(k)$ and $\mathcal{R}_k \equiv \mathcal{R}(k)$ for any $k \geq 0$. Note that by (2.23), for any $S_{k-1} < t \leq S_k$, we have $\mathcal{K}_t = k-1$ and $\mathcal{R}(\mathcal{K}_t) = \mathcal{R}(k-1) = \mathcal{R}_{k-1}$. The previous observation together with (2.27) then imply that, for any $1 \leq k \leq \mathcal{K}_N$,

$$\Theta(S_k) = \sum_{\tau=1}^{S_k} \tau^2 [\mathcal{R}(\mathcal{K}_\tau)]^2 = \sum_{i=1}^{k} \sum_{\tau=S_{i-1}+1}^{S_i} \tau^2 [\mathcal{R}_{i-1}]^2 = \frac{2\mathcal{V}_0}{\mu} \sum_{i=1}^{k} \left(\frac{1}{2^{i-1}} \sum_{\tau=S_{i-1}+1}^{S_i} \tau^2 \right)$$

Moreover, using the fact that

$$\frac{1}{2^{i-1}} = \frac{1}{2^{k-1}} + \left(\frac{1}{2^{k-1}} + \frac{1}{2^{k-2}} + \ldots + \frac{1}{2^i} \right), \quad \forall\, 1 \leq i \leq k,$$

we have

$$\begin{aligned}
\sum_{i=1}^{k} \left(\frac{1}{2^{i-1}} \sum_{\tau=S_{i-1}+1}^{S_i} \tau^2 \right) &= \frac{1}{2^{k-1}} \sum_{\tau=1}^{S_k} \tau^2 + \sum_{i=1}^{k-1} \left[\left(\frac{1}{2^{k-1}} + \frac{1}{2^{k-2}} + \ldots + \frac{1}{2^i} \right) \sum_{\tau=S_{i-1}+1}^{S_i} \tau^2 \right] \\
&= \frac{1}{2^{k-1}} \sum_{\tau=1}^{S_k} \tau^2 + \frac{1}{2^{k-1}} \sum_{\tau=1}^{S_{k-1}} \tau^2 + \sum_{i=1}^{k-2} \left[\left(\frac{1}{2^{k-2}} + \ldots + \frac{1}{2^i} \right) \sum_{\tau=S_{i-1}+1}^{S_i} \tau^2 \right] \\
&= \frac{1}{2^{k-1}} \sum_{\tau=1}^{S_k} \tau^2 + \sum_{i=1}^{k-1} \left(\frac{1}{2^i} \sum_{\tau=1}^{S_i} \tau^2 \right).
\end{aligned}$$

Combining the previous two observations and using the definition of $S(k)$ in (2.23), we have

$$\begin{aligned}
\Theta(S_k) &\leq \frac{2\mathcal{V}_0}{\mu} \left[\frac{1}{2^{k-1}} \sum_{\tau=1}^{S_k} \tau^2 + \sum_{i=1}^{k-1} \left(\frac{1}{2^i} \sum_{\tau=1}^{S_i} \tau^2 \right) \right] \\
&\leq \frac{2\mathcal{V}_0}{\mu} \left[\frac{1}{3 \cdot 2^{k-1}} (S_k+1)^3 + \sum_{i=1}^{k-1} \left(\frac{1}{3 \cdot 2^i} (S_i+1)^3 \right) \right] \\
&\leq \frac{2\mathcal{V}_0}{\mu} \left(\frac{8S_k^3}{3 \cdot 2^{k-1}} + \sum_{i=1}^{k-1} \frac{8S_i^3}{3 \cdot 2^i} \right) = \frac{16\mathcal{V}_0 \hat{S}^3}{3\mu} \left(4^{k-1} + \frac{1}{8} \sum_{i=1}^{k-1} 4^i \right) \leq \frac{14\mathcal{V}_0 \hat{S}^3}{9\mu} 4^k, \quad (2.39)
\end{aligned}$$

where the second and third inequality follow from the facts that $\sum_{i=1}^{s} i^2 \leq (s+1)^3/3$ and $\mathcal{S}_i \geq 1$ for $i \geq 1$, respectively. Using the above conclusion, the fact that $N \leq \mathcal{S}(\mathcal{K}_N + 1)$ due to (2.23) and the simple observation that $\Theta(\cdot)$ is non-decreasing, we conclude

$$\Theta(N) \leq \Theta(\mathcal{S}(\mathcal{K}_N+1)) \leq \frac{14\mathcal{V}_0 \hat{S}^3}{9\mu} 4^{\mathcal{K}_N+1} = \frac{56\mathcal{V}_0 \hat{S}^3}{9\mu} 4^{\mathcal{K}_N} \leq \frac{56\mathcal{V}_0 \hat{S}^3}{9\mu} \left(\frac{2N}{\hat{S}} \right)^2 = \frac{224\mathcal{V}_0 \hat{S} N^2}{9\mu}, \quad (2.40)$$

where the last inequality follows from the fact that $N \geq \mathcal{S}(\mathcal{K}_N) = \hat{S} 2^{\mathcal{K}_N - 1}$ due to (2.23). Hence, by using the above inequality, (2.34) and (2.38), we obtain

$$\mathcal{B}_p^s(N) \leq \frac{2\sigma}{N^2} [\Theta(N)]^{\frac{1}{2}} + \frac{8\sigma^2}{\nu\mu N} \leq \frac{32\sqrt{14}\,\sigma \mathcal{V}_0^{\frac{1}{2}} \hat{S}^{\frac{1}{2}}}{3\mu^{\frac{1}{2}} N} + \frac{8\sigma^2}{\nu\mu N} = \mathcal{D}_p^s(N).$$

We have thus shown that (2.33) holds.

To show (2.35), we first claim that

$$\mathrm{Prob}\left\{\Psi(x_{S_k}^{ag}) - \Psi^* + \mu V(x_{S_k}, x^*) \geq \hat{\mathcal{D}}_e^s(k) + \lambda \hat{\mathcal{D}}_p^s(k) | E_{k-1}\right\} \leq \exp(-\lambda^2/3) + \exp(-\lambda). \tag{2.41}$$

Indeed, in view of Lemma 3, it suffices to show that $\mathcal{B}_e(S_k) \leq \hat{\mathcal{D}}_e^s(k)$ and that $\mathcal{B}_p^s(S_k) \leq \hat{\mathcal{D}}_p^s(k)$. Note that by (2.4), (1.2) and the fact that $\Psi(x_0) - \Psi^* \leq \mathcal{V}_0$, we have

$$V(x_0, x^*) \leq \|x_0 - x^*\|^2/2 \leq [\Psi(x_0) - \Psi^*]/\mu \leq \mathcal{V}_0/\mu,$$

which, in view of (2.16), (2.23) and (2.36), then implies that $\mathcal{D}_e(t) \leq \hat{\mathcal{D}}_e^s(k)$. Using this observation and the fact $\mathcal{B}_e(t) \leq \mathcal{D}_e(t)$, we have $\mathcal{B}_e(S_k) \leq \hat{\mathcal{D}}_e^s(k)$. Moreover, it follows from (2.38), (2.39) and (2.36) that, for any $1 \leq k \leq \mathcal{K}_N$,

$$\mathcal{B}_p^s(S_k) \leq \frac{2\sigma}{\mathcal{S}_k^2}[\Theta(\mathcal{S}_k)]^{\frac{1}{2}} + \frac{8\sigma^2}{\nu\mu\mathcal{S}_k} \leq \frac{8\sqrt{14}\sigma\mathcal{V}_0^{\frac{1}{2}}}{3\mu^{\frac{1}{2}}\hat{S}^{\frac{1}{2}}2^k} + \frac{16\sigma^2}{\nu\mu\hat{S}\,2^k} = \hat{\mathcal{D}}_p^s(k).$$

