Bregman Divergence for Stochastic Variance Reduction: Saddle-Point and Adversarial Prediction

Zhan Shi **Xinhua Zhang**
University of Illinois at Chicago
Chicago, Illinois 60661
{zshi22,zhangx}@uic.edu

Yaoliang Yu
University of Waterloo
Waterloo, ON, N2L3G1
yaoliang.yu@uwaterloo.ca

Abstract

Adversarial machines, where a learner competes against an adversary, have regained much recent interest in machine learning. They are naturally in the form of saddle-point optimization, often with *separable* structure but sometimes also with unmanageably large dimension. In this work we show that adversarial prediction under multivariate losses can be solved much faster than they used to be. We first reduce the problem size *exponentially* by using appropriate sufficient statistics, and then we adapt the new stochastic variance-reduced algorithm of Balamurugan & Bach (2016) to allow any Bregman divergence. We prove that the same linear rate of convergence is retained and we show that for adversarial prediction using KL-divergence we can further achieve a speedup of #example times compared with the Euclidean alternative. We verify the theoretical findings through extensive experiments on two example applications: adversarial prediction and LPboosting.

1 Introduction

Many algorithmic advances have been achieved in machine learning by finely leveraging the *separability* in the model. For example, stochastic gradient descent (SGD) algorithms typically exploit the fact that the objective is an expectation of a random function, with each component corresponding to a training example. A "dual" approach partitions the problem into blocks of coordinates and processes them in a stochastic fashion [1]. Recently, by exploiting the finite-sum structure of the model, variance-reduction based stochastic methods have surpassed the well-known sublinear lower bound of SGD. Examples include SVRG [2], SAGA [3], SAG [4], Finito [5], MISO [6], and SDCA [7, 8], just to name a few. Specialized algorithms have also been proposed for accommodating proximal terms [9], and for further acceleration through the condition number [10–13].

However, not all empirical risks are separable in its plain form, and in many cases dualization is necessary for achieving separability. This leads to a composite saddle-point problem with convex-concave (saddle) functions K and M:

$$(x^*, y^*) = \arg\min_x \max_y K(x,y) + M(x,y), \text{ where } K(x,y) = \tfrac{1}{n}\textstyle\sum_{k=1}^n \psi_k(x,y). \quad (1)$$

Most commonly used supervised losses for linear models can be written as $g^\star(X\mathbf{w})$, where $g^\star$ is the Fenchel dual of a convex function g, X is the design matrix, and $\mathbf{w}$ is the model vector. So the regularized risk minimization can be naturally written as $\min_{\mathbf{w}} \max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}' X\mathbf{w} + \Omega(\mathbf{w}) - g(\boldsymbol{\alpha})$, where Ω is a regularizer. This fits into our framework (1) with a bilinear function K and a decoupled function M. Optimization for this specific form of saddle-point problems has been extensively studied. For example, [14] and [15] performed batch updates on $\mathbf{w}$ and stochastic updates on $\boldsymbol{\alpha}$, while [16] and [17] performed *doubly* stochastic updates on *both* $\mathbf{w}$ and $\boldsymbol{\alpha}$, achieving $O(\frac{1}{\epsilon})$ and $O(\log\frac{1}{\epsilon})$ rates respectively. The latter two also studied the more general form (1). Our interest in this paper is double stochasticity, aiming to maximally harness the power of separability and stochasticity.

Adversarial machines, where the learner competes against an adversary, have re-gained much recent interest in machine learning [18–20]. On one hand they fit naturally into the saddle-point optimization framework (1) but on the other hand they are known to be notoriously challenging to solve. The central message of this work is that certain adversarial machines can be solved significantly faster than they used to be. Key to our development is a new extension of the stochastic variance-reduced algorithm in [17] such that it is compatible with any Bregman divergence, hence opening the possibility to largely reduce the *quadratic* condition number in [17] by better adapting to the underlying geometry using non-Euclidean norms and Bregman divergences.

Improving condition numbers by Bregman divergence has long been studied in (stochastic, proximal) gradient descent [21, 22]. The best known algorithm is arguably stochastic mirror descent [23], which was extended to saddle-points by [16] and to ADMM by [24]. However, they can only achieve the sublinear rate $O(1/\epsilon)$ (for an ϵ-accurate solution). On the other hand, many recent stochastic variance-reduced methods [2–6, 9, 17] that achieve the much faster linear rate $O(\log 1/\epsilon)$ rely inherently on the Euclidean structure, and their extension to Bregman divergence, although conceptually clear, remains challenging in terms of the analysis. For example, the analysis of [17] relied on the resolvent of monotone operators [25] and is hence restricted to the Euclidean norm. In §2 we extend the notion of Bregman divergence to saddle functions and we prove a new Pythagorean theorem that may be of independent interest for analyzing first order algorithms. In §4 we introduce a fundamentally different proof technique (details relegated to Appendix C) that overcomes several challenges arising from a general Bregman divergence (e.g. asymmetry and unbounded gradient on bounded domain), and we recover similar quantitative linear rate of convergence as [17] but with the flexibility of using suitable Bregman divergences to reduce the condition number.

The new stochastic variance-reduced algorithm Breg-SVRG is then applied to the adversarial prediction framework (with multivariate losses such as F-score) [19, 20]. Here we make three novel contributions: (a) We provide a significant reformulation of the adversarial prediction problem that reduces the dimension of the optimization variable from 2^n to n^2 (where n is the number of samples), hence making it amenable to stochastic variance-reduced optimization (§3). (b) We develop a new efficient algorithm for computing the proximal update with a separable saddle KL-divergence (§5). (c) We verify that Breg-SVRG accelerates its Euclidean alternative by a factor of n in both theory and practice (§6), hence confirming again the uttermost importance of adapting to the underlying problem geometry. To our best knowledge, this is the first time stochastic variance-reduced methods have been shown with great promise in optimizing adversarial machines.

Finally, we mention that we expect our algorithm Breg-SVRG to be useful for solving many other saddle-point problems, and we provide a second example (LPboosting) in experiments (§6).

2 Bregman Divergence and Saddle Functions

In this section we set up some notations, recall some background materials, and extend Bregman divergences to saddle functions, a key notion in our later analysis.

Bregman divergence. For any convex and differentiable function ψ over some closed convex set $C \subseteq \mathbb{R}^d$, its induced Bregman divergence is defined as:

$$\forall x \in \operatorname{int}(C), x' \in C, \;\; \Delta_\psi(x', x) := \psi(x') - \psi(x) - \langle \nabla\psi(x), x' - x \rangle , \tag{2}$$

where $\nabla\psi$ is the gradient and $\langle \cdot, \cdot \rangle$ is the standard inner product in $\mathbb{R}^d$. Clearly, $\Delta_\psi(x', x) \geq 0$ since ψ is convex. We mention two familiar examples of Bregman divergence.

- Squared Euclidean distance: $\Delta_\psi(x', x) = \frac{1}{2}\|x' - x\|_2^2, \psi(x) = \frac{1}{2}\|x\|_2^2$, where $\|\cdot\|_2$ is ℓ_2 norm.
- (Unnormalized) KL-divergence: $\Delta_\psi(x', x) = \sum_i x'_i \log \frac{x'_i}{x_i} - x'_i + x_i, \psi(x) = \sum_i x_i \log x_i$.

Strong convexity. Following [26] we call a function f ψ-convex if $f - \psi$ is convex, i.e. for all x, x'

$$f(x') \geq f(x) + \langle \partial f(x), x' - x \rangle + \Delta_\psi(x', x). \tag{3}$$

Smoothness. A function f is L-smooth wrt a norm $\|\cdot\|$ if its gradient ∇f is L-Lipschitz continuous, i.e., for all x and x', $\|\nabla f(x') - \nabla f(x)\|_* \leq L\|x' - x\|$, where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. The change of a smooth function, in terms of its induced Bregman divergence, can be upper bounded by the change of its input and lower bounded by the change of its slope, cf. Lemma 2 in Appendix A.

Saddle functions. Recall that a function $\phi(x, y)$ over $C_{\mathsf{z}} = C_{\mathsf{x}} \times C_{\mathsf{y}}$ is called a saddle function if it is convex in x for any $y \in C_{\mathsf{y}}$, and concave in y for any $x \in C_{\mathsf{x}}$. Given a saddle function ϕ, we call (x^*, y^*) its saddle point if

$$\forall x \in C_{\mathsf{x}},\ \forall y \in C_{\mathsf{y}}, \quad \phi(x^*, y) \leq \phi(x^*, y^*) \leq \phi(x, y^*), \tag{4}$$

or equivalently $(x^*, y^*) \in \arg\min_{x \in C_{\mathsf{x}}} \max_{y \in C_{\mathsf{y}}} \phi(x, y)$. Assuming ϕ is differentiable, we denote

$$\mathsf{G}_\phi(x, y) := [\partial_x \phi(x, y); -\partial_y \phi(x, y)]. \tag{5}$$

Note the negation sign due to the concavity in y. We can quantify the notion of "saddle": A function $f(x, y)$ is called ϕ-saddle iff $f - \phi$ is a saddle function, or equivalently, $\Delta_f(z', z) \geq \Delta_\phi(z', z)$ (see below). Note that any saddle function ϕ is 0-saddle and ϕ-saddle.

Bregman divergence for saddle functions. We now define the Bregman divergence induced by a saddle function ϕ: for $z = (x, y)$ and $z' = (x', y')$ in C_{z},

$$\Delta_\phi(z', z) := \Delta_{\phi_y}(x', x) + \Delta_{-\phi_x}(y', y) = \phi(x', y) - \phi(x, y') - \langle \mathsf{G}_\phi(z), z' - z \rangle, \tag{6}$$

where $\phi_y(x) = \phi(x, y)$ is a convex function of x for any fixed y, and similarly $\phi_x(y) = \phi(x, y)$ is a concave (hence the negation) function of y for any fixed x. The similarity between (6) and the usual Bregman divergence Δ_ψ in (2) is apparent. However, ϕ is never evaluated at z' but z (for G) and the cross pairs (x', y) and (x, y'). Key to our subsequent analysis is the following lemma that extends a result of [27] to saddle functions (proof in Appendix A).

Lemma 1. *Let f and g be ϕ-saddle and φ-saddle respectively, with one of them being differentiable. Then, for any $z = (x, y)$ and any saddle point (if exists) $z^* := (x^*, y^*) \in \arg\min_x \max_y \{f(z) + g(z)\}$, we have $f(x, y^*) + g(x, y^*) \geq f(x^*, y) + g(x^*, y) + \Delta_{\phi+\varphi}(z, z^*)$.*

Geometry of norms. In the sequel, we will design two convex functions $\psi_{\mathsf{x}}(x)$ and $\psi_{\mathsf{y}}(y)$ such that their induced Bregman divergences are "distance enforcing" (a.k.a. 1-strongly convex), that is, w.r.t. two norms $\|\cdot\|_{\mathsf{x}}$ and $\|\cdot\|_{\mathsf{y}}$ that we also design, the following inequality holds:

$$\Delta_{\mathsf{x}}(x, x') := \Delta_{\psi_{\mathsf{x}}}(x, x') \geq \tfrac{1}{2} \|x - x'\|_{\mathsf{x}}^2, \ \Delta_{\mathsf{y}}(y, y') := \Delta_{\psi_{\mathsf{y}}}(y, y') \geq \tfrac{1}{2} \|y - y'\|_{\mathsf{y}}^2. \tag{7}$$

Further, for $z = (x, y)$, we define

$$\Delta_{\mathsf{z}}(z, z') := \Delta_{\psi_{\mathsf{x}} - \psi_{\mathsf{y}}}(z, z') \geq \tfrac{1}{2} \|z - z'\|_{\mathsf{z}}^2, \quad \text{where} \quad \|z\|_{\mathsf{z}}^2 := \|x\|_{\mathsf{x}}^2 + \|y\|_{\mathsf{y}}^2 \tag{8}$$

When it is clear from the context, we simply omit the subscripts and write Δ, $\|\cdot\|$, and $\|\cdot\|_*$.

3 Adversarial Prediction under Multivariate Loss

A number of saddle-point based machine learning problems have been listed in [17]. Here we give another example (adversarial prediction under multivariate loss) that is naturally formulated as a saddle-point problem but also requires a careful adaptation to the underlying geometry—a challenge that was not addressed in [17] since their algorithm inherently relies on the Euclidean norm. We remark that adaptation to the underlying geometry has been studied in the (stochastic) mirror descent framework [23], with significant improvements on condition numbers or gradient norm bounds. Surprisingly, no analogous efforts have been attempted in the stochastic variance reduction framework—a gap we intend to fill in this work.

The adversarial prediction framework [19, 20, 28], arising naturally as a saddle-point problem, is a convex alternative to the generative adversarial net [18]. Given a training sample $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ and $\tilde{\mathbf{y}} = [\tilde{y}_1, \ldots, \tilde{y}_n] \in \{0, 1\}^n$, adversarial prediction optimizes the following saddle function that is an expectation of some multivariate loss $\ell(\mathbf{y}, \mathbf{z})$ (e.g. F-score) over the labels $\mathbf{y}, \mathbf{z} \in \{0, 1\}^n$ of all data points:

$$\min_{p \in \Delta^{2^n}} \left[\max_{q \in \Delta^{2^n}} \mathop{\mathbb{E}}_{\mathbf{y} \sim p, \mathbf{z} \sim q} \ell(\mathbf{y}, \mathbf{z}), \ \text{s.t.} \mathop{\mathbb{E}}_{\mathbf{z} \sim q} \left(\tfrac{1}{n} X \mathbf{z}\right) = \tfrac{1}{n} X \tilde{\mathbf{y}} \right] \tag{9}$$

Here the proponent tries to find a distribution $p(\cdot)$ over the labeling on the entire training set in order to minimize the loss (Δ^{2^n} is the 2^n dimensional probability simplex). An opponent in contrast tries to maximize the expected loss by finding another distribution $q(\cdot)$, but his strategy is subject to the constraint that the feature expectation matches that of the empirical distribution. Introducing a

Lagrangian variable $\boldsymbol{\theta}$ to remove the feature expectation constraint and specializing the problem to F-score where $\ell(\mathbf{y},\mathbf{z}) = \frac{2\mathbf{y}'\mathbf{z}}{\mathbf{1}'\mathbf{y}+\mathbf{1}'\mathbf{z}}$ and $\ell(\mathbf{0},\mathbf{0}) := 1$, the partial dual problem can be written as

$$\max_{\boldsymbol{\theta}} \; -\tfrac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 + \tfrac{1}{n}\boldsymbol{\theta}' X\tilde{\mathbf{y}} + \min_{p\in\Delta^{2^n}} \max_{q\in\Delta^{2^n}} \mathop{\mathbb{E}}_{\mathbf{y}\sim p,\mathbf{z}\sim q}\left[\tfrac{2\mathbf{y}'\mathbf{z}}{\mathbf{1}'\mathbf{y}+\mathbf{1}'\mathbf{z}} - \tfrac{1}{n}\boldsymbol{\theta}'X\mathbf{y}\right], \tag{10}$$

where we use $\mathbf{y}'\mathbf{z}$ to denote the standard inner product and we followed [19] to add an ℓ_2^2 regularizer on $\boldsymbol{\theta}$ penalizing the dual variables on the constraints over the training data. It appears that solving (10) can be quite challenging, because the variables p and q in the inner minimax problem have 2^n entries! A constraint sampling algorithm was adopted in [19] to address this challenge, although no formal guarantee was established. Note that we can maximize the outer unconstrained variable $\boldsymbol{\theta}$ (with dimension the same as the number of features) relatively easily using for instance gradient ascent, provided that we can solve the inner minimax problem quickly—a significant challenge to which we turn our attention below.

