Comparison of Algorithms for Noisy First Order Gradients in Stochastic Accelerated Optimization

Chenyang Zhu
chenyang.zhu@berkeley.edu

May 15, 2019

Abstract

In this project, we study stochastic optimization problems with noisy first order gradient descents. We compare the related algorithms on convergence rate in terms of their bias and variance bounds. Two computational examples are computed for strongly convex and non-strongly convex cases to compare methods that are designed for optimal convergence. We also applied the flexible step-size method proposed in Multistage Accelerated Stochastic Gradient Descent (MASG, [1]) to Accelerated Stochastic Approximation Algorithm (ACSA, [2]). Our results show that while the stabilization in MASG seems to work well, it is not yet a universal method and that a more rigorous proof might be necessary before applying this idea to other methods. For reproducible results, We uploaded all the codes to Github[1].

1 Noisy First Order Gradients

In this project, we study optimization problems that deal with noisy first order gradients. People care about the noisy first order gradients because noises tend to accumulate during the convergence, which will lead to unstable convergence or not converging at all. Most noises are caused by stochastic optimization methods or they are involved already in the data set. Many algorithms have been designed to mitigate this noisy influence on the convergence path. In other papers (see, e.g., [3]), they intentionally use noisy gradients for data privacy.

The scope of this project is to compare algorithms that deal with noisy first order gradients in an accelerated gradient descent framework. One can see that though accelerated gradient descent converges faster than the vanilla gradient descent, it's also less stable during the convergence. Many algorithms look seriously into this trade-off and derived optimal conditions on both strongly convex and non-strongly convex cases. The problem can be formulated into the following optimization problem

$$\min_{x\in\mathcal{X}} \quad f(x)+\psi(x) \tag{1}$$

[1]Project Github: https://github.com/chenyangzhu/noisy-gradients

where $\mathcal{X}$ is a closed convex set. $f : \mathcal{X} \to \mathbb{R}$ is a differential and (strongly) convex function. We will keep the definition to allow both convex and strongly convex cases, but one may see in later discussion that some algorithms only allow strongly convex functions. Nevertheless, all functions of $f(x)$ have L-lipschitz continuous gradients. In the mathematical form

$$f(y) - f(x) \geq \nabla f(y)^T(y - x) + \frac{L}{2}||x - y||^2 \tag{2}$$

$$f(x) - f(y) \geq \nabla f(y)^T(x - y) + \frