SIAM J. OPTIM.
Vol. 19, No. 4, pp. 1574–1609

ROBUST STOCHASTIC APPROXIMATION APPROACH TO STOCHASTIC PROGRAMMING*

A. NEMIROVSKI†, A. JUDITSKY‡, G. LAN†, AND A. SHAPIRO†

Abstract. In this paper we consider optimization problems where the objective function is given in a form of the expectation. A basic difficulty of solving such stochastic optimization problems is that the involved multidimensional integrals (expectations) cannot be computed with high accuracy. The aim of this paper is to compare two computational approaches based on Monte Carlo sampling techniques, namely, the stochastic approximation (SA) and the sample average approximation (SAA) methods. Both approaches, the SA and SAA methods, have a long history. Current opinion is that the SAA method can efficiently use a specific (say, linear) structure of the considered problem, while the SA approach is a crude subgradient method, which often performs poorly in practice. We intend to demonstrate that a properly modified SA approach can be competitive and even significantly outperform the SAA method for a certain class of convex stochastic problems. We extend the analysis to the case of convex-concave stochastic saddle point problems and present (in our opinion highly encouraging) results of numerical experiments.

Key words. stochastic approximation, sample average approximation method, stochastic programming, Monte Carlo sampling, complexity, saddle point, minimax problems, mirror descent algorithm

AMS subject classifications. 90C15, 90C25

DOI. 10.1137/070704277

1. Introduction.

In this paper we first consider the following stochastic optimization problem:

$$\min_{x\in X}\big\{f(x)=\mathbb{E}[F(x,\xi)]\big\}, \tag{1.1}$$

and then we deal with an extension of the analysis to stochastic saddle point problems. Here $X\subset\mathbb{R}^n$ is a nonempty bounded closed convex set, ξ is a random vector whose probability distribution P is supported on set $\Xi\subset\mathbb{R}^d$ and $F: X\times\Xi\to\mathbb{R}$. We assume that the expectation

$$\mathbb{E}[F(x,\xi)]=\textstyle\int_{\Xi}F(x,\xi)dP(\xi) \tag{1.2}$$

is well defined and finite valued for every $x\in X$. Moreover, we assume that the expected value function $f(\cdot)$ is *continuous* and *convex* on X. Of course, if for every $\xi\in\Xi$ the function $F(\cdot,\xi)$ is convex on X, then it follows that $f(\cdot)$ is convex. With these assumptions, (1.1) becomes a convex programming problem.

A basic difficulty of solving stochastic optimization problem (1.1) is that the multidimensional integral (expectation) (1.2) cannot be computed with a high accuracy for dimension d, say, greater than five. The aim of this paper is to compare two

*Received by the editors October 1, 2007; accepted for publication (in revised form) August 26, 2008; published electronically January 21, 2009.

http://www.siam.org/journals/siopt/19-4/70427.html

†Georgia Institute of Technology, Atlanta, Georgia 30332 (nemirovs@isye.gatech.edu, glan@isye.gatech.edu, ashapiro@isye.gatech.edu). Research of the first author was partly supported by NSF award DMI-0619977. Research of the third author was partially supported by NSF award CCF-0430644 and ONR award N00014-05-1-0183. Research of the fourth author was partly supported by NSF awards DMS-0510324 and DMI-0619977.

‡Université J. Fourier, B.P. 53, 38041 Grenoble Cedex 9, France (Anatoli.Juditsky@imag.fr).

computational approaches based on Monte Carlo sampling techniques, namely, the *stochastic approximation* (SA) and the *sample average approximation* (SAA) methods. To this end we make the following assumptions.

(A1) It is possible to generate an independent identically distributed (iid) sample $\xi_1, \xi_2, \ldots$, of realizations of random vector ξ.

(A2) There is a mechanism (an oracle), which, for a given input point $(x, \xi) \in X \times \Xi$ returns *stochastic subgradient*—a vector $\mathsf{G}(x, \xi)$ such that $\mathsf{g}(x) := \mathbb{E}[\mathsf{G}(x, \xi)]$ is well defined and is a subgradient of $f(\cdot)$ at x, i.e., $\mathsf{g}(x) \in \partial f(x)$.

Recall that if $F(\cdot, \xi)$, $\xi \in \Xi$, is convex and $f(\cdot)$ is finite valued in a neighborhood of a point x, then (cf. Strassen [28])

$$\partial f(x) = \mathbb{E}\left[\partial_x F(x, \xi)\right]. \tag{1.3}$$

In that case we can employ a measurable selection $\mathsf{G}(x, \xi) \in \partial_x F(x, \xi)$ as a stochastic subgradient. At this stage, however, this is not important, we shall see later other relevant ways for constructing stochastic subgradients.

Both approaches, the SA and SAA methods, have a long history. The SA method is going back to the pioneering paper by Robbins and Monro [21]. Since then SA algorithms became widely used in stochastic optimization (see, e.g., [3, 6, 7, 20, 22] and references therein) and, due to especially low demand for computer memory, in signal processing . In the classical analysis of the SA algorithm (it apparently goes back to the works [5] and [23]) it is assumed that $f(\cdot)$ is twice continuously differentiable and strongly convex and in the case when the minimizer of f belongs to the interior of X, exhibits asymptotically optimal rate[1] of convergence $\mathbb{E}[f(x_t) - f_*] = O(t^{-1})$ (here x_t is tth iterate and f_* is the minimal value of $f(x)$ over $x \in X$). This algorithm, however, is very sensitive to a choice of the respective stepsizes. Since "asymptotically optimal" stepsize policy can be very bad in the beginning, the algorithm often performs poorly in practice (e.g., [27, section 4.5.3.]).

An important improvement of the SA method was developed by Polyak [18] and Polyak and Juditsky [19], where longer stepsizes were suggested with consequent averaging of the obtained iterates. Under the outlined "classical" assumptions, the resulting algorithm exhibits the same optimal $O(t^{-1})$ asymptotical convergence rate, while using an easy to implement and "robust" stepsize policy. It should be mentioned that the main ingredients of Polyak's scheme—long steps and averaging—were, in a different form, proposed already in Nemirovski and Yudin [15] for the case of problems (1.1) with general-type Lipschitz continuous convex objectives and for convex-concave saddle point problems. The algorithms from [15] exhibit, in a nonasymptotical fashion, the $O(t^{-1/2})$ rate of convergence. It is possible to show that in the general convex case (without assuming smoothness and strong convexity of the objective function), this rate of $O(t^{-1/2})$ is unimprovable. For a summary of early results in this direction, see Nemirovski and Yudin [16].

The SAA approach was used by many authors in various contexts under different names. Its basic idea is rather simple: generate a (random) sample $\xi_1, \ldots, \xi_N$, of size N, and approximate the "true" problem (1.1) by the sample average problem

$$\min_{x \in X} \left\{ \hat{f}_N(x) = N^{-1} \sum_{j=1}^{N} F(x, \xi_j) \right\}. \tag{1.4}$$

[1] Throughout the paper, we speak about convergence in terms of the objective value.

Note that the SAA method is not an algorithm; the obtained SAA problem (1.4) still has to be solved by an appropriate numerical procedure. Recent theoretical studies (cf. [11, 25, 26]) and numerical experiments (see, e.g., [12, 13, 29]) show that the SAA method coupled with a good (deterministic) algorithm could be reasonably efficient for solving certain classes of two-stage stochastic programming problems. On the other hand, classical SA-type numerical procedures typically performed poorly for such problems.

We intend to demonstrate in this paper that a properly modified SA approach can be competitive and even significantly outperform the SAA method for a certain class of stochastic problems. The mirror descent SA method we propose here is a direct descendent of the stochastic mirror descent method of Nemirovski and Yudin [16]. However, the method developed in this paper is more flexible than its "ancestor": the iteration of the method is exactly the prox-step for a chosen prox-function, and the choice of prox-type function is not limited to the norm-type distance-generating functions. Close techniques, based on subgradient averaging, have been proposed in Nesterov [17] and used in [10] to solve the stochastic optimization problem (1.1). Moreover, the results on large deviations of solutions and applications of the mirror descent SA to saddle point problems, to the best of our knowledge, are new.

The rest of this paper is organized as follows. In section 2 we focus on theory of the SA method applied to (1.1). We start with outlining the relevant–to-our-goals part of the classical "$O(t^{-1})$" SA theory (section 2.1), along with its "$O(t^{-1/2})$" modifications (section 2.2). Well-known and simple results presented in these sections pave the road to our main developments carried out in section 2.3. In section 3 we extend the constructions and results of section 2.3 to the case of the convex-concave stochastic saddle point problem. In concluding section 4 we present results (in our opinion, highly encouraging) of numerical experiments with the SA algorithm (sections 2.3 and 3) applied to large-scale stochastic convex minimization and saddle point problems. Section 5 gives a short conclusion for the presented results. Finally, some technical proofs are given in the appendix.

Throughout the paper, we use the following notation. By $\|x\|_p$, we denote the ℓ_p norm of vector $x \in \mathbb{R}^n$, in particular, $\|x\|_2 = \sqrt{x^T x}$ denotes the Euclidean norm, and $\|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\}$. By Π_X, we denote the metric projection operator onto the set X, that is, $\Pi_X(x) = \arg\min_{x' \in X} \|x - x'\|_2$. Note that Π_X is a nonexpanding operator, i.e.,

$$\|\Pi_X(x') - \Pi_X(x)\|_2 \le \|x' - x\|_2 \ \ \forall x', x \in \mathbb{R}^n. \tag{1.5}$$

By $O(1)$, we denote positive absolute constants. The notation $\lfloor a \rfloor$ stands for the largest integer less than or equal to $a \in \mathbb{R}$ and $\lceil a \rceil$ for the smallest integer greater than or equal to $a \in \mathbb{R}$. By $\xi_{[t]} = (\xi_1, \ldots, \xi_t)$, we denote the history of the process $\xi_1, \ldots,$ up to time t. Unless stated otherwise, all relations between random variables are supposed to hold almost surely.

2. Stochastic approximation, basic theory. In this section we discuss theory and implementations of the SA approach to the minimization problem (1.1).

2.1. Classical SA algorithm. The classical SA algorithm solves (1.1) by mimicking the simplest subgradient descent method. That is, for chosen $x_1 \in X$ and a sequence $\gamma_j > 0$, $j = 1, \ldots,$ of stepsizes, it generates the iterates by the formula

$$x_{j+1} = \Pi_X(x_j - \gamma_j \mathsf{G}(x_j, \xi_j)). \tag{2.1}$$

Of course, the crucial question of that approach is how to choose the stepsizes γ_j. Let x_* be an optimal solution of (1.1). Note that since the set X is compact and $f(x)$ is continuous, (1.1) has an optimal solution. Note also that the iterate $x_j = x_j(\xi_{[j-1]})$ is a function of the history $\xi_{[j-1]} = (\xi_1, \ldots, \xi_{j-1})$ of the generated random process and hence is random.

Denote

$$A_j = \tfrac{1}{2}\|x_j - x_*\|_2^2 \quad \text{and} \quad a_j = \mathbb{E}[A_j] = \tfrac{1}{2}\mathbb{E}\left[\|x_j - x_*\|_2^2\right]. \tag{2.2}$$

By using (1.5) and since $x_* \in X$ and hence $\Pi_X(x_*) = x_*$, we can write

$$\begin{aligned} A_{j+1} &= \tfrac{1}{2}\left\|\Pi_X(x_j - \gamma_j \mathsf{G}(x_j, \xi_j)) - x_*\right\|_2^2 \\ &= \tfrac{1}{2}\left\|\Pi_X(x_j - \gamma_j \mathsf{G}(x_j, \xi_j)) - \Pi_X(x_*)\right\|_2^2 \\ &\le \tfrac{1}{2}\left\|x_j - \gamma_j \mathsf{G}(x_j, \xi_j) - x_*\right\|_2^2 \\ &= A_j + \tfrac{1}{2}\gamma_j^2 \|\mathsf{G}(x_j, \xi_j)\|_2^2 - \gamma_j (x_j - x_*)^T \mathsf{G}(x_j, \xi_j). \end{aligned} \tag{2.3}$$

Since $x_j = x_j(\xi_{[j-1]})$ is independent of ξ_j, we have

$$\begin{aligned} \mathbb{E}\left[(x_j - x_*)^T \mathsf{G}(x_j, \xi_j)\right] &= \mathbb{E}\left\{\mathbb{E}\left[(x_j - x_*)^T \mathsf{G}(x_j, \xi_j) \,|\xi_{[j-1]}\right]\right\} \\ &= \mathbb{E}\left\{(x_j - x_*)^T \mathbb{E}\left[\mathsf{G}(x_j, \xi_j) \,|\xi_{[j-1]}\right]\right\} \\ &= \mathbb{E}\left[(x_j - x_*)^T \mathsf{g}(x_j)\right]. \end{aligned} \tag{2.4}$$

Assume now that there is a positive number M such that

$$\mathbb{E}\left[\|\mathsf{G}(x, \xi)\|_2^2\right] \le M^2 \quad \forall x \in X. \tag{2.5}$$

Then, by taking expectation of both sides of (2.3) and using (2.4), we obtain

$$a_{j+1} \le a_j - \gamma_j \mathbb{E}\left[(x_j - x_*)^T \mathsf{g}(x_j)\right] + \tfrac{1}{2}\gamma_j^2 M^2. \tag{2.6}$$

Suppose further that the expectation function $f(x)$ is differentiable and strongly convex on X, i.e., there is constant $c > 0$ such that

$$f(x') \ge f(x) + (x' - x)^T \nabla f(x) + \tfrac{1}{2}c\|x' - x\|_2^2, \quad \forall x', x \in X,$$

or equivalently that

$$(x' - x)^T (\nabla f(x') - \nabla f(x)) \ge c\|x' - x\|_2^2 \quad \forall x', x \in X. \tag{2.7}$$

Note that strong convexity of $f(x)$ implies that the minimizer x_* is unique. By optimality of x_*, we have that

$$(x - x_*)^T \nabla f(x_*) \ge 0 \quad \forall x \in X,$$

which together with (2.7) implies that $(x - x_*)^T \nabla f(x) \ge c\|x - x_*\|_2^2$. In turn, it follows that $(x - x_*)^T g \ge c\|x - x_*\|_2^2$ for all $x \in X$ and $g \in \partial f(x)$, and hence

$$\mathbb{E}\left[(x_j - x_*)^T \mathsf{g}(x_j)\right] \ge c\mathbb{E}\left[\|x_j - x_*\|_2^2\right] = 2ca_j.$$

Therefore, it follows from (2.6) that

$$a_{j+1} \le (1 - 2c\gamma_j)a_j + \tfrac{1}{2}\gamma_j^2 M^2. \tag{2.8}$$

Let us take stepsizes $\gamma_j = \theta/j$ for some constant $\theta > 1/(2c)$. Then, by (2.8), we have

$$a_{j+1} \leq (1 - 2c\theta/j)a_j + \tfrac{1}{2}\theta^2 M^2/j^2.$$

It follows by induction that

$$\mathbb{E}\left[\|x_j - x_*\|_2^2\right] = 2a_j \leq Q(\theta)/j, \tag{2.9}$$

where

$$Q(\theta) = \max\left\{\theta^2 M^2 (2c\theta - 1)^{-1}, \|x_1 - x_*\|_2^2\right\}. \tag{2.10}$$

Suppose further that x_* is an *interior* point of X and $\nabla f(x)$ is Lipschitz continuous, i.e., there is constant $L > 0$ such that

$$\|\nabla f(x') - \nabla f(x)\|_2 \leq L\|x' - x\|_2 \ \ \forall x', x \in X. \tag{2.11}$$

Then

$$f(x) \leq f(x_*) + \tfrac{1}{2}L\|x - x_*\|_2^2, \ \ \forall x \in X, \tag{2.12}$$

and hence

$$\mathbb{E}\big[f(x_j) - f(x_*)\big] \leq \tfrac{1}{2}L\mathbb{E}\left[\|x_j - x_*\|_2^2\right] \leq \tfrac{1}{2}LQ(\theta)/j, \tag{2.13}$$

where $Q(\theta)$ is defined in (2.10).

Under the specified assumptions, it follows from (2.9) and (2.13), respectively, that after t iterations, the expected error of the current solution in terms of the distance to x_* is of order $O(t^{-1/2})$, and the expected error in terms of the objective value is of order $O(t^{-1})$, provided that $\theta > 1/(2c)$. The simple example of $X = \{x : \|x\|_2 \leq 1\}$, $f(x) = \frac{1}{2}cx^T x$, and $\mathsf{G}(x, \xi) = \nabla f(x) + \xi$, with ξ having standard normal distribution $\mathcal{N}(0, I_n)$, demonstrates that the outlined upper bounds on the expected errors are tight within factors independent of t.