Surprisingly, we show here that the inner minimax problem in (10) can be significantly simplified. The key observation is that the expectation in the objective depends only on a few sufficient statistics of p and q. Indeed, by interpreting p and q as probability distributions over $\{0,1\}^n$ we have:

$$\mathbb{E}\frac{2\mathbf{y}'\mathbf{z}}{\mathbf{1}'\mathbf{y}+\mathbf{1}'\mathbf{z}} = p(\{\mathbf{0}\})q(\{\mathbf{0}\}) + \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{E}\left(\tfrac{2\mathbf{y}'\mathbf{z}}{\mathbf{1}'\mathbf{y}+\mathbf{1}'\mathbf{z}}[\![\mathbf{1}'\mathbf{y}=i]\!][\![\mathbf{1}'\mathbf{z}=j]\!]\right) \tag{11}$$

$$= p(\{\mathbf{0}\})q(\{\mathbf{0}\}) + \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{2ij}{i+j}\cdot\underbrace{\frac{1}{i}\mathbb{E}\left(\mathbf{y}[\![\mathbf{1}'\mathbf{y}=i]\!]\right)}_{\boldsymbol{\alpha}_i}{}'\cdot\underbrace{\frac{1}{j}\mathbb{E}\left(\mathbf{z}[\![\mathbf{1}'\mathbf{z}=j]\!]\right)}_{\boldsymbol{\beta}_j}, \tag{12}$$

where $[\![\cdot]\!] = 1$ if $\cdot$ is true, and 0 otherwise. Crucially, the variables $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_j$ are sufficient for re-expressing (10), since

$$\mathbf{1}'\boldsymbol{\alpha}_i = \frac{1}{i}\mathbb{E}\left(\mathbf{1}'\mathbf{y}[\![\mathbf{1}'\mathbf{y}=i]\!]\right) = \mathbb{E}[\![\mathbf{1}'\mathbf{y}=i]\!] = p(\{\mathbf{1}'\mathbf{y}=i\}), \tag{13}$$

$$\sum_i i\boldsymbol{\alpha}_i = \sum_i \mathbb{E}\left(\mathbf{y}[\![\mathbf{1}'\mathbf{y}=i]\!]\right) = \mathbb{E}\mathbf{y}, \tag{14}$$

and similar equalities also hold for $\boldsymbol{\beta}_j$. In details, the inner minimax problem of (10) simplifies to:

$$\min_{\boldsymbol{\alpha}\in S}\max_{\boldsymbol{\beta}\in S}\; \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\big[\underbrace{\tfrac{2ijn^2}{i+j}\boldsymbol{\alpha}_i'\boldsymbol{\beta}_j + n^2\boldsymbol{\alpha}_i'\mathbf{1}\mathbf{1}'\boldsymbol{\beta}_j}_{f_{ij}(\boldsymbol{\alpha}_i,\boldsymbol{\beta}_j)} - n\mathbf{1}'\boldsymbol{\alpha}_i - n\mathbf{1}'\boldsymbol{\beta}_j - \boldsymbol{\theta}'Xi\boldsymbol{\alpha}_i\big] + \Omega(\boldsymbol{\alpha}) - \Omega(\boldsymbol{\beta}), \tag{15}$$

$$\text{where } S = \{\boldsymbol{\alpha}\geq\mathbf{0} : \mathbf{1}'\boldsymbol{\alpha}\leq 1, \forall i,\; \|i\boldsymbol{\alpha}_i\|_\infty \leq \|\boldsymbol{\alpha}_i\|_1\}, \quad \Omega(\boldsymbol{\alpha}) = \mu\textstyle\sum_{i,j}\alpha_{ij}\log(\alpha_{ij}). \tag{16}$$

Importantly, $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1;\ldots,\boldsymbol{\alpha}_n]$ (resp. $\boldsymbol{\beta}$) has n^2 entries, which is significantly smaller than the 2^n entries of p (resp. q) in (10). For later purpose we have also incorporated an entropy regularizer for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively in (15).

To justify the constraint set S, note from (12) and (13) that for any distribution p of $\mathbf{y}$:

$$\text{since } \boldsymbol{\alpha}\geq 0 \text{ and } \mathbf{y}\in\{0,1\}^n,\; \|i\boldsymbol{\alpha}_i\|_\infty \leq \mathbb{E}\|\mathbf{y}[\![\mathbf{1}'\mathbf{y}=i]\!]\|_\infty \leq \mathbb{E}[\![\mathbf{1}'\mathbf{y}=i]\!] = \|\boldsymbol{\alpha}_i\|_1. \tag{17}$$

Conversely, for any $\boldsymbol{\alpha}\in S$, we can construct a distribution p such that $i\alpha_{ij} = \mathbb{E}\left(y_j[\![\mathbf{1}'\mathbf{y}=i]\!]\right) = p(\{1'\mathbf{y}=i, y_j=1\})$ in the following algorithmic way: Fix i and for each j define $Y_j = \{\mathbf{y}\in\{0,1\}^n : \mathbf{1}'\mathbf{y}=i, y_j=1\}$. Let $U=\{1,\ldots,n\}$. Find an index j in U that minimizes α_{ij} and set $p(\{\mathbf{y}\}) = i\alpha_{ij}/|Y_j|$ for each $\mathbf{y}\in Y_j$. Perform the following updates:

$$U \leftarrow U\setminus\{j\},\;\; \forall k\neq j, Y_k \leftarrow Y_k\setminus Y_j,\;\; \alpha_{ik}\leftarrow\alpha_{ik} - \alpha_{ij}|Y_k\cap Y_j|/|Y_j| \tag{18}$$

Continue this procedure until U is empty. Due to the way we choose j, $\boldsymbol{\alpha}$ remains nonnegative and by construction $\alpha_{ij} = p(\{1'\mathbf{y}=i, y_j=1\})$ once we remove j from U.

The objective function in (15) fits naturally into the framework of (1), with $\Omega(\boldsymbol{\alpha}) - \Omega(\boldsymbol{\beta})$ and constraints corresponding to M, and the rest terms to K. The entropy function Ω is convex wrt the KL-divergence, which is in turn distance enforcing wrt the ℓ_1 norm over the probability simplex [23]. In the next section we propose the SVRG algorithm with Bregman divergence (Breg-SVRG) that (a) provably optimizes strongly convex saddle function with a linear convergence rate, and (b) adapts to the underlying geometry by choosing an appropriate Bregman divergence. Then, in §5 we apply Breg-SVRG to (15) and achieve a factor of n speedup over a straightforward instantiation of [17].

4 Breg-SVRG for Saddle-Point

In this section we propose an efficient algorithm for solving the general saddle-point problem in (1) and prove its linear rate of convergence. Our main assumption is:

Assumption 1. *There exist two norms $\|\cdot\|_{\mathsf{x}}$ and $\|\cdot\|_{\mathsf{y}}$ such that each ψ_k is a saddle function and L-smooth; M is $(\psi_{\mathsf{x}}, \psi_{\mathsf{y}})$-saddle; and ψ_{x} and ψ_{y} are distance enforcing (cf. (7)).*

Note that w.l.o.g. we have scaled the norms so that the usual strong convexity parameter of M is 1.

Algorithm 1: Breg-SVRG for Saddle-Point

```
1 Initialize $z_0$ randomly. Set $\tilde{z} = z_0$.
2 for $s = 1, 2, \ldots$ do                          ▷ epoch index
3     $\tilde{\mu} \leftarrow \tilde{\mu}^s := \nabla K(\tilde{z})$, $z_0 \leftarrow z_0^s := z_m$
4     for $t = 1, \ldots, m$ do                      ▷ iter index
5         Randomly pick $\xi \in \{1, \ldots, n\}$.
6         Compute $v_t$ using (20).
7         Update $z_t$ using (21).
8     $\tilde{z} \leftarrow \tilde{z}^s := \sum_{t=1}^{m} (1+\eta)^t z_t \Big/ \sum_{t=1}^{m} (1+\eta)^t$.
```

Recall we defined $\|z\|_{\mathsf{z}}$ and Δ_{z} in (8). For saddle-point optimization, it is common to define a signed gradient $\mathsf{G}(z) := [\partial_x K(z); -\partial_y K(z)]$ (since K is concave in y). Recall $J = K + M$, and (x^*, y^*) is a saddle-point of J. Using Assumption 1, we measure the gap of an iterate $z_t = (x_t, y_t)$ as follows:

$$\epsilon_t = \epsilon(z_t) = J(x_t, y^*) - J(x^*, y_t) \geq \Delta(z_t, z^*) \geq \tfrac{1}{2} \|z_t - z^*\|^2 \geq 0. \quad (19)$$

Inspired by [2, 9, 17], we propose in Algorithm 1 a new stochastic variance-reduced algorithm for solving the saddle-point problem (1) using Bregman divergences. The algorithm proceeds in epochs. In each epoch, we first compute the following stochastic estimate of the signed gradient $G(z_t)$ by drawing a random component from K:

$$v_t = \begin{pmatrix} v_x(z_t) \\ -v_y(z_t) \end{pmatrix} \quad \text{where} \quad \begin{cases} v_x(z_t) := \partial_x \psi_\xi(z_t) - \partial_x \psi_\xi(\tilde{z}) + \partial_x K(\tilde{z}) \\ v_y(z_t) := \partial_y \psi_\xi(z_t) - \partial_y \psi_\xi(\tilde{z}) + \partial_y K(\tilde{z}) \end{cases}. \quad (20)$$

Here $\tilde{z}$ is the pivot chosen after completing the previous epoch. We make two important observations: (1) By construction the stochastic gradient v_t is unbiased: $\mathbb{E}_\xi[v_t] = \mathsf{G}(z_t)$; (2) The expensive gradient evaluation $\partial K(\tilde{z})$ need only be computed once in each epoch since $\tilde{z}$ is held unchanged. If $\tilde{z} \to z^*$, then the variance of v_t would be largely reduced hence faster convergence may be possible.

Next, Algorithm 1 performs the following *joint* proximal update:

$$(x_{t+1}, y_{t+1}) = \arg\min_x \max_y \eta \langle v_x(z_t), x \rangle + \eta \langle v_y(z_t), y \rangle + \eta M(x, y) + \Delta(x, x_t) - \Delta(y, y_t), \quad (21)$$

where we have the flexibility in choosing a suitable Bregman divergence to better adapt to the underlying geometry. When $\Delta(x, x_t) = \frac{1}{2}\|x - x_t\|_2^2$, we recover the special case in [17]. However, to handle the asymmetry in a general Bregman divergence (which does not appear for the Euclidean distance), we have to choose the pivot $\tilde{z}$ in a significantly different way than [2, 9, 17].

We are now ready to present our main convergence guarantee for Breg-SVRG in Algorithm 1.

Theorem 1. *Let Assumption 1 hold, and choose a sufficiently small $\eta > 0$ such that $m := \left\lceil \log\left(\frac{1-\eta L}{18\eta L^2} - \eta - 1\right) / \log(1+\eta) \right\rceil \geq 1$. Then Breg-SVRG enjoys linear convergence in expectation:*

$$\mathbb{E}\epsilon(\tilde{z}^s) \leq (1+\eta)^{-ms}[\Delta(z^*, z_0) + c(Z+1)\epsilon(z_0)], \text{ where } Z = \textstyle\sum_{t=0}^{m-1}(1+\eta)^t,\ c = \frac{18\eta^2 L^2}{1-\eta L}. \quad (22)$$

For example, we may set $\eta = \frac{1}{45L^2}$, which leads to $c = O(1/L^2)$, $m = \Theta\left(L^2\right)$, $(1+\eta)^m \geq \frac{64}{45}$, and $Z = O(L^2)$. Therefore, between epochs, the gap $\epsilon(\tilde{z}^s)$ decays (in expectation) by a factor of $\frac{45}{64}$, and each epoch needs to conduct the proximal update (21) for $m = \Theta(L^2)$ number of times. (We remind that w.l.o.g. we have scaled the norms so that the usual strong convexity parameter is 1.) In total, to reduce the gap below some threshold ϵ, Breg-SVRG needs to call the proximal update (21) $O(L^2 \log \frac{1}{\epsilon})$ number of times, plus a similar number of *component* gradient evaluations.

Discussions. As mentioned, Algorithm 1 and Theorem 1 extend those in [17] which in turn extend [2, 9] to saddle-point problems. However, [2, 9, 17] all heavily exploit the Euclidean structure (in particular symmetry) hence their proofs cannot be applied to an asymmetric Bregman divergence. Our innovations here include: (a) A new Pythagorean theorem for the newly introduced saddle Bregman divergence (Lemma 1). (b) A moderate extension of the variance reduction lemma in [9] to accommodate any norm (Appendix B). (c) A different pivot $\tilde{z}$ is adopted in each epoch to handle asymmetry. (d) A new analysis technique through introducing a crucial auxiliary variable that enables us to bound the function gap directly. See our proof in Appendix C for more details. Compared with classical mirror descent algorithms [16, 23] that can also solve saddle-point problems with Bregman divergences, our analysis is fundamentally different and we achieve the significantly stronger rate $O(\log(1/\epsilon)$ than the sublinear $O(1/\epsilon)$ rate of [16], at the expense of a squared instead of linear dependence on L. Similar tradeoff also appeared in [17]. We will return to this issue in Section 5.

Variants and acceleration. Our analysis also supports to use different ξ in v_x and v_y. The standard acceleration methods such as universal catalyst [10] and non-uniform sampling can be applied directly (see Appendix E where L, the largest smoothness constant over all pieces, is replaced by their mean).

5 Application of Breg-SVRG to Adversarial Prediction

The quadratic dependence on L, the smoothness parameter, in Theorem 1 reinforces the need to choose suitable Bregman divergences. In this section we illustrate how this can be achieved for the adversarial prediction problem in Section 3. As pointed out in [17], the factorization of K is important, and we consider three schemes: (a) $\psi_k = f_{ij}$; (b) $\psi_k = \frac{1}{n}\sum_{j=1}^{n} f_{k,j}$; and (c) $\psi_k = \frac{1}{n}\sum_{i=1}^{n} f_{i,k}$. W.l.o.g. let us fix the μ in (16) to 1.

Comparison of smoothness constant. Both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are n^2-dimensional, and the bilinear function f_{ij} can be written as $\boldsymbol{\alpha}' A_{ij} \boldsymbol{\beta}$, where $A_{ij} \in \mathbb{R}^{n^2 \times n^2}$ is an n-by-n block matrix, with the (i, j)-th block being $n^2(\frac{2ij}{i+j} I + \mathbf{1}\mathbf{1}')$ and all other blocks being $\mathbf{0}$. The linear terms in (15) can be absorbed into the regularizer Ω without affecting the smoothness parameter.

For scheme (a), the smoothness constant L_2 under ℓ_2 norm depends on the spectral norm of A_{ij}: $L_2 = \max_{i,j} n^2(n + \frac{2ij}{i+j})) = \Theta(n^3)$. In contrast the smoothness constant L_1 under ℓ_1 norm depends on the absolute value of the entries in A_{ij}: $L_1 = \max_{i,j} n^2(1+\frac{2ij}{i+j}) = \Theta(n^3)$; no saving is achieved.

For scheme (b), the bilinear function ψ_k corresponds to $\frac{1}{n}\boldsymbol{\alpha}' \sum_{j=1}^{n} A_{kj}\boldsymbol{\beta}$. Then $L_1 = O(n^2)$ while

$$L_2^2 = \frac{1}{n^2} \max_k \max_{\mathbf{v}:\|\mathbf{v}\|_2=1} \sum\nolimits_{j=1}^{n} \|A_{kj}\mathbf{v}\|_2^2 \geq n^2 \max_{\|\mathbf{v}\|_2=1} \sum\nolimits_{j=1}^{n} \|\mathbf{1}\mathbf{1}'\mathbf{v}\|^2 = n^5. \tag{23}$$

Therefore, L_1^2 saves a factor of n compared with L_2^2.

Comparison of smoothness constant for the overall problem. By strong duality, we may push the maximization over $\boldsymbol{\theta}$ to the innermost level of (10), arriving at an overall problem in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ only:

$$\min_{\{\boldsymbol{\alpha}_i\}\in S} \max_{\{\boldsymbol{\beta}_j\}\in S} \frac{1}{n^2} \sum_{i=1}^{n}\sum_{j=1}^{n} \left[f_{ij}(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_j) - \frac{i}{\lambda n}\mathbf{c}' X \boldsymbol{\alpha}_i + \frac{ij}{2\lambda}\boldsymbol{\alpha}_i' X' X \boldsymbol{\alpha}_j + \frac{1}{2\lambda n^2}\|\mathbf{c}\|_2^2 \right]. \tag{24}$$

where $\mathbf{c} = X\tilde{\mathbf{y}}$. The quadratic term w.r.t. $\boldsymbol{\alpha}$ can be written as $\boldsymbol{\alpha}' B_{ij}\boldsymbol{\alpha}$, where $B_{ij} \in \mathbb{R}^{n^2\times n^2}$ is an n-by-n block matrix, with its (i, j)-th block being $\frac{ij}{2\lambda} X'X$ and all other blocks being $\mathbf{0}$. And we assume each $\|\mathbf{x}_i\|_2 \leq 1$. The smoothness constant can be bounded separately from A_{ij} and B_{ij}; see (128) in Appendix F.

For scheme (a), the smoothness constant square L_2^2 under ℓ_2 norm is upper bounded by the sum of spectral norm square of A_{ij} and B_{ij}. So $L_2^2 \geq \max_{i,j} \left(\frac{ij}{2\lambda} n\right)^2 = \Omega(n^6)$, i.e. $L_2 = \Theta(n^3)$. In contrast the smoothness constant square L_1^2 under ℓ_1 norm is at most the sum of square of maximum absolute value of the entries in A_{ij} and B_{ij}. Hence $L_1^2 \leq \max_{i,j} \left(n^2(1+\frac{2ij}{i+j})\right)^2 + \max_{i,j}\left(\frac{ij}{2\lambda}\right)^2 = \Theta(n^6)$, i.e. $L_1 = \Theta(n^3)$. So no saving is achieved here.

For scheme (b), ψ_k corresponds to $\frac{1}{n}(\boldsymbol{\alpha}' \sum_{j=1}^{n} A_{kj}\boldsymbol{\beta} + \boldsymbol{\alpha}' \sum_{j=1}^{n} B_{kj}\boldsymbol{\alpha})$. Then

$$L_1^2 \leq \frac{1}{n^2} \max_k \left[\max_{\mathbf{v}:\|\mathbf{v}\|_1=1} \left\| \sum\nolimits_{j=1}^{n} A_{kj}\mathbf{v} \right\|_\infty^2 + \max_{\mathbf{v}:\|\mathbf{v}\|_1=1} \left\| \sum\nolimits_{j=1}^{n} B_{kj}\mathbf{v} \right\|_\infty^2 \right] \quad \text{(by (128))} \tag{25}$$

$$\leq \frac{1}{n^2} \max_k \max_j \left[\left(n^2(1+\tfrac{2kj}{k+j})\right)^2 + \left(\tfrac{kj}{2}\right)^2 \right] = n^4, \tag{26}$$

and by setting $\boldsymbol{\beta}$ to $\mathbf{0}$ in (126), we get $L_2^2 \geq n^5$ similar to (23). Therefore, L_1^2 saves a factor of n compared with L_2^2. Similar results apply to scheme (c) too. We also tried non-uniform sampling, but it does not change the order in n. It can also be shown that if our scheme randomly samples n entries from $\{A_{ij}, B_{ij}\}$, the above L_1 and L_2 cannot be improved by further engineering the factorization.

Computational complexity. We finally seek efficient algorithms for the proximal update (21) used by Breg-SVRG. When $M(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \Omega(\boldsymbol{\alpha}) - \Omega(\boldsymbol{\beta})$ as in (16), we can solve $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ separately as:

$$\min_{\boldsymbol{\alpha}} \sum_{ik} \alpha_{ik} \log(\alpha_{ik}/b_{ik}) - c_{ik}, \quad \text{s.t.} \quad \mathbf{1}'\boldsymbol{\alpha} \le 1,\ \forall i\, \forall k,\ 0 \le i\alpha_{ik} \le \mathbf{1}'\boldsymbol{\alpha}_i. \tag{27}$$

where b_{ik} and c_{ik} are constants. In Appendix D we designe an efficient "closed form" algorithm which finds an ϵ accurate solution in $O(n^2 \log^2 \frac{1}{\epsilon})$ time, which is also on par with that for computing the stochastic gradient in schemes (b) and (c). Although scheme (a) reduces the cost of gradient computation to $O(n)$, its corresponding smoothness parameter L_1^2 is increased by n^2 times, hence not worthwhile. We did manage to design an $\tilde{O}(n)$ algorithm for the proximal update in scheme (a), but empirically the overall convergence is rather slow.

If we use the Euclidean squared distance as the Bregman divergence, then a term $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_t\|_2^2$ needs to be added to the objective (27). No efficient "closed form" solution is available, and so in experiments we simply absorbed M into K, and then the proximal update becomes the Euclidean projection onto S, which does admit a competitive $O(n^2 \log^2(1/\epsilon))$ time solution.

6 Experimental Results

Our major goal here is to show that empirically Entropy-SVRG (Breg-SVRG with KL divergence) is significantly more efficient than Euclidean-SVRG (Breg-SVRG with squared Euclidean distance) on some learning problems, especially those with an entropic regularizer and a simplex constraint.

6.1 Entropy regularized LPBoost

We applied Breg-SVRG to an extension of LP Boosting using entropy regularization [29]. In a binary classification setting, the base hypotheses over the training set can be compactly represented as $U = (y_1\mathbf{x}_1, \ldots, y_n\mathbf{x}_n)'$. Then the model considers a minimax game between a distribution $\mathbf{d} \in \Delta^n$ over training examples and a distribution $\mathbf{w} \in \Delta^m$ over the hypotheses:

$$\min_{\mathbf{d}\in\Delta^n, d_i \le \nu} \ \max_{\mathbf{w}\in\Delta^m} \mathbf{d}'U\mathbf{w} + \lambda\Omega(\mathbf{d}) - \gamma\Omega(\mathbf{w}). \tag{28}$$

Here $\mathbf{w}$ tries to combine the hypotheses to maximize the edge (prediction confidence) $y_i\mathbf{x}_i'\mathbf{w}$, while the adversary $\mathbf{d}$ tries to place more weights (bounded by ν) on "hard" examples to reduce the edge.

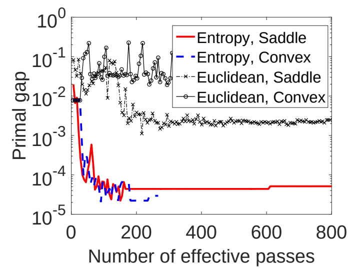
Settings. We experimented on the adult dataset from the UCI repository, which we partitioned into $n = 32,561$ training examples and 16,281 test examples, with $m = 123$ features. We set $\lambda = \gamma = 0.01$ and $\nu = 0.1$ due to its best prediction accuracy. We tried a range of values of the step size η, and the best we found was 10^{-3} for Entropy-SVRG and 10^{-6} for Euclidean-SVRG (larger step size for Euclidean-SVRG fluctuated even worse). For both methods, $m = 32561/50$ gave good results.

The stochastic gradient in $\mathbf{d}$ was computed by $U_{:j}w_j$, where $U_{:j}$ is the j-th column and j is randomly sampled. The stochastic gradient in $\mathbf{w}$ is $d_iU_{i:}'$. We tried with $U_{ij}w_j$ and $U_{ij}d_i$ (scheme (a) in §5), but they performed worse. We also tried with the universal catalyst in the same form as [17], which can be directly extended to Entropy-SVRG. Similarly we used the non-uniform sampling based on the ℓ_2 norm of the rows and columns of U. It turned out that the Euclidean-SVRG can benefit slightly from it, while Entropy-SVRG does not. So we only show the "accelerated" results for the former.

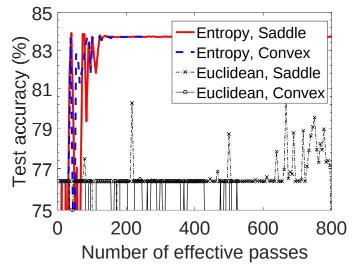
To make the computational cost comparable across machines, we introduced a counter called effective number of passes: #pass. Assume the proximal operator has been called #po number of times, then

$$\text{\#pass} \ := \ \text{number of epochs so far} \ + \tfrac{n+m}{nm} \cdot \text{\#po}. \tag{29}$$

We also compared with a "convex" approach. Given $\mathbf{d}$, the optimal $\mathbf{w}$ in (28) obviously admits a closed-form solution. General saddle-point problems certainly do not enjoy such a convenience. However, we hope to take advantage of this opportunity to study the following question: suppose we solve (28) as a convex optimization in $\mathbf{d}$ and the stochastic gradient were computed from the optimal

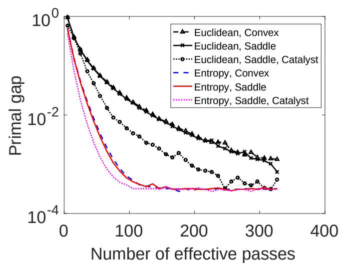


(a) Primal gap v.s. #pass

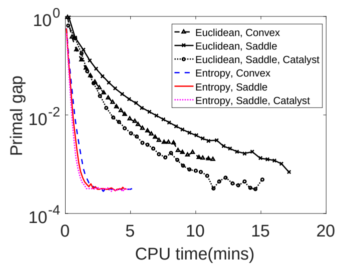


(b) Test accuracy v.s. #pass

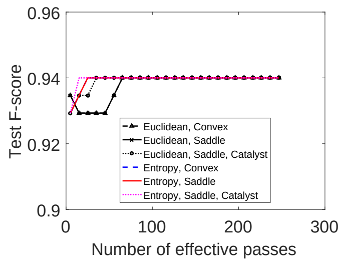
Figure 1: Entropy Regularized LPBoost on adult



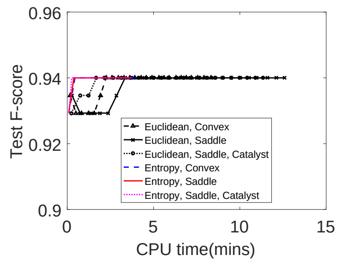
(a) Primal gap v.s. #pass



(b) Primal gap v.s. CPU time



(c) Test F-score v.s. #pass



(d) Test F-score v.s. CPU time

Figure 2: Adversarial Prediction on the synthetic dataset.

$\mathbf{w}$, would it be faster than the saddle SVRG? Since solving $\mathbf{w}$ requires visiting the entire U, strictly speaking the term $\frac{n+m}{nm}$ ·#po in the definition of #pass in (29) should be replaced by #po. However, we stuck with (29) because our interest is whether a more accurate stochastic gradient in $\mathbf{d}$ (based on the optimal $\mathbf{w}$) can outperform doubly stochastic (saddle) optimization. We emphasize that this comparison is only for conceptual understanding, because generally optimizing the inner variable requires costly iterative methods.

Results. Figure 1(a) demonstrated how fast the *primal gap* (with $\mathbf{w}$ optimized out for each $\mathbf{d}$) is reduced as a function of the number of effective passes. Methods based on entropic prox are clearly much more efficient than Euclidean prox. This corroborates our theory that for problems like (28), Entropy-SVRG is more suitable for the underlying geometry (entropic regularizer with simplex constraints).

We also observed that using entropic prox, our doubly stochastic method is as efficient as the "convex" method, meaning that although at each iteration the $\mathbf{w}$ in saddle SVRG is not the optimal for the current $\mathbf{d}$, it still allows the overall algorithm to perform as fast as if it were. This suggests that for general saddle-point problems where no closed-form inner solution is available, our method will still be efficient and competitive. Note this "convex" method is similar to the optimizer used by [29].

Finally, we investigated the increase of test accuracy as more passes over the data are performed. Figure 1(b) shows, once more, that the entropic prox does allow the accuracy to be improved much faster than Euclidean prox. Again, the convex and saddle methods perform similarly.

As a final note, the Euclidean/entropic proximal operator for both $\mathbf{d}$ and $\mathbf{w}$ can be solved in either closed form, or by a 1-D line search based on partial Lagrangian. So their computational cost differ in the same order of magnitude as multiplication v.s. exponentiation, which is much smaller than the difference of #pass shown in Figure 1.

6.2 Adversarial prediction with F-score

Datasets. Here we considered two datasets. The first is a synthetic dataset where the positive examples are drawn from a 200 dimensional normal distribution with mean $0.1 \cdot \mathbf{1}$ and covariance $0.5 \cdot I$, and negative examples are drawn from $\mathcal{N}(-0.1 \cdot \mathbf{1},\ 0.5 \cdot I)$. The training set has $n = 100$ samples, half are positive and half are negative. The test set has 200 samples with the same class ratio. Notice that $n = 100$ means we are optimizing over two 100-by-100 matrices constrained to a challenging set S. So the optimization problem is indeed not trivial.

The second dataset, ionosphere, has 211 training examples (122 pos and 89 neg). 89 examples were used for testing (52 pos and 37 neg). Each example has 34 features.

Methods. To apply saddle SVRG, we used strong duality to push the optimization over $\boldsymbol{\theta}$ to the inner-most level of (10), and then eliminated $\boldsymbol{\theta}$ because it is a simple quadratic. So we ended up with the convex-concave optimization as shown in (24), where the K part of (15) is augmented with a quadratic term in $\boldsymbol{\alpha}$. The formulae for computing the stochastic gradient using scheme (b) are detailed in Appendix G. We fixed $\mu = 1$, $\lambda = 0.01$ for the ionosphere dataset, and $\mu = 1$, $\lambda = 0.1$ for the synthetic dataset.

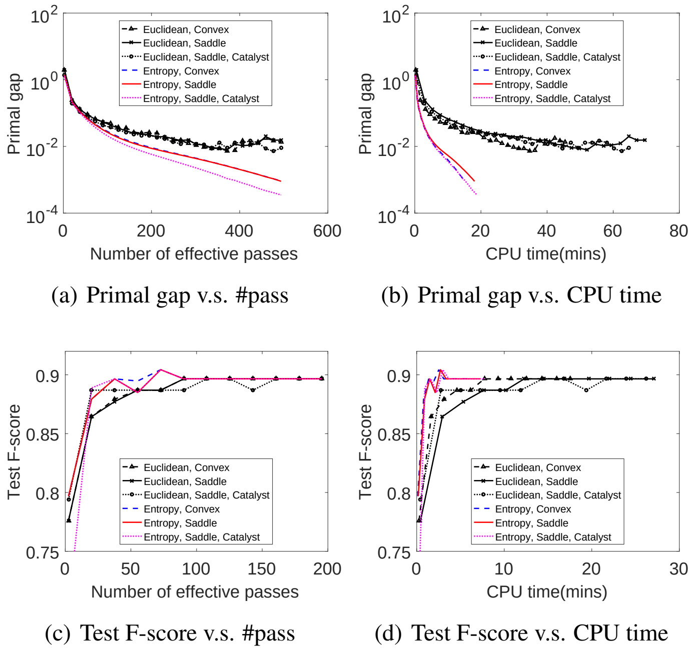


Figure 3: Adversarial Prediction on the ionosphere dataset.

We also tried the universal catalyst along with non-uniform sampling where each i was sampled with a probability proportional to $\sum_{k=1}^{n} \|A_{ik}\|_F^2$, and similarly for j. Here $\|\cdot\|_F$ is the Frobenious norm.

Parameter Tuning. Since each entry in the $n \times n$ matrix $\boldsymbol{\alpha}$ is relatively small when n is large, we needed a relatively small step size. When $n = 100$, we used 10^{-2} for Entropy-SVRG and 10^{-6} for Euclidean-SVRG (a larger step size makes it over-fluctuate). When applying catalyst, the catalyst regularizor can suppress the noise from larger step size. After a careful trade off between catalyst regularizor parameter and larger step size, we managed to achieve faster convergence empirically.

Results. The results on the two datasets are shown in Figures 2 and 3 respectively. We truncated the #pass and CPU time in subplots (c) and (d) because the F-score has stabilized and we would rather zoom in to see the initial growing phase. In terms of primal gap versus #pass (subplot a), the entropy based method is significantly more effective than Euclidean methods on both datasets (Figure 2(a) and 3(a)). Even with catalyst, Euclidean-Saddle is still much slower than the entropy based methods on the synthetic dataset in Figure 2(a). The CPU time comparisons (subplot b) follow the similar trend, except that the "convex methods" should be ignored because they are introduced only to compare #pass.

The F-score is noisy because, as is well known, it is not monotonic with the primal gap and glitches can appear. In subplots 2(d) and 3(d), the entropy based methods achieve higher F-score significantly faster than the plain Euclidean based methods on both datasets. In terms of passes (subplots 2(c) and 3(c)), Euclidean-Saddle and Entropy-Saddle achieved a similar F-score at first because their primal gaps are comparable at the beginning. After 20 passes, the F-score of Euclidean-Saddle is overtaken by Entropy-Saddle as the primal gap of Entropy-Saddle become much smaller than Euclidean-Saddle.

7 Conclusions and Future Work

We have proposed Breg-SVRG to solve saddle-point optimization and proved its linear rate of convergence. Application to adversarial prediction confirmed its effectiveness. For future work, we are interested in relaxing the (potentially hard) proximal update in (21). We will also derive similar reformulations for DCG and precision@k, with a quadratic number of variables and with a finite sum structure that is again amenable to Breg-SVRG, leading to a similar reduction of the condition number compared to Euclidean-SVRG. These reformulations, however, come with different constraint sets, and new proximal algorithms with similar complexity as for the F-score can be developed.

References

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

[2] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *In NIPS*. 2013.

[3] A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *In NIPS*. 2014.

[4] M. Schmidt, N. L. Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 2016.

[5] A. J. Defazio, T. S. Caetano, and J. Domke. Finito: A faster, permutable incremental gradient method for big data problems. *In ICML*. 2014.

[6] J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

[7] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *J. Mach. Learn. Res.*, 14:567–599, 2013.

[8] S. Shalev-Shwartz. SDCA without duality, regularization, and individual convexity. *In ICML*. 2016.

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

[10] H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. *In NIPS*. 2015.

[11] A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. *In NIPS*. 2014.

[12] S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *In ICML*. 2014.

[13] R. Babanezhad, M. O. Ahmed, A. Virani, M. Schmidt, J. Konečný, and S. Sallinen. Stop wasting my gradients: Practical svrg. *In NIPS*. 2015.

[14] Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *In ICML*. 2015.

[15] Z. Zhu and A. J. Storkey. Adaptive stochastic primal-dual coordinate descent for separable saddle point problems. *In Machine Learning and Knowledge Discovery in Databases*, pp. 645–658. 2015.

[16] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[17] P. Balamurugan and F. Bach. Stochastic variance reduction methods for saddle-point problems. *In NIPS*. 2016.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *In NIPS*. 2014.

[19] H. Wang, W. Xing, K. Asif, and B. D. Ziebart. Adversarial prediction games for multivariate losses. *In NIPS*. 2015.

[20] F. Farnia and D. Tse. A minimax approach to supervised learning. *In NIPS*. 2016.

[21] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[22] Y. Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM J. on Optimization*, 16(1):235–249, 2005. ISSN 1052-6234.

[23] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[24] H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. *In NIPS*. 2014.

[25] R. T. Rockafellar. Monotone operators associated with saddle functions and minimax problems. *Nonlinear Functional Analysis*, 18(part 1):397–407, 1970.

[26] J. C. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. *In Proc. Annual Conf. Computational Learning Theory*. 2010.

[27] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*, 2009.

[28] K. Asif, W. Xing, S. Behpour, and B. D. Ziebart. Adversarial cost-sensitive classification. *In UAI*. 2015.

[29] M. K. Warmuth, K. A. Glocer, and S. V. N. Vishwanathan. Entropy regularized LPBoost. *In* Y. Freund, Y. L. Györfi, and G. Turàn, eds., *Proc. Intl. Conf. Algorithmic Learning Theory*, no. 5254 in Lecture Notes in Artificial Intelligence, pp. 256 – 271. Springer-Verlag, Budapest, October 2008.

[30] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, 49:185–212, 2011.

A Proofs for Section 2

The following result that sanwiches the Bregman divergence induced by an L-smooth convex function is well-known:

Lemma 2. *If f is convex, and L-smooth w.r.t. $\|\cdot\|$, then*

$$\tfrac{1}{2L}\left\|\nabla f(x') - \nabla f(x)\right\|_*^2 \le \Delta_f(x', x) \le \tfrac{L}{2}\left\|x' - x\right\|^2. \tag{30}$$

Proof. The second inequality is obvious. To show the first inequality, define

$$g(x') := \Delta_f(x', x) = f(x') - f(x) - \langle \nabla f(x), x' - x\rangle . \tag{31}$$

Then g is minimized at x with $g(x) = 0$, and g is also L-smooth. Therefore for any $\bar{x}$

$$g(\bar{x}) \le g(x') + \langle \nabla g(x'), \bar{x} - x'\rangle + \frac{L}{2}\left\|\bar{x} - x'\right\|^2 \le g(x') + \left\|\nabla g(x')\right\|_* \left\|\bar{x} - x'\right\| + \frac{L}{2}\left\|\bar{x} - x'\right\|^2 . \tag{32}$$

Now take minimization over $\bar{x}$ on both sides:

$$0 = g(x) \le g(x') - \frac{1}{2L}\left\|\nabla g(x')\right\|_*^2 . \tag{33}$$

Plugging in the definition of g and noticing $\nabla g(x') = \nabla f(x') - \nabla f(x)$, we get the first inequality. □

The following result is crucial for our later analysis, and extends a result of [27].

Lemma 1. *Let f and g be ϕ-saddle and φ-saddle respectively, with one of them being differentiable. Then, for any $z = (x, y)$ and any saddle point (if exists) $z^* := (x^*, y^*) \in \arg\min_x \max_y \{f(z) + g(z)\}$, we have $f(x, y^*) + g(x, y^*) \ge f(x^*, y) + g(x^*, y) + \Delta_{\phi+\varphi}(z, z^*)$.*

Proof. We first recall the following slight generalization of a result of [27]:

Claim. Let h and k be respectively ψ_1- and ψ_2-convex, with one of them being differentiable. Let $x^* \in \arg\min_x h(x) + k(x)$, then for all x, $h(x) + k(x) \ge h(x^*) + k(x^*) + \Delta_{\psi_1+\psi_2}(x, x^*)$.

Indeed, using the optimality of x^*, we have $\mathbf{0} \in \partial(h + k)(x^*) = \partial h(x^*) + \partial k(x^*)$, where the last equality is due to the differentiable assumption (on one of h and k). Since h is ψ_1-convex and k is ψ_2-convex, we have

$$h(x) \ge h(x^*) + \langle x - x^*, \partial h(x^*)\rangle + \Delta_{\psi_1}(x, x^*) \tag{34}$$

$$k(x) \ge k(x^*) + \langle x - x^*, \partial k(x^*)\rangle + \Delta_{\psi_2}(x, x^*). \tag{35}$$

Adding the above two inequalities and noting that $\Delta_{\psi_1} + \Delta_{\psi_2} = \Delta_{\psi_1+\psi_2}$ completes the proof of our claim.

Now to prove Lemma 1, we note that if (x^*, y^*) is a saddle point of $f + g$, then $x^* \in \arg\min_x f(x, y^*) + g(x, y^*)$ and also $y^* \in \arg\min_y -f(x^*, y) - g(x^*, y)$. Note also that if f is ϕ-saddle, then $f_y(x) = f(x, y)$ is ϕ_y-convex. Similarly, if g is φ-saddle, then $-g_x(y) = -g(x, y)$ is $(-\varphi_x)$-convex. Applying the above claim twice we have:

$$f_{y^*}(x) + g_{y^*}(x) \ge f_{y^*}(x^*) + g_{y^*}(x^*) + \Delta_{\phi_{y^*}+\varphi_{y^*}}(x, x^*) \tag{36}$$

$$-f_{x^*}(y) - g_{x^*}(y) \ge -f_{x^*}(y^*) - g_{x^*}(y^*) + \Delta_{-\phi_{x^*}-\varphi_{x^*}}(y, y^*). \tag{37}$$

Adding the above two equations and noting that $\Delta_{\phi_{y^*}+\varphi_{y^*}}(x, x^*) + \Delta_{-\phi_{x^*}-\varphi_{x^*}}(y, y^*) = \Delta_{\phi+\varphi}(z, z^*)$ completes our proof. □

Algorithm 2: SVRG with Bregman Divergence

```
1 Initialize $x_0$ randomly. Set $\tilde{x} = x_0$.
2 for $s = 1, 2, \dots$ do                                        // epoch index
3     $\tilde{\mu} \leftarrow \tilde{\mu}^s := \nabla P(\tilde{x})$, $x_0 \leftarrow x_0^s := x_m^{s-1}$
4     for $t = 1, \dots, m$ do                                    // iter index
5         Randomly pick $\xi \in \{1, \dots, n\}$.
6         Compute $v_t$ using (38).
7         Update $x_{t+1}$ using (38).
8     Denote $x_m^s = x_m$.
9     $\tilde{x} \leftarrow \tilde{x}^s := \frac{\sum_{t=1}^m (1+\eta\lambda)^t x_t}{\sum_{t=1}^m (1+\eta\lambda)^t}$.
```

B Bregman Divergence for Convex SVRG

Prior to saddle-point optimization, it is illustrative to see how variance reduction methods can be extended to Bregman divergence in convex optimization. Let us consider a proximal objective

$$J(x) = P(x) + \Omega(x) = \tfrac{1}{n}\textstyle\sum_{k=1}^n \psi_k(x) + \Omega(x).$$

Here each ψ_k is convex and L-smooth (w.r.t. some norm), and Ω is Δ-convex for some Bregman divergence Δ. Breg-SVRG extends the vanilla SVRG by employing the following proximal operator [30] which we assume is efficiently computable:

$$x_{t+1} = \arg\min_x \left\{\eta \langle v_t, x\rangle + \eta\Omega(x) + \Delta(x, x_t)\right\}, \text{ where } \; v_t = \nabla\psi_\xi(x_t) - \nabla\psi_\xi(\tilde{x}) + \tilde{\mu}. \quad (38)$$

Here ξ is sampled uniformly at random from $\{1, \dots, n\}$, $\tilde{x}$ is the pivot found after completing the last epoch, and $\tilde{\mu} = \nabla P(\tilde{x})$. The whole procedure, which we call Breg-SVRG, is detailed in Algorithm 2. To ease notation, the x_t here always refers to the t-th step of the current epoch s, and we will include the epoch index s only when necessary.

Let us define the gap $\epsilon(x) := J(x) - J(x_*)$ for some x_* that minimizes J. Our first convergence result for Algorithm 2 is as follows:

Theorem 2. *Assume each ψ_k is convex and L-smooth wrt $\|\cdot\|$, and P and Ω are $(\gamma\Delta)$- and $(\lambda\Delta)$-convex wrt some Bregman divergence Δ, respectively. Let η be sufficiently small such that $m := \left\lceil \log(\frac{1}{8\eta L} - \frac{1}{8} - \rho) \big/ \log\rho \right\rceil \geq 1$. Then Breg-SVRG enjoys linear convergence in expectation:*

$$\mathbb{E}\epsilon(\tilde{x}^s) \leq \rho^{-ms}[\Delta(x_*, x_0) + c(Z+1)\epsilon(x_0)],$$

where $\rho = \frac{1+\eta\lambda}{1-\eta\gamma}$, $c = \frac{8\eta^2 L}{(1-\eta L)(1-\eta\gamma)}$ and $Z = \sum_{t=0}^{m-1} \rho^t$.

For example, we may set $\eta = \frac{1}{18L}$, which leads to $c = \frac{4}{153L}$, $m = \frac{\log\left(\frac{9}{8} - \frac{\lambda}{18L}\right)}{\log\left(1+\frac{\lambda}{18L}\right)} = \Theta\left(\frac{L}{\lambda}\right)$, $(1 + \eta\lambda)^m = \frac{9}{8} - \frac{\lambda}{18L} \geq \frac{9}{8} - \frac{1}{18} = \frac{77}{72}$, and $Z = \frac{9L}{4\lambda} - 1$. Therefore, between epochs, the gap decays by a factor of $\frac{72}{77}$, and each epoch needs to call (38) for $\Theta(L/\lambda)$ times. In total, to reduce the gap below some tolerance ϵ, the proximal operator (38) needs to be called for $O\left(\frac{L}{\lambda}\log\frac{1}{\epsilon}\right)$ times. If the norm $\|\cdot\|$ is chosen to be Euclidean, then the above guarantee reduces to that of SVRG [9]. The condition number L/λ, however, can change significantly w.r.t. the chosen norm (which reflects the underlying problem geometry).

We need the following variance reduction lemma that extends a result in [9] to any norm.

Lemma 3. *The variance of v_t can be bounded by: $\mathbb{E}_\xi \|v_t - \nabla P(x_t)\|_*^2 \leq 16L \cdot [\epsilon(x_t) + \epsilon(\tilde{x})]$.*

Proof. Clearly, for any norm, $\|a + b\|^2 \leq 2(\|a\|^2 + \|b\|^2)$. Besides, for any random variable X and norm $\|\cdot\|$, $\mathbb{E}\,\|X - \mathbb{E}[X]\|^2 \leq 2\mathbb{E}[\|X\|^2 + \|\mathbb{E}X\|^2] \leq 4\mathbb{E}\,\|X\|^2$. It bounds the "variance" of a random variable, under any norm, by four times its "second moment."

Using these two inequalities and conditional on x_t, we have

$$\begin{aligned}
&\mathbb{E}_\xi \left\|v_t - \nabla P(x_t)\right\|_*^2 = \mathbb{E}_\xi \|(\nabla\psi_\xi(x_t) - \nabla\psi_\xi(\tilde{x})) - (\nabla P(x_t) - \nabla P(\tilde{x}))\|_*^2 \\
&\leq 4 \cdot \mathbb{E}_\xi \left\|\nabla\psi_\xi(x_t) - \nabla\psi_\xi(\tilde{x})\right\|_*^2 = 4\mathbb{E}_\xi \|\nabla\psi_\xi(x_t) - \nabla\psi_\xi(x_*) - (\nabla\psi_\xi(\tilde{x}) - \nabla\psi_\xi(x_*))\|_*^2 \\
&\leq 8 \cdot \mathbb{E}_\xi \left\|\nabla\psi_\xi(x_t) - \nabla\psi_\xi(x_*)\right\|_*^2 + 8 \cdot \mathbb{E}_\xi \left\|\nabla\psi_\xi(\tilde{x}) - \nabla\psi_\xi(x_*)\right\|_*^2. \qquad (39)
\end{aligned}$$

We can next invoke Lemma 2 to upper bound the first part of (39):

$$\tfrac{1}{2L}\mathbb{E}_\xi \left\|\nabla\psi_\xi(x_t) - \nabla\psi_\xi(x_*)\right\|_*^2 \leq \mathbb{E}\Delta_{\psi_\xi}(x_t, x_*) = \Delta_P(x_t, x_*) \leq \epsilon(x_t),$$

where the last inequality is due to (a special case of) Lemma 1. The second part of (39) can be bounded similarly. □

Proof of Theorem 2. We apply Lemma 1 to the update (38), with $g = \eta\Omega(x) + \Delta(x, x_t)$, $\phi = 0$, and $\varphi = \lambda\Delta$:

$$\begin{aligned}
&\eta \langle v_t, x_{t+1}\rangle + \eta\Omega(x_{t+1}) + \Delta(x_{t+1}, x_t) && (40)\\
&\leq \eta \langle v_t, x^*\rangle + \eta\Omega(x^*) + \Delta(x^*, x_t) - \eta\Delta_\Omega(x^*, x_{t+1}) - \Delta(x^*, x_{t+1}) && (41)\\
&\leq \eta \langle v_t, x^*\rangle + \eta\Omega(x^*) + \Delta(x^*, x_t) - \eta\lambda\Delta(x^*, x_{t+1}) - \Delta(x^*, x_{t+1}). && (42)
\end{aligned}$$

Therefore

$$\begin{aligned}
&\eta\Omega(x_{t+1}) + (1 + \eta\lambda)\Delta(x^*, x_{t+1}) && (43)\\
&\leq \Delta(x^*, x_t) + \eta \langle v_t, x^* - x_{t+1}\rangle + \eta\Omega(x^*) - \Delta(x_{t+1}, x_t) && (44)\\
&\leq \Delta(x^*, x_t) + \eta \langle v_t, x^* - x_{t+1}\rangle + \eta\Omega(x^*) - \frac{1}{2}\|x_{t+1} - x_t\|^2, && (45)
\end{aligned}$$

where the last inequality is because Δ is distance enforcing w.r.t. the norm $\|\cdot\|$. Since J is L-smooth, we obtain

$$0 \leq P(x_t) - P(x_{t+1}) + \langle\nabla P(x_t), x_{t+1} - x_t\rangle + \frac{L}{2}\left\|x_{t+1} - x_t\right\|^2. \qquad (46)$$

Multiplying the above by $\eta > 0$ and adding to (45) we get

$$\begin{aligned}
&\eta\Omega(x_{t+1}) + (1 + \eta\lambda)\Delta(x^*, x_{t+1}) && (47)\\
\leq&\Delta(x^*, x_t) + \eta \langle v_t, x^* - x_{t+1}\rangle + \eta\Omega(x^*) - \frac{1 - \eta L}{2}\left\|x_{t+1} - x_t\right\|^2 && (48)\\
&+ \eta P(x_t) - \eta P(x_{t+1}) + \eta \langle\nabla P(x_t), x_{t+1} - x_t\rangle && (49)\\
=&\Delta(x^*, x_t) + \eta \langle v_t - \nabla P(x_t), x_t - x_{t+1}\rangle - \frac{1 - \eta L}{2}\left\|x_{t+1} - x_t\right\|^2 && (50)\\
&+ \eta P(x_t) - \eta P(x_{t+1}) + \eta\Omega(x^*) + \eta \langle v_t, x^* - x_t\rangle && (51)\\
\leq&\Delta(x^*, x_t) + \frac{\eta^2}{2(1 - \eta L)}\left\|v_t - \nabla P(x_t)\right\|_*^2 && (52)\\
&+ \eta[\langle v_t, x^* - x_t\rangle + P(x_t) - P(x_{t+1}) + \Omega(x^*)]. && (53)
\end{aligned}$$

Conditional on x_t we take expectation over ξ on both sides:

$$\begin{aligned}
(1 + \eta\lambda)\mathbb{E}\Delta(x^*, x_{t+1}) \leq (1 - \eta\gamma)\Delta(x^*, x_t) + \frac{\eta^2}{2(1 - \eta L)}\mathbb{E}\left\|v_t - \nabla P(x_t)\right\|_*^2 && (54)\\
+ \eta[J(x^*) - \mathbb{E}J(x_{t+1})], && (55)
\end{aligned}$$

where we have also used the assumption that P is $(\gamma\Delta)$-convex. Using Lemma 3 we take expectation over x_t again on both sides, leading to

$$\rho\Delta_{t+1} \leq \Delta_t + c\left(\delta_t + \tilde{\delta}^{s-1}\right) - \kappa\delta_{t+1}. \qquad (56)$$

where $\rho := \frac{1+\eta\lambda}{1-\eta\gamma}$, $c := \frac{8\eta^2 L}{(1-\eta L)(1-\eta\gamma)}$, $\kappa := \frac{\eta}{1-\eta\gamma}$, $\delta_t := \mathbb{E}\epsilon(x_t)$, $\Delta_t := \mathbb{E}\Delta(x^*, x_t)$, $\tilde{\delta}^{s-1} := \mathbb{E}\epsilon(\tilde{x}^{s-1})$. Multiplying both sides by ρ^t and telescoping from $t = 0$ to $m - 1$, we obtain

$$\rho^m\Delta_m \leq \Delta_0 + c\sum_{t=1}^{m}\rho^{t-1}\delta_{t-1} + c\tilde{\delta}^{s-1}\sum_{t=1}^{m}\rho^{t-1} - \kappa\sum_{t=1}^{m}\rho^{t-1}\delta_t. \qquad (57)$$

Rearranging, we get

$$\rho^m \Delta_m + c\rho^m \delta_m + (\kappa - c\rho) \sum_{t=1}^{m} \rho^{t-1} \delta_t \leq \Delta_0 + c\delta_0 + c\tilde{\delta}^{s-1} \sum_{t=1}^{m} \rho^{t-1}. \tag{58}$$

Now define the representer of epoch s as

$$\tilde{x}^s = \frac{1}{Z} \sum_{t=1}^{m} \rho^{t-1} x_t, \quad \text{where} \quad Z = \sum_{t=1}^{m} \rho^{t-1}. \tag{59}$$

Note that $\rho > 1$ hence the most recent iterate gets a bigger weight. Also, we can equivalently use $\tilde{x}^s = \frac{1}{Z'} \sum_{t=1}^{m} \rho^t x_t$ where $Z' = \sum_{t=1}^{m} \rho^t$, see Algorithm 2. Then, noting that J is convex and $\tilde{\delta}^s = \mathbb{E}[J(\tilde{x}^s) - J(x^*)]$, we obtain

$$\rho^m(\Delta_m + c\delta_m) + (\kappa - c\rho) Z \tilde{\delta}^s \leq \rho^m(\Delta_m + c\delta_m) + (\kappa - c\rho) \sum_{t=1}^{m} \rho^{t-1} \delta_t \tag{60}$$

$$\leq (\Delta_0 + c\delta_0) + cZ\tilde{\delta}^{s-1}. \tag{61}$$

Now pick m such that

$$\rho^m = \frac{(\kappa - c\rho)Z}{cZ} = \frac{\kappa - c\rho}{c} = \frac{1 - \eta L}{8\eta L} - \frac{1 + \eta\lambda}{1 - \eta\gamma} = \frac{1}{8\eta L} - \frac{1}{8} - \frac{1 + \eta\lambda}{1 - \eta\gamma}. \tag{62}$$

Therefore,

$$\mathbb{E}\Delta(x^*, x_m^s) + c(\mathbb{E}J(x_m^s) - J(x^*)) + cZ(J(\tilde{x}^s) - J(x^*)) \tag{63}$$

$$\leq \rho^{-m}[\mathbb{E}\Delta(x^*, x_m^{s-1}) + c(\mathbb{E}J(x_m^{s-1}) - J(x^*)) + cZ(J(\tilde{x}^{s-1}) - J(x^*))]. \tag{64}$$

So there is a decay of factor ρ^{-m} between epochs. Set $\eta = \frac{\alpha}{L}$ and we obtain

$$\rho^m = \frac{1}{8\alpha} - \frac{1}{8} - \frac{L + \alpha\lambda}{L - \alpha\gamma} > 1 \tag{65}$$

for α sufficiently small. Moreover, $\rho = \frac{L+\alpha\lambda}{L-\alpha\gamma}$ hence

$$m = \frac{\log\left(\frac{1}{8\alpha} - \frac{1}{8} - \frac{L+\alpha\lambda}{L-\alpha\gamma}\right)}{\log \frac{L+\alpha\lambda}{L-\alpha\gamma}} = \Theta\left(\frac{1}{\alpha} \log \frac{1}{\alpha} \cdot \frac{L}{\lambda + \gamma}\right). \tag{66}$$

So between epochs, we decrease the gap by a constant factor that is strictly smaller than 1, and the number of iterations per epoch is $\Theta\left(\frac{L}{\lambda+\gamma}\right)$. In total, to find an ϵ accurate solution, the computational cost is $\Theta\left(\frac{L}{\lambda+\gamma} \log \frac{1}{\epsilon}\right)$. □

C Rates for Proximal Saddle-Point Optimization in Section 4

The proof of [17] relies on resolvent operators, which is inherently restricted to the Euclidean norm. Besides, their bound is on $\|z_t - z^*\|^2$, and it was claimed that "the convex minimization analysis does not apply and we use the notion of monotone operators to prove convergence". We show here that by introducing an auxiliary variable, our analysis in Appendix B can be largely reused for SVRG with Bregman divergence in saddle-point problems, and the bound is directly on function values.

Theorem 1. *Let Assumption 1 hold, and choose a sufficiently small $\eta > 0$ such that $m := \left\lceil \log\left(\frac{1-\eta L}{18\eta L^2} - \eta - 1\right) / \log(1+\eta) \right\rceil \geq 1$. Then Breg-SVRG enjoys linear convergence in expectation:*

$$\mathbb{E}\epsilon(\tilde{z}^s) \leq (1+\eta)^{-ms}[\Delta(z^*, z_0) + c(Z+1)\epsilon(z_0)], \text{ where } Z = \sum_{t=0}^{m-1}(1+\eta)^t, \ c = \frac{18\eta^2 L^2}{1-\eta L}. \tag{22}$$

Proof. Our key innovation in analysis is the introduction of an auxiliary variable: $u_t = \begin{pmatrix} \partial_x K(x_t, y^*) \\ -\partial_y K(x^*, y_t) \end{pmatrix}$. Note that $\mathbb{E}_\xi v_t \neq u_t$.

Recall that

$$\epsilon_t^M := \epsilon^M(z_t) := M(x_t, y^*) - M(x^*, y_t), \qquad \epsilon_t^K := \epsilon^K(z_t) := K(x_t, y^*) - K(x^*, y_t) \tag{67}$$

$$\epsilon_t^K := \epsilon^K(z_t) := K(x_t, y^*) - K(x^*, y_t) \in \mathbb{R} \tag{68}$$

$$\epsilon_t = \epsilon(z_t) = J(x_t, y^*) - J(x^*, y_t) = [J(x_t, y^*) - J(x^*, y^*)] + [J(x^*, y^*) - J(x^*, y_t)] \tag{69}$$

$$\geq \Delta(z_t, z^*) \geq \tfrac{1}{2} \|z_t - z^*\|^2 \geq 0. \tag{70}$$

The first step of our proof is to invoke Lemma 1 on the update (21):

$$(1+\eta)\Delta(z^*, z_{t+1}) \leq \Delta(z^*, z_t) - \eta\epsilon_{t+1}^M - \Delta(z_{t+1}, z_t) + \eta \langle v_t, z^* - z_t\rangle + \eta \langle v_t, z_t - z_{t+1}\rangle$$
$$= \Delta(z^*, z_t) - \eta\epsilon_{t+1}^M - \Delta(z_{t+1}, z_t) + \eta \langle v_t, z^* - z_t\rangle + \eta \langle v_t - u_t, z_t - z_{t+1}\rangle + \eta \langle u_t, z_t - z_{t+1}\rangle .$$

It is easy to bound $\langle u_t, z_t - z_{t+1}\rangle$ as K is L-smooth:

$$\langle u_t, z_t - z_{t+1}\rangle = \langle \partial_x K(x_t, y^*), x_t - x_{t+1}\rangle - \langle \partial_y K(x^*, y_t), y_t - y_{t+1}\rangle$$
$$\leq K(x_t, y^*) - K(x_{t+1}, y^*) + \tfrac{L}{2} \|x_t - x_{t+1}\|^2 + K(x^*, y_{t+1}) - K(x^*, y_t) + \tfrac{L}{2} \|y_t - y_{t+1}\|^2$$
$$= \epsilon_t^K - \epsilon_{t+1}^K + \tfrac{L}{2} \|z_t - z_{t+1}\|^2 . \tag{71}$$

So we can proceed by

$$(1+\eta)\Delta(z^*, z_{t+1})$$
$$\leq \Delta(z^*, z_t) - \eta\epsilon_{t+1} + \eta\epsilon_t^K + \eta \langle v_t, z^* - z_t\rangle + \eta \langle v_t - u_t, z_t - z_{t+1}\rangle - \tfrac{1-\eta L}{2} \|z_t - z_{t+1}\|^2$$
$$\leq \Delta(z^*, z_t) - \eta\epsilon_{t+1} + \eta\epsilon_t^K + \eta \langle v_t, z^* - z_t\rangle + \frac{\eta^2 \|v_t - u_t\|_*^2}{2(1-\eta L)}.$$

Take expectation over ξ on both sides (conditional on z_t). Since $\mathbb{E}_\xi[v_t] = \mathsf{G}(z_t)$, we may apply the inequality $K(x, y') - K(x', y) \leq \langle \mathsf{G}(z), z - z'\rangle$:

$$(1+\eta)\mathbb{E}\Delta(z^*, z_{t+1}) \leq \Delta(z^*, z_t) - \eta\mathbb{E}\epsilon_{t+1} + \frac{\eta^2}{2(1-\eta L)} \mathbb{E} \|v_t - u_t\|_*^2 . \tag{72}$$

Finally we bound $\mathbb{E} \|v_t - u_t\|_*^2$:

$$\mathbb{E} \|v_t - u_t\|_*^2 = \mathbb{E} \|v_t - \mathsf{G}(z_t) + \mathsf{G}(z_t) - u_t\|_*^2 \leq 2\mathbb{E} \|v_t - \mathsf{G}(z_t)\|_*^2 + 2 \|\mathsf{G}(z_t) - u_t\|_*^2 . \tag{73}$$

Notice that by the L-smoothness of K,

$$\|\mathsf{G}(z_t) - u_t\|_*^2 = \|\partial_x K(x_t, y_t) - \partial_x K(x_t, y^*)\|_*^2 + \|\partial_y K(x_t, y_t) - \partial_y K(x^*, y_t)\|_*^2$$
$$\leq L^2 \|y_t - y^*\|^2 + L^2 \|x_t - x^*\|^2 . \tag{74}$$

Again using $\mathbb{E} \|X - \mathbb{E}[X]\|^2 \leq 4\mathbb{E} \|X\|^2$ and L-smoothness of ψ_k,

$$\mathbb{E} \|v_t - \mathsf{G}(z_t)\|_*^2 = \mathbb{E}\|\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z}) - \mathbb{E}[\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z})]\|_*^2$$
$$\leq 4\mathbb{E} \|\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z})\|_*^2 \leq 4L^2 \|z_t - \tilde{z}\|^2$$
$$\leq 8L^2 \|z_t - z^*\|^2 + 8L^2 \|\tilde{z} - z^*\|^2. \tag{75}$$

Plug (74) and (75) into (73), and then into (72). Using (67), we finally arrive at (expectation on ξ)

$$(1+\eta)\mathbb{E}\Delta(z^*, z_{t+1}) \leq \Delta(z^*, z_t) - \eta\mathbb{E}\epsilon_{t+1} + \frac{18\eta^2 L^2}{1-\eta L}(\epsilon_t + \epsilon(\tilde{z}^{s-1})). \tag{76}$$

Taking expectation of the whole history on both sides, we obtain

$$\rho\Delta_{t+1} \leq \Delta_t + c' \left(\delta_t + \tilde{\delta}^{s-1}\right) - \eta\delta_{t+1}. \tag{77}$$

where $\rho := 1+\eta$, $c' := \frac{18\eta^2 L^2}{1-\eta L}$, $\delta_t := \mathbb{E}\epsilon(z_t)$, $\Delta_t := \mathbb{E}\Delta(z^*, z_t)$, $\tilde{\delta}^{s-1} := \mathbb{E}\epsilon(\tilde{z}^{s-1})$. This has exactly the same shape as (56), and therefore the rest derivation is almost identical, except that in c', we have $\eta^2 L^2$ rather that $\eta^2 L$ as under (56). So almost all the derivation can be shared. Let us set $\eta = \frac{1}{45L^2}$, and we obtain

$$\rho^m = \frac{\eta - c\rho}{c} = \frac{45 - 1/L}{18} - \frac{1}{45L^2} - 1 \geq \frac{45-1}{18} - \frac{1}{45} - 1 = \frac{64}{45}. \tag{78}$$

Since $\rho = 1 + \frac{1}{45L^2}$, we derive

$$m = \frac{\log\left(\frac{45-1/L}{18} - \frac{1}{45L^2} - 1\right)}{\log\left(1 + \frac{1}{45L^2}\right)} = \Theta\left(L^2\right). \tag{79}$$

So between epochs, the decay is by a factor of $\frac{45}{64}$, and the number of iterations per epoch is $\Theta(L^2)$. The total computational cost is therefore $O\left(L^2 \log \frac{1}{\epsilon}\right)$. $\square$

D Efficient Proximal Operator for Solving (27)

Given a set of variables $\{\boldsymbol{\alpha}_i\}$ which are *not* necessarily in S, we need to project it to S based on Bregman divergence. Here we show how this can be done in $O(n^2)$ time for both Euclidean and entropic projections.

D.1 Euclidean projection to S

Given a set $\{\boldsymbol{\alpha}_k\}$, the projection to S requires solving

$$\min_{\{\mathbf{x}_k\}} \tfrac{1}{2} \textstyle\sum_k \|\mathbf{x}_k - \boldsymbol{\alpha}_k\|_2^2 \text{ s.t. } \mathbf{x}_k \in C_k, \ \sum_k \mathbf{1}'\mathbf{x}_k \leq 1 \tag{80}$$

$$\text{where} \quad C_k := \{\mathbf{x} \in [0, \tfrac{r_k}{k}]^n : r_k \geq 0, \ \mathbf{1}'\mathbf{x} = r_k\} \tag{81}$$

Introducing a Lagrange variable ρ that corresponds to the last constraint, we obtain the partial Lagrangian:

$$\max_{\rho \geq 0} -\rho + \sum_k \min_{\mathbf{x}_k \in C_k} \left\{ \frac{1}{2} \|\mathbf{x}_k - \boldsymbol{\alpha}_k\|_2^2 + \rho \mathbf{1}'\mathbf{x}_k \right\}. \tag{82}$$

Since the optimal $\mathbf{x}_k$ is unique by strong convexity, we can solve ρ by any smooth solver such as BFGS, proximal bundle method (PBM, http://napsu.karmitsa.fi/proxbundle/), or even bi-section. Given a ρ and its optimal $\mathbf{x}_k(\rho)$, the gradient in ρ can be easily computed as $-1 + \sum_k \mathbf{1}'\mathbf{x}_k(\rho)$. Therefore it suffices to optimize $\mathbf{x}_k$ separately. In the sequel, we will first present the optimization procedure without worrying about the computational cost. After that, we will show how to reduce the complexity to $\tilde{O}(n)$.

Fixing ρ, the optimal $\mathbf{x}_k$ can be found by solving the following problem. Here we dropped all subscripts k to lighten the notation.

$$\min_{r \geq 0} \min_{\mathbf{x} \in [0, \frac{r}{k}]^n, \mathbf{1}'\mathbf{x} = r} \frac{1}{2} \|\mathbf{x} - \boldsymbol{\alpha}\|_2^2 + \rho \mathbf{1}'\mathbf{x}. \tag{83}$$

Introducing a Lagrange multiplier μ for the $\mathbf{1}'\mathbf{x} = r$ constraint, we dualize the inner problem as

$$\begin{aligned}
&\min_{r \geq 0} \min_{\mathbf{x} \in [0, \frac{r}{k}]^n} \max_{\mu} \frac{1}{2} \|\mathbf{x} - \boldsymbol{\alpha}\|_2^2 + \rho \mathbf{1}'\mathbf{x} - \mu(\mathbf{1}'\mathbf{x} - r) \\
&= \min_{r \geq 0} \max_{\mu} \left\{ \mu r + \min_{\mathbf{x} \in [0, \frac{r}{k}]^n} \frac{1}{2} \|\mathbf{x} - \boldsymbol{\alpha}\|_2^2 + \rho \mathbf{1}'\mathbf{x} - \mu \mathbf{1}'\mathbf{x} \right\} && (84) \\
&= \min_{r \geq 0} \max_{\mu} \left\{ \mu r + \min_{\mathbf{y} \in [0,1]^n} \frac{1}{2} \left\| \tfrac{r}{k}\mathbf{y} - \boldsymbol{\alpha} \right\|_2^2 + (\rho - \mu) \tfrac{r}{k} \mathbf{1}'\mathbf{y} \right\}. && (85)
\end{aligned}$$

Now r does not appear in the constraint and so the once we have the optimal μ and $\mathbf{y}$ (or optimal $\mathbf{x}$ based on which we get the optimal $\mathbf{y} = \frac{k}{r}\mathbf{x}$), the gradient in r can be written as

$$\mu + \mathbf{y}'(\tfrac{r}{k}\mathbf{y} - \boldsymbol{\alpha})/k + (\rho - \mu)\mathbf{1}'\mathbf{y}/k. \tag{86}$$

which can be calculated in constant time via our efficient update rule.

Algorithm 3: Euclidean projection of $\{\boldsymbol{\alpha}_k\}$ on S

```
1  $\rho^* = $ minimize(obj_rho, $[0, +\infty)$)
2  [$\sim$, $\sim$, $\{\mathbf{x}_k\}$] = obj_rho($\rho^*$).
3  Return $\{\mathbf{x}_k\}$

   Function [$f$, $g$, $\{\mathbf{x}_k\}$] = obj_rho($\rho$)
4  for $k = 1, \ldots, n$ do
5      $r_k$ = minimize(@(r)obj_r(r, $k$, $\rho$),
         $[0, +\infty)$)
6      [$f_k$, $\sim$, $\mathbf{x}_k$] = obj_r($r_k$, $k$, $\rho$)
                        ▷ $\mathbf{x}_k = \mathbf{x}_k(r_k) = \mathbf{x}_k(\rho)$
7  $f = \rho - \sum_{k=1}^n f_k$
8  $g = 1 - \sum_{k=1}^n \mathbf{1}'\mathbf{x}_k$
   end function

   Function [$f$, $g$, $\mathbf{x}$] = obj_r($r$, $k$, $\rho$)
9  $\mu_{\min} = \rho - \max_s \alpha_{ks}$,
     $\mu_{\max} = \rho - \min_s \alpha_{ks} + \frac{r}{k}$
10 while true do                     ▷ bi-section search
11     $\mu = (\mu_{\min} + \mu_{\max})/2$
12     $\mathbf{x} = \mathrm{MED}(\boldsymbol{\alpha}_k + \mu\mathbf{1} - \rho\mathbf{1}, \mathbf{0}, \frac{r}{k}\mathbf{1})$
13     if $\mathbf{1}'\mathbf{x} > r + 10^{-5}$ then
14         $\mu_{\max} = \mu$
15     else if $\mathbf{1}'\mathbf{x} < r - 10^{-5}$ then
16         $\mu_{\min} = \mu$
17     else
18         break
19 $f = \frac{1}{2}\|\mathbf{x} - \boldsymbol{\alpha}_k\|_2^2 + \rho\mathbf{1}'\mathbf{x}$  ▷ Now $\mathbf{x} = \mathbf{x}_k(r)$
20 $g = \mu + \mathbf{y}'(\frac{r}{k}\mathbf{y} - \boldsymbol{\alpha})/k + (\rho - \mu)\mathbf{1}'\mathbf{y}/k$  ▷
     Now $\mu = \mu_k(r)$
   end function
```

Algorithm 4: Entropic projection of $\{\boldsymbol{\alpha}_k\}$ on S

```
1  $\rho^* = $ minimize(obj_rho, $[0, +\infty)$)
2  [$\sim$, $\sim$, $\{\mathbf{x}_k\}$] = obj_rho($\rho^*$).
3  Return $\{\mathbf{x}_k\}$

   Function [$f$, $g$, $\{\mathbf{x}_k\}$] = obj_rho($\rho$)
4  for $k = 1, \ldots, n$ do
5      $r_k$ = minimize(@(r)obj_r(r, $k$, $\rho$),
         $[0, +\infty)$)
6      [$f_k$, $\sim$, $\mathbf{x}_k$] = obj_r($r_k$, $k$, $\rho$)
                        ▷ $\mathbf{x}_k = \mathbf{x}_k(r_k) = \mathbf{x}_k(\rho)$
7  $f = \rho - \sum_{k=1}^n f_k$
8  $g = 1 - \sum_{k=1}^n \mathbf{1}'\mathbf{x}_k$
   end function

   Function [$f$, $g$, $\mathbf{x}$] = obj_r($r$, $k$, $\rho$)
9  $\mu_{\min} = -50$,
     $\mu_{\max} = \rho + \log(\frac{r}{k * \min_s \alpha_{ks}})$
10 while true do                ▷ bi-section search
11     $\mu = (\mu_{\min} + \mu_{\max})/2$
12     $\mathbf{x} = \mathrm{MIN}(\boldsymbol{\alpha}\exp(\mu - \rho), \frac{r}{k}\mathbf{1})$
13     if $\mathbf{1}'\mathbf{x} > r + 10^{-5}$ then
14         $\mu_{\max} = \mu$
15     else if $\mathbf{1}'\mathbf{x} < r - 10^{-5}$ then
16         $\mu_{\min} = \mu$
17     else
18         break
19 $f = \rho\mathbf{1}'\mathbf{x} + \sum_s Q$    ▷ Now $\mathbf{x} = \mathbf{x}_k(r)$
20 $g = \mu + \sum_s \frac{y_s}{k}\log\frac{y_s r}{\alpha_s k} + (\rho - \mu)\frac{y_s}{k}$  ▷
     Now $\mu = \mu_k(r)$
   end function
```

Given r and μ, the optimal $\mathbf{x}$ admits a closed form

$$\mathbf{x} = \mathrm{MED}(\boldsymbol{\alpha} + \mu\mathbf{1} - \rho\mathbf{1}, \mathbf{0}, \tfrac{r}{k}\mathbf{1}), \tag{87}$$

where MED stands for the elementwise median. Given r, the optimal μ is the one that ensures $\mathbf{1}'\mathbf{x} = r$ (*not* necessarily unique). Since each x in (87) is non-decreasing in μ, a simple bi-section search can find such a $\mu(r)$ by probing $O(\log n)$ values of μ. With $\mu(r)$ in hand, the optimal $\mathbf{x}(r)$ for the inner problem in (83) can be recovered by (87).

In hindsight, we observe that although the optimal $\mathbf{x}$ in (83) is unique, the objective function in r is not necessarily smooth because r also appears in the constraints of $\mathbf{x}$. Therefore we resort to a nonsmooth solver (e.g. PBM) for optimizing over r.

The overall procedure for Euclidean projection is summarized in Algorithm 3. We assumed without loss of generality that for each $\boldsymbol{\alpha}_k$ all its elements are already sorted increasingly. The binary search over μ can be refined, with μ only probing kink points corresponding to the entries in $\boldsymbol{\alpha}_k$. This will ensure the bi-section terminates in $O(\log\min\{n, 1/\epsilon\})$ iterations.

D.2 Entropic projection to S

Given a set $\{\boldsymbol{\alpha}_k\}$, the entropic projection to S requires solving

$$\min_{\{\mathbf{x}_k\}} \sum_{ks} x_{ks}\log\frac{x_{ks}}{\alpha_{ks}} + \alpha_{ks} - x_{ks} \tag{88}$$

$$s.t.\ \mathbf{x}_k \in C_k,\ \sum_{ks} x_{ks} \le 1 \quad \text{where} \quad C_k := \{\mathbf{x} \in [0, \frac{r_k}{k}]^n : r_k \ge 0,\ \mathbf{1}'\mathbf{x} = r_k\}$$

Introducing a Lagrange variable ρ that corresponds to the last constraint, we obtain the partial Lagrangian:

$$\max_{\rho\geq 0} -\rho + \sum_{ks} \min_{\mathbf{x}_k \in C_k} \left\{ x_{ks} \log \frac{x_{ks}}{\alpha_{ks}} + \alpha_{ks} - x_{ks} + \rho x_{ks} \right\}. \tag{89}$$

Since the optimal $\mathbf{x}_k$ is unique by strong convexity, we can solve ρ by any smooth solver such as BFGS, PBM, or even bi-section. Given a ρ and its optimal $\mathbf{x}_k(\rho)$, the gradient in ρ can be easily computed as $-1 + \sum_k \mathbf{1}'\mathbf{x}_k(\rho)$. Therefore it suffices to optimize $\mathbf{x}_k$ separately.

Fixing ρ, the optimal $\mathbf{x}_k$ can be found by solving the following problem. Here we dropped all subscripts k to lighten the notation.

$$\min_{r\geq 0} \min_{\mathbf{x}\in[0,\frac{r}{k}]^n,\, \mathbf{1}'\mathbf{x}=r} \sum_s \{Q + \rho x_s\} \quad \text{where} \quad Q = x_s \log \frac{x_s}{\alpha_s} + \alpha_s - x_s \tag{90}$$

Introducing a Lagrange multiplier μ for the $\mathbf{1}'\mathbf{x} = r$ constraint, we dualize the inner problem as

$$\min_{r\geq 0} \max_{\mu} \left\{ \mu r + \min_{x_s\in[0,\frac{r}{k}]} \sum_s \{Q + \rho x_s - \mu x_s\} \right\} \tag{91}$$

$$= \min_{r\geq 0} \max_{\mu} \left\{ \mu r + \min_{y_s\in[0,1]} \sum_s \{Q_y + (\rho - \mu)\frac{r}{k} y_s\} \right\} \tag{92}$$

$$\text{where } Q_y = \frac{r}{k} y_s \log \frac{y_s r}{\alpha_s k} + \alpha_s - \frac{r}{k} y_s$$

the gradient in r can be written as

$$\mu + \sum_s \frac{y_s}{k} \log \frac{y_s r}{\alpha_s k} + (\rho - \mu)\frac{y_s}{k}. \tag{93}$$

which can be calculated in constant time via our efficient update rule.

Given r and μ, the optimal $\mathbf{x}$ admits a closed form

$$\mathbf{x} = \text{MIN}\left(\boldsymbol{\alpha} \exp(\mu - \rho), \frac{r}{k}\mathbf{1}\right), \tag{94}$$

where MIN stands for the elementwise minimum. Given r, the optimal μ is the one that ensures $\mathbf{1}'\mathbf{x} = r$ (*not* necessarily unique). Since each x in (94) is non-decreasing in μ, a simple bi-section search can find such a $\mu(r)$ by probing $O(\log n)$ values of μ. With $\mu(r)$ in hand, the optimal $\mathbf{x}(r)$ for the inner problem in (90) can be recovered by (94).

The overall procedure for Entropic projection is summarized in Algorithm 4. We assumed without loss of generality that for each $\boldsymbol{\alpha}_k$ all its elements are already sorted increasingly. The binary search over μ can be refined, with μ only probing kink points corresponding to the entries in $\boldsymbol{\alpha}_k$. This will ensure the bi-section terminates in $O(\log n) < O(\log 1/\epsilon)$ iterations.

E Rates for Proximal Saddle-Point Optimization (Non-uniform)

Problem. We consider the following problem

$$(x^*, y^*) = \arg\min_x \max_y K(x,y) + M(x,y), \quad \text{where} \quad K(x,y) = \frac{1}{n}\sum_{i=1}^n \psi_i(x,y). \tag{95}$$

Assumption 2. We assume each ψ_i is a saddle function that is L_i-smooth as follows:

$$L_i = \sup_{z\neq z'} \frac{\|B_i(z) - B_i(z')\|_*}{\|z - z'\|}, \quad \text{where} \quad B_i(z) = [\partial_x \psi_i(x,y); -\partial_y \psi_i(x,y)] \tag{96}$$

and K is L_{avg}-smooth:

$$L_{avg} = \sup_{z\neq z'} \frac{\|B(z) - B(z')\|_*}{\|z - z'\|}, \quad \text{where} \quad B(z) = [\partial_x \psi(x,y); -\partial_y \psi(x,y)] \tag{97}$$

Then we can define $\bar{L}$ adapted to our sampling schemes:

$$\bar{L}(\pi)^2 = \sup_{z \neq z'} \frac{\sum_{i=1}^n \frac{1}{n^2 \pi_i} \|B_i(z) - B_i(z')\|_*^2}{\|z - z'\|^2}, \quad \text{where} \quad B_i(z) = [\partial_x \psi_i(x, y); -\partial_y \psi_i(x, y)] \tag{98}$$

and π is a probability vector that sums to 1. We always have the bound:

$$L_{avg}^2 \leq \bar{L}(\pi)^2 \leq \max_{i=1}^n L_i^2 \times \sum_{i=1}^n \frac{1}{n^2 \pi_i}. \tag{99}$$

Algorithm. Let us define a variant of variance-reduced stochastic gradient for saddle-point problems:

$$v_t := [v_x(z_t); -v_y(z_t)], \tag{100}$$

$$\text{where } v_x(z_t) := \frac{1}{n\pi_\xi}(\partial_x \psi_\xi(z_t) - \partial_x \psi_\xi(\tilde{z})) + \partial_x K(\tilde{z}) \tag{101}$$

$$v_y(z_t) := \frac{1}{n\pi_\xi}(\partial_y \psi_\xi(z_t) - \partial_y \psi_\xi(\tilde{z})) + \partial_y K(\tilde{z}). \tag{102}$$

Here $\tilde{z}$ is the pivot chosen after completing the last epoch, ξ is randomly choose from probability vector π. Clearly, $\mathbb{E}_\xi[v_t] = \mathsf{G}(z_t)$ (unbiased). The stochastic algorithm then performs the proximal update at each step:

$$(x_{t+1}, y_{t+1}) = \arg\min_x \max_y \eta \langle v_x(z_t), x \rangle + \eta \langle v_y(z_t), y \rangle + \eta M(x, y) + \Delta(x, x_t) - \Delta(y, y_t). \tag{103}$$

Theorem 3. *With the above modification, the same guarantee in Theorem 1 with L (in fact, this L is* $\max_{i=1}^n L_i$ *) replaced by* $\bar{L} = \bar{L}(\pi)$ *holds.*

Proof. Our key innovation in analysis is the introduction of an auxiliary variable:

$$u_t = \begin{pmatrix} \partial_x K(x_t, y^*) \\ -\partial_y K(x^*, y_t) \end{pmatrix}. \tag{104}$$

Note that $\mathbb{E}_\xi v_t \neq u_t$. The first step of our proof is to invoke Lemma 1 on the update (103):

$$(1 + \eta)\Delta(z^*, z_{t+1}) \leq \Delta(z^*, z_t) - \eta\epsilon_{t+1}^M - \Delta(z_{t+1}, z_t) + \eta \langle v_t, z - z_t \rangle + \eta \langle v_t, z_t - z_{t+1} \rangle \tag{105}$$

$$\begin{aligned} &= \Delta(z^*, z_t) - \eta\epsilon_{t+1}^M - \Delta(z_{t+1}, z_t) + \eta \langle v_t, z - z_t \rangle \\ &\quad + \eta \langle v_t - u_t, z_t - z_{t+1} \rangle + \eta \langle u_t, z_t - z_{t+1} \rangle . \end{aligned} \tag{106}$$

It is easy to bound $\langle u_t, z_t - z_{t+1} \rangle$ as K is L_{avg}-smooth:

$$\begin{aligned} \langle u_t, z_t - z_{t+1} \rangle &= \langle \partial_x K(x_t, y^*), x_t - x_{t+1} \rangle - \langle \partial_y K(x^*, y_t), y_t - y_{t+1} \rangle && (107) \\ &\leq K(x_t, y^*) - K(x_{t+1}, y^*) + \tfrac{L_{avg}}{2} \|x_t - x_{t+1}\|^2 \\ &\qquad + K(x^*, y_{t+1}) - K(x^*, y_t) + \tfrac{L_{avg}}{2} \|y_t - y_{t+1}\|^2 && (108) \\ &= \epsilon_t^K - \epsilon_{t+1}^K + \tfrac{L_{avg}}{2} \|z_t - z_{t+1}\|^2 && (109) \\ &= \epsilon_t^K - \epsilon_{t+1}^K + \tfrac{\bar{L}}{2} \|z_t - z_{t+1}\|^2 . && (110) \end{aligned}$$

The last inequality is due to $L_{avg}^2 \leq \bar{L}^2$.

So we can proceed by

$$\begin{aligned} (1 + \eta)\Delta(z^*, z_{t+1}) &\leq \Delta(z^*, z_t) - \eta\epsilon_{t+1} + \eta\epsilon_t^K + \eta \langle v_t, z^* - z_t \rangle + \eta \langle v_t - u_t, z_t - z_{t+1} \rangle && (111) \\ &\qquad - \tfrac{1-\eta\bar{L}}{2} \|z_t - z_{t+1}\|^2 \\ &\leq \Delta(z^*, z_t) - \eta\epsilon_{t+1} + \eta\epsilon_t^K + \eta \langle v_t, z^* - z_t \rangle + \frac{\eta^2 \|v_t - u_t\|_*^2}{2(1 - \eta\bar{L})}. && (112) \end{aligned}$$

Take expectation over ξ on both sides (conditional on z_t). Since $\mathbb{E}_\xi[v_t] = \mathsf{G}(z_t)$, we apply the inequality $K(x, y') - K(x', y) \le \langle \mathsf{G}(z), z - z' \rangle$ and get:

$$(1+\eta)\mathbb{E}\Delta(z^*, z_{t+1}) \le \Delta(z^*, z_t) - \eta \mathbb{E}\epsilon_{t+1} + \frac{\eta^2}{2(1-\eta\bar{L})}\mathbb{E}\left\|v_t - u_t\right\|_*^2. \tag{113}$$

Finally we bound $\mathbb{E}\left\|v_t - u_t\right\|_*^2$:

$$\begin{aligned}\mathbb{E}\left\|v_t - u_t\right\|_*^2 &= \mathbb{E}\left\|v_t - \mathsf{G}(z_t) + \mathsf{G}(z_t) - u_t\right\|_*^2 \\ &\le 2\mathbb{E}\left\|v_t - \mathsf{G}(z_t)\right\|_*^2 + 2\left\|\mathsf{G}(z_t) - u_t\right\|_*^2. \end{aligned} \tag{114}$$

Notice that by the L-smoothness of K,

$$\left\|\mathsf{G}(z_t) - u_t\right\|_*^2 = \left\|\partial_x K(x_t, y_t) - \partial_x K(x_t, y^*)\right\|_*^2 + \left\|\partial_y K(x_t, y_t) - \partial_y K(x^*, y_t)\right\|_*^2 \tag{115}$$

$$\le L_{avg}^2\left\|y_t - y^*\right\|^2 + L_{avg}^2\left\|x_t - x^*\right\|^2 \tag{116}$$

$$\le \bar{L}^2\left\|y_t - y^*\right\|^2 + \bar{L}^2\left\|x_t - x^*\right\|^2. \tag{117}$$

Again using the definition (98),

$$\begin{aligned}\mathbb{E}\left\|v_t - \mathsf{G}(z_t)\right\|_*^2 &= \mathbb{E}\|\frac{1}{n\pi_\xi}(\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z})) - \mathbb{E}[\frac{1}{n\pi_\xi}(\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z}))]\|_*^2 \\ &\le 4\mathbb{E}\left\|\frac{1}{n\pi_\xi}(\nabla\psi_\xi(z_t) - \nabla\psi_\xi(\tilde{z}))\right\|_*^2 \\ &\le 4\bar{L}^2\left\|z_t - \tilde{z}\right\|^2 \quad \text{(by } \bar{L}\text{-smoothness)} \\ &\le 8\bar{L}^2\left\|z_t - z^*\right\|^2 + 8\bar{L}^2\left\|\tilde{z} - z^*\right\|^2. \end{aligned} \tag{118}$$

Plug (116) and (118) into (114), and then into (113). Using (67), we finally arrive at (expectation is only over ξ)

$$(1+\eta)\mathbb{E}\Delta(z^*, z_{t+1}) \le \Delta(z^*, z_t) - \eta\mathbb{E}\epsilon_{t+1} + \frac{18\bar{L}^2\eta^2}{1-\eta\bar{L}}(\epsilon_t + \epsilon(\tilde{z}^{s-1})). \tag{119}$$

Taking expectation of the whole history on both sides, we obtain

$$\rho\Delta_{t+1} \le \Delta_t + c'\left(\delta_t + \tilde{\delta}^{s-1}\right) - \eta\delta_{t+1}. \tag{120}$$

where $\rho := 1+\eta$, $c' := \frac{18\bar{L}^2\eta^2}{1-\eta\bar{L}}$, $\delta_t := \mathbb{E}\epsilon(z_t)$, $\Delta_t := \mathbb{E}\Delta(z^*, z_t)$, $\tilde{\delta}^{s-1} := \mathbb{E}\epsilon(\tilde{z}^{s-1})$. This has exactly the same shape as (56), and therefore the rest derivation is almost identical, except that in c', we have $\bar{L}^2\eta^2$ rather that $\eta^2 L_Q$ as under (56). So almost all the derivation can be shared. Let us set $\eta = \frac{1}{45\bar{L}^2}$, and we obtain

$$\rho^m = \frac{\eta - c\rho}{c} = \frac{45 - 1/\bar{L}}{18} - \frac{1}{45\bar{L}^2} - 1 \ge \frac{45-1}{18} - \frac{1}{45} - 1 = \frac{64}{45}. \tag{121}$$

Since $\rho = 1 + \frac{1}{45\bar{L}^2}$, we derive

$$m = \frac{\log\left(\frac{45-1/\bar{L}}{18} - \frac{1}{45\bar{L}^2} - 1\right)}{\log\left(1 + \frac{1}{45\bar{L}^2}\right)} = \Theta\left(\bar{L}^2\right). \tag{122}$$

So between epochs, the decay is by a factor of $\frac{45}{64}$, and the number of iterations per epoch is $\Theta(\bar{L}^2)$. The total computational cost is therefore $O\left(\bar{L}^2\log\frac{1}{\epsilon}\right)$.

For uniform sampling, $\pi_i = 1/n$ for all $i = 1, \ldots, n$, we can recover Theorem 1 as $\bar{L} = \max_i L_i$ in this case. The smallest possible value for $\bar{L}$ is $\bar{L} = L_{avg} = (1/n)\sum_{i=1}^n L_i$, achieved at $\pi_i = L_i / \sum_{j=1}^n L_j$, i.e., the sampling probabilities for the component functions are proportional to their Lipschitz constants.

□

F More about L

Let us consider both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as n^2 dimensional vectors. Denote $\mathbf{z} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{pmatrix}$. Then the bilinear part of $\frac{1}{n^2}\sum_{ij} f_{ij}(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_j)$ can be written as $F(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\alpha}' A \boldsymbol{\beta} = \frac{1}{2}\mathbf{z}' \begin{pmatrix} \mathbf{0} & A \\ A' & \mathbf{0} \end{pmatrix} \mathbf{z}$. Then $\nabla F(\mathbf{z}) = \begin{pmatrix} \mathbf{0} & A \\ A' & \mathbf{0} \end{pmatrix} \mathbf{z} = \begin{pmatrix} A\boldsymbol{\beta} \\ A'\boldsymbol{\alpha} \end{pmatrix}$

Recall that $\|\mathbf{z}\|^2 = \|\boldsymbol{\alpha}\|^2 + \|\boldsymbol{\beta}\|^2$ and similarly for their dual norms. We could use subscripts for these norms to highlight that $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\mathbf{z}$ employ different norms. But we omit these subscripts because they are clear from the context.

So the L^2 of F can be computed as

$$L^2 = \max_{\|\mathbf{z}\| \le 1} \left\| \begin{pmatrix} A\boldsymbol{\beta} \\ A'\boldsymbol{\alpha} \end{pmatrix} \right\|_*^2 = \max_{\|\boldsymbol{\alpha}\|^2 + \|\boldsymbol{\beta}\|^2 \le 1} \|A\boldsymbol{\beta}\|_*^2 + \|A'\boldsymbol{\alpha}\|_*^2 . \tag{123}$$

The objective function here is obviously convex in $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ jointly. Since we are maximizing, the optimal solution must be attained at some extreme points of the domain. There can be only two types of extreme points: a) $\|\boldsymbol{\alpha}\| = 1$ and $\boldsymbol{\beta} = \mathbf{0}$; and b) $\boldsymbol{\alpha} = \mathbf{0}$ and $\|\boldsymbol{\beta}\| = 1$. So

$$L^2 = \max \left\{ \max_{\|\boldsymbol{\alpha}\|=1} \|A'\boldsymbol{\alpha}\|_*^2 ,\ \max_{\|\boldsymbol{\beta}\|=1} \|A\boldsymbol{\beta}\|_*^2 \right\}, \tag{124}$$

where the first term corresponds to case a), and the second term to case b). It is not hard to see from definition that these two terms are equal, both being exactly $\max_{\|\boldsymbol{\alpha}\| = \|\boldsymbol{\beta}\| = 1} \boldsymbol{\alpha}' A \boldsymbol{\beta}$.

Now let us add the quadratic term in $\boldsymbol{\alpha}$, so that $G(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\alpha}' A \boldsymbol{\beta} + \frac{1}{2}\boldsymbol{\alpha}' B \boldsymbol{\alpha}$. Then

$$\nabla G(\mathbf{z}) = \begin{pmatrix} B & A \\ A' & \mathbf{0} \end{pmatrix} \mathbf{z} = \begin{pmatrix} B\boldsymbol{\alpha} + A\boldsymbol{\beta} \\ A'\boldsymbol{\alpha} \end{pmatrix} \tag{125}$$

So we can compute the L of G by:

$$L^2 = \max_{\|\mathbf{z}\| \le 1} \left\| \begin{pmatrix} B\boldsymbol{\alpha} + A\boldsymbol{\beta} \\ A'\boldsymbol{\alpha} \end{pmatrix} \right\|_*^2 = \max_{\|\boldsymbol{\alpha}\|^2 + \|\boldsymbol{\beta}\|^2 \le 1} \|B\boldsymbol{\alpha} + A\boldsymbol{\beta}\|_*^2 + \|A'\boldsymbol{\alpha}\|_*^2 \tag{126}$$

$$\le \max \left\{ \max_{\|\boldsymbol{\alpha}\|=1} \|B\boldsymbol{\alpha}\|_*^2 + \|A'\boldsymbol{\alpha}\|_*^2 ,\ \max_{\|\boldsymbol{\beta}\|=1} \|A\boldsymbol{\beta}\|_*^2 \right\} \tag{127}$$

$$\le \max_{\|\boldsymbol{\alpha}\|=1} \|B\boldsymbol{\alpha}\|_*^2 + \max_{\|\boldsymbol{\alpha}\|=1} \|A'\boldsymbol{\alpha}\|_*^2 . \tag{128}$$

where the last equality again used the fact that $\max_{\|\boldsymbol{\alpha}\|=1} \|A'\boldsymbol{\alpha}\|_*^2 = \max_{\|\boldsymbol{\beta}\|=1} \|A\boldsymbol{\beta}\|_*^2$, and $\max_{\boldsymbol{\alpha}}\{f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha})\} \le \max_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) + \max_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha})$.

So now bounding $\max_{\|\boldsymbol{\alpha}\|=1} \|B\boldsymbol{\alpha}\|_*^2$ can be done in the similar way as bounding $\max_{\|\boldsymbol{\alpha}\|=1} \|A'\boldsymbol{\alpha}\|_*^2$.

G Stochastic Updates of SVRG on F-score game

The original problem for F-score is

$$\min_{\{\boldsymbol{\alpha}_i\} \in S} \max_{\{\boldsymbol{\beta}_j\} \in S} \left\{ \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n f_{ij}(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_j) + \max_{\boldsymbol{\theta}} -\frac{\lambda}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{\boldsymbol{\theta}'}{n} X \tilde{\mathbf{y}} - \frac{\boldsymbol{\theta}'}{n} X \sum_{i=1}^n i \boldsymbol{\alpha}_i \right\}. \tag{129}$$

Then the optimal $\boldsymbol{\theta}$ admits a closed form solution

$$\boldsymbol{\theta}(\boldsymbol{\alpha}) = \frac{1}{\lambda n} X \left(\tilde{\mathbf{y}} - \sum_{i=1}^n i \boldsymbol{\alpha}_i \right). \tag{130}$$

Plugging it back and denoting $\mathbf{c} = X\tilde{\mathbf{y}}$, we arrive at the overall problem in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ only:

$$\min_{\{\boldsymbol{\alpha}_i\}\in S} \max_{\{\boldsymbol{\beta}_j\}\in S} \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[f_{ij}(\boldsymbol{\alpha}_i,\boldsymbol{\beta}_j) - \frac{i}{\lambda n}\mathbf{c}'X\boldsymbol{\alpha}_i + \frac{ij}{2\lambda}\boldsymbol{\alpha}_i'X'X\boldsymbol{\alpha}_j + \frac{1}{2\lambda n^2}\left\|\mathbf{c}\right\|_2^2\right]. \tag{131}$$

$$= \min_{\{\boldsymbol{\alpha}_i\}\in S} \max_{\{\boldsymbol{\beta}_j\}\in S} \frac{1}{n}\sum_{j=1}^{n}\psi_j(\boldsymbol{\alpha},\boldsymbol{\beta}_j). \tag{132}$$

where

$$\psi_j(\boldsymbol{\alpha},\boldsymbol{\beta}_j) = \frac{1}{n}\sum_{i=1}^{n}\left[f_{ij}(\boldsymbol{\alpha}_i,\boldsymbol{\beta}_j) - \frac{i}{\lambda n}\mathbf{c}'X\boldsymbol{\alpha}_i + \frac{ij}{2\lambda}\boldsymbol{\alpha}_i'X'X\boldsymbol{\alpha}_j + \frac{1}{2\lambda n^2}\left\|\mathbf{c}\right\|_2^2\right]$$

Denoting $\mathbf{e}_j = (0, 0, \dots, 1, \dots, 0)$, i.e. the canonical row vector for jth dimension. Thus,

1. The stochastic gradient over $\boldsymbol{\alpha}^t$ at iteration t is

$$\begin{aligned}
&\nabla\psi_j(\boldsymbol{\alpha}^t,\boldsymbol{\beta}_j^t)\\
=&\nabla\frac{1}{n}\sum_{i=1}^{n}\left[f_{ij}(\boldsymbol{\alpha}_i^t,\boldsymbol{\beta}_j^t) - \frac{i}{\lambda n}\mathbf{c}'X\boldsymbol{\alpha}_i^t + \frac{ij}{2\lambda}\boldsymbol{\alpha}_i^{t'}X'X\boldsymbol{\alpha}_j^t + \frac{1}{2\lambda n^2}\left\|\mathbf{c}\right\|_2^2\right]\\
=&\nabla\left[\frac{1}{n}\sum_{i=1}^{n}f_{ij}(\boldsymbol{\alpha}_i^t,\boldsymbol{\beta}_j^t)\right] - \frac{X'\mathbf{c}\cdot(1:n)}{\lambda n^2} + \frac{jX'X[\boldsymbol{\alpha}_j^t\cdot(1:n) + (\sum_{i=1}^{n} i\boldsymbol{\alpha}_i^t)\mathbf{e}_j]}{2\lambda n}
\end{aligned}$$

2. The delayed stochastic gradient over anchor variable $\hat{\boldsymbol{\alpha}}$ is

$$\begin{aligned}
&\nabla\psi_j(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}}_j)\\
=&\nabla\left[\frac{1}{n}\sum_{i=1}^{n}f_{ij}(\hat{\boldsymbol{\alpha}}_i,\hat{\boldsymbol{\beta}}_j)\right] - \frac{X'\mathbf{c}\cdot(1:n)}{\lambda n^2} + \frac{jX'X[\hat{\boldsymbol{\alpha}}_j\cdot(1:n) + (\sum_{i=1}^{n} i\hat{\boldsymbol{\alpha}}_i)\mathbf{e}_j]}{2\lambda n}
\end{aligned}$$

3. Full gradient in each epoch is

$$\begin{aligned}
\hat{\mu} &= \frac{1}{n}\sum_{j=1}^{n}\nabla\psi_j(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}}_j)\\
&= \nabla\left[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}f_{ij}(\hat{\boldsymbol{\alpha}}_i,\hat{\boldsymbol{\beta}}_j)\right] - \frac{X'\mathbf{c}\cdot(1:n)}{\lambda n^2} + \frac{X'X(\sum_{j=1}^{n} j\hat{\boldsymbol{\alpha}}_j)\cdot(1:n)}{\lambda n^2}\\
&= \nabla\left[\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}f_{ij}(\hat{\boldsymbol{\alpha}}_i,\hat{\boldsymbol{\beta}}_j)\right] - X'\boldsymbol{\theta}(\hat{\boldsymbol{\alpha}}) * \frac{(1:n)}{n}
\end{aligned}$$

So the euclidean SVRG update in each iteration is

$$\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t - \eta\left[\hat{\mu} + \nabla\psi_j(\boldsymbol{\alpha}^t,\boldsymbol{\beta}_j^t) - \nabla\psi_j(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}}_j)\right] \tag{133}$$