We have thus shown that relation (2.41) holds. Now observe that by (2.5) and (2.25), x_t^+ can be written as a convex combination of x_t^{ag} and x_t, which, in view of the convexity of $\|\cdot\|^2$ and the strong-convexity of $\Psi(x)$ and $\omega(x)$, implies that

$$\begin{aligned}\|x_t^+ - x^*\|^2 &\leq \|x_t^{ag} - x^*\|^2 + \|x_t - x^*\|^2 \leq \frac{2}{\mu}[\Psi(x_t^{ag}) - \Psi^*] + \frac{2}{\nu}V(x_t, x^*) \\ &\leq \frac{2}{\mu\min\{1,\nu\}}[\Psi(x_t^{ag}) - \Psi^* + \mu V(x_t, x^*)],\ \forall\, t \geq 1.\end{aligned}$$

Using the above conclusion and (2.41), we immediately obtain (2.35). ∎

We are now ready to establish the iteration complexity of the shrinking AC-SA algorithm with stepsize policy (2.15).

Proposition 5 *Let $\{x_t^{ag}\}_{t\geq 1}$ be computed by the shrinking AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$ with stepsize policy (2.15). Assume that $\mu > 0$ and condition (2.4) holds. Also suppose that, for a given confidence level $\Lambda \in (0,1)$, the number of iterations N satisfies*

$$N > \hat{S} \equiv \left\lceil \max\left\{ \left(\frac{32L}{\nu\mu\min\{1,\nu\}}\right)^{\frac{1}{2}}, \frac{64\max\{M^2+\sigma^2, \tilde{\lambda}\sigma^2\}}{\nu\mu\mathcal{V}_0\min\{1,\nu\}}, \frac{\sigma^2}{\mu\mathcal{V}_0}\left(\frac{32\sqrt{14}\,\tilde{\lambda}}{3\min\{1,\nu\}}\right)^2 \right\} \right\rceil \tag{2.42}$$

where $\tilde{\lambda}$ is chosen such that

$$\exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda}) \leq \frac{\Lambda}{2N+1}. \tag{2.43}$$

Then under Assumptions A1 and A2,

$$\mathrm{Prob}\left\{\Psi(x_N^{ag}) - \Psi^* \geq \mathcal{D}_e(N) + \tilde{\lambda}\,\mathcal{D}_p^s(N)\right\} \leq \Lambda, \tag{2.44}$$

where $\mathcal{D}_e(\cdot)$ and $\mathcal{D}_p^s(\cdot)$ are given by (2.16) and (2.34), respectively.

Proof. Let E_k, $0 \le k \le \mathcal{K}_N$, be the random event defined in (2.32), and $\bar{E}_k$ denote its complement. It can be easily seen from (2.5) and (2.25) that $x_0^+ = x_0$. The observation, in view of the strong-convexity of Ψ, then implies that $\|x^* - x_0^+\|^2 \le 2\mathcal{V}_0/\mu = [\mathcal{R}(0)]^2$ and hence that $\mathrm{Prob}[\bar{E}_0] = 0$. By (2.23), (2.42) and (2.36), we have, for any $1 \le k \le \mathcal{K}_N$,

$$\frac{2}{\mu \min\{1,\nu\}}\left[\hat{\mathcal{D}}_e^s(k) + \tilde{\lambda}\,\hat{\mathcal{D}}_p^s(k)\right] \le \frac{2}{\mu \min\{1,\nu\}}\left[\frac{\min\{1,\nu\}\mathcal{V}_0}{2^{k+1}} + \frac{\min\{1,\nu\}\mathcal{V}_0}{2^{k+1}}\right] = [\mathcal{R}(k)]^2,$$

which, in view of (2.35), implies that for any $1 \le k \le \mathcal{K}_N$,

$$\mathrm{Prob}\left\{\|x^* - x^+_{\mathcal{S}(k)}\|^2 \ge [\mathcal{R}(k)]^2 | E_{k-1}\right\} \le \exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda}).$$

Hence, for any $1 \le k \le \mathcal{K}_N$,

$$\begin{aligned}
\mathrm{Prob}\left\{\bar{E}_k\right\} &\le \mathrm{Prob}\left\{\|x^* - x^+_{\mathcal{S}(k)}\|^2 \ge [\mathcal{R}(k)]^2\right\} + \mathrm{Prob}\left\{\bar{E}_{k-1}\right\} \\
&\le \mathrm{Prob}\left\{\|x^* - x^+_{\mathcal{S}(k)}\|^2 \ge [\mathcal{R}(k)]^2 | E_{k-1}\right\} + 2\,\mathrm{Prob}\left\{\bar{E}_{k-1}\right\} \\
&\le \left[\exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda})\right] + 2\,\mathrm{Prob}\left\{\bar{E}_{k-1}\right\}.
\end{aligned}$$

Using the previous conclusion inductively and noting that $\mathrm{Prob}\left\{\bar{E}_0\right\} = 0$, we conclude

$$\mathrm{Prob}\left\{\bar{E}_{\mathcal{K}_N}\right\} \le \left[\exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda})\right]\sum_{k=0}^{\mathcal{K}_N - 1} 2^k \le \left[\exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda})\right] 2^{\mathcal{K}_N},$$

which together with (2.33), (2.43) and the fact that $2^{\mathcal{K}_N} \le 2N$ then imply that

$$\begin{aligned}
&\mathrm{Prob}\left\{\Psi(x_N^{ag}) - \Psi^* \ge \mathcal{D}_e(N) + \tilde{\lambda}\,\mathcal{D}_p^s(N)\right\} \\
&\le \mathrm{Prob}\left\{\Psi(x_N^{ag}) - \Psi^* \ge \mathcal{D}_e(N) + \tilde{\lambda}\,\mathcal{D}_p^s(N) | E_{\mathcal{K}_N}\right\} + \mathrm{Prob}\left\{\bar{E}_{\mathcal{K}_N}\right\} \\
&\le \left[\exp(-\tilde{\lambda}^2/3) + \exp(-\tilde{\lambda})\right](1 + 2^{\mathcal{K}_N}) \le \Lambda.
\end{aligned}$$

■

By using Proposition 5, we can estimate a bound on the number of iterations performed by the shrinking AC-SA algorithm to find an (ϵ, Λ)-solution $\bar{x} \in X$ such that $\mathrm{Prob}\{\Psi(\bar{x}) - \Psi^* < \epsilon\} \ge 1 - \Lambda$. For the sake of simplicity, let us focus on the dependence of this iteration-complexity bound on ϵ and Λ, while disregarding its dependence on other constants including $\mathcal{V}_0$, L, M, μ and σ. Clearly, by (2.43), we have $\tilde{\lambda} = \Omega\left(\log(N/\Lambda)\right)$. Also note that by (2.42), we have $\hat{S} = \Omega(\tilde{\lambda}^2)$. Using these two observations, (2.16), (2.34) and (2.44), we conclude that the total number of iterations performed by the shrinking AC-SA algorithm to find an (ϵ, Λ)-solution of (1.1) can be bounded by

$$\mathcal{O}\left\{\frac{1}{\epsilon}\left(\log\frac{1}{\epsilon\Lambda}\right)^2\right\}.$$

The above theoretical iteration-complexity bound is significantly better than the one in (2.20) in terms of its dependence on ϵ.