We have arrived at the $O(t^{-1})$ rate of convergence in terms of the expected value of the objective mentioned in the Introduction. Note, however, that the result is highly sensitive to a priori information on c. What would happen if the parameter c of strong convexity is overestimated? As a simple example, consider $f(x) = x^2/10$, $X = [-1, 1] \subset \mathbb{R}$, and assume that there is no noise, i.e., $\mathsf{G}(x, \xi) \equiv \nabla f(x)$. Suppose, further that we take $\theta = 1$ (i.e., $\gamma_j = 1/j$), which will be the optimal choice for $c = 1$, while actually here $c = 0.2$. Then the iteration process becomes

$$x_{j+1} = x_j - f'(x_j)/j = \left(1 - \frac{1}{5j}\right) x_j,$$

and hence starting with $x_1 = 1$,

$$\begin{aligned} x_j = \prod_{s=1}^{j-1}\left(1 - \frac{1}{5s}\right) &= \exp\left\{-\sum_{s=1}^{j-1} \ln\left(1 + \frac{1}{5s-1}\right)\right\} > \exp\left\{-\sum_{s=1}^{j-1}\frac{1}{5s-1}\right\} \\ &> \exp\left\{-\left(0.25 + \int_1^{j-1}\frac{1}{5t-1}dt\right)\right\} > \exp\left\{-0.25 + 0.2\ln 1.25 - \frac{1}{5}\ln j\right\} \\ &> 0.8 j^{-1/5}. \end{aligned}$$

That is, the convergence is extremely slow. For example, for $j = 10^9$, the error of the iterated solution is greater than 0.015. On the other hand, for the optimal stepsize factor of $\theta = 1/c = 5$, the optimal solution $x_* = 0$ is found in one iteration.

It could be added that the stepsizes $\gamma_j = \theta/j$ may become completely unacceptable when f loses strong convexity. For example, when $f(x) = x^4$, $X = [-1, 1]$, and there is no noise, these stepsizes result in a disastrously slow convergence: $|x_j| \geq O([\ln(j+1)]^{-1/2})$. The precise statement here is that with $\gamma_j = \theta/j$ and $0 < x_1 \leq \frac{1}{6\sqrt{\theta}}$, we have that $x_j \geq \frac{x_1}{\sqrt{1+32\theta x_1^2[1+\ln(j+1)]}}$ for $j = 1, 2, \ldots$.

We see that in order to make the SA "robust"—applicable to general convex objectives rather than to strongly convex ones—one should replace the classical stepsizes $\gamma_j = O(j^{-1})$, which can be too small to ensure a reasonable rate of convergence even in the "no noise" case, with "much larger" stepsizes. At the same time, a detailed analysis shows that "large" stepsizes poorly suppress noise. As early as in [15] it was realized that in order to resolve the arising difficulty, it makes sense to separate collecting information on the objective from generating approximate solutions. Specifically, we can use large stepsizes, say, $\gamma_j = O(j^{-1/2})$ in (2.1), thus avoiding too slow motion at the cost of making the trajectory "more noisy." In order to suppress, to some extent, this noisiness, we take, as approximate solutions, appropriate averages of the search points x_j rather than these points themselves.

2.2. Robust SA approach. Results of this section go back to Nemirovski and Yudin [15, 16]. Let us look again at the basic relations (2.2), (2.5), and (2.6). By convexity of $f(x)$, we have that $f(x) \geq f(x_t) + (x - x_t)^T \mathsf{g}(x_t)$ for any $x \in X$, and hence

$$\mathbb{E}\big[(x_t - x_*)^T \mathsf{g}(x_t)\big] \geq \mathbb{E}\big[f(x_t) - f(x_*)\big].$$

Together with (2.6), this implies (recall that $a_t = \mathbb{E}\left[\frac{1}{2}\|x_t - x_*\|_2^2\right]$)

$$\gamma_t \mathbb{E}\big[f(x_t) - f(x_*)\big] \leq a_t - a_{t+1} + \tfrac{1}{2}\gamma_t^2 M^2.$$

It follows that whenever $1 \leq i \leq j$, we have

$$\sum_{t=i}^{j} \gamma_t \mathbb{E}\big[f(x_t) - f(x_*)\big] \leq \sum_{t=i}^{j} [a_t - a_{t+1}] + \tfrac{1}{2}M^2 \sum_{t=i}^{j} \gamma_t^2 \leq a_i + \tfrac{1}{2}M^2 \sum_{t=i}^{j} \gamma_t^2, \tag{2.14}$$

and hence, setting $\nu_t = \frac{\gamma_t}{\sum_{\tau=i}^{j} \gamma_\tau}$,

$$\mathbb{E}\left[\sum_{t=i}^{j} \nu_t f(x_t) - f(x_*)\right] \leq \frac{a_i + \frac{1}{2}M^2 \sum_{t=i}^{j} \gamma_t^2}{\sum_{t=i}^{j} \gamma_t}. \tag{2.15}$$

Note that $\nu_t \geq 0$ and $\sum_{t=i}^{j} \nu_t = 1$. Consider the points

$$\tilde{x}_i^j = \sum_{t=i}^{j} \nu_t x_t, \tag{2.16}$$

and let

$$D_X = \max_{x \in X} \|x - x_1\|_2. \tag{2.17}$$

By convexity of X, we have $\tilde{x}_i^j \in X$, and, by convexity of f, we have $f(\tilde{x}_i^j) \leq \sum_{t=i}^j \nu_t f(x_t)$. Thus, by (2.15) and in view of $a_1 \leq D_X^2$ and $a_i \leq 4D_X^2$, $i > 1$, we get

$$\begin{aligned} &\text{(a)} \quad \mathbb{E}\left[f(\tilde{x}_1^j) - f(x_*)\right] \leq \frac{D_X^2 + M^2 \sum_{t=1}^j \gamma_t^2}{2\sum_{t=1}^j \gamma_t} \quad \text{for} \quad 1 \leq j, \\ &\text{(b)} \quad \mathbb{E}\left[f(\tilde{x}_i^j) - f(x_*)\right] \leq \frac{4D_X^2 + M^2 \sum_{t=i}^j \gamma_t^2}{2\sum_{t=i}^j \gamma_t} \quad \text{for} \quad 1 < i \leq j. \end{aligned} \tag{2.18}$$

Based on the resulting bounds on the expected inaccuracy of approximate solutions $\tilde{x}_i^j$, we can now develop "reasonable" stepsize policies along with the associated efficiency estimates.

Constant stepsizes and basic efficiency estimate. Assume that the number N of iterations of the method is fixed in advance and that $\gamma_t = \gamma$, $t = 1, \ldots, N$. Then it follows by (2.18(a)) that

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \leq \frac{D_X^2 + M^2 N \gamma^2}{2N\gamma}. \tag{2.19}$$

Minimizing the right-hand side of (2.19) over $\gamma > 0$, we arrive at the *constant* stepsize policy

$$\gamma_t = \frac{D_X}{M\sqrt{N}}, \quad t = 1, \ldots, N, \tag{2.20}$$

along with the associated efficiency estimate

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \leq \frac{D_X M}{\sqrt{N}}. \tag{2.21}$$

With the constant stepsize policy (2.20), we also have, for $1 \leq K \leq N$,

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right) - f(x_*)\right] \leq \frac{D_X M}{\sqrt{N}} \left[\frac{2N}{N - K + 1} + \frac{1}{2}\right]. \tag{2.22}$$

When $K/N \leq 1/2$, the right-hand side of (2.22) coincides, within an absolute constant factor, with the right-hand side of (2.21). Finally, for a constant $\theta > 0$, passing from the stepsizes (2.20) to the stepsizes

$$\gamma_t = \frac{\theta D_X}{M\sqrt{N}}, \quad t = 1, \ldots, N, \tag{2.23}$$

the efficiency estimate becomes

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right) - f(x_*)\right] \leq \max\left\{\theta, \theta^{-1}\right\} \frac{D_X M}{\sqrt{N}} \left[\frac{2N}{N - K + 1} + \frac{1}{2}\right], \ 1 \leq K \leq N. \tag{2.24}$$

Discussion. We conclude that the expected error in terms of the objective of *Robust SA* algorithm (2.1), (2.16), with constant stepsize policy (2.20), after N iterations is of order $O(N^{-1/2})$ in our setting. Of course, this is worse than the rate $O(N^{-1})$ for the classical SA algorithm as applied to a smooth strongly convex function attaining minimum at a point from the interior of the set X. However, the error bounds (2.21)

and (2.22) are guaranteed independently of any smoothness and/or strong convexity assumptions on f. All that matters is the convexity of f on the convex compact set X and the validity of (2.5). Moreover, scaling the stepsizes by positive constant θ affects the error bound (2.24) *linearly* in $\max\{\theta, \theta^{-1}\}$. This can be compared with a possibly disastrous effect of such scaling in the classical SA algorithm discussed in section 2.1. These observations, in particular the fact that there is no necessity in "fine tuning" the stepsizes to the objective function f, explain the adjective "robust" in the name of the method. Finally, it can be shown that without additional, as compared to convexity and (2.5), assumptions on f, the accuracy bound (2.21) within an absolute constant factor is the best one allowed by statistics (cf. [16]).

Varying stepsizes. When the number of steps is not fixed in advance, it makes sense to replace constant stepsizes with the stepsizes

$$\gamma_t = \frac{\theta D_X}{M\sqrt{t}}, \quad t = 1, 2, \ldots . \tag{2.25}$$

From (2.18(b)) it follows that with this stepsize policy, one has, for $1 \le K \le N$,

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right) - f(x_*)\right] \le \frac{D_X M}{\sqrt{N}} \left[\frac{2}{\theta}\left(\frac{N}{N-K+1}\right) + \frac{\theta}{2}\sqrt{\frac{N}{K}}\right]. \tag{2.26}$$

Choosing K as a fixed fraction of N, i.e., setting $K = \lceil rN \rceil$, with a fixed $r \in (0,1)$, we get the efficiency estimate

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right) - f(x_*)\right] \le C(r) \max\left\{\theta, \theta^{-1}\right\} \frac{D_X M}{\sqrt{N}}, \quad N = 1, 2, \ldots, \tag{2.27}$$

with an easily computable factor $C(r)$ depending solely on r. This bound, up to a factor depending solely on r and θ, coincides with the bound (2.21), with the advantage that our new stepsize policy should not be adjusted to a fixed-in-advance number of steps N.

2.3. Mirror descent SA method. On a close inspection, the robust SA algorithm from section 2.2 is intrinsically linked to the Euclidean structure of $\mathbb{R}^n$. This structure plays the central role in the very construction of the method (see (2.1)), the same as in the associated efficiency estimates, like (2.21) (since the quantities D_X, M participating in the estimates are defined in terms of the Euclidean norm, see (2.17) and (2.5)). By these reasons, from now on, we refer to the algorithm from section 2.2 as the (robust) *Euclidean SA* (E-SA). In this section we develop a substantial generalization of the E-SA approach allowing us to adjust, to some extent, the method to the geometry, not necessary Euclidean, of the problem in question. We shall see in the meantime that we can gain a lot, both theoretically and numerically, from such an adjustment. A rudimentary form of the generalization to follow can be found in Nemirovski and Yudin [16], from where the name "mirror descent" originates.

Let $\|\cdot\|$ be a (general) norm on $\mathbb{R}^n$ and $\|x\|_* = \sup_{\|y\|\le 1} y^T x$ be its dual norm. We say that a function $\omega : X \to \mathbb{R}$ is a *distance-generating function* modulus $\alpha > 0$ with respect to $\|\cdot\|$, if ω is convex and continuous on X, the set

$$X^o = \{x \in X : \partial\omega(x) \ne \emptyset\} \tag{2.28}$$

is convex (note that X^o always contains the relative interior of X) and restricted to X^o, ω is continuously differentiable and strongly convex with parameter α with

respect to $\|\cdot\|$, i.e.,

$$(x'-x)^T(\nabla\omega(x')-\nabla\omega(x)) \geq \alpha\|x'-x\|^2 \ \ \forall x', x \in X^o. \tag{2.29}$$

A simple example of a distance-generating function is $\omega(x) = \frac{1}{2}\|x\|_2^2$ (modulus 1 with respect to $\|\cdot\|_2$, $X^o = X$).

Let us define function $V : X^o \times X \to \mathbb{R}_+$ as follows:

$$V(x,z) = \omega(z) - \left[\omega(x) + \nabla\omega(x)^T(z-x)\right]. \tag{2.30}$$

In what follows we shall refer to $V(\cdot,\cdot)$ as *prox-function* associated with distance-generating function $\omega(x)$ (it is also called Bregman distance [4]). Note that $V(x,\cdot)$ is nonnegative and is a strongly convex modulus α with respect to the norm $\|\cdot\|$. Let us define *prox-mapping* $P_x : \mathbb{R}^n \to X^o$, associated with ω and a point $x \in X^o$, viewed as a parameter, as follows:

$$P_x(y) = \arg\min_{z\in X}\left\{y^T(z-x) + V(x,z)\right\}. \tag{2.31}$$

Observe that the minimum in the right-hand side of (2.31) is attained since ω is continuous on X and X is compact, and all the minimizers belong to X^o, whence the minimizer is unique, since $V(x,\cdot)$ is strongly convex on X^o. Thus, the prox-mapping is well defined.

For $\omega(x) = \frac{1}{2}\|x\|_2^2$, we have $P_x(y) = \Pi_X(x-y)$ so that (2.1) is the recurrence

$$x_{j+1} = P_{x_j}(\gamma_j \mathsf{G}(x_j,\xi_j)), \quad x_1 \in X^o. \tag{2.32}$$

Our goal is to demonstrate that the main properties of the recurrence (2.1) (which from now on we call the E-SA recurrence) are inherited by (2.32), *whatever be the underlying distance-generating function* $\omega(x)$.

The statement of the following lemma is a simple consequence of the optimality conditions of the right-hand side of (2.31) (proof of this lemma is given in the appendix).

LEMMA 2.1. *For every $u \in X, x \in X^o$, and $y \in \mathbb{R}^n$, one has*

$$V(P_x(y),u) \leq V(x,u) + y^T(u-x) + \frac{\|y\|_*^2}{2\alpha}. \tag{2.33}$$

Using (2.33) with $x = x_j$, $y = \gamma_j \mathsf{G}(x_j,\xi_j)$, and $u = x_*$, we get

$$\gamma_j(x_j - x_*)^T\mathsf{G}(x_j,\xi_j) \leq V(x_j,x_*) - V(x_{j+1},x_*) + \frac{\gamma_j^2}{2\alpha}\|\mathsf{G}(x_j,\xi_j)\|_*^2. \tag{2.34}$$

Note that with $\omega(x) = \frac{1}{2}\|x\|_2^2$, one has $V(x,z) = \frac{1}{2}\|x-z\|_2^2$, $\alpha = 1$, $\|\cdot\|_* = \|\cdot\|_2$. That is, (2.34) becomes nothing but the relation (2.6), which played a crucial role in all the developments related to the E-SA method. We are about to process, in a completely similar fashion, the relation (2.34) in the case of a general distance-generating function, thus arriving at the mirror descent SA. Specifically, setting

$$\Delta_j = \mathsf{G}(x_j,\xi_j) - \mathsf{g}(x_j), \tag{2.35}$$

we can rewrite (2.34), with j replaced by t, as

$$\gamma_t(x_t - x_*)^T\mathsf{g}(x_t) \leq V(x_t,x_*) - V(x_{t+1},x_*) - \gamma_t\Delta_t^T(x_t - x_*) + \frac{\gamma_t^2}{2\alpha}\|\mathsf{G}(x_t,\xi_t)\|_*^2. \tag{2.36}$$

Summing up over $t = 1, \ldots, j$, and taking into account that $V(x_{j+1}, u) \geq 0$, $u \in X$, we get

$$\sum_{t=1}^{j} \gamma_t (x_t - x_*)^T \mathsf{g}(x_t) \leq V(x_1, x_*) + \sum_{t=1}^{j} \frac{\gamma_t^2}{2\alpha} \|\mathsf{G}(x_t, \xi_t)\|_*^2 - \sum_{t=1}^{j} \gamma_t \Delta_t^T (x_t - x_*). \tag{2.37}$$

