JMLR: Workshop and Conference Proceedings vol 30 (2013) 1–3

Open Problem: Fast Stochastic Exp-Concave Optimization

Tomer Koren TOMERK@TECHNION.AC.IL
Technion—Israel Institute of Technology

Abstract

Stochastic exp-concave optimization is an important primitive in machine learning that captures several fundamental problems, including linear regression, logistic regression and more. The exp-concavity property allows for fast convergence rates, as compared to general stochastic optimization. However, current algorithms that attain such rates scale poorly with the dimension n and run in time $O(n^4)$, even on very simple instances of the problem. The question we pose is whether it is possible to obtain fast rates for exp-concave functions using more computationally-efficient algorithms.

Consider the problem of minimizing a convex function F over a convex set $\mathcal{K} \subseteq \mathbb{R}^n$ where our only access to F is via a stochastic gradient oracle, that given a point $x \in \mathcal{K}$ returns a random vector $\hat{g}_x$ for which $\mathbf{E}[\hat{g}_x] = \nabla F(x)$. We make the following assumptions:

(i) F is α-exp-concave and twice differentiable; that is, if $g_x = \nabla F(x)$ and $H_x = \nabla^2 F(x)$ are the gradient and Hessian at some point $x \in \mathcal{K}$, then $H_x \succeq \alpha\, g_x g_x^\top$.

(ii) The gradient oracle has $\|\hat{g}_x\|_2 \leq G$ with probability 1 at any point $x \in \mathcal{K}$, for some positive constant G.

(iii) For concreteness, we assume the case that $\mathcal{K} = \{x \in \mathbb{R}^n : \|x\|_2 \leq 1\}$ is the Euclidean unit ball.

An important special case is when F is given as an expectation $F(x) = \mathbf{E}_{z\sim\mathcal{D}}[f(x,z)]$ over an unknown distribution $\mathcal{D}$ of parameters z, where for every fixed parameter value z the function $f(x,z)$ is α-exp-concave with gradients bounded by G. Indeed, this implies that F is itself α-exp-concave (see Appendix A). Given the ability to sample from the distribution $\mathcal{D}$, we can implement a gradient oracle by setting $\hat{g}_x = \nabla f(x,z)$ where $z \sim \mathcal{D}$.

For example, $f(x,(a,b)) = \frac{1}{2}(a^\top x - b)^2$ corresponds to linear regression. In a learning scenario it is reasonable to assume that $f(x,(a,b)) \leq M$ with probability 1 for some constant M, which also guarantees that f is exp-concave with $\alpha = 1/M$. Additional examples include the log-loss $f(x,a) = -\log(a^\top x)$ and the logistic loss $f(x,(a,b)) = \log(1+\exp(-b\cdot a^\top x))$, both are exp-concave provided that a, b and x are properly bounded.

The goal of an optimization algorithm, given a target accuracy ε, is to compute a point $\bar{x}$ for which $F(\bar{x}) - \min_{x\in\mathcal{K}} F(x) \leq \varepsilon$ (either in expectation, or with high probability). The standard approach to general stochastic optimization, namely the Stochastic Gradient Descent algorithm, computes an ε-approximate solution using $O(1/\varepsilon^2)$ oracle queries. Since each iteration runs in linear time[1], the total runtime of this approach is $O(n/\varepsilon^2)$.

1. We assume that an oracle query runs in time $O(1)$.

However, the exp-concavity property of F allows for better convergence rates. A standard online-to-batch conversion of the Online Newton Step (ONS) algorithm (Hazan et al., 2007) yields an algorithm[2] that attains a rate of $\tilde{O}(n/\varepsilon)$ for exp-concave functions. Nevertheless, the runtime per iteration of this algorithm is $O(n^2)$, which implies a total runtime of $\tilde{O}(n^3/\varepsilon)$ ignoring projections. When considering the time required to compute a "generalized projection" used by ONS, the runtime becomes as high as $\tilde{O}(n^4/\varepsilon)$, even for very simple domains $\mathcal{K}$ (such as the unit ball). For the technical details, refer to Appendix B.

The poor dependence of the ONS algorithm on the dimension n hinders it from becoming relevant to practical learning applications. Thus, we propose the following problem.

Open Problem: Is it possible to find an optimization algorithm that attains the rate of $\tilde{O}(n/\varepsilon)$ for exp-concave objectives, with only linear-time computation per iteration? Is it possible to perform any better than $\tilde{O}(n^4/\varepsilon)$ overall?