3 Optimal AC-SA algorithms for strongly convex SCO

In this section, we show that the generic AC-SA framework described in Subsection 2.1 can yield an optimal algorithm for solving strongly convex SCO problems. More specifically, we first introduce an AC-SA algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ and then present an optimal algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$ with $\mu>0$ obtained by properly restarting the AC-SA algorithms for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$. We also discuss how to improve the large-deviation properties associated with the optimal expected rate of convergence for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$.

Observe that, if μ is set to 0 in the generic AC-SA algorithm, then the identities (2.5) and (2.6), respectively, reduce to

$$x_t^{md} = (1-\alpha_t)x_{t-1}^{ag}+\alpha_t x_{t-1}, \tag{3.1}$$

$$x_t = \arg\min_{x\in X}\{\alpha_t[\langle G_t,x\rangle+\mathcal{X}(x)]+\gamma_t V(x_{t-1},x)\}. \tag{3.2}$$

For the sake of future reference, we call these algorithms *the AC-SA algorithms for* $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$. The following result regarding the AC-SA algorithms for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ has been shown in [9, 4].

Proposition 6 *Let $\{x_t^{ag}\}_{t\geq 1}$ be computed by the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ with*

$$\alpha_t=\frac{2}{t+1} \quad \textit{and} \quad \gamma_t=\frac{4\gamma}{\nu t(t+1)}, \quad \forall\, t\geq 1, \tag{3.3}$$

for some $\gamma\geq 2L$. Then, under Assumption A1, we have $\mathbb{E}[\Psi(x_t^{ag})-\Psi^*]\leq \mathcal{C}_{e,1}(t)$, $\forall t\geq 1$, where

$$\mathcal{C}_{e,1}(t)\equiv\mathcal{C}_{e,1}(x_0,\gamma,t):=\frac{4\gamma V(x_0,x^*)}{\nu t(t+1)}+\frac{8(M^2+\sigma^2)(t+1)}{3\gamma}. \tag{3.4}$$

If Assumption A2 holds, then, $\forall\lambda>0,\forall t\geq 1$,

$$\text{Prob}\{\Psi(x_t^{ag})-\Psi^*>\mathcal{C}_{e,1}(t)+\lambda\,\mathcal{C}_{p,1}(t)\}\leq \exp\{-\lambda^2/3\}+\exp\{-\lambda\}, \tag{3.5}$$

where

$$\mathcal{C}_{p,1}(t)\equiv\mathcal{C}_{p,1}(\gamma,t):=\frac{2\sigma R_X(x^*)}{\sqrt{3t}}+\frac{8\sigma^2(t+1)}{3\gamma}. \tag{3.6}$$

We demonstrated in [4] how to properly specify the value of γ in (3.3) so as to obtain an optimal algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$. Below we will use the aforementioned AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ as a subroutine for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$ with $\mu>0$. We observe that the stepsize parameter γ in (3.3) will assume different values than the one used in [4] and that the prox-function $V(x,z)$ in this algorithm is now assumed to grow quadratically.

A multi-stage AC-SA algorithm

0) Let a prox-function $V(x,z)$ satisfying condition (2.4), an point $p_0\in X$, and a bound $\mathcal{V}_0$ such that $\Psi(p_0)-\Psi(x^*)\leq\mathcal{V}_0$ be given.

1) For $k=1,2,\ldots$

1.a) Run N_k iterations of the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ with input $x_0 = p_{k-1}$, $V(x,z)$, and $\{\alpha_t\}_{t\geq 1}$ and $\{\gamma_t\}_{t\geq 1}$ set to (3.3) with $\gamma = \gamma_k$, where

$$N_k = \left\lceil \max\left\{4\sqrt{\frac{2L}{\nu\mu}}, \frac{128(M^2+\sigma^2)}{3\nu\mu\mathcal{V}_0 2^{-(k+1)}}\right\}\right\rceil, \tag{3.7}$$

$$\gamma_k = \max\left\{2L, \left[\frac{\nu\mu(M^2+\sigma^2)}{3\mathcal{V}_0 2^{-(k-1)}}N_k(N_k+1)(N_k+2)\right]^{\frac{1}{2}}\right\}; \tag{3.8}$$

1.b) Set $p_k = x^{ag}_{N_k}$, where $x^{ag}_{N_k}$ is the solution obtained in Step 1.a).

We say that a stage of the algorithm described above, referred to as *the multi-stage AC-SA*, occurs whenever the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$ is called in Step 1.a). Clearly, the kth stage of this algorithm consists of N_k iterations of the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,0)$, which are also called iterations of the multi-stage AC-SA algorithm for the sake of notational convenience. The following proposition summarizes the convergence properties of the multi-stage AC-SA algorithm.

Proposition 7 *Let $\{p_k\}_{k\geq 1}$ be computed by the multi-stage AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$. Then under Assumption A1,*

$$\mathbb{E}[\Psi(p_k) - \Psi^*] \leq \mathcal{V}_k \equiv \mathcal{V}_0 2^{-k}, \quad \forall\, k \geq 0. \tag{3.9}$$

As a consequence, the multi-stage AC-SA algorithm will find a solution $\bar{x} \in X$ of (1.1) such that $\mathbb{E}[\Psi(\bar{x}) - \Psi^*] \leq \epsilon$ for any $\epsilon \in (0, \mathcal{V}_0)$ in at most $K := \lceil \log \mathcal{V}_0/\epsilon \rceil$ stages. Moreover, the total number of iterations performed by this algorithm to find such a solution is bounded by $\mathcal{O}\left(\mathcal{T}_1(\epsilon)\right)$, where

$$\mathcal{T}_1(\epsilon) := \sqrt{\frac{L}{\nu\mu}}\max\left(1, \log\frac{\mathcal{V}_0}{\epsilon}\right) + \frac{M^2+\sigma^2}{\nu\mu\epsilon}. \tag{3.10}$$

Proof. We first show that (3.9) holds by using induction. Clearly (3.9) holds for $k = 0$. Assume that for some $k \geq 1$, $\mathbb{E}[\Psi(p_{k-1}) - \Psi^*] \leq \mathcal{V}_{k-1} = \mathcal{V}_0 2^{-(k-1)}$. This assumption together with (1.2) and (2.4) clearly imply that

$$\mathbb{E}[V(p_{k-1}, x^*)] \leq \mathbb{E}\left[\frac{1}{2}\|p_{k-1} - x^*\|^2\right] \leq \mathbb{E}\left[\frac{\Psi(p_{k-1}) - \Psi^*}{\mu}\right] \leq \frac{\mathcal{V}_{k-1}}{\mu}. \tag{3.11}$$

Also note that by the definitions of N_k and $\mathcal{V}_k$, respectively, in (3.7) and (3.9), we have

$$\mathcal{Q}_1(N_k) \equiv \frac{8L\mathcal{V}_{k-1}}{\mu\nu N_k(N_k+1)} \leq \frac{8L\mathcal{V}_{k-1}}{\mu\nu N_k^2} = \frac{16L\mathcal{V}_k}{\mu\nu N_k^2} \leq \frac{1}{2}\mathcal{V}_k, \tag{3.12}$$

$$\mathcal{Q}_2(N_k) \equiv \frac{(M^2+\sigma^2)(N_k+2)\mathcal{V}_{k-1}}{3\nu\mu N_k(N_k+1)} \leq \frac{2(M^2+\sigma^2)\mathcal{V}_{k-1}}{3\nu\mu N_k} \leq \frac{\mathcal{V}_0 2^{-(k+1)}\mathcal{V}_{k-1}}{64} = \frac{\mathcal{V}_k^2}{64}. \tag{3.13}$$