Setting $\nu_t = \frac{\gamma_t}{\sum_{i=1}^{j} \gamma_i}$, $t = 1, \ldots, j$, and

$$\tilde{x}_1^j = \sum_{t=1}^{j} \nu_t x_t \tag{2.38}$$

and invoking convexity of $f(\cdot)$, we have

$$\begin{aligned} \sum_{t=1}^{j} \gamma_t (x_t - x_*)^T \mathsf{g}(x_t) &\geq \sum_{t=1}^{j} \gamma_t \left[f(x_t) - f(x_*) \right] \\ &= \left(\sum_{t=1}^{j} \gamma_t \right) \left[\sum_{t=1}^{j} \nu_t f(x_t) - f(x_*) \right] \\ &\geq \left(\sum_{t=1}^{j} \gamma_t \right) \left[f(\tilde{x}_j) - f(x_*) \right], \end{aligned}$$

which combines with (2.37) to imply that

$$f(\tilde{x}_1^j) - f(x_*) \leq \frac{V(x_1, x_*) + \sum_{t=1}^{j} \frac{\gamma_t^2}{2\alpha} \|\mathsf{G}(x_t, \xi_t)\|_*^2 - \sum_{t=1}^{j} \gamma_t \Delta_t^T (x_t - x_*)}{\sum_{t=1}^{j} \gamma_t}. \tag{2.39}$$

Let us suppose, as in the previous section (cf. (2.5)), that we are given a positive number M_* such that

$$\mathbb{E}\left[\|\mathsf{G}(x, \xi)\|_*^2 \right] \leq M_*^2 \ \ \forall x \in X. \tag{2.40}$$

Taking expectations of both sides of (2.39) and noting that (i) x_t is a deterministic function of $\xi_{[t-1]} = (\xi_1, \ldots, \xi_{t-1})$, (ii) conditional on $\xi_{[t-1]}$, the expectation of Δ_t is 0, and (iii) the expectation of $\|\mathsf{G}(x_t, \xi_t)\|_*^2$ does not exceed M_*^2, we obtain

$$\mathbb{E}\left[f(\tilde{x}_1^j) - f(x_*) \right] \leq \frac{\max_{u \in X} V(x_1, u) + (2\alpha)^{-1} M_*^2 \sum_{t=1}^{j} \gamma_t^2}{\sum_{t=1}^{j} \gamma_t}. \tag{2.41}$$

Assume from now on that the method starts with the minimizer of ω:

$$x_1 = \operatorname{argmin}_X \omega(x).$$

Then, from (2.30), it follows that

$$\max_{z \in X} V(x_1, z) \leq D_{\omega, X}^2, \tag{2.42}$$

where

$$D_{\omega, X} := \left[\max_{z \in X} \omega(z) - \min_{z \in X} \omega(z) \right]^{1/2}. \tag{2.43}$$

Consequently, (2.41) implies that

$$\mathbb{E}\left[f(\tilde{x}_1^j) - f(x_*) \right] \leq \frac{D_{\omega, X}^2 + \frac{1}{2\alpha} M_*^2 \sum_{t=1}^{j} \gamma_t^2}{\sum_{t=1}^{j} \gamma_t}. \tag{2.44}$$

Constant stepsize policy. Assuming that the total number of steps N is given in advance and $\gamma_t = \gamma$, $t = 1, \ldots, N$, optimizing the right-hand side of (2.44) over $\gamma > 0$ we arrive at the constant stepsize policy

$$\gamma_t = \frac{\sqrt{2\alpha}D_{\omega,X}}{M_*\sqrt{N}}, \quad t = 1, \ldots, N \tag{2.45}$$

and the associated efficiency estimate

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \le D_{\omega,X}M_*\sqrt{\frac{2}{\alpha N}} \tag{2.46}$$

(cf. (2.20), (2.21)). For a constant $\theta > 0$, passing from the stepsizes (2.45) to the stepsizes

$$\gamma_t = \frac{\theta\sqrt{2\alpha}D_{\omega,X}}{M_*\sqrt{N}}, \quad t = 1, \ldots, N, \tag{2.47}$$

the efficiency estimate becomes

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \le \max\left\{\theta, \theta^{-1}\right\} D_{\omega,X}M_*\sqrt{\frac{2}{\alpha N}}. \tag{2.48}$$

We refer to the method (2.32), (2.38), and (2.47) as the (robust) *mirror descent SA* algorithm with constant stepsize policy.

Probabilities of large deviations. So far, all our efficiency estimates were upper bounds on the expected nonoptimality, in terms of the objective, of approximate solutions generated by the algorithms. Here we complement these results with bounds on probabilities of large deviations. Observe that by Markov inequality, (2.48) implies that

$$\mathrm{Prob}\left\{f\left(\tilde{x}_1^N\right) - f(x_*) > \varepsilon\right\} \le \frac{\sqrt{2}\max\left\{\theta, \theta^{-1}\right\} D_{\omega,X}M_*}{\varepsilon\sqrt{\alpha N}} \quad \forall \varepsilon > 0. \tag{2.49}$$

It is possible, however, to obtain much finer bounds on deviation probabilities when imposing more restrictive assumptions on the distribution of $\mathsf{G}(x,\xi)$. Specifically, assume that

$$\mathbb{E}\left[\exp\left\{\|\mathsf{G}(x,\xi)\|_*^2/M_*^2\right\}\right] \le \exp\{1\} \quad \forall x \in X. \tag{2.50}$$

Note that condition (2.50) is stronger than (2.40). Indeed, if a random variable Y satisfies $\mathbb{E}[\exp\{Y/a\}] \le \exp\{1\}$ for some $a > 0$, then by Jensen inequality, $\exp\{\mathbb{E}[Y/a]\} \le \mathbb{E}[\exp\{Y/a\}] \le \exp\{1\}$, and therefore, $\mathbb{E}[Y] \le a$. Of course, condition (2.50) holds if $\|\mathsf{G}(x,\xi)\|_* \le M_*$ for all $(x,\xi) \in X \times \Xi$.

PROPOSITION 2.2. *In the case of* (2.50) *and for the constant stepsizes* (2.47)*, the following holds for any* $\Omega \ge 1$*:*

$$\mathrm{Prob}\left\{f\left(\tilde{x}_1^N\right) - f(x_*) > \frac{\sqrt{2}\max\left\{\theta, \theta^{-1}\right\} M_*D_{\omega,X}(12 + 2\Omega)}{\sqrt{\alpha N}}\right\} \le 2\exp\{-\Omega\}. \tag{2.51}$$

Proof of this proposition is given in the appendix.

Varying stepsizes. Same as in the case of E-SA, we can modify the mirror descent SA algorithm to allow for time-varying stepsizes and "sliding averages" of the search points x_t in the role of approximate solutions, thus getting rid of the necessity to fix in advance the number of steps. Specifically, consider

$$\begin{aligned}\overline{D}_{\omega,X} &:= \sqrt{2}\sup_{x\in X^o, z\in X}\left[\omega(z)-\omega(x)-(z-x)^T\nabla\omega(x)\right]^{1/2}\\ &= \sup_{x\in X^o,z\in X}\sqrt{2V(x,z)}\end{aligned} \tag{2.52}$$

and assume that $\overline{D}_{\omega,X}$ is finite. This is definitely so when ω is continuously differentiable on the entire X. Note that for the E-SA, that is, with $\omega(x)=\frac{1}{2}\|x\|_2^2$, $\overline{D}_{\omega,X}$ is the Euclidean diameter of X.

In the case of (2.52), setting

$$\tilde{x}_i^j = \frac{\sum_{t=i}^j \gamma_t x_t}{\sum_{t=i}^j \gamma_t}, \tag{2.53}$$

summing up inequalities (2.34) over $K\le t\le N$, and acting exactly as when deriving (2.39), we get for $1\le K\le N$,

$$f\left(\tilde{x}_K^N\right)-f(x_*)\le \frac{V(x_K,x_*)+\sum_{t=K}^N\frac{\gamma_t^2}{2\alpha}\|\mathsf{G}(x_t,\xi_t)\|_*^2-\sum_{t=K}^N\gamma_t\Delta_t^T(x_t-x_*)}{\sum_{t=K}^N\gamma_t}.$$

Noting that $V(x_K,x_*)\le\frac{1}{2}\overline{D}_{\omega,X}^2$ and taking expectations, we arrive at

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right)-f(x_*)\right]\le\frac{\frac{1}{2}\overline{D}_{\omega,X}^2+\frac{1}{2\alpha}M_*^2\sum_{t=K}^N\gamma_t^2}{\sum_{t=K}^N\gamma_t} \tag{2.54}$$

(cf. (2.44)). It follows that with a decreasing stepsize policy

$$\gamma_t=\frac{\theta\overline{D}_{\omega,X}\sqrt{\alpha}}{M_*\sqrt{t}},\quad t=1,2,\dots, \tag{2.55}$$

one has for $1\le K\le N$,

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right)-f(x_*)\right]\le\frac{\overline{D}_{\omega,X}M_*}{\sqrt{\alpha}\sqrt{N}}\left[\frac{2}{\theta}\frac{N}{N-K+1}+\frac{\theta}{2}\sqrt{\frac{N}{K}}\right] \tag{2.56}$$

(cf. (2.26)). In particular, with $K=\lceil rN\rceil$ for a fixed $r\in(0,1)$, we get an efficiency estimate

$$\mathbb{E}\left[f\left(\tilde{x}_K^N\right)-f(x_*)\right]\le C(r)\max\left\{\theta,\theta^{-1}\right\}\frac{\overline{D}_{\omega,X}M_*}{\sqrt{\alpha}\sqrt{N}}, \tag{2.57}$$

completely similar to the estimate (2.27) for the E-SA.

Discussion. Comparing (2.21) to (2.46) and (2.27) to (2.57), we see that for both the Euclidean and the mirror descent robust SA, the expected inaccuracy, in terms of the objective, of the approximate solution built in course of N steps is $O(N^{-1/2})$. A benefit of the mirror descent over the Euclidean algorithm is in the

potential possibility to reduce the constant factor hidden in $O(\cdot)$ by adjusting the norm $\|\cdot\|$ and the distance-generating function $\omega(\cdot)$ to the geometry of the problem.

Example. Let $X = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1,\ x \geq 0\}$ be a standard simplex. Consider two setups for the mirror descent SA:

— *Euclidean setup*, where $\|\cdot\| = \|\cdot\|_2$ and $\omega(x) = \frac{1}{2}\|x\|_2^2$, and

— ℓ_1*-setup*, where $\|\cdot\| = \|\cdot\|_1$, with $\|\cdot\|_* = \|\cdot\|_\infty$ and ω is the *entropy* function

$$\omega(x) = \sum_{i=1}^n x_i \ln x_i. \tag{2.58}$$

The Euclidean setup leads to the Euclidean robust SA, which is easily implementable (computing the prox-mapping requires $O(n \ln n)$ operations) and guarantees that

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \leq O(1) \max\left\{\theta, \theta^{-1}\right\} M N^{-1/2}, \tag{2.59}$$

with $M^2 = \sup_{x \in X} \mathbb{E}\left[\|\mathsf{G}(x,\xi)\|_2^2\right]$, provided that the constant M is known and the stepsizes (2.23) are used (see (2.24), (2.17), and note that the Euclidean diameter of X is $\sqrt{2}$).

The ℓ_1-setup corresponds to $X^o = \{x \in X : x > 0\}$, $D_{\omega,X} = \sqrt{\ln n}$, $\alpha = 1$, and $x_1 = \operatorname{argmin}_X \omega = n^{-1}(1, \ldots, 1)^T$ (see appendix). The associated mirror descent SA is easily implementable: the prox-function here is

$$V(x,z) = \sum_{i=1}^n z_i \ln \frac{z_i}{x_i},$$

and the prox-mapping $P_x(y) = \operatorname{argmin}_{z \in X}\left[y^T(z-x) + V(x,z)\right]$ can be computed in $O(n)$ operations according to the explicit formula

$$[P_x(y)]_i = \frac{x_i e^{-y_i}}{\sum_{k=1}^n x_k e^{-y_k}},\ \ i = 1, \ldots, n.$$

The efficiency estimate guaranteed with the ℓ_1-setup is

$$\mathbb{E}\left[f\left(\tilde{x}_1^N\right) - f(x_*)\right] \leq O(1) \max\left\{\theta, \theta^{-1}\right\} \sqrt{\ln n} M_* N^{-1/2}, \tag{2.60}$$

with

$$M_*^2 = \sup_{x \in X} \mathbb{E}\left[\|\mathsf{G}(x,\xi)\|_\infty^2\right],$$

provided that the constant M_* is known and the constant stepsizes (2.47) are used (see (2.48) and (2.40)). To compare (2.60) and (2.59), observe that $M_* \leq M$, and the ratio M_*/M can be as small as $n^{-1/2}$. Thus, the efficiency estimate for the ℓ_1-setup never is much worse than the estimate for the Euclidean setup, and for large n, can be *far better* than the latter estimate:

$$\sqrt{\frac{1}{\ln n}} \leq \frac{M}{\sqrt{\ln n} M_*} \leq \sqrt{\frac{n}{\ln n}},\ N = 1, 2, \ldots,$$

both the upper and the lower bounds being achievable. Thus, when X is a standard simplex of large dimension, we have strong reasons to prefer the ℓ_1-setup to the usual Euclidean one.

Note that $\|\cdot\|_1$-norm can be coupled with "good" distance-generating functions different from the entropy one, e.g., with the function

$$\omega(x) = (\ln n)\sum_{i=1}^{n} |x_i|^{1+\frac{1}{\ln n}}, \quad n \geq 3. \tag{2.61}$$

Whenever $0 \in X$ and $\mathrm{Diam}_{\|\cdot\|_1}(X) \equiv \max_{x,y\in X} \|x-y\|_1$ equal to 1 (these conditions can always be ensured by scaling and shifting X), for the just-outlined setup, one has $\overline{D}_{\omega,X} = O(1)\sqrt{\ln n}$, $\alpha = O(1)$, so that the associated mirror descent robust SA guarantees that with $M_*^2 = \sup_{x\in X} \mathbb{E}\left[\|\mathsf{G}(x,\xi)\|_\infty^2\right]$ and $N \geq 1$,

$$\mathbb{E}\left[f\left(\tilde{x}^N_{\lceil rN\rceil}\right) - f(x_*)\right] \leq C(r)\frac{M_*\sqrt{\ln n}}{\sqrt{N}} \tag{2.62}$$

(see (2.57)), while the efficiency estimate for the Euclidean robust SA is

$$\mathbb{E}\left[f(\tilde{x}^N_{\lceil rN\rceil}) - f(x_*)\right] \leq C(r)\frac{M\,\mathrm{Diam}_{\|\cdot\|_2}(X)}{\sqrt{N}}, \tag{2.63}$$

with

$$M^2 = \sup_{x\in X} \mathbb{E}\left[\|\mathsf{G}(x,\xi)\|_2^2\right] \ \text{ and } \ \mathrm{Diam}_{\|\cdot\|_2}(X) = \max_{x,y\in X} \|x-y\|_2.$$

Ignoring logarithmic in n factors, the second estimate (2.63) can be much better than the first estimate (2.62) only when $\mathrm{Diam}_{\|\cdot\|_2}(X) \ll 1 = \mathrm{Diam}_{\|\cdot\|_1}(X)$, as it is the case, e.g., when X is an Euclidean ball. On the other hand, when X is an $\|\cdot\|_1$-ball or its nonnegative part (which is the simplex), so that the $\|\cdot\|_1$- and $\|\cdot\|_2$-diameters of X are of the same order, the first estimate (2.62) is much more attractive than the estimate (2.63) due to potentially much smaller constant M_*.

Comparison with the SAA approach. We compare now theoretical complexity estimates for the robust mirror descent SA and the SAA methods. Consider the case when (i) $X \subset \mathbb{R}^n$ is contained in the $\|\cdot\|_p$-ball of radius R, $p = 1, 2$, and the SA in question is either the E-SA ($p = 2$), or the SA associated with $\|\cdot\|_1$ and the distance-generating function[2] (2.61), (ii) in SA, the constant stepsize rule (2.45) is used, and (iii) the "light tail" assumption (2.50) takes place.

Given $\varepsilon > 0$, $\delta \in (0, 1/2)$, let us compare the number of steps $N = N_{\mathrm{SA}}$ of SA, which, with probability $\geq 1-\delta$, results in an approximate solution $\tilde{x}_1^N$ such that $f(\tilde{x}_1^N) - f(x_*) \leq \varepsilon$, with the sample size $N = N_{\mathrm{SAA}}$ for the SAA resulting in the same accuracy guarantees. According to Proposition 2.2 we have that $\mathrm{Prob}\left[f(\tilde{x}_1^N) - f(x_*) > \varepsilon\right] \leq \delta$ for

$$N_{\mathrm{SA}} = O(1)\varepsilon^{-2}D^2_{\omega,X}M_*^2\ln^2(1/\delta), \tag{2.64}$$

where M_* is the constant from (2.50) and $D_{\omega,X}$ is defined in (2.43). Note that the constant M_* depends on the chosen norm, $D^2_{\omega,X} = O(1)R^2$ for $p = 2$, and $D^2_{\omega,X} = O(1)\ln(n)R^2$ for $p = 1$.