Note that in the case of *strongly convex* functions (which is a subclass of exp-concave functions, see Hazan et al. 2007), it is well known that an ε-approximate solution can be computed in time $\tilde{O}(n/\varepsilon)$ (Hazan et al., 2007; Kakade and Shalev-Shwartz, 2009). Recently, Hazan and Kale (2011) proposed an optimal algorithm for this case that runs in time $O(n/\varepsilon)$, avoiding any logarithmic terms.

We conjecture that better runtimes should be possible for exp-concave functions as well, using algorithms that are based on simple gradient steps. Our intuition is as follows: an exp-concave function can be thought of as a function which is strongly convex in the direction of its gradient. As a concrete example, consider the simple case of linear regression with the squared loss in which the Hessian $H = \nabla^2 F$ is fixed and does not depend on x. Any small eigenvalue of H (that prevents F from being strongly convex) corresponds to a direction in space that does not affect the objective value much, and so the algorithm need not converge quickly in that direction. In other directions, the "directional strong convexity" property should enable a gradient-based algorithm to converge in the fast $1/\varepsilon$ rate. The difficulty is in tuning (perhaps adaptively) the learning rate of such algorithm so as to attain fast convergence in all directions simultaneously.

References

Elad Hazan and Satyen Kale. Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization. *Journal of Machine Learning Research-Proceedings Track*, 19:421–436, 2011.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

Sham Kakade and Shai Shalev-Shwartz. Mind the duality gap: Logarithmic regret algorithms for online optimization. *Advances in Neural Information Processing Systems 21*, pages 1457–1464, 2009.

2. Here we assume that F is an expectation of exp-concave functions $f(x, z)$ and the algorithm has access to a gradient oracle of the form $\hat{g}_x = \nabla f(x, z)$ with $z \sim \mathcal{D}$.

Appendix A. Exp-concavity of $\mathbf{E}_{z\sim\mathcal{D}}[f(x,z)]$

Lemma 1 *If $f(x,z)$ is α-exp-concave for any z, then $F(x)=\mathbf{E}_{z\sim\mathcal{D}}[f(x,z)]$ is also α-exp-concave.*

Proof Fix a point x and denote $g=\nabla F(x)$ and $H=\nabla^2 F(x)$. For all z, let $g_z=\nabla f(x,z)$ and $H_z=\nabla^2 f(x,z)$. Then clearly $\mathbf{E}_{z\sim\mathcal{D}}[g_z]=g$ and $\mathbf{E}_{z\sim\mathcal{D}}[H_z]=H$. In addition, since $f(\cdot,z)$ is exp-concave, we have $\tilde{H}_z\succeq\alpha\tilde{g}_z\tilde{g}_z^\top$ for all z which means that $u^\top\tilde{H}_z u\geq\alpha u^\top\tilde{g}_z\tilde{g}_z^\top u$ for all $u\in\mathbb{R}^n$ with probability 1. Hence, for all u,

$$\begin{aligned}u^\top H_z u &= u^\top\mathbf{E}[\tilde{H}_z]u\\ &\geq\alpha\mathbf{E}[u^\top\tilde{g}_z\tilde{g}_z^\top u]=\alpha\mathbf{E}[\|\tilde{g}_z^\top u\|^2]\\ &\geq\alpha\|\mathbf{E}[\tilde{g}_z^\top u]\|^2=\alpha\|g_z^\top u\|^2\\ &=\alpha u^\top g_z g_z^\top u\end{aligned}$$

which implies that $H_z\succeq\alpha\, g_z g_z^\top$, i.e. F is α-exp-concave. ■

Appendix B. The Online Newton Step (ONS) Algorithm

We give a high-level overview of the ONS algorithm and demonstrate its high computational complexity; for a detailed description of ONS, see Hazan et al. (2007). The algorithm maintains an intermediate solution x_t and a matrix A_t, and uses an update rule of the form

$$x_{t+1}\leftarrow\Pi_{A_t}(x_t-\eta A_t^{-1}g_t),\qquad A_{t+1}\leftarrow A_t+g_t g_t^\top,$$

where g_t is the vector returned by the oracle on iteration t and the "general projection" operator Π_{A_t} is the projection onto the set $\mathcal{K}$ with respect to the norm induced by the matrix A_t, defined as $\|x\|_{A_t}=\sqrt{x^\top A_t x}$. The regret bound of ONS directly implies that $\tilde{O}(n/\varepsilon)$ iterations of the above form are sufficient for convergence.

The matrix A_t, which stands as a proxy for the Hessian of F, is the source of the high computational complexity. Up to generalized projections, each update can be implemented in $O(n^2)$ time[3]. In the case that $\mathcal{K}$ is the Euclidean unit ball, the projection Π_{A_t} requires matrix factorizations and thus computed in time $O(n^3)$.

3. The inverse A_t^{-1} can be updated efficiently via the Sherman-Morrison formula.