We then conclude from Proposition 6, (3.8), (3.11), (3.12) and (3.13) that

$$\begin{aligned}
\mathbb{E}[\Psi(p_k) - \Psi^*] &\leq \frac{4\gamma_k\mathbb{E}[V(p_{k-1},x^*)]}{\nu N_k(N_k+1)} + \frac{4(M^2+\sigma^2)(N_k+2)}{3\gamma_k} \\
&\leq \frac{4\gamma_k\mathcal{V}_{k-1}}{\mu\nu N_k(N_k+1)} + \frac{4(M^2+\sigma^2)(N_k+2)}{3\gamma_k} \\
&\leq \max\left\{\mathcal{Q}_1(N_k), 4\sqrt{\mathcal{Q}_2(N_k)}\right\} + 4\sqrt{\mathcal{Q}_2(N_k)} \leq \mathcal{V}_k.
\end{aligned}$$

We have thus shown that (3.9) holds. Now suppose that that the multi-stage AC-SA algorithm is run for K stages. By (3.9), we have $\mathbb{E}[\Psi(p_K) - \Psi^*] \leq \mathcal{V}_0 2^{-K} \leq \mathcal{V}_0 2^{\log \frac{\epsilon}{\mathcal{V}_0}} = \epsilon$. Moreover, it follows from (3.7) that the total number of iterations can be bounded by

$$\begin{aligned}
\sum_{k=1}^{K} N_k &\leq \sum_{k=1}^{K} \left[4\sqrt{\frac{2L}{\nu\mu}} + \frac{128(M^2+\sigma^2)}{3\nu\mu\mathcal{V}_0 2^{-(k+1)}} + 1 \right] \\
&= K\left(4\sqrt{\frac{2L}{\nu\mu}} + 1 \right) + \frac{128(M^2+\sigma^2)}{3\nu\mu\mathcal{V}_0} \sum_{k=1}^{K} 2^{k+1} \leq K\left(4\sqrt{\frac{2L}{\nu\mu}} + 1 \right) + \frac{128(M^2+\sigma^2)}{3\nu\mu\mathcal{V}_0} 2^{K+2} \\
&\leq \left(4\sqrt{\frac{2L}{\nu\mu}} + 1 \right) \left\lceil \log \frac{\mathcal{V}_0}{\epsilon} \right\rceil + \frac{384(M^2+\sigma^2)}{\nu\mu\epsilon},
\end{aligned}$$

which clearly implies bound (3.10). ∎

A few remarks about the results in Proposition 7 are in place. First, in view of (1.3), the multi-stage AC-SA algorithm achieves the optimal expected rate of convergence for solving $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, \mu)$. Note that, since $\mathcal{V}_0$ only appears inside the logarithmic term of (3.10), the selection of the initial point $p_0 \in X$ has little affect on the efficiency of this algorithm. Second, suppose that we run the multi-stage AC-SA algorithm for $K_\Lambda := \lceil \log \mathcal{V}_0/(\Lambda\epsilon) \rceil$ stages for a given confidence level $\Lambda \in (0,1)$. Then, by (3.9) and Markov's inequality, we have $\text{Prob}[\Psi(p_{K_\Lambda}) - \Psi^* > \epsilon] \leq \mathbb{E}[\Psi(p_{K_\Lambda}) - \Psi^*]/\epsilon \leq \Lambda$, which implies that the total number of iterations performed by the multi-stage AC-SA algorithm for finding an (ϵ, Λ)-solution of (1.1) can be bounded by $\mathcal{O}\left(\mathcal{T}_1(\Lambda\epsilon)\right)$.

Let us suppose now that Assumption A2 holds. Similar to the shrinking AC-SA algorithm in Section 2.2, we introduce a shrinking multi-stage AC-SA algorithm which has a significantly better iteration-complexity bound in terms of the dependence on Λ, for finding an (ϵ, Λ)-solution of (1.1) than the previous multi-stage AC-SA algorithm. It is worth noting that the AC-SA algorithms for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, 0)$ with stepsize policy either (3.3) or a different stepsize policy in [4] can be used here to update iterate p_k of the shrinking multi-stage AC-SA, although we focus on the former one in this paper. Also note that it is possible to relax the assumption that the prox-function grows quadratically. We make this assumption here only for the sake of simplicity.

The shrinking multi-stage AC-SA:

0) Let a prox-function $V(x, z)$ satisfying condition (2.4), an point $p_0 \in X$, and a bound $\mathcal{V}_0$ such that $\Psi(p_0) - \Psi(x^*) \leq \mathcal{V}_0$ be given. Set $\bar{K} := \lceil \log(2\mathcal{V}_0/\epsilon) \rceil$ and $\lambda := \lambda(\bar{K}) > 0$ such that $\exp\{-\lambda^2/3\} + \exp\{-\lambda\} \leq \Lambda/\bar{K}$.

1) For $k = 1, \ldots, \bar{K}$

   1.a) Run $\hat{N}_k$ iterations of the AC-SA algorithm for $\mathcal{F}_{X,\mathcal{X}}(L, M, \sigma, 0)$ applied to $\min_{x \in \hat{X}_k} \{\Psi(x)\}$,

with input $x_0 = p_{k-1}$, $V(z,x)$, and $\{\alpha_t\}_{t\geq 1}$ and $\{\gamma_t\}_{t\geq 1}$ set to (3.3) with $\gamma = \hat{\gamma}_k$. Here,

$$\hat{X}_k := \left\{x \in X : \|x - p_{k-1}\|^2 \leq \hat{R}_{k-1}^2 := \frac{2\mathcal{V}_0}{\mu 2^{k-1}}\right\}, \tag{3.14}$$

$$\hat{N}_k = \left\lceil \max\left\{8\sqrt{\frac{L}{\nu\mu}}, \frac{128\max\left\{4(M^2+\sigma^2), \lambda^2\sigma^2(1+\sqrt{\nu})^2\right\}}{3\nu\mu\mathcal{V}_0 2^{-(k+1)}}\right\}\right\rceil, \tag{3.15}$$

$$\hat{\gamma}_k = \max\left\{2L, \left[\frac{\nu\mu(M^2+\sigma^2)}{3\mathcal{V}_0 2^{-(k-1)}}\hat{N}_k(\hat{N}_k+1)(\hat{N}_k+2)\right]^{\frac{1}{2}}\right\}; \tag{3.16}$$

1.b) Set $p_k = x_{\hat{N}_k}^{ag}$, where $x_{\hat{N}_k}^{ag}$ is the solution obtained in Step 1.a).

Note that in the shrinking multi-stage AC-SA algorithm, the stage limit $\bar{K}$ is computed for a given accuracy ϵ. The value of $\bar{K}$ is then used in the computation of $\lambda(\bar{K})$ and subsequently in $\hat{N}_k$ and $\hat{\gamma}_k$ (c.f. (3.15) and (3.16)). This is in contrast to the multi-stage AC-SA algorithm without shrinkage, in which the definitions of N_k and γ_k in (3.7) and (3.8) do not depend on the target accuracy ϵ.

The following result shows some convergence properties of the shrinking multi-stage AC-SA algorithm.