This can be compared with the estimate of the sample size (cf. [25, 26])

$$N_{\mathrm{SAA}} = O(1)\varepsilon^{-2}R^2M_*^2\left[\ln(1/\delta) + n\ln\left(RM_*/\varepsilon\right)\right]. \tag{2.65}$$

[2]In the second case, we apply the SA after the variables are scaled to make X the unit $\|\cdot\|_1$-ball.

We see that both SA and SAA methods have logarithmic in δ and quadratic (or nearly so) in $1/\varepsilon$ complexity in terms of the corresponding sample sizes. It should be noted, however, that the SAA method requires solution of the corresponding (deterministic) problem, while the SA approach is based on simple calculations as long as stochastic subgradients could be easily computed.

3. Stochastic saddle point problem. We show in this section how the mirror descent SA algorithm can be modified to solve a convex-concave stochastic saddle point problem. Consider the following minimax (saddle point) problem:

$$\min_{x\in X}\max_{y\in Y}\big\{\phi(x,y)=\mathbb{E}[\Phi(x,y,\xi)]\big\}. \tag{3.1}$$

Here $X\subset\mathbb{R}^n$ and $Y\subset\mathbb{R}^m$ are nonempty bounded closed convex sets, ξ is a random vector whose probability distribution P is supported on set $\Xi\subset\mathbb{R}^d$, and $\Phi: X\times Y\times\Xi\to\mathbb{R}$. We assume that for every $(x,y)\in X\times Y$, the expectation

$$\mathbb{E}[\Phi(x,y,\xi)]=\int_{\Xi}\Phi(x,y,\xi)dP(\xi)$$

is well defined and finite valued and that the expected value function $\phi(x,y)$ is *convex* in $x\in X$ and *concave* in $y\in Y$. It follows that (3.1) is a *convex-concave saddle point* problem. In addition, we assume that $\phi(\cdot,\cdot)$ is *Lipschitz continuous* on $X\times Y$. It is well known that, in the above setting, (3.1) is solvable, i.e., the corresponding "primal" and "dual" optimization problems

$$\min_{x\in X}\left[\max_{y\in Y}\phi(x,y)\right]\quad\text{and}\quad\max_{y\in Y}\left[\min_{x\in X}\phi(x,y)\right],$$

respectively, are solvable with equal optimal values, denoted ϕ^*, and pairs (x^*,y^*) of optimal solutions to the respective problems form the set of saddle points of $\phi(x,y)$ on $X\times Y$.

As in the case of the minimization problem (1.1), we assume that neither the function $\phi(x,y)$ nor its sub/supergradients in x and y are available explicitly. However, we make the following assumption.

(A′2) We have at our disposal an oracle which, given an input of point $(x,y,\xi)\in X\times Y\times\Xi$, returns a *stochastic subgradient*, that is, $(n+m)$-dimensional vector $\mathsf{G}(x,y,\xi)=\left[\begin{smallmatrix}\mathsf{G}_x(x,y,\xi)\\ -\mathsf{G}_y(x,y,\xi)\end{smallmatrix}\right]$ such that vector

$$\mathsf{g}(x,y)=\begin{bmatrix}\mathsf{g}_x(x,y)\\ -\mathsf{g}_y(x,y)\end{bmatrix}=\begin{bmatrix}\mathbb{E}[\mathsf{G}_x(x,y,\xi)]\\ -\mathbb{E}[\mathsf{G}_y(x,y,\xi)]\end{bmatrix}$$

is well defined, $\mathsf{g}_x(x,y)\in\partial_x\phi(x,y)$, and $-\mathsf{g}_y(x,y)\in\partial_y(-\phi(x,y))$.

For example, if for every $\xi\in\Xi$ the function $\Phi(\cdot,\cdot,\xi)$ is *convex-concave* and the respective subdifferential and integral operators are interchangeable, we ensure (A′2) by setting

$$\mathsf{G}(x,y,\xi)=\begin{bmatrix}\mathsf{G}_x(x,y,\xi)\\ -\mathsf{G}_y(x,y,\xi)\end{bmatrix}\in\begin{bmatrix}\partial_x\Phi(x,y,\xi)\\ \partial_y(-\Phi(x,y,\xi))\end{bmatrix}.$$

Let $\|\cdot\|_x$ be a norm on $\mathbb{R}^n$ and $\|\cdot\|_y$ be a norm on $\mathbb{R}^m$, and let $\|\cdot\|_{*,x}$ and $\|\cdot\|_{*,y}$ stand for the corresponding dual norms. As in section 2.1, the basic assumption we make about the stochastic oracle (aside from its unbiasedness, which we have already postulated) is that we know positive constants $M_{*,x}^2$ and $M_{*,y}^2$ such that

$$\mathbb{E}\left[\|\mathsf{G}_x(u,v,\xi)\|_{*,x}^2\right]\le M_{*,x}^2\ \text{ and }\ \mathbb{E}\left[\|\mathsf{G}_y(u,v,\xi)\|_{*,y}^2\right]\le M_{*,y}^2\ \ \forall(u,v)\in X\times Y. \tag{3.2}$$

3.1. Mirror SA algorithm for saddle point problems. We equip X and Y with distance-generating functions $\omega_x : X \to \mathbb{R}$ modulus α_x with respect to $\|\cdot\|_x$, and $\omega_y : Y \to \mathbb{R}$ modulus α_y with respect to $\|\cdot\|_y$. Let $D_{\omega_x,X}$ and $D_{\omega_y,Y}$ be the respective constants (see definition (2.42)). We equip $\mathbb{R}^n \times \mathbb{R}^m$ with the norm

$$\|(x,y)\| = \sqrt{\tfrac{\alpha_x}{2D^2_{\omega_x,X}}\|x\|_x^2 + \tfrac{\alpha_y}{2D^2_{\omega_y,Y}}\|y\|_y^2}, \tag{3.3}$$

so that the dual norm is

$$\|(\zeta,\eta)\|_* = \sqrt{\tfrac{2D^2_{\omega_x,X}}{\alpha_x}\|\zeta\|_{*,x}^2 + \tfrac{2D^2_{\omega_y,Y}}{\alpha_y}\|\eta\|_{*,y}^2} \tag{3.4}$$

and set

$$M_*^2 = \frac{2D^2_{\omega_x,X}}{\alpha_x}M_{*,x}^2 + \frac{2D^2_{\omega_y,Y}}{\alpha_y}M_{*,y}^2. \tag{3.5}$$

It follows by (3.2) that

$$\mathbb{E}\big[\|\mathsf{G}(x,y,\xi)\|_*^2\big] \le M_*^2. \tag{3.6}$$

We use the notation $z = (x,y)$ and equip the set $Z = X \times Y$ with the distance-generating function

$$\omega(z) = \frac{\omega_x(x)}{2D^2_{\omega_x,X}} + \frac{\omega_y(y)}{2D^2_{\omega_y,Y}}.$$

It is immediately seen that ω indeed is a distance-generating function for Z modulus $\alpha = 1$ with respect to the norm $\|\cdot\|$ and that $Z^o = X^o \times Y^o$ and $D_{\omega,Z} = 1$. In what follows, $V(z,u) : Z^o \times Z \to \mathbb{R}$ and $P_z(\zeta) : \mathbb{R}^{n+m} \to Z^o$ are the prox-function and prox-mapping associated with ω and Z (see (2.30), (2.31)).

We are ready now to present the mirror SA algorithm for saddle point problems. This is the iterative procedure (compare with (2.32))

$$z_{j+1} = P_{z_j}(\gamma_j \mathsf{G}(z_j,\xi_j)), \tag{3.7}$$

where the initial point $z_1 \in Z$ is chosen to be the minimizer of $\omega(z)$ on Z. As before (compare with (2.38)), we define approximate solution $\tilde z_1^j = (\tilde x_1^j, \tilde y_1^j)$ of (3.1) after j iterations as

$$\tilde z_1^j = \frac{\sum_{t=1}^j \gamma_t z_t}{\sum_{t=1}^j \gamma_t}. \tag{3.8}$$

We refer to the procedure (3.7), (3.8) as the *saddle point mirror SA* algorithm.

Let us analyze convergence properties of the algorithm. We measure quality of an approximate solution $\tilde z = (\tilde x, \tilde y)$ by the error

$$\epsilon_\phi(\tilde z) := \left[\max_{y\in Y}\phi(\tilde x, y) - \phi_*\right] + \left[\phi_* - \min_{x\in X}\phi(x,\tilde y)\right] = \max_{y\in Y}\phi(\tilde x, y) - \min_{x\in X}\phi(x,\tilde y).$$

By convexity of $\phi(\cdot, y)$, we have

$$\phi(x_t,y_t) - \phi(x,y_t) \le (x_t - x)^T \mathsf{g}_x(x_t,y_t) \ \ \forall x \in X$$

and by concavity of $\phi(x,\cdot)$,

$$\phi(x_t,y)-\phi(x_t,y_t)\le (y-y_t)^T\mathtt{g}_y(x_t,y_t)\ \ \forall y\in Y$$

so that for all $z=(x,y)\in Z$,

$$\phi(x_t,y)-\phi(x,y_t)\le (x_t-x)^T\mathtt{g}_x(x_t,y_t)+(y-y_t)^T\mathtt{g}_y(x_t,y_t)=(z_t-z)^T\mathtt{g}(z_t).$$

Using once again the convexity-concavity of ϕ, we write

$$\begin{aligned}\epsilon_\phi(\tilde z_1^j)&=\max_{y\in Y}\phi\left(\tilde x_1^j,y\right)-\min_{x\in X}\phi\left(x,\tilde y_1^j\right)\\&\le\left[\sum_{t=1}^j\gamma_t\right]^{-1}\left[\max_{y\in Y}\sum_{t=1}^j\gamma_t\phi(x_t,y)-\min_{x\in X}\sum_{t=1}^j\gamma_t\phi(x,y_t)\right]\\&\le\left[\sum_{t=1}^j\gamma_t\right]^{-1}\max_{z\in Z}\sum_{t=1}^j\gamma_t(z_t-z)^T\mathtt{g}(z_t).\end{aligned}\tag{3.9}$$

To bound the right-hand side of (3.9), we use the result of the following lemma (its proof is given in the appendix).

LEMMA 3.1. *In the above setting, for any $j\ge 1$, the following inequality holds:*

$$\mathbb{E}\left[\max_{z\in Z}\sum_{t=1}^j\gamma_t(z_t-z)^T\mathtt{g}(z_t)\right]\le 2+\frac{5}{2}M_*^2\sum_{t=1}^j\gamma_t^2.\tag{3.10}$$

Now to get an error bound for the solution $\tilde z_1^j$, it suffices to substitute inequality (3.10) into (3.9) to obtain

$$\mathbb{E}\left[\epsilon_\phi\left(\tilde z_1^j\right)\right]\le\left[\sum_{t=1}^j\gamma_t\right]^{-1}\left[2+\frac{5}{2}M_*^2\sum_{t=1}^j\gamma_t^2\right].\tag{3.11}$$

Constant stepsizes and basic efficiency estimates. For a fixed number of steps N, with the constant stepsize policy

$$\gamma_t=\frac{2\theta}{M_*\sqrt{5N}},\ \ t=1,\dots,N,\tag{3.12}$$

condition (3.6) and estimate (3.11) imply that

$$\begin{aligned}\epsilon_\phi\left(\tilde z_1^N\right)&\le 2\max\left\{\theta,\theta^{-1}\right\}M_*\sqrt{\tfrac{5}{N}}\\&=2\max\left\{\theta,\theta^{-1}\right\}\sqrt{\tfrac{10\left[\alpha_yD^2_{\omega_x,X}M^2_{*,x}+\alpha_xD^2_{\omega_y,Y}M^2_{*,y}\right]}{\alpha_x\alpha_yN}}.\end{aligned}\tag{3.13}$$

Variable stepsizes. Same as in the minimization case, assuming that

$$\begin{aligned}\overline{D}_{\omega,Z}&:=\sqrt{2}\sup_{z\in Z^o,w\in Z}\left[\omega(w)-\omega(z)-(w-z)^T\nabla\omega(z)\right]^{1/2}\\&=\sqrt{2}\left[\sup_{z\in Z^o,w\in Z}V(z,w)\right]^{1/2}\end{aligned}\tag{3.14}$$

is finite, we can pass from constant stepsizes on a fixed "time horizon" to decreasing stepsize policy

$$\gamma_t = \frac{\theta \overline{D}_{\omega,Z}}{M_* \sqrt{t}}, \quad t = 1, 2, \ldots$$

(compare with (2.55) and take into account that we are in the situation of $\alpha = 1$), and from the averaging of all iterates to the "sliding averaging"

$$\tilde{z}_i^j = \frac{\sum_{t=i}^j \gamma_t z_t}{\sum_{t=i}^j \gamma_t},$$

arriving at the efficiency estimates (compare with (2.56) and (2.57))

$$\begin{aligned} \epsilon\left(\tilde{z}_K^N\right) &\le \frac{\overline{D}_{\omega,Z} M_*}{\sqrt{N}} \left[\frac{2}{\theta} \frac{N}{N-K+1} + \frac{5\theta}{2}\sqrt{\frac{N}{K}}\right], \quad 1 \le K \le N, \\ \epsilon\left(\tilde{z}_{\lceil rN \rceil}^N\right) &\le C(r) \max\left\{\theta, \theta^{-1}\right\} \frac{\overline{D}_{\omega,Z} M_*}{\sqrt{N}}, \qquad r \in (0,1). \end{aligned} \tag{3.15}$$

Probabilities of large deviations. Assume that instead of (3.2), the following stronger assumption holds:

$$\begin{aligned} \mathbb{E}\left[\exp\left\{\|\mathsf{G}_x(u,v,\xi)\|_{*,x}^2 / M_{*,x}^2\right\}\right] &\le \exp\{1\}, \\ \mathbb{E}\left[\exp\left\{\|\mathsf{G}_y(x,y,\xi)\|_{*,y}^2 / M_{*,y}^2\right\}\right] &\le \exp\{1\}. \end{aligned} \tag{3.16}$$

Proposition 3.2. *In the case of* (3.16)*, with the stepsizes given by* (3.12) and (3.6)*, one has, for any* $\Omega > 1$*,*

$$\text{Prob}\left\{\epsilon_\phi\left(\tilde{z}_1^N\right) > \frac{(8+2\Omega)\max\left\{\theta,\theta^{-1}\right\}\sqrt{5} M_*}{\sqrt{N}}\right\} \le 2\exp\{-\Omega\}. \tag{3.17}$$

Proof of this proposition is given in the appendix.

3.2. Application to minimax stochastic problems. Consider the following minimax stochastic problem:

$$\min_{x \in X} \max_{1 \le i \le m} \left\{f_i(x) = \mathbb{E}[F_i(x,\xi)]\right\}, \tag{3.18}$$

where $X \subset \mathbb{R}^n$ is a nonempty bounded closed convex set, ξ is a random vector whose probability distribution P is supported on set $\Xi \subset \mathbb{R}^d$, and $F_i : X \times \Xi \to \mathbb{R}$, $i = 1, \ldots, m$. We assume that the expected value functions $f_i(\cdot)$, $i = 1, \ldots, m$, are well defined, finite valued, *convex*, and *Lipschitz continuous* on X. Then the minimax problem (3.18) can be formulated as the following saddle point problem:

$$\min_{x \in X} \max_{y \in Y} \left\{\phi(x,y) = \sum_{i=1}^m y_i f_i(x)\right\}, \tag{3.19}$$

where $Y = \{y \in \mathbb{R}^m : \sum_{i=1}^m y_i = 1,\ y \ge 0\}$.

Assume that we are able to generate independent realizations $\xi_1, \ldots,$ of random vector ξ, and, for given $x \in X$ and $\xi \in \Xi$, we can compute $F_i(x,\xi)$ and its *stochastic subgradient* $\mathsf{G}_i(x,\xi)$ such that $\mathsf{g}_i(x) = \mathbb{E}[\mathsf{G}_i(x,\xi)]$ is well defined and $\mathsf{g}_i(x) \in \partial f_i(x)$,

$x \in X$, $i = 1, \ldots, m$. In other words, we have a stochastic oracle for the problem (3.19) such that assumption (A′2) holds, with

$$\mathsf{G}(x, y, \xi) = \begin{bmatrix} \sum_{i=1}^m y_i \mathsf{G}_i(x, \xi) \\ -\big(F_1(x, \xi), \ldots, F_m(x, \xi)\big) \end{bmatrix} \tag{3.20}$$

and

$$\mathsf{g}(x, y) = \mathbb{E}[\mathsf{G}(x, y, \xi)] = \begin{bmatrix} \sum_{i=1}^m y_i \mathsf{g}_i(x) \\ -\big(f_1(x), \ldots, f_m(x)\big) \end{bmatrix} \in \begin{bmatrix} \partial_x \phi(x, y) \\ -\partial_y \phi(x, y) \end{bmatrix}. \tag{3.21}$$

Suppose that the set X is equipped with norm $\|\cdot\|_x$, whose dual norm is $\|\cdot\|_{*,x}$, and a distance-generating function ω modulus α_x with respect to $\|\cdot\|_x$, and let $R_x^2 = \frac{D^2_{\omega_x, X}}{\alpha_x}$. We equip the set Y with the norm $\|\cdot\|_y = \|\cdot\|_1$, so that $\|\cdot\|_{*,y} = \|\cdot\|_\infty$, and with the distance-generating function

$$\omega_y(y) = \sum_{i=1}^m y_i \ln y_i$$

and set $R_y^2 = \frac{D^2_{\omega_y, Y}}{\alpha_y} = \ln m$. Next, following (3.3), we set

$$\|(x, y)\| = \sqrt{\frac{\|x\|_x^2}{2R_x^2} + \frac{\|y\|_1^2}{2R_y^2}},$$

and hence

$$\|(\zeta, \eta)\|_* = \sqrt{2R_x^2 \|\zeta\|_{*,x}^2 + 2R_y^2 \|\eta\|_\infty^2}.$$

Let us assume uniform bounds:

$$\max_{1 \le i \le m} \mathbb{E}\left[\|\mathsf{G}_i(x, \xi)\|_{*,x}^2\right] \le M_{*,x}^2, \quad \mathbb{E}\left[\max_{1 \le i \le m} |F_i(x, \xi)|^2\right] \le M_{*,y}^2, \quad i = 1, \ldots, m.$$

Note that

$$\mathbb{E}\left[\|\mathsf{G}(x, y, \xi)\|_*^2\right] = 2R_x^2 \, \mathbb{E}\left[\Big\| \sum_{i=1}^m y_i \mathsf{G}_i(x, \xi) \Big\|_{*,x}^2\right] + 2R_y^2 \, \mathbb{E}\left[\|F(x, \xi)\|_\infty^2\right],$$

and since $y \in Y$,

$$\left\| \sum_{i=1}^m y_i \mathsf{G}_i(x, \xi) \right\|_{*,x}^2 \le \left(\sum_{i=1}^m y_i \|\mathsf{G}_i(x, \xi)\|_{*,x} \right)^2 \le \sum_{i=1}^m y_i \|\mathsf{G}_i(x, \xi)\|_{*,x}^2.$$

It follows that

$$\mathbb{E}\left[\|\mathsf{G}(x, y, \xi)\|_*^2\right] \le M_*^2, \tag{3.22}$$

where

$$M_*^2 = 2R_x^2 M_{*,x}^2 + 2R_y^2 M_{*,y}^2 = 2R_x^2 M_{*,x}^2 + 2M_{*,y}^2 \ln m.$$

Let us now use the saddle point mirror SA algorithm (3.7)–(3.8) with the constant stepsize policy

$$\gamma_t = \frac{2}{M_*\sqrt{5N}}, \quad t = 1, 2, \ldots, N.$$

When substituting the value of M_*, we obtain the following from (3.13):

$$\begin{aligned} \mathbb{E}\left[\epsilon_\phi\left(\tilde{z}_1^N\right)\right] &= \mathbb{E}\left[\max_{y\in Y}\phi\left(\hat{x}_1^N, y\right) - \min_{x\in X}\phi\left(x, \hat{y}_1^N\right)\right] \\ &\le 2M_*\sqrt{\tfrac{5}{N}} \le 2\sqrt{\tfrac{10\left[R_x^2 M_{*,x}^2 + M_{*,x}^2 \ln m\right]}{N}}. \end{aligned} \tag{3.23}$$

Discussion. Looking at the bound (3.23), one can make the following important observation. The error of the saddle point mirror SA algorithm in this case is "almost independent" of the number m of constraints (it grows as $O(\sqrt{\ln m})$ as m increases). The interested reader can easily verify that if an E-SA algorithm were used in the same setting (i.e., the algorithm tuned to the norm $\|\cdot\|_y = \|\cdot\|_2$), the corresponding bound would grow with m much faster (in fact, our error bound would be $O(\sqrt{m})$ in that case).

Note that properties of the saddle point mirror SA can be used to reduce significantly the arithmetic cost of the algorithm implementation. To this end let us look at the definition (3.20) of the stochastic oracle: In order to obtain a realization $\mathsf{G}(x, y, \xi)$, one has to compute m random subgradients $\mathsf{G}_i(x, \xi)$, $i = 1, \ldots, m$, and then their convex combination $\sum_{i=1}^m y_i \mathsf{G}_i(x, \xi)$. Now let η be an independent of ξ and uniformly distributed on $[0, 1]$ random variable, and let $\imath(\eta, y) : [0, 1] \times Y \to \{1, \ldots, m\}$ equal to i when $\sum_{s=1}^{i-1} y_s < \eta \le \sum_{s=1}^{i} y_s$. That is, random variable $\hat{\imath} = \imath(\eta, y)$ takes values $1, \ldots, m$ with probabilities $y_1, \ldots, y_m$. Consider random vector

$$\mathsf{G}(x, y, (\xi, \eta)) = \begin{bmatrix} \mathsf{G}_{\imath(\eta,y)}(x, \xi) \\ -\big(F_1(x, \xi), \ldots, F_m(x, \xi)\big) \end{bmatrix}. \tag{3.24}$$

We refer to $\mathsf{G}(x, y, (\xi, \eta))$ as a *randomized oracle* for problem (3.19), the corresponding random parameter being (ξ, η). By construction, we still have $\mathbb{E}\big[\mathsf{G}(x, y, (\xi, \eta))\big] = \mathsf{g}(x, y)$, where g is defined in (3.21), and, moreover, the same bound (3.22) holds for $\mathbb{E}\big[\|\mathsf{G}(x, y, (\xi, \eta))\|_*^2\big]$. We conclude that the accuracy bound (3.23) holds for the error of the saddle point mirror SA algorithm with randomized oracle. On the other hand, in the latter procedure only one randomized subgradient $\mathsf{G}_{\hat{\imath}}(x, \xi)$ per iteration is computed. This simple idea is further developed in another interesting application of the saddle point mirror SA algorithm to bilinear matrix games, which we discuss next.

3.3. Application to bilinear matrix games. Consider the standard matrix game problem, that is, problem (3.1) with

$$\phi(x, y) = y^T A x + b^T x + c^T y,$$

where $A \in \mathbb{R}^{m\times n}$, and X and Y are the standard simplexes:

$$X = \left\{x \in \mathbb{R}^n : x \ge 0, \sum_{j=1}^n x_j = 1\right\}, \quad Y = \left\{y \in \mathbb{R}^m : y \ge 0, \sum_{i=1}^m y_i = 1\right\}.$$

In the case in question it is natural to equip X (respectively, Y) with the $\|\cdot\|_1$-norm on $\mathbb{R}^n$ (respectively, $\mathbb{R}^m$). We choose entropies as the corresponding distance-generating functions:

$$\omega_x(x) = \sum_{i=1}^n x_i \ln x_i, \quad \omega_y(x) = \sum_{i=1}^m y_i \ln y_i \qquad \left[\Rightarrow \tfrac{D^2_{\omega_x,X}}{\alpha_x} = \ln n, \tfrac{D^2_{\omega_y,Y}}{\alpha_y} = \ln m\right].$$

According to (3.3), we set

$$\|(x,y)\| = \sqrt{\frac{\|x\|_1^2}{2\ln n} + \frac{\|y\|_1^2}{2\ln m}} \Rightarrow \|(\zeta,\eta)\|_* = \sqrt{2\|\zeta\|_\infty^2 \ln n + 2\|\eta\|_\infty^2 \ln m}. \tag{3.25}$$

In order to compute the estimates $\mathsf{G}(x,y,\xi)$ of $\mathsf{g}(x,y) = (b + A^T y, -c - Ax)$, to be used in the saddle point mirror SA iterations (3.7), we use the *randomized oracle*

$$\mathsf{G}(x,y,\xi) = \begin{bmatrix} c + A^{\imath(\xi_1,y)} \\ -b - A_{\imath(\xi_2,x)} \end{bmatrix}, \tag{3.26}$$

where ξ_1 and ξ_2 are independent uniformly distributed on $[0,1]$ random variables and $\hat{j} = \imath(\xi_1, y)$, $\hat{\imath} = \imath(\xi_2, x)$ are defined as in (3.24) (i.e., $\hat{j}$ can take values $1,\dots,m$, with probabilities $y_1,\dots,y_m$ and $\hat{\imath}$ can take values $1,\dots,n$, with probabilities $x_1,\dots,x_n$), and A_j, $[A^i]^T$ are jth column and ith row in A, respectively.

Note that

$$\mathsf{g}(x,y) \equiv \mathbb{E}\left[\mathsf{G}\left(x,y,\left(\hat{j},\hat{\imath}\right)\right)\right] \in \begin{bmatrix} \partial_x \phi(x,y) \\ \partial_y(-\phi(x,y)) \end{bmatrix}. \tag{3.27}$$

Besides this,

$$\begin{array}{ll} |\mathsf{G}(x,y,\xi)_i| \le \max_{1\le j\le m} \|A^j + b\|_\infty, & 1 \le i \le n, \\ |\mathsf{G}(x,y,\xi)_i| \le \max_{1\le j\le n} \|A_j + c\|_\infty, & n+1 \le i \le n+m, \end{array}$$

whence, invoking (3.25), for any $x \in X$, $y \in Y$, and ξ,

$$\|\mathsf{G}(x,y,\xi)\|_*^2 \le M_*^2 = 2\ln n \max_{1\le j\le m} \|A^j + b\|_\infty^2 + 2\ln m \max_{1\le j\le n} \|A_j + c\|_\infty^2. \tag{3.28}$$

The bottom line is that our stochastic gradients along with the just-defined M_* satisfy both (A′2) and (3.16), and therefore with the constant stepsize policy (3.12), we have

$$\mathbb{E}\left[\epsilon_\phi\left(\tilde{z}_1^N\right)\right] = \mathbb{E}\left[\max_{y\in Y} \phi\left(\tilde{x}_1^N, y\right) - \min_{x\in X} \phi\left(x, \tilde{y}_1^N\right)\right] \le 2M_* \sqrt{\frac{5}{N}} \tag{3.29}$$

(cf. (3.13)). In our present situation, Proposition 3.2 in a slightly refined form (for proof, see the appendix) reads as follows.

PROPOSITION 3.3. *With the constant stepsize policy* (3.12)*, for the just-defined algorithm, one has for any* $\Omega \ge 1$*, that*

$$\mathrm{Prob}\left\{\epsilon_\phi\left(\tilde{z}_1^N\right) > 2M_* \sqrt{\tfrac{5}{N}} + \tfrac{4\overline{M}}{\sqrt{N}}\Omega\right\} \le \exp\left\{-\Omega^2/2\right\}, \tag{3.30}$$

where

$$\overline{M} = \max_{1\le j\le m} \|A^j + b\|_\infty + \max_{1\le j\le n} \|A_j + c\|_\infty. \tag{3.31}$$

Discussion. Consider a bilinear matrix game with $n \geq m$, $\ln(m) = O(1)\ln(n)$, and $b = c = 0$ (so that $M_* = O(1)\sqrt{\ln n}\,\overline{M}$ and $\overline{M} = \max_{i,j} |A_{ij}|$; see (3.28), (3.31)). Suppose that we are interested to solve it within a fixed relative accuracy ρ, that is, to ensure that the (perhaps random) approximate solution $\tilde{z}_1^N$, which we get after N iterations, satisfies the error bound

$$\epsilon_\phi(\tilde{z}_N) \leq \rho \max_{1 \leq i,\, j \leq n} |A_{ij}|$$

with probability at least $1 - \delta$. According to (3.30), to this end, one can use the randomized saddle point mirror SA algorithm (3.7), (3.8), (3.26) with stepsizes (3.12), (3.28) and with

$$N = O(1)\frac{\ln n + \ln(1/\delta)}{\rho^2}. \tag{3.32}$$

The computational cost of building $\tilde{z}_1^N$ with this approach is

$$\mathcal{C}(\rho) = O(1)\frac{[\ln n + \ln(1/\delta)]\,[\mathcal{R} + n]}{\rho^2}$$

arithmetic operations, where $\mathcal{R}$ is the arithmetic cost of extracting a column/row from A given the index of this column/row. The total number of rows and columns visited by the algorithm does not exceed the number of steps N as given in (3.32) so that the total number of entries in A used in the course of the entire computation does not exceed

$$M = O(1)\frac{n(\ln n + \ln(1/\delta))}{\rho^2}.$$

When ρ is fixed, $m = O(1)n$ and n is large, M is incomparably less than the total number mn of entries in A. Thus, our algorithm exhibits *sublinear-time behavior*: it produces reliable solutions of prescribed quality to large-scale matrix games by inspecting a negligible, as $n \to \infty$, part of randomly selected data. Note that randomization here is critical.[3] It can be seen that a deterministic algorithm, which is capable to find a solution with (deterministic) relative accuracy $\rho \leq 0.1$, has to "see" in the worst case *at least* $O(1)n$ rows/columns of A.

4. Numerical results. In this section, we report the results of our computational experiments where we compare the performance of the robust mirror descent SA method and the SAA method applied to three stochastic programming problems, namely: a stochastic utility problem, a stochastic max-flow problem, and a network planning problem with random demand. We also present a small simulation study of the performance of randomized mirror SA algorithm for bilinear matrix games.

The algorithms we were testing are the two variants of the robust mirror descent SA. The first variant, the E-SA, is as described in section 2.2; in terms of section 2.3, this is nothing but mirror descent robust SA with Euclidean setup; see the example in section 2.3. The second variant, referred to as the *non-Euclidean* SA (N-SA), is the mirror descent robust SA with ℓ_1-setup; see, the example in section 2.3.

[3] The possibility to solve matrix games in a sublinear-time fashion by a randomized algorithm was discovered by Grigoriadis and Khachiyan [9]. Their "ad hoc" algorithm is similar, although not completely identical to ours, and possesses the same complexity bounds.

TABLE 4.1
Selecting stepsize policy.

[method: N-SA, N:2,000, K:10,000, instance: L1]

	θ			
Policy	0.1	1	5	10
Variable	−7.4733	−7.8865	−7.8789	−7.8547
Constant	−6.9371	−7.8637	−7.9037	−7.8971

These two variants of the SA method are compared with the SAA approach in the following way: fixing an iid. sample (of size N) for the random variable ξ, we apply the three aforementioned methods to obtain approximate solutions for the test problem under consideration, and then the quality of the solutions yielded by these algorithms is evaluated using another iid. sample of size $K >> N$. It should be noted that SAA itself is not an algorithm, and in our experiments, it was coupled with the non-Euclidean restricted memory level (NERML) [2]—a powerful deterministic algorithm for solving the sample average problem (1.4).

4.1. Preliminaries.

Algorithmic schemes. Both E-SA and N-SA were implemented according to the description in section 2.3, the number of steps N being the parameter of a particular experiment. In such an experiment, we generated $\approx \log_2 N$ candidate solutions $\tilde{x}_i^N$, with $N-i+1 = \min[2^k, N]$, $k = 0, 1, \ldots, \lceil \log_2 N \rceil$. We then used an additional sample to estimate the objective at these candidate solutions in order to choose the best of these candidates, specifically, as follows: we used a relatively short sample to choose the two "most promising" of the candidate solutions, and then a large sample (of size $K \gg N$) to identify the best of these two candidates, thus getting the "final" solution. The computational effort required by this simple postprocessing is *not* reflected in the tables to follow.

The stepsizes. At the "pilot stage" of our experimentation, we made a decision on which stepsize policy—(2.47) or (2.55)—to choose and how to identify the underlying parameters M_* and θ. In all our experiments, M_* was estimated by taking the maxima of $\|\mathsf{G}(\cdot,\cdot)\|_*$ over a small (just 100) calls to the stochastic oracle at randomly generated feasible solutions. As about the value of θ and type of the stepsize policy ((2.47) or (2.55)), our choice was based on the results of experimentation with a single test problem (instance L1 of the utility problem, see below); some results of this experimentation are presented in Table 4.1. We have found that the constant stepsize policy (2.47) with $\theta = 0.1$ for the E-SA and $\theta = 5$ for the N-SA slightly outperforms other variants we have considered. This particular policy, combined with the aforementioned scheme for estimating M_*, was used in all subsequent experiments.

Format of test problems. All our test problems are of the form $\min_{x\in X} f(x)$, $f(x) = \mathbb{E}[F(x,\xi)]$, where the domain X either is a standard simplex $\{x \in \mathbb{R}^n : x \geq 0, \sum_i x_i = 1\}$ or can be converted into such a simplex by scaling of the original variables.

Notation in the tables. Below,

- n is the design dimension of an instance,
- N is the sample size (i.e., the number of steps in SA, and the size of the sample used to build the stochastic average in SAA),
- `Obj` is the empirical mean of the random variable $F(x,\xi)$, x being the approximate solution generated by the algorithm in question. The empirical means are taken over a large ($K = 10^4$ elements) dedicated sample,
- `CPU` is the CPU time in seconds.

TABLE 4.2
SA versus SAA on the stochastic utility problem.

-		L1: $n = 500$		L2: $n = 1{,}000$		L3: $n = 2{,}000$		L4: $n = 5{,}000$	
ALG.	N	Obj	CPU	Obj	CPU	Obj	CPU	Obj	CPU
N-SA	100	−7.7599	0	−5.8340	0	−7.1419	1	−5.4688	3
	1,000	−7.8781	2	−5.9152	2	−7.2312	6	−5.5716	13
	2,000	−7.8987	2	−5.9243	5	−7.2513	10	−5.5847	25
	4,000	−7.9075	5	−5.9365	12	−7.2595	20	−5.5935	49
E-SA	100	−7.6895	0	−5.7988	1	−7.0165	1	−4.9364	4
	1,000	−7.8559	2	−5.8919	4	−7.2029	7	−5.3895	20
	2,000	−7.8737	3	−5.9067	7	−7.2306	15	−5.4870	39
	4,000	−7.8948	7	−5.9193	13	−7.2441	29	−5.5354	77
SAA	100	−7.6571	7	−5.6346	8	−6.9748	19	−5.3360	44
	1,000	−7.8821	31	−5.9221	68	−7.2393	134	−5.5656	337
	2,000	−7.9100	72	−5.9313	128	−7.2583	261	−5.5878	656
	4,000	−7.9087	113	−5.9384	253	−7.2664	515	−5.5967	1,283

TABLE 4.3
The variability for the stochastic utility problem.

-		N-SA			E-SA			SAA		
		Obj		CPU	Obj		CPU	Obj		CPU
Inst	N	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)
L2	1,000	−5.9159	0.0025	2.63	−5.8925	0.0024	4.99	−5.9219	0.0047	67.31
L2	2,000	−5.9258	0.0022	5.03	−5.9063	0.0019	7.09	−5.9328	0.0028	131.25

4.2. A stochastic utility problem. Our first experiment was carried out with the utility model

$$\min_{x\in X}\left\{f(x)=\mathbb{E}\left[\phi\Big(\sum_{i=1}^{n}(i/n+\xi_i)x_i\Big)\right]\right\}, \tag{4.1}$$

where $X=\{x\in\mathbb{R}^n : x\geq 0, \sum_{i=1}^n x_i = 1\}$, $\xi_i \sim \mathcal{N}(0,1)$ are independent and $\phi(\cdot)$ is a piecewise linear convex function given by $\phi(t)=\max\{v_1+s_1t,\ldots,v_m+s_mt\}$, where v_k and s_k are certain constants. In our experiment, we used $m=10$ breakpoints, all located on $[0,1]$. The four instances L1, L2, L3, L4 we dealt with were of dimension varying from 500 to 2,000, each instance—with its own randomly generated function ϕ. All the algorithms were coded in ANSI C, and the experiments were conducted on an Intel PIV 1.6GHz machine with Microsoft windows XP professional.

We run each of the three aforementioned methods with various sample sizes on every one of the instances. The results are reported in Table 4.2.

In order to evaluate stability of the algorithms, we run each of them 100 times; the resulting statistics are shown in Table 4.3. In this relatively time-consuming experiment, we restrict ourselves with a single instance (L2) and just two sample sizes (N = 1,000 and 2,000). In Table 4.3, "Mean" and "Dev" are, respectively, the mean and the deviation, over 100 runs, of the objective value Obj at the resulting approximate solution.

The experiments demonstrate that as far as the quality of approximate solutions is concerned, N-SA outperforms E-SA and is almost as good as SAA. At the same time, the solution time for N-SA is significantly smaller than the one for SAA.

4.3. Stochastic max-flow problem. In the second experiment, we consider simple two-stage stochastic linear programming, namely, a stochastic max-flow problem. The problem is to optimize the capacity expansion of a stochastic network. Let

TABLE 4.4
SA versus SAA on the stochastic max-flow problem.

-		F1		F2		F3		F4	
(m,n)		(50,500)		(100, 1,000)		(100, 2,000)		(250, 5,000)	
ALG.	N	Obj	CPU	Obj	CPU	Obj	CPU	Obj	CPU
N-SA	100	0.1140	0	0.0637	0	0.1296	1	0.1278	3
	1,000	0.1254	1	0.0686	3	0.1305	6	0.1329	15
	2,000	0.1249	3	0.0697	6	0.1318	11	0.1338	29
	4,000	0.1246	5	0.0698	11	0.1331	21	0.1334	56
E-SA	100	0.0840	0	0.0618	1	0.1277	2	0.1153	7
	1,000	0.1253	3	0.0670	6	0.1281	16	0.1312	39
	2,000	0.1246	5	0.0695	13	0.1287	28	0.1312	72
	4,000	0.1247	9	0.0696	24	0.1303	53	0.1310	127
SAA	100	0.1212	5	0.0653	12	0.1310	20	0.1253	60
	1,000	0.1223	35	0.0694	84	0.1294	157	0.1291	466
	2,000	0.1223	70	0.0693	170	0.1304	311	0.1284	986
	4,000	0.1221	140	0.0693	323	0.1301	636	0.1293	1,885

$G = (N, A)$ be a diagraph with a source node s and a sink node t. Each arc $(i, j) \in A$ has an existing capacity $p_{ij} \geq 0$ and a random implementing/operating level ξ_{ij}. Moreover, there is a common random degrading factor η for all arcs in A. The goal is to determine how much capacity to add to the arcs, subject to a budget constraint, in order to maximize the expected maximum flow from s to t. Denoting by x_{ij} the capacity to be added to arc (i, j), the problem reads

$$\max_x \left\{ f(x) = \mathbb{E}[F(x;\xi,\eta)] : \sum_{(i,j)\in A} c_{ij}x_{ij} \leq b, x_{ij} \geq 0, \quad \forall (i,j) \in A \right\}, \tag{4.2}$$

where c_{ij} is the per unit cost for the capacity to be added, b is the total available budget, and $F(x;\xi,\eta)$ denotes the maximum $s-t$ flow in the network when the capacity of an arc (i, j) is $\eta\xi_{ij}(p_{ij} + x_{ij})$. Note that the above is a maximization rather than a minimization problem.

We assume that the random variables ξ_{ij}, θ are independent and uniformly distributed on $[0, 1]$ and $[0.5, 1]$, respectively, and consider the case of $p_{ij} = 0$, $c_{ij} = 1$ for all $(i, j) \in E$, and $b = 1$. We randomly generated 4 network instances (referred to as F1, F2, F3, and F4) using the network generator GRIDGEN available on DIMACS challenge. The push-relabel algorithm [8] was used to solve the second stage max-flow problem.

In the first test, each algorithm (N-SA, E-SA, SAA) was run once at each test instance; the results are reported in Table 4.4, where m, n stand for the number of nodes, respectively, arcs in G. Similar to the stochastic utility problem, we investigate the stability of the methods by running each of them 100 times. The resulting statistics is presented in Table 4.5, whose columns have exactly the same meaning as in Table 4.3.

This experiment fully supports the conclusions on the methods suggested by the experiments with the utility problem.

4.4. A network planning problem with random demand. In the last experiment, we consider the so-called SSN problem of Sen, Doverspike, and Cosares [24]. This problem arises in telecommunications network design where the owner of the network sells private-line services between pairs of nodes in the network, and the demands are treated as random variables based on the historical demand patterns.

TABLE 4.5
The variability for the stochastic max-flow problem.

-		N-SA			E-SA			SAA		
		Obj		Avg.	Obj		Avg.	Obj		Avg.
Inst	N	Mean	Dev	CPU	Mean	Dev	CPU	Mean	Dev	CPU
F2	1,000	0.0691	0.0004	3.11	0.0688	0.0006	4.62	0.0694	0.0003	90.15
F2	2,000	0.0694	0.0003	6.07	0.0692	0.0002	6.91	0.0695	0.0003	170.45

The optimization problem is to decide where to add capacity to the network to minimize the expected rate of unsatisfied demands. Since this problem has been studied by several authors (see, e.g., [12, 24]), it could be interesting to compare the results. Another purpose of this experiment is to investigate the behavior of the SA method when the Latin hyperplane sampling (LHS) variance reduction technique (introduced in [14]) is applied.

The problem has been formulated as a two-stage stochastic linear programming as follows:

$$\min_x \left\{ f(x) = \mathbb{E}[F(x,\xi)] : x \geq 0, \sum_i x_i = b \right\}, \tag{4.3}$$

where x is the vector of capacities to be added to the arcs of the network, b (the budget) is the total amount of capacity to be added, ξ denotes the random demand, and $F(x,\xi)$ represents the number of unserved requests, specifically,

$$F(x,\xi) = \min_{s,f} \left\{ \sum_i s_i : \begin{array}{l} \sum_i \sum_{r \in R(i)} A_r f_{ir} \leq x + c \\ \sum_{r \in R(i)} f_{ir} + s_i = \xi^i, \quad \forall i \\ f_{ir} \geq 0, s_i \geq 0, \quad \forall i, r \in R(i) \end{array} \right\}. \tag{4.4}$$

Here,

- $R(i)$ is the set of routes used for traffic i (traffic between the source-sink pair of nodes # i),
- ξ^i is the (random) demand for traffic i,
- A_r are the route-arc incidence vectors (so that jth component of A_r is 1 or 0 depending on whether arc j belongs to the route r),
- c is the vector of current capacities, f_{ir} is the fraction of traffic i transferred via route r, and s is the vector of unsatisfied demands.

In the SSN instance, there are $\dim x = 89$ arcs and $\dim \xi = 86$ source-sink pairs, and components of ξ are independent random variables with known discrete distributions (from 3 to 7 possible values per component), which result in $\approx 10^{70}$ possible demand scenarios.

In the first test with the SSN instance, each of our 3 algorithms was run once without and once with the LHS technique; the results are reported in Table 4.6. We then tested the stability of algorithms by running each of them 100 times; see statistics in Table 4.7. Note that experiments with the SSN problem were conducted on a more powerful computer: Intel Xeon 1.86GHz with Red Hat Enterprize Linux.

As far as comparison of our three algorithms is concerned, the conclusions are in full agreement with those for the utility and the max-flow problem. We also see that for our particular example, the LHS does not yield much of an improvement, especially when a larger sample size is applied. This result seems to be consistent with the observation in [12].

TABLE 4.6
SA versus SAA on the SSN problem.

-		Without LHS		With LHS	
Alg.	N	Obj	CPU	Obj	CPU
N-SA	100	11.0984	1	10.1024	1
	1,000	10.0821	6	10.0313	7
	2,000	9.9812	12	9.9936	12
	4,000	9.9151	23	9.9428	22
E-SA	100	10.9027	1	10.3860	1
	1,000	10.1268	6	10.0984	6
	2,000	10.0304	12	10.0552	12
	4,000	9.9662	23	9.9862	23
SAA	100	11.8915	24	11.0561	23
	1,000	10.0939	215	10.0488	216
	2,000	9.9769	431	9.9872	426
	4,000	9.8773	849	9.9051	853

TABLE 4.7
The variability for the SSN problem.

-		N-SA			E-SA			SAA		
		Obj		Avg.	Obj		Avg.	Obj		Avg.
N	LHS	Mean	Dev	CPU	Mean	Dev	CPU	Mean	Dev	CPU
1,000	no	10.0624	0.1867	6.03	10.1730	0.1826	6.12	10.1460	0.2825	215.06
1,000	yes	10.0573	0.1830	6.16	10.1237	0.1867	6.14	10.0135	0.2579	216.10
2,000	no	9.9965	0.2058	11.61	10.0853	0.1887	11.68	9.9943	0.2038	432.93
2,000	yes	9.9978	0.2579	11.71	10.0486	0.2066	11.74	9.9830	0.1872	436.94

4.5. N-SA versus E-SA. The data in Tables 4.3, 4.4, and 4.6 demonstrate that with the same sample size N, the N-SA somehow outperforms the E-SA in terms of both the quality of approximate solutions and the running time.[4] The difference in solutions' quality, at the first glance, seems slim, and one could think that adjusting the SA algorithm to the "geometry" of the problem in question (in our case, to minimization over a standard simplex) is of minor importance. We, however, do believe that such a conclusion would be wrong. In order to get a better insight, let us come back to the stochastic utility problem. This test problem has an important advantage—we can easily compute the value of the objective $f(x)$ at a given candidate solution x analytically.[5] Moreover, it is easy to minimize $f(x)$ over the simplex—on a closest inspection, this problem reduces to minimizing an easy-to-compute *univariate* convex function so that we can approximate the true optimal value f_* to high accuracy by bisection. Thus, in the case in question, we can compare solutions x generated by various algorithms in terms of their "true inaccuracy" $f(x) - f_*$, and this is the rationale behind our "Gaussian setup." We can now exploit this advantage of the stochastic utility problem for comparing properly N-SA and E-SA. In Table 4.8, we present the true values of the objective $f(\bar{x})$ at the approximate solutions $\bar{x}$ generated by N-SA and E-SA as applied to the instances L1 and L4 of the utility problem (cf. Table 4.3) along with the inaccuracies $f(\bar{x}) - f_*$ and the Monte Carlo estimates $\widehat{f}(\bar{x})$ of $f(\bar{x})$ obtained via 50,000-element samples. We see that the difference in

[4]The difference in running times can be easily explained: with X being a simplex, the prox-mapping for E-SA takes $O(n \ln n)$ operations versus $O(n)$ operations for N-SA.

[5]Indeed, $(\xi_1, \ldots, \xi_n) \sim \mathcal{N}(0, I_n)$, so that the random variable $\xi_x = \sum_i (a_i + \xi_i) x_i$ is normal with easily computable mean and variance, and since ϕ is piecewise linear, the expectation $f(x) = \mathbb{E}[\phi(\xi_x)]$ can be immediately expressed via the error function.

TABLE 4.8
N-SA versus E-SA.

Method	Problem	$\widehat{f}(\bar{x}), f(\bar{x})$	$f(\bar{x}) - f_*$	Time
N-SA, $N = 2{,}000$	L2: $n = 1{,}000$	$-5.9232/-5.9326$	0.0113	5.00
E-SA, $N = 2{,}000$	L2	$-5.8796/-5.8864$	0.0575	6.60
E-SA, $N = 10{,}000$	L2	$-5.9059/-5.9058$	0.0381	39.80
E-SA, $N = 20{,}000$	L2	$-5.9151/-5.9158$	0.0281	74.50
N-SA, $N = 2{,}000$	L4: $n = 5{,}000$	$-5.5855/-5.5867$	0.0199	25.00
E-SA, $N = 2{,}000$	L4	$-5.5467/-5.5469$	0.0597	44.60
E-SA, $N = 10{,}000$	L4	$-5.5810/-5.5812$	0.0254	165.10
E-SA, $N = 20{,}000$	L4	$-5.5901/-5.5902$	0.0164	382.00

the inaccuracy $f(\bar{x}) - f_*$ of the solutions produced by the algorithms is much more significant than is suggested by the data in Table 4.3 (where the actual inaccuracy is "obscured" by the estimation error and summation with f_*). Specifically, at the common for both algorithm sample sizes $N = 2{,}000$, the inaccuracy yielded by N-SA is 3–5 times less than the one for E-SA and in order to compensate for this difference, one should increase the sample size for E-SA (and hence the running time) by factor 5–10. It should be added that in light of theoretical complexity analysis carried out in Example 2.3, the outlined significant difference in performances of N-SA and E-SA is not surprising; the surprising fact is that E-SA works at all.