Lemma 8 *Let $\{p_k\}_{k\geq 1}$ be computed by the shrinking multi-stage AC-SA algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$. Also for any $k \geq 0$, let $\mathcal{V}_k \equiv \mathcal{V}_0 2^{-k}$ and denote the event $A_k := \{\Psi(p_k) - \Psi^* \leq \mathcal{V}_k\}$. Then under Assumption A2,*

$$\text{Prob}[\Psi(p_k) - \Psi^* \geq \mathcal{V}_k | A_{k-1}] \leq \frac{\Lambda}{\bar{K}}, \quad \forall\, 1 \leq k \leq \bar{K}. \tag{3.17}$$

Proof. By the conditional assumption in (3.17), we have $\Psi(p_{k-1}) - \Psi^* \leq \mathcal{V}_{k-1}$, which together with the strong-convexity of f and the definition of $\hat{R}_{k-1}$ in (3.14) imply that

$$\|p_{k-1} - x^*\|^2 \leq \frac{2[\Psi(p_{k-1}) - \Psi^*]}{\mu} \leq \frac{2\mathcal{V}_{k-1}}{\mu} = \hat{R}_{k-1}^2. \tag{3.18}$$

Hence, the restricted problem $\min_{x\in\hat{X}_k}\{\Psi(x)\}$ has the same solution as (1.1). We then conclude from Proposition 6 applied to the previous restricted problem that

$$\text{Prob}[\Psi(p_k) - \Psi^* > \hat{\mathcal{C}}_{e,1}(\hat{N}_k) + \lambda\hat{\mathcal{C}}_{p,1}(\hat{N}_k)|A_{k-1}] \leq \exp\{-\lambda^2/3\} + \exp\{-\lambda\} \leq \frac{\Lambda}{\bar{K}}, \tag{3.19}$$

where

$$\hat{\mathcal{C}}_{e,1}(\hat{N}_k) := \frac{4\hat{\gamma}_k V(p_{k-1},x^*)}{\nu\hat{N}_k(\hat{N}_k+1)} + \frac{4(M^2+\sigma^2)(\hat{N}_k+2)}{3\hat{\gamma}_k} \text{ and } \hat{\mathcal{C}}_{p,1}(\hat{N}_k) := \frac{2\sigma R_{X_k}(x^*)}{\sqrt{3\hat{N}_k}} + \frac{4\sigma^2(\hat{N}_k+2)}{3\hat{\gamma}_k}.$$

Note that by (2.4) and (3.18), $V(p_{k-1},x^*) \leq \|p_{k-1}-x^*\|^2/2 \leq \mathcal{V}_{k-1}/\mu$. Now let $\mathcal{Q}_1(\cdot)$ and $\mathcal{Q}_2(\cdot)$ be defined in (3.12) and (3.13), respectively. Note that by the definition of $\hat{N}_k$ in (3.15), we have

$\mathcal{Q}_1(\hat{N}_k) \le \mathcal{V}_k/4$ and $\mathcal{Q}_2(\hat{N}_k) \le \mathcal{V}_k^2/256$. Using the previous observations and the definition of $\hat{\gamma}_k$ in (3.16), we obtain

$$\begin{aligned}\hat{\mathcal{C}}_{e,1}(\hat{N}_k) &\le \frac{4\hat{\gamma}_k\mathcal{V}_{k-1}}{\mu\nu\hat{N}_k(\hat{N}_k+1)} + \frac{4(M^2+\sigma^2)(\hat{N}_k+2)}{3\hat{\gamma}_k} \\ &\le \max\left\{\mathcal{Q}_1(\hat{N}_k), 4\sqrt{\mathcal{Q}_2(\hat{N}_k)}\right\} + 4\sqrt{\mathcal{Q}_2(\hat{N}_k)} \le \frac{\mathcal{V}_k}{2}. \end{aligned} \tag{3.20}$$

Moreover, note that by (3.14) and (3.18), we have for any $x \in X_k$,

$$\|x - p_{k-1}\| \le \sqrt{\frac{2\mathcal{V}_{k-1}}{\mu}} \quad \text{and} \quad \|x - x^*\| \le \|x - p_{k-1}\| + \|p_{k-1} - x^*\| \le 2\sqrt{\frac{2\mathcal{V}_{k-1}}{\mu}},$$

and hence that $R_{X_k}(x^*) \le 2\sqrt{2\mathcal{V}_{k-1}/\mu}$, which together with (3.15) and (3.16) then imply that

$$\begin{aligned}\hat{\mathcal{C}}_{p,1}(\hat{N}_k) &\le 4\sigma\sqrt{\frac{2\mathcal{V}_{k-1}}{3\mu\hat{N}_k}} + 4\left[\frac{\sigma^2(\hat{N}_k+2)\mathcal{V}_{k-1}}{3\nu\mu\hat{N}_k(\hat{N}_k+1)}\right]^{\frac{1}{2}} \le 4\sigma\sqrt{\frac{2\mathcal{V}_{k-1}}{3\mu\hat{N}_k}} + 4\sigma\sqrt{\frac{2\mathcal{V}_{k-1}}{3\nu\mu\hat{N}_k}} \\ &= 4\sigma\left(1+\frac{1}{\sqrt{\nu}}\right)\sqrt{\frac{2\mathcal{V}_{k-1}}{3\mu\hat{N}_k}} \le \frac{\sqrt{\mathcal{V}_{k-1}\mathcal{V}_0 2^{-(k+1)}}}{2\lambda} = \frac{\mathcal{V}_k}{2\lambda}. \end{aligned} \tag{3.21}$$

Combining (3.19), (3.20) and (3.21), we obtain (3.17). ∎

The following proposition establishes the iteration complexity of the shrinking multi-stage AC-SA algorithm.

Proposition 9 *Let $\{p_k\}_{k\ge1}$ be computed by the shrinking multi-stage AC-SA algorithm for solving $\mathcal{F}_{X,\mathcal{X}}(L,M,\sigma,\mu)$. Then under Assumption A2, we have*

$$\text{Prob}[\Psi(p_{\bar{K}}) - \Psi^* > \epsilon] \le \Lambda. \tag{3.22}$$

Moreover, the total number of iterations performed by the algorithm to find such a solution is bounded by $\mathcal{O}\left(\mathcal{T}_2(\epsilon,\Lambda)\right)$, where

$$\mathcal{T}_2(\epsilon,\Lambda) := \sqrt{\frac{L}{\nu\mu}}\max\left(1, \log\frac{\mathcal{V}_0}{\epsilon}\right) + \frac{M^2+\sigma^2}{\nu\mu\epsilon} + \left[\ln\frac{\log(\mathcal{V}_0/\epsilon)}{\Lambda}\right]^2\frac{\sigma^2(1+\sqrt{\nu})^2}{\nu\mu\epsilon}. \tag{3.23}$$

Proof. Denote $\mathcal{V}_k = \mathcal{V}_0 2^{-k}$. Let A_k denote the event of $\{\Psi(p_k) - \Psi^* \le \mathcal{V}_k\}$ and $\bar{A}_k$ be its complement. Clearly, we have $\text{Prob}(A_0) = 1$. It can also be easily seen that

$$\begin{aligned}\text{Prob}[\Psi(p_k) - \Psi^* > 2\mathcal{V}_k] &\le \text{Prob}[\Psi(p_k) - \Psi^* > 2\mathcal{V}_k | A_{k-1}] + \text{Prob}[\bar{A}_{k-1}] \\ &\le \frac{\Lambda}{\bar{K}} + \text{Prob}[\Psi(p_{k-1}) - \Psi^* > 2\mathcal{V}_{k-1}], \quad \forall\, 1 \le k \le \bar{K}\end{aligned}$$

where the last inequality follows from Lemma 8 and the definition of $\bar{A}_{k-1}$. Summing up both sides of the above inequality from $k = 1$ to $\bar{K}$, we obtain (3.22). Now, by (3.15), the total number of

AC-SA iterations can be bounded by

$$\begin{aligned}\sum_{k=1}^{\bar{K}}\hat{N}_k &\leq \sum_{k=1}^{\bar{K}}\left\{8\sqrt{\frac{L}{\nu\mu}}+\frac{128\max\left\{4(M^2+\sigma^2),\lambda^2\sigma^2(1+\sqrt{\nu})^2\right\}}{3\nu\mu\mathcal{V}_0 2^{-(k+1)}}+1\right\}\\ &= \bar{K}\left(8\sqrt{\frac{L}{\nu\mu}}+1\right)+\frac{128\max\left\{4(M^2+\sigma^2),\lambda^2\sigma^2(1+\sqrt{\nu})^2\right\}}{3\nu\mu\mathcal{V}_0}\sum_{k=1}^{\bar{K}}2^{k+1}\\ &\leq \bar{K}\left(8\sqrt{\frac{L}{\nu\mu}}+1\right)+\frac{128\max\left\{4(M^2+\sigma^2),\lambda^2\sigma^2(1+\sqrt{\nu})^2\right\}}{3\nu\mu\mathcal{V}_0}2^{\bar{K}+2}.\end{aligned}$$

Using the above conclusion, the fact that $\bar{K}=\lceil\log(2\mathcal{V}_0/\epsilon)\rceil$, the observation that $\lambda=\mathcal{O}\{\log(\bar{K}/\Lambda)\}$ and (3.23), we conclude that the total number of AC-SA iterations is bounded by $\mathcal{O}\left(\mathcal{T}_2(\epsilon,\lambda)\right)$. ∎

4 Numerical Results

In this section, we present the results of our computational experiments carried out with the on-line Ridge regression problem, i.e.,

$$\min_{x\in\mathbb{R}^d}\mathbb{E}_{u,v}[(\langle x,u\rangle-v)^2]+\rho\|x\|_2^2, \tag{4.1}$$

where $\rho>0$ is a user-defined parameter. Moreover, we assume that u is uniformly distributed over $[0,1]^d$ and v is given by $v=\langle\bar{x},u\rangle+\xi$, where $\xi\sim N(0,\tilde{\sigma})$ is a random variable independent of u, and $\langle\bar{x},\cdot\rangle$ is the genuine linear relation between u and v. Note that, for a given training data sample $S=\{(u_i,v_i)\}_{i=1}^m$, associate with (4.1) is a batch-learning model (or sample average approximation) given by

$$\min_{x\in\mathbb{R}^d}\frac{1}{m}\sum_{i=1}^m(\langle x,u_i\rangle-v_i)^2+\rho\|x\|_2^2. \tag{4.2}$$

Observe that the explicit solution of (4.2) is readily given by its first-order optimality condition. However, assembling the Hessian of the objective function of (4.2) and computing its inverse can be computationally expensive if both d and m is large. Moreover, this approach is not applicable to the on-line learning setting, see for example [23], the theoretical comparison of on-line and batch learning.

We compare the single-stage AC-SA algorithms, the multi-stage AC-SA algorithms, the classic SA (see, e.g., [13]), the mirror descent SA ([9]) and the batch-learning approach in the following way. At first, we use an initial i.i.d. sample for all the algorithms to estimate problem parameters, namely, L, σ and μ (note that $M=0$ and $\mathcal{X}(\cdot)=0$), whenever these parameters are needed. Second, fixing an i.i.d. sample (of size N) for the random variable (u,v), we apply the aforementioned methods to obtain approximate solutions for problem (4.1), and then the quality of the solutions yielded by these algorithms is evaluated using another i.i.d. sample of size $k>>N$ (by using the sample average approximation). Also we use $\omega(x)=\|x\|_2^2/2$ as the prox function in our experiments. Note that we used the enhanced version of mirror descent SA in [9] in view of the fact that the objective function of (4.1) is smooth.

Our first experiment is to compare the solution quality of these algorithms and their sensitivity with respect to the parameter ρ and noise level $\tilde{\sigma}$. More specifically, we consider six different groups.

Table 1: Problem instances

Group	Instance	d	$\tilde{\sigma}$	ρ	L	μ
group 1	Reg-11	20	2	1	12.43	2.08
	Reg-12			0.1	10.63	0.28
	Reg-13			0.01	10.45	0.10
group 2	Reg-21	20	5	1	11.96	2.09
	Reg-22			0.1	10.16	0.29
	Reg-23			0.01	9.98	0.11
group 3	Reg-31	100	2	1	52.02	2.02
	Reg-32			0.1	50.28	0.22
	Reg-33			0.01	50.04	0.04
group 4	Reg-41	100	5	1	52.66	2.02
	Reg-42			0.1	50.86	0.22
	Reg-43			0.01	50.68	0.04
group 5	Reg-51	400	2	1	204.08	2.00
	Reg-52			0.1	202.28	0.20
	Reg-53			0.01	202.10	0.02
group 6	Reg-61	400	5	1	202.07	2.00
	Reg-62			0.1	200.28	0.20
	Reg-63			0.01	200.09	0.02

Within each group, we generated three instances having the same problem size (d) and different noise levels ($\tilde{\sigma}$). Also, the parameter ρ is set to $\rho = 1, 0.1$ and 0.01 for each group. Table 1 shows these problem instances and estimation of problem parameters, namely, L and μ, for each instance, which are computed for sample average approximation problem (4.2) with $m = 200$. Then, we run the algorithms for each problem instance with different number of iterations $N = 2,000$ and $N = 4,000$ and evaluate the solution quality with $K = 30,000$. Note that in these experiments, we fix the initial point $x_0 = R * \underline{x}$ for all the algorithms, where $\underline{x}$ is randomly chosen from the box $[0, 1]^d$ and R is set to 10. Our algorithms are implemented in MATLAB R2007b and the CPU times for the classic SA, the mirror descent SA and the AC-SA algorithms are all comparable. We repeated each run of the algorithms fifty times and then take the average of results as shown in Table 2 through Table 4. The following conclusions regarding the comparison of the solution quality can be drawn from these numerical results.

- **AC-SA vs. classic SA:** the AC-SA algorithms substantially outperform the classic SA algorithm for all test instances. In particular, the classic SA is much more sensitive to ρ than the AC-SA algorithms. As shown in Tables 2-4, the classic SA may actually diverge when d increases or as ρ decreases.

- **AC-SA vs. mirror descent SA:** the solution quality of the AC-SA algorithms is at least comparable to that of the mirror descent SA algorithm for all the test instances. It can considerably outperform the latter approach when d increases and/or ρ decreases (see Table 4). One possible explanation is that the AC-SA algorithm can handle more effectively the smooth component of the objective function, which plays a more significant role when d increases (the Lipschitz constant becomes larger) and/or as ρ decreases (the strong convexity vanishes).