4.6. Bilinear matrix game. We consider here a bilinear matrix game

$$\min_{x\in X}\max_{y\in Y} y^T Ax,$$

where both feasible sets are the standard simplexes in $\mathbb{R}^n$: $Y = X = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1, x \geq 0\}$. We consider two versions of the randomized mirror SA algorithm (3.7), (3.8) for the saddle point problem. The first algorithm, the E-SA, uses $\frac{1}{2}\|x\|_2^2$ as ω_x, ω_y and $\|\cdot\|_2$ as $\|\cdot\|_x$, $\|\cdot\|_y$. The second algorithm, the N-SA, uses the entropy function (2.58) as ω_x, ω_y and the norm $\|\cdot\|_1$ as $\|\cdot\|_x$, $\|\cdot\|_y$. To compare the two procedures, we compute the corresponding approximate solutions $\tilde{z}_1^N$ and compute the exact values of the error:

$$\epsilon\left(\tilde{z}_1^N\right) = \max_{y\in Y} y^T A\tilde{x}_1^N - \min_{x\in X}\left[\tilde{y}_1^N\right]^T Ax, \quad i = 1, 2.$$

In our experiments we consider symmetric matrices A of two kinds. The matrices of the first family, parameterized by $\alpha > 0$, are given by

$$A_{ij} = \left(\frac{i+j-1}{2n-1}\right)^\alpha, \ \ 1 \leq i, j \leq n.$$

The second family of matrices, which are also parameterized by $\alpha > 0$, is given by

$$A_{ij} = \left(\frac{|i-j|+1}{2n-1}\right)^\alpha, \ \ 1 \leq i, j \leq n.$$

We use the notations $E_1(\alpha)$ and $E_2(\alpha)$ to refer to the experiments with the matrices of the first and second kind with parameter α. We present in Table 4.9 the results of experiments conducted for the matrices A of size $10^4 \times 10^4$. We made 100 simulation runs in each experiment and present the average error (column Mean), standard

TABLE 4.9
SA for bilinear matrix games.

	$E_2(2)$, $\epsilon(\tilde{z}_1)=0.500$			$E_2(1)$, $\epsilon(\tilde{z}_1)=0.500$			$E_2(0.5)$, $\epsilon(\tilde{z}_1)=0.390$		
N-SA	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU
N	Mean	Dev	CPU	Mean	Dev	CPU	Mean	Dev	CPU
100	0.0121	3.9 e−4	0.58	0.0127	1.9 e−4	0.69	0.0122	4.3 e−4	0.81
1,000	0.00228	3.7 e−5	5.8	0.00257	2.2 e−5	7.3	0.00271	4.5 e−5	8.5
2,000	0.00145	2.1 e−5	11.6	0.00166	1.0 e−5	13.8	0.00179	2.7 e−5	16.4
E-SA	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU
N	Mean	DEV	(Avg.)	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)
100	0.00952	1.0 e−4	1.27	0.0102	5.1 e−5	1.77	0.00891	1.1 e−4	1.94
1,000	0.00274	1.3 e−5	11.3	0.00328	7.8 e−6	17.6	0.00309	1.6 e−5	20.9
2,000	0.00210	7.4 e−6	39.7	0.00256	4.6 e−6	36.7	0.00245	7.8 e−6	39.2
	$E_1(2)$, $\epsilon(\tilde{z}_1)=0.0625$			$E_1(1)$, $\epsilon(\tilde{z}_1)=0.125$			$E_1(0.5)$, $\epsilon(\tilde{z}_1)=0.138$		
N-SA	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU
N	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)
100	0.00817	0.0016	0.58	0.0368	0.0068	0.66	0.0529	0.0091	0.78
1,000	0.00130	2.7 e−4	6.2	0.0115	0.0024	6.5	0.0191	0.0033	7.6
2,000	0.00076	1.6 e−4	11.4	0.00840	0.0014	11.7	0.0136	0.0018	13.8
E-SA	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU	$\epsilon\left(\tilde{z}_1^N\right)$		CPU
N	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)	Mean	Dev	(Avg.)
100	0.00768	0.0012	1.75	0.0377	0.0062	2.05	0.0546	0.0064	2.74
1,000	0.00127	2.2 e−4	17.2	0.0125	0.0022	19.9	0.0207	0.0020	18.4
2,000	0.00079	1.6 e−4	35.0	0.00885	0.0015	36.3	0.0149	0.0020	36.7

deviation (column Dev) and the average running time (with excluded time to compute the error of the resulting solution). For comparison, we also present the error of the initial solution $\tilde{z}_1 = (x_1, y_1)$.

Our basic observation is as follows: Both N-SA and E-SA succeed to reduce the solution error reasonably fast. The N-SA implementation is preferable as it is more efficient in terms of running time. For comparison, it takes MATLAB from 10 (for the simplest problem) to 35 seconds (for the hardest one) to compute just one answer $\mathrm{g}(x,y) = \begin{bmatrix} A^T y \\ -Ax \end{bmatrix}$ of the deterministic oracle.

5. Conclusions. It is shown that for a certain class of convex stochastic optimization and saddle point problems, robust versions of the SA approach have similar theoretical estimates of computational complexity, in terms of the required sample size, to the SAA method. Numerical experiments, reported in section 4, confirm this conclusion. These results demonstrate that for considered problems, a properly implemented mirror descent SA algorithm produces solutions of comparable accuracy to the SAA method for the same sample size of generated random points. On the other hand, the implementation (computational) time of the SA method is significantly smaller with a factor of up to 30–40 for considered problems. Thus, both theoretical and numerical results suggest that the robust mirror descent SA is a viable alternative to the SAA approach, an alternative which at least deserves testing in particular applications. It is also shown that the robust mirror SA approach can be applied as a randomization algorithm to large-scale deterministic saddle point problems (in particular, to minimax optimization problems and bilinear matrix games) with encouraging results.

6. Appendix. *Proof of Lemma* 2.1. Let $x \in X^o$ and $v = P_x(y)$. Note that $v \in \operatorname{argmin}_{z\in X}[\omega(z) + p^T z]$, where $p = \nabla\omega(x) - y$. Thus, ω is differentiable at v and $v \in$

X^o. As $\nabla_v V(x,v) = \nabla\omega(v) - \nabla\omega(x)$, the optimality conditions for (2.31) imply that

$$(\nabla\omega(v) - \nabla\omega(x) + y)^T(v-u) \le 0 \quad \forall u \in X. \tag{6.1}$$

For $u \in X$, we therefore have

$$\begin{aligned} V(v,u) - V(x,u) &= \big[\omega(u) - \nabla\omega(v)^T(u-v) - \omega(v)\big] \\ &\quad - \big[\omega(u) - \nabla\omega(x)^T(u-x) - \omega(x)\big] \\ &= \nabla\omega(v) - \nabla\omega(x) + y)^T(v-u) + y^T(u-v) \\ &\le y^T(u-v) - V(x,v), \end{aligned}$$

where the last inequality is due to (6.1). By Young's inequality,[6] we have

$$y^T(x-v) \le \frac{\|y\|_*^2}{2\alpha} + \frac{\alpha}{2}\|x-v\|^2,$$

while $V(x,v) \ge \frac{\alpha}{2}\|x-v\|^2$, due to the strong convexity of $V(x,\cdot)$. We get

$$\begin{aligned} V(v,u) - V(x,u) &\le y^T(u-v) - V(x,v) = y^T(u-x) + y^T(x-v) - V(x,v) \\ &\le y^T(u-x) + \frac{\|y\|_*^2}{2\alpha}, \end{aligned}$$

as required in (2.33). ∎

Entropy as a distance-generating function on the standard simplex. The only property which is not immediately evident is that the entropy $w(x) = \sum_{i=1}^n x_i \ln x_i$ is strongly convex, modulus 1 with respect to $\|\cdot\|_1$-norm, on the standard simplex $X = \big\{x \in \mathbb{R}^n : x \ge 0,\ \sum_{i=1}^n x_i\big\}$. We are in the situation where $X^o = \{x \in X : x > 0\}$ and in order to establish the property in question, it suffices to verify that $h^T\nabla^2\omega(x)h \ge \|h\|_1^2$ for every $x \in X^o$. Here is the computation:

$$\left[\sum_i |h_i|\right]^2 = \left[\sum_i (x_i^{-1/2}|h_i|)x_i^{1/2}\right]^2 \le \left[\sum_i h_i^2 x_i^{-1}\right]\left[\sum_i x_i\right] = \sum_i h_i^2 x_i^{-1} = h^T\nabla^2\omega(x)h,$$

where the inequality follows by Cauchy inequality.

Proof of Lemma 3.1. By (2.33), we have, for any $u \in Z$, that

$$\gamma_t(z_t - u)^T\mathsf{G}(z_t,\xi_t) \le V(z_t,u) - V(z_{t+1},u) + \frac{\gamma_t^2}{2}\|\mathsf{G}(z_t,\xi_t)\|_*^2 \tag{6.2}$$

(recall that we are in the situation of $\alpha = 1$). This relation implies that for every $u \in Z$, one has

$$\gamma_t(z_t - u)^T\mathsf{g}(z_t) \le V(z_t,u) - V(z_{t+1},u) + \frac{\gamma_t^2}{2}\|\mathsf{G}(z_t,\xi_t)\|_*^2 - \gamma_t(z_t-u)^T\Delta_t, \tag{6.3}$$

where $\Delta_t = \mathsf{G}(z_t,\xi_t) - \mathsf{g}(z_t)$. Summing up these inequalities over $t = 1,\dots,j$, we get

$$\sum_{t=1}^j \gamma_t(z_t-u)^T\mathsf{g}(z_t) \le V(z_1,u) - V(z_{t+1},u) + \sum_{t=1}^j \frac{\gamma_t^2}{2}\|\mathsf{G}(z_t,\xi_t)\|_*^2 - \sum_{t=1}^j \gamma_t(z_t-u)^T\Delta_t.$$

Now we need the following simple lemma.

[6] For any $u, v \in \mathbb{R}^n$, we have, by the definition of the dual norm, that $\|u\|_*\|v\| \ge u^Tv$, and hence $(\|u\|_*^2/\alpha + \alpha\|v\|^2)/2 \ge \|u\|_*\|v\| \ge u^Tv$.

LEMMA 6.1. *Let $\zeta_1, \dots, \zeta_j$ be a sequence of elements of $\mathbb{R}^{n+m}$. Define the sequence v_t, $t = 1, 2, \dots$ in Z^o as follows: $v_1 \in Z^o$ and*

$$v_{t+1} = P_{v_t}(\zeta_t),\ 1 \le t \le j.$$

Then, for any $u \in Z$, the following holds:

$$\sum_{t=1}^{j} \zeta_t^T (v_t - u) \le V(v_1, u) + \tfrac{1}{2} \sum_{t=1}^{j} \|\zeta_t\|_*^2. \tag{6.4}$$

Proof. Using the bound (2.33) of Lemma 2.1 with $y = \zeta_t$ and $x = v_t$ (so that $v_{t+1} = P_{v_t}(\zeta_t)$) and recalling that we are in the situation of $\alpha = 1$, we obtain the following for any $u \in Z$:

$$V(v_{t+1}, u) \le V(v_t, u) + \zeta_t^T (u - v_t) + \tfrac{1}{2} \|\zeta_t\|_*^2.$$

Summing up from $t = 1$ to $t = j$, we conclude that

$$V(v_{j+1}, u) \le V(v_1, u) + \sum_{t=1}^{j} \zeta_t^T (u - v_t) + \tfrac{1}{2} \sum_{t=1}^{j} \|\zeta_t\|_*^2,$$

which implies (6.4) due to $V(v, u) \ge 0$ for any $v \in Z^o, u \in Z$. ∎

Applying Lemma 6.1 with $v_1 = z_1$, $\zeta_t = -\gamma_t \Delta_t$, we get

$$\sum_{t=1}^{j} \gamma_t \Delta_t^T (u - v_t) \le V(z_1, u) + \frac{1}{2} \sum_{t=1}^{j} \gamma_t^2 \|\Delta_t\|_*^2 \ \ \forall u \in Z. \tag{6.5}$$

Observe that

$$\mathbb{E}\|\Delta_t\|_*^2 \le 4\mathbb{E}\|\mathsf{G}(z_t, \xi_t)\|_*^2 \le 4 \left(\frac{2D_{\omega_x, X}^2}{\alpha_x} M_{*,x}^2 + \frac{2D_{\omega_y, Y}^2}{\alpha_y} M_{*,y}^2 \right) = 4M_*^2$$

so that when taking the expectation of both sides of (6.5), we get

$$\mathbb{E}\left[\sup_{u \in Z} \left\{ \sum_{t=1}^{j} \gamma_t \Delta_t^T (u - v_t) \right\} \right] \le 1 + 2M_*^2 \sum_{t=1}^{j} \gamma_t^2 \tag{6.6}$$

(recall that $V(z_1, \cdot)$ is bounded by 1 on Z). Now we proceed exactly as in section 2.2: we sum up (6.3) from $t = 1$ to j to obtain

$$\begin{aligned} &\sum_{t=1}^{j} \gamma_t (z_t - u)^T \mathsf{g}(z_t) \le V(z_1, u) + \sum_{t=1}^{j} \frac{\gamma_t^2}{2} \|\mathsf{G}(z_t, \xi_t)\|_*^2 - \sum_{t=1}^{j} \gamma_t (z_t - u)^T \Delta_t \\ &= V(z_1, u) + \sum_{t=1}^{j} \frac{\gamma_t^2}{2} \|\mathsf{G}(z_t, \xi_t)\|_*^2 - \sum_{t=1}^{j} \gamma_t (z_t - v_t)^T \Delta_t + \sum_{t=1}^{j} \gamma_t (u - v_t)^T \Delta_t. \end{aligned} \tag{6.7}$$

When taking into account that z_t and v_t are deterministic functions of $\xi_{[t-1]} = (\xi_1, \dots, \xi_{t-1})$ and that the conditional expectation of Δ_t, $\xi_{[t-1]}$ being given, vanishes, we conclude that $\mathbb{E}[(z_t - v_t)^T \Delta_t] = 0$. We take now suprema in $u \in Z$ and then

expectations on both sides of (6.7):

$$\begin{aligned}\mathbb{E}\left[\sup_{u\in Z}\sum_{t=1}^{j}\gamma_t(z_t-u)^T\mathsf{g}(z_t)\right] &\le \sup_{u\in Z}V(z_1,u)+\sum_{t=1}^{j}\frac{\gamma_t^2}{2}\mathbb{E}\|\mathsf{G}(z_t,\xi_t)\|_*^2\\ &\quad+\sup_{u\in Z}\sum_{t=1}^{j}\gamma_t(u-v_t)^T\Delta_t\\ \text{(by (6.6))}\quad &\le 1+\frac{M_*^2}{2}\sum_{t=1}^{j}\gamma_t^2+\left[1+2M_*^2\sum_{t=1}^{j}\gamma_t^2\right]\\ &= 2+\frac{5}{2}M_*^2\sum_{t=1}^{j}\gamma_t^2,\end{aligned}$$

and we arrive at (3.10).

Proof of Propositions 2.2 and 3.2. We provide here the proof of Proposition 3.2 only. The proof of Proposition 2.2 follows the same lines and can be easily reconstructed using the bound (2.39) instead of the relations (6.5) and (6.7) in the proof below.

First of all, with M_* given by (3.6), one has

$$\forall(z\in Z):\mathbb{E}\left[\exp\{\|\mathsf{G}(z,\xi)\|_*^2/M_*^2\}\right]\le\exp\{1\}. \tag{6.8}$$

Indeed, setting $p_x=\frac{2D^2_{\omega_x,X}M^2_{*,x}}{\alpha_x M_*^2}$, $p_y=\frac{2D^2_{\omega_y,Y}M^2_{*,y}}{\alpha_y M_*^2}$, we have $p_x+p_y=1$, whence, invoking (3.4),

$$\mathbb{E}\left[\exp\{\|\mathsf{G}(z,\xi)\|_*^2/M_*^2\}\right]=\mathbb{E}\left[\exp\{p_x\|\mathsf{G}_x(z,\xi)\|_{*,x}^2/M_{*,x}^2+p_y\|\mathsf{G}_y(z,\xi)\|_{*,y}^2/M_{*,y}^2\}\right],$$

and (6.8) follows from (3.16) by the Hölder inequality.