Table 2: Numerical results for group 1 and group 2 ($d = 20$)

Number of iterations: $N = 2,000$						
	Reg-11	Reg-12	Reg-13	Reg-21	Reg-22	Reg-23
Classic SA	37.47	152.12	4.67e+5	1025.91	48.46	5.02e+5
Mirror descent SA	8.19	4.62	4.25	29.47	26.08	26.41
Single-stage AC-SA	8.19	4.64	4.27	29.55	26.04	25.83
Shrinking Single-stage AC-SA	8.18	4.60	4.23	29.55	25.98	26.17
Multi-stage AC-SA	8.24	5.06	4.42	29.62	26.53	27.34
Shrinking multi-stage AC-SA	8.27	5.02	4.75	29.70	26.94	31.61
Batch-learning	8.19	4.58	4.10	29.45	25.71	25.27
Number of iterations: $N = 4,000$						
Classic SA	34.32	134.08	3.62e+5	961.43	47.40	4.13e+5
Mirror descent SA	8.18	4.58	4.15	29.50	25.77	25.71
Single-stage AC-SA	8.19	4.59	4.15	29.46	25.82	25.65
Shrinking Single-stage AC-SA	8.19	4.58	4.17	29.47	25.80	25.59
Multi-stage AC-SA	8.20	4.64	4.24	29.54	26.10	26.75
Shrinking multi-stage AC-SA	8.23	4.83	4.45	29.57	26.18	28.55
Batch-learning	8.18	4.56	4.08	29.48	25.67	25.10

- **AC-SA vs. batch-learning:** the solution quality of the AC-SA algorithms are comparable to that of the batch-learning approach in many cases. When ρ is really small or the variance of noise is big, the solution quality of the AC-SA algorithms is worse than that of batch-learning approach (see Table 4). This can be viewed as an indication that more iterations should be run for the AC-SA algorithms. It should also be noted that the AC-SA algorithms usually outperform other SA algorithms whenever these situations happen.

- **Different variants of AC-SA:** Firstly, the multi-stage AC-SA algorithm is as good as the single-stage AC-SA in many cases. However, it requires the estimation of more problem parameters, such as σ and $\mathcal{V}_0$. Moreover, no significant difference between the AC-SA algorithms and their shrinking counterparts is observed except for one instance. A plausible explanation is that the iterates for the AC-SA algorithm (without shrinkage) get clustered in a smaller and smaller area. In that case, shrinking the domain does not affect the execution of the algorithms in practice, although it does help to significantly improve the theoretical complexity bounds. Finally, by increasing number of iterations, one can improve in the solution quality of the AC-SA algorithms. Such improvement is most noticeable in Table 4, where the obtained solutions are still not too close to the optimal ones since d is relatively large.

In the second experiment, we investigate the sensitivity of different algorithms with respect to the selection of the initial solution. We set the initial point $x_0 = R * \underline{x}$ as before, but change R from 10 to $5,000$. A group of test results for the instance Reg-42 with $N = 2,000$ is shown in Figure 1, where the vertical axis represents the final objective value and the horizontal axis displays the values of R. As it can be easily seen, there is no significant difference in final solutions of multi-stage AC-SA algorithms when the initial solution changes. For single stage algorithms, the final solutions become worse when R increases. Both the classic SA and mirror descent SA are very sensitive to

Table 3: Numerical results for group 3 and group 4 ($d = 100$)

Number of iterations: $N = 2,000$						
	Reg-31	Reg-32	Reg-33	Reg-41	Reg-42	Reg-43
Classic SA	774.31	4450.35	5.28e+14	914.20	5497.25	1.33e+15
Mirror descent SA	30.65	8.66	8.80	43.48	30.53	34.28
Single stage AC-SA	30.49	7.55	6.79	43.32	30.41	33.67
Shrinking single-stage AC-SA	30.51	7.46	6.66	43.50	30.29	35.87
Multi-stage AC-SA	30.56	8.65	5.61	44.44	34.03	30.81
Shrinking multi-stage AC-SA	30.71	11.39	6.71	47.47	40.92	31.36
Batch-learning	30.45	7.13	4.52	43.15	27.78	26.35
Number of iterations: $N = 4,000$						
Classic SA	748.86	4243.66	4.72e+14	867.83	5382.95	9.50e+14
Mirror descent SA	30.49	7.52	5.74	43.20	28.55	28.96
Single-stage AC-SA	30.46	7.24	5.46	43.19	28.41	32.97
Shrinking Single-stage AC-SA	30.46	7.22	5.59	43.26	28.78	32.58
Multi-stage AC-SA	30.47	7.85	6.53	43.95	30.31	33.02
Shrinking multi-stage AC-SA	30.55	9.06	13.89	45.26	34.37	35.83
Batch-learning	30.43	7.07	4.44	43.08	27.45	25.87

Table 4: Numerical results for group 5 and group 6 ($d = 400$)

Number of iterations: $N = 2,000$						
	Reg-51	Reg-52	Reg-53	Reg-61	Reg-62	Reg-63
Classic SA	Inf	Inf	Inf	Inf	Inf	Inf
Mirror descent SA	121.52	108.65	182.53	139.35	125.34	209.14
Single-stage AC-SA	109.41	18.96	54.02	128.50	49.25	84.13
Shrinking single-stage AC-SA	109.24	18.99	85.18	128.59	47.64	120.82
Multi-stage AC-SA	109.67	24.36	47.57	130.55	70.46	76.58
Shrinking multi-stage AC-SA	110.41	24.51	47.62	134.52	111.50	75.63
Batch-learning	108.92	16.29	5.97	127.39	38.63	31.37
Number of iterations: $N = 4,000$						
Classic SA	1.38e+8	Inf	Inf	Inf	Inf	Inf
Mirror descent SA	95.83	37.09	67.68	130.29	61.20	92.32
Single-stage AC-SA	92.86	15.40	24.53	127.61	44.11	66.62
Shrinking single-stage AC-SA	92.85	15.35	46.64	127.47	41.89	103.40
Multi-stage AC-SA	93.03	17.50	9.57	128.90	58.23	41.00
Shrinking multi-stage AC-SA	93.44	20.18	9.50	132.40	56.42	40.29
Batch-learning	92.73	14.40	5.55	127.15	37.60	28.66

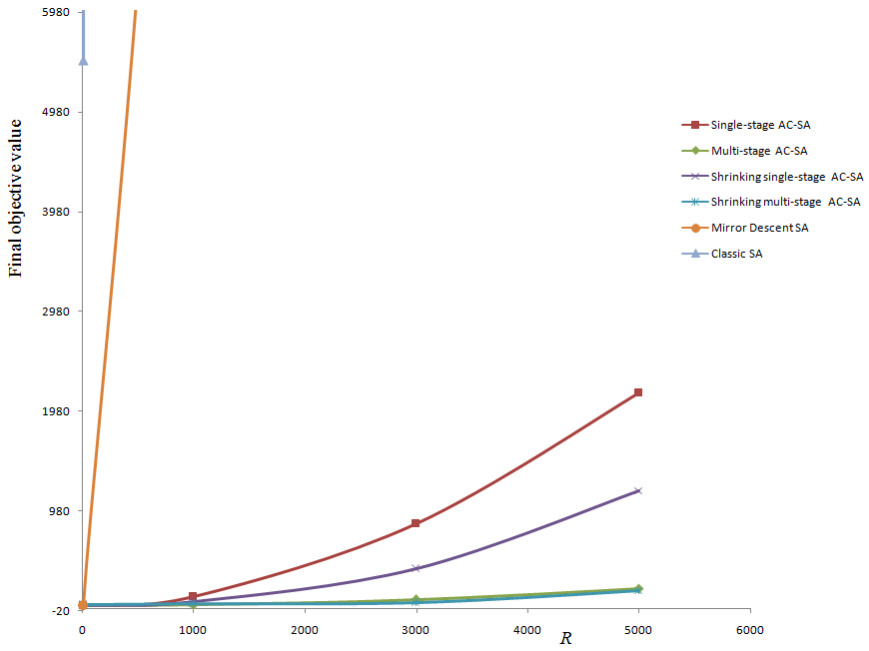

Figure 1: Stability of algorithms with respect to initial solutions

the selection of the initial solution. In particular,the classic SA does not even converge in the case of bad initial solutions. It should be mentioned that such phenomenon has been consistently observed for all the test instances in Table 1, although we only show the results of one representative instance.

In the third experiment, we compute the online lower bounds for the optimal value according to the validation analysis presented in [4]. Table 5 shows these lower bounds for the single stage AC-SA algorithm at the $t = 1,000$, $2,000$, $3,000$ and $4,000$ iteration. As it can be seen for this table, these lower bounds converge to the optimal values. Moreover, Figure 2 shows the convergence of online lower and upper bounds (see Section 5 of [4]) for the single stage AC-SA algorithm applied to two representative instances. It should be noted that we have not used a trivial lower bound, i.e., 0, on the optimal value of (4.1) in order to demonstrate the convergence behaviour of these online lower and upper bounds.

Acknowledgement: The authors would like to express sincere appreciation to Professor Arkadi Nemirovski for some discussions on the complexity issues for stochastic optimization. The authors are very grateful to the associate editor and two anonymous referees for their very useful suggestions for improving the paper.

References

[1] A. Benveniste, M. Métivier, and P. Priouret. *Algorithmes adaptatifs et approximations stochastiques*. Masson, 1987. English translation: *Adaptive Algorithms and Stochastic Approximations*, Springer Verlag (1993).

Table 5: Online lower bounds for the single stage AC-SA algorithm

Instance	Iteration			
	$1,000$	$2,000$	$3,000$	$4,000$
Reg-31	27.18	29.29	30.02	30.14
Reg-32	1.66	3.97	5.17	6.01
Reg-33	-28.27	-3.77	-0.54	1.41
Reg-41	39.20	41.44	41.69	42.13
Reg-42	4.72	18.87	22.15	23.52
Reg-43	-124.39	-6.41	-3.71	11.86
Reg-51	35.67	76.96	85.51	88.43
Reg-52	-112.37	-18.86	-1.49	4.88
Reg-53	-2488.30	-402.00	-195.28	-65.03
Reg-61	65.02	110.36	116.31	122.45
Reg-62	-97.86	-11.03	9.74	18.89
Reg-63	-4137.60	-514.71	-250.43	-156.67

[2] Y. Ermoliev. Stochastic quasigradient methods and their application to system optimization. *Stochastics*, 9:1–36, 1983.

[3] A. Gaivoronski. Nonstationary stochastic programming problems. *Kybernetika*, 4:89–92, 1978.

[4] S. Ghadimi and G. Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, I: a generic algorithmic framework. Manuscript, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, USA, 2010. Submitted to *SIAM Journal on Optimization*.

[5] C. Hu, J. T. Kwok, and W. Pan. Accelerated gradient methods for stochastic optimization and online learning. In *NIPS*, 2009.

[6] A. Juditsky and Y. Nesterov. Primal-dual subgradient methods for minimizing uniformly convex functions. Manuscript.

[7] A. J. Kleywegt, A. Shapiro, and T. Homem de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12:479–502, 2001.

[8] H. J. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*, volume 35 of *Applications of Mathematics*. Springer-Verlag, New York, 2003.

[9] G. Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 2010. Forthcoming, online first, DOI: 10.1007/s10107-010-0434-y.

[10] G. Lan, A. Nemirovski, and A. Shapiro. Validation analysis of mirror descent stochastic approximation method. *Mathematical Programming*, 2011. Forthcoming, online first.

[11] S. Lee and S. Wright. Manifold identification in dual averaging for regularized stochastic online learning. Manuscript, University of Wisconsin-Madison, Wisconsin, July 2011.

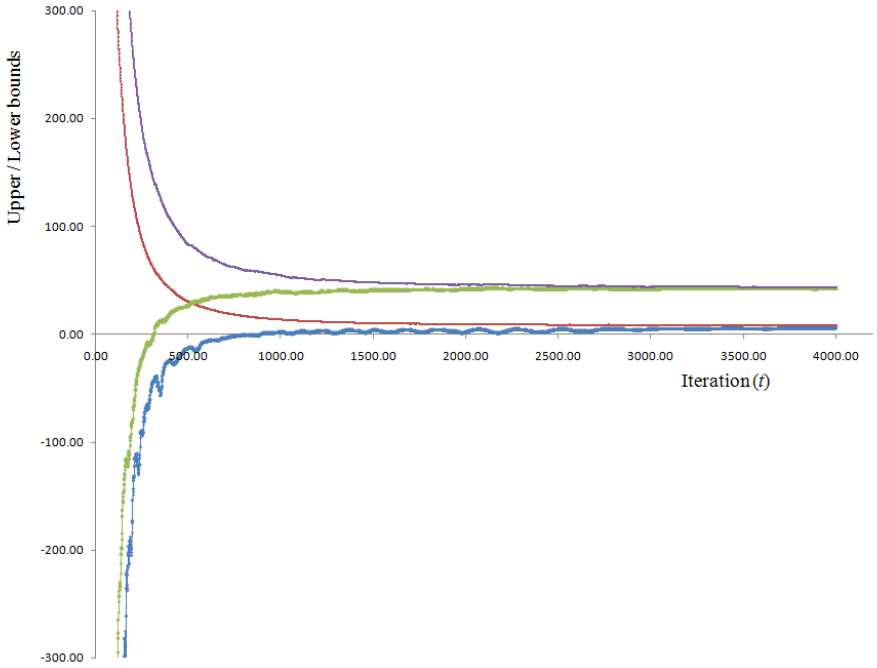

Figure 2: Convergence of online lower and upper bounds for the single-stage AC-SA algorithm: The first set of two curves (green and magenta) are the results for the instance Reg-41 converging to the optimal value near 43; The second set of two curves (blue and red) are the results for the instance Reg-32 converging to the optimal value near 7.

[12] Q. Lin, X. Chen, and J. Peña. A sparsity preserving stochastic gradient method for composite optimization. Manuscript, Carnegie Mellon University, PA 15213, April 2011.

[13] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19:1574–1609, 2009.

[14] A. Nemirovski and D. Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience Series in Discrete Mathematics. John Wiley, XV, 1983.

[15] Y. E. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. *Doklady AN SSSR*, 269:543–547, 1983.

[16] Y. E. Nesterov. *Introductory Lectures on Convex Optimization: a basic course*. Kluwer Academic Publishers, Massachusetts, 2004.

[17] G.C. Pflug. Optimization of stochastic models. In *The Interface Between Simulation and Optimization*. Kluwer, Boston, 1996.

[18] B.T. Polyak. New stochastic approximation type procedures. *Automat. i Telemekh.*, 7:98–107, 1990.

[19] B.T. Polyak and A.B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM J. Control and Optimization*, 30:838–855, 1992.

[20] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[21] A. Ruszczyński and W. Sysk. A method of aggregate stochastic subgradients with on-line stepsize rules for convex stochastic programming problems. *Mathematical Programming Study*, 28:113–131, 1986.

[22] A. Shapiro. Monte carlo sampling methods. In A. Ruszczyński and A. Shapiro, editors, *Stochastic Programming*. North-Holland Publishing Company, Amsterdam, 2003.

[23] S. Smale and Y. Yao. Online learning algorithms. *Found. Comp. Math*, 6:145–170, 2005.

[24] J.C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. John Wiley, Hoboken, NJ, 2003.

[25] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *J. Mach. Learn. Res.*, pages 2543–2596, 2010.

[26] D.B. Yudin and A.S. Nemirovskii. Computational complexity of strictly convex programming. *Eknomika i Matematicheskie Metody*, 3:550–569, 1977.