Setting $\Gamma_N=\sum_{t=1}^N\gamma_t$ and using the notation from the proof of Lemma 3.1, relations (3.9), (6.5), and (6.7) combined with the fact that $V(z_1,u)\le 1$ for $u\in Z$, imply that

$$\Gamma_N\epsilon_\phi(\tilde z_N)\le 2+\underbrace{\tfrac12\sum_{t=1}^{N}\gamma_t^2\left[\|\mathsf{G}(z_t,\xi_t)\|_*^2+\|\Delta_t\|_*^2\right]}_{\alpha_N}+\underbrace{\sum_{t=1}^{N}\gamma_t(v_t-z_t)^T\Delta_t}_{\beta_N}. \tag{6.9}$$

Now, from (6.8), it follows straightforwardly that

$$\mathbb{E}\left[\exp\left\{\|\Delta_t\|_*^2/(2M_*)^2\right\}\right]\le\exp\{1\},\quad \mathbb{E}\left[\exp\left\{\|\mathsf{G}(z_t,\xi_t)\|_*^2/M_*^2\right\}\right]\le\exp\{1\}, \tag{6.10}$$

which, in turn, implies that

$$\mathbb{E}[\exp\{\alpha_N/\sigma_\alpha\}]\le\exp\{1\},\quad \sigma_\alpha=\frac{5}{2}M_*^2\sum_{t=1}^{N}\gamma_t^2, \tag{6.11}$$

and therefore, by Markov inequality, for any $\Omega>0$,

$$\mathrm{Prob}\{\alpha_N\ge(1+\Omega)\sigma_\alpha\}\le\exp\{-\Omega\}. \tag{6.12}$$

Indeed, we have by (6.8)

$$\|\mathsf{g}(z_t)\|_*=\|\mathbb{E}[\mathsf{G}(z_t,\xi_t)|\xi_{[t-1]}]\|_*\le\sqrt{\mathbb{E}(\|\mathsf{G}(z_t,\xi_t)\|_*^2|\xi_{[t-1]})}\le M_*$$

and

$$\|\Delta_t\|_*^2 = \|\mathsf{G}(z_t,\xi_t) - \mathsf{g}(z_t)\|_*^2 \le (\|\mathsf{G}(z_t,\xi_t)\|_* + \|\mathsf{g}(z_t)\|_*)^2 \le 2\|\mathsf{G}(z_t,\xi_t)\|_*^2 + 2M_*^2,$$

which implies that

$$\alpha_N \le \sum_{t=1}^N \frac{\gamma_t^2}{2}\left[3\|\mathsf{G}(z_t,\xi_t)\|_*^2 + 2M_*^2\right].$$

Further, by the Hölder inequality, we have the following from (6.8):

$$\mathbb{E}\left[\exp\left\{\frac{\gamma_t^2\left[\frac{3}{2}\|\mathsf{G}(z_t,\xi_t)\|_*^2 + M_*^2\right]}{\frac{5}{2}\gamma_t^2 M_*^2}\right\}\right] \le \exp(1).$$

Observe that if $r_1,\dots,r_i$ are nonnegative random variables such that $\mathbb{E}[\exp\{r_t/\sigma_t\}] \le \exp\{1\}$ for some deterministic $\sigma_t > 0$, then, by convexity of the exponent, $w(s) = \exp\{s\}$ and

$$\mathbb{E}\left[\exp\left\{\frac{\sum_{t\le i} r_t}{\sum_{t\le i}\sigma_t}\right\}\right] \le \mathbb{E}\left[\sum_{t\le i}\frac{\sigma_t}{\sum_{\tau\le i}\sigma_\tau}\exp\{r_t/\sigma_t\}\right] \le \exp\{1\}. \tag{6.13}$$

Now applying (6.13) with $r_t = \gamma_t^2\left[\frac{3}{2}\|\mathsf{G}(z_t,\xi_t)\|_*^2 + M_*^2\right]$ and $\sigma_t = \frac{5}{2}\gamma_t^2 M_*^2$, we obtain (6.11).

Now let $\zeta_t = \gamma_t(v_t - z_t)^T\Delta_t$. Observing that v_t, z_t are deterministic functions of $\xi_{[t-1]}$, while $\mathbb{E}[\Delta_t|\xi_{[t-1]}] = 0$, we see that the sequence $\{\zeta_t\}_{t=1}^N$ of random real variables forms a martingale difference. Besides this, by strong convexity of ω with modulus 1 w.r.t. $\|\cdot\|$ and due to $D_{\omega,Z} \le 1$, we have

$$u \in Z \Rightarrow 1 \ge V(z_1,u) \ge \frac{1}{2}\|u - z_1\|^2,$$

whence the $\|\cdot\|$-diameter of Z does not exceed $2\sqrt{2}$ so that $|\zeta_t| \le 2\sqrt{2}\gamma_t\|\Delta_t\|_*$, and therefore

$$\mathbb{E}\left[\exp\left\{|\zeta_t|^2/\left(32\gamma_t^2M_*^2\right)\right\}|\xi_{[t-1]}\right] \le \exp\{1\}$$

by (6.10). Applying Cramer's deviation bound, we obtain, for any $\Omega > 0$,

$$\text{Prob}\left\{\beta_N > 4\Omega M_*\sqrt{\sum_{t=1}^N \gamma_t^2}\right\} \le \exp\{-\Omega^2/4\}. \tag{6.14}$$

Indeed, for $0 \le \gamma$, setting $\sigma_t = 4\sqrt{2}\gamma_t M_*$ and taking into account that ζ_t is a deterministic function of $\xi_{[t]}$, with $\mathbb{E}[\zeta_t|\xi_{[t-1]}] = 0$ and $\mathbb{E}[\exp\{\zeta_t^2/\sigma_t^2\}|\xi_{[t-1]}] \le \exp\{1\}$, we have

$$\begin{aligned}
&0 < \gamma\sigma_t \le 1 \Rightarrow \quad \left(\text{as } e^x \le x + e^{x^2}\right)\\
&\mathbb{E}[\exp\{\gamma\zeta_t\}|\xi_{[t-1]}] \le \mathbb{E}[\exp\left\{\gamma^2\zeta_t^2\right\}|\xi_{[t-1]}]\\
&\qquad\qquad \le \mathbb{E}\left[\left(\exp\left\{\zeta_t^2/\sigma_t^2\right\}\right)^{\gamma^2\sigma_t^2}|\xi_{[t-1]}\right] \le \exp\left\{\gamma^2\sigma_t^2\right\};\\
&\gamma\sigma_t > 1 \Rightarrow\\
&\mathbb{E}[\exp\{\gamma\zeta_t\}|\xi_{[t-1]}] \le \mathbb{E}\left[\exp\left\{\left[\tfrac{1}{2}\gamma^2\sigma_t^2 + \tfrac{1}{2}\zeta_t^2/\sigma_t^2\right\}|\xi_{[t-1]}\right]\\
&\qquad\qquad \le \exp\left\{\tfrac{1}{2}\gamma^2\sigma_t^2 + \tfrac{1}{2}\right\} \le \exp\left\{\gamma^2\sigma_t^2\right\},
\end{aligned}$$

that is, in both cases, $\mathbb{E}[\exp\{\gamma\zeta_t\}|\xi_{[t-1]}] \le \exp\{\gamma^2\sigma_t^2\}$. Therefore,

$$\mathbb{E}[\exp\{\gamma\beta_i\}] = \mathbb{E}\left[\exp\{\gamma\beta_{i-1}\}\mathbb{E}[\exp\{\gamma\zeta_i\}|\xi_{[i-1]}]\right] \le \exp\left\{\gamma^2\sigma_i^2\right\}\mathbb{E}[\exp\{\gamma\beta_{i-1}\}],$$

whence $\mathbb{E}[\exp\{\gamma\beta_N\}] \le \exp\{\gamma^2\sum_{t=1}^N\sigma_t^2\}$, and thus, by Markov inequality for every $\Omega > 0$, it holds

$$\mathrm{Prob}\left\{\beta_N > \Omega\sqrt{\sum_{t=1}^N\sigma_t^2}\right\} \le \exp\left\{\gamma^2\sum_{t=1}^N\sigma_t^2\right\}\exp\left\{-\gamma\Omega\sqrt{\sum_{t=1}^N\sigma_t^2}\right\}.$$

When choosing $\gamma = \frac{1}{2}\Omega\left(\sum_{t=1}^N\sigma_t^2\right)^{-1/2}$, we arrive at (6.14).

Combining (6.9), (6.10), and (6.14), we get the following for any positive Ω and Θ:

$$\mathrm{Prob}\left\{\Gamma_N\,\epsilon_\phi(\tilde z_t) > 2 + \frac{5}{2}(1+\Omega)M_*^2\sum_{t=1}^N\gamma_t^2 + 4\sqrt{2}\Theta M_*\sqrt{\sum_{t=1}^N\gamma_t^2}\right\} \le \exp\{-\Omega\} + \exp\left\{-\frac{1}{4}\Theta^2\right\}.$$

When setting $\Theta = 2\sqrt{\Omega}$ and substituting (3.12), we obtain (3.17). ∎

Proof of Proposition 3.3. As in the proof of Proposition 3.2, when setting $\Gamma_N = \sum_{t=1}^N\gamma_t$ and using the relations (3.9), (6.5), and (6.7), combined with the fact that $\|\mathsf{G}(z,\xi_y)\|_* \le M_*$, we obtain

$$\begin{aligned}\Gamma_N\,\epsilon_\phi(\tilde z_N) &\le 2 + \sum_{t=1}^N\frac{\gamma_t^2}{2}\left[\|\mathsf{G}(z_t,\xi_t)\|_*^2 + \|\Delta_t\|_*^2\right] + \sum_{t=1}^N\gamma_t(v_t - z_t)^T\Delta_t \\ &\le 2 + \frac{5}{2}M_*^2\sum_{t=1}^N\gamma_t^2 + \underbrace{\sum_{t=1}^N\gamma_t(v_t - z_t)^T\Delta_t}_{\alpha_N}. \qquad (6.15)\end{aligned}$$

Recall that by definition of Δ_t, $\|\Delta_t\|_* = \|\mathsf{G}(z_t,\xi_t) - \mathsf{g}(z_t)\|_* \le \|\mathsf{G}(z_t,\xi_t)\| + \|\mathsf{g}(z_t)\|_* \le 2M_*$.

Note that $\zeta_t = \gamma_t(v_t - z_t)^T\Delta_t$ is a bounded martingale difference, i.e., $\mathbb{E}(\zeta_t|\xi_{[t-1]}) = 0$ and $|\zeta_t| \le 4\gamma_t\overline{M}$ (here $\overline{M}$ is defined in (3.31)). Then, by Azuma–Hoeffding's inequality [1] for any $\Omega \ge 0$,

$$\mathrm{Prob}\left(\alpha_N > 4\Omega\overline{M}\sqrt{\sum_{t=1}^N\gamma_t^2}\right) \le e^{-\Omega^2/2}. \qquad (6.16)$$

Indeed, let us denote $v_t = (v_t^{(x)}, v_t^{(y)})$ and $\Delta_t = (\Delta_t^{(x)}, \Delta_t^{(y)})$. When taking into account that $\|v_t^{(x)}\|_1 \le 1$, $\|v_t^{(y)}\|_1 \le 1$ and $\|x_t\|_1 \le 1$, $\|y_t\|_1 \le 1$, we conclude that

$$\begin{aligned}|(v_t - z_t)^T\Delta_t| &\le \left|\left(v_t^{(x)} - x_t\right)^T\Delta_t^{(x)}\right| + \left|\left(v_t^{(y)} - y_t\right)^T\Delta_t^{(y)}\right| \\ &\le 2\left\|\Delta_t^{(x)}\right\|_\infty + 2\left\|\Delta_t^{(y)}\right\|_\infty \le 4\max_{1\le j\le m}\|A^j + b\|_\infty + 4\max_{1\le j\le n}\|A_j + c\|_\infty \\ &= 4\overline{M}.\end{aligned}$$

We conclude from (6.15) and (6.16) that

$$\text{Prob}\left(\Gamma_N\,\epsilon_\phi(\tilde{z}_N) > 2 + \frac{5}{2}M_*^2\sum_{t=1}^N\gamma_t^2 + 4\Omega\overline{M}\sqrt{\sum_{t=1}^N\gamma_t^2}\right) \le e^{-\Omega^2/2},$$

and the bound (3.30) of the proposition can be easily obtained by substituting the constant stepsizes γ_t as defined in (3.12). □

REFERENCES

[1] K. Azuma, *Weighted sums of certain dependent random variables*, Tökuku Math. J., 19 (1967), pp. 357–367.

[2] A. Ben-Tal and A. Nemirovski, *Non-Euclidean restricted memory level method for large-scale convex optimization*, Math. Program., 102 (2005), pp. 407–456.

[3] A. Benveniste, M. Métivier, and P. Priouret, *Algorithmes Adaptatifs et Approximations Stochastiques*, Masson, Paris, 1987 (in French). Adaptive Algorithms and Stochastic Approximations, Springer, New York, 1993 (in English).

[4] L.M. Bregman, *The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming*, Comput. Math. Math. Phys., 7 (1967), pp. 200–217.

[5] K.L. Chung, *On a stochastic approximation method*, Ann. Math. Statist., 25 (1954), pp. 463–483.

[6] Y. Ermoliev, *Stochastic quasigradient methods and their application to system optimization*, Stochastics, 9 (1983), pp. 1–36.

[7] A.A. Gaivoronski, *Nonstationary stochastic programming problems*, Kybernetika, 4 (1978), pp. 89–92.

[8] A.V. Goldberg and R.E. Tarjan, *A new approach to the maximum flow problem*, J. ACM, 35 (1988), pp. 921–940.

[9] M.D. Grigoriadis and L.G. Khachiyan, *A sublinear-time randomized approximation algorithm for matrix games*, Oper. Res. Lett., 18 (1995), pp. 53–58.

[10] A. Juditsky, A. Nazin, A. Tsybakov, and N. Vayatis, *Recursive aggregation of estimators by the mirror descent algorithm with averaging*, Probl. Inf. Transm., 41 (2005), pp. 368–384.

[11] A.J. Kleywegt, A. Shapiro, and T. Homem-de-Mello, *The sample average approximation method for stochastic discrete optimization*, SIAM J. Optim., 12 (2002), pp. 479–502.

[12] J. Linderoth, A. Shapiro, and S. Wright, *The empirical behavior of sampling methods for stochastic programming*, Ann. Oper. Res., 142 (2006), pp. 215–241.

[13] W.K. Mak, D.P. Morton, and R.K. Wood, *Monte Carlo bounding techniques for determining solution quality in stochastic programs*, Oper. Res. Lett., 24 (1999), pp. 47–56.

[14] M.D. McKay, R.J. Beckman, and W.J. Conover, *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code*, Technometrics, 21 (1979), pp. 239–245.

[15] A. Nemirovski and D. Yudin, *On Cezari's convergence of the steepest descent method for approximating saddle point of convex-concave functions*, Dokl. Akad. Nauk SSSR, 239 (1978) (in Russian). Soviet Math. Dokl., 19 (1978) (in English).

[16] A. Nemirovski and D. Yudin, *Problem Complexity and Method Efficiency in Optimization*, Wiley-Intersci. Ser. Discrete Math. 15, John Wiley, New York, 1983.

[17] Y. Nesterov, *Primal-dual subgradient methods for convex problems*, Math. Program., Ser. B, http://www.springerlink.com content-b441795t5254m533.

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

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

[20] G.C. Pflug, *Optimization of Stochastic Models*, The Interface Between Simulation and Optimization, Kluwer, Boston, 1996.

[21] H. Robbins and S. Monro, *A stochastic approximation method*, Ann. Math. Stat., 22 (1951), pp. 400–407.

[22] A. Ruszczyński and W. Syski, *A method of aggregate stochastic subgradients with on-line stepsize rules for convex stochastic programming problems*, Math. Prog. Stud., 28 (1986), pp. 113–131.

[23] J. Sacks, *Asymptotic distribution of stochastic approximation*, Ann. Math. Stat., 29 (1958), pp. 373–409.
[24] S. Sen, R.D. Doverspike, and S. Cosares, *Network planning with random demand*, Telecomm. Syst., 3 (1994), pp. 11–30.
[25] A. Shapiro, *Monte Carlo sampling methods*, in Stochastic Programming, Handbook in OR & MS, Vol. 10, A. Ruszczyński and A. Shapiro, eds., North-Holland, Amsterdam, 2003.
[26] A. Shapiro and A. Nemirovski, *On complexity of stochastic programming problems*, in Continuous Optimization: Current Trends and Applications, V. Jeyakumar and A.M. Rubinov, eds., Springer, New York, 2005, pp. 111–144.
[27] J.C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*, John Wiley, Hoboken, NJ, 2003.
[28] V. Strassen, *The existence of probability measures with given marginals*, Ann. Math. Statist., 38 (1965), pp. 423–439.
[29] B. Verweij, S. Ahmed, A.J. Kleywegt, G. Nemhauser, and A. Shapiro, *The sample average approximation method applied to stochastic routing problems: A computational study*, Comput. Optim. Appl., 24 (2003), pp. 289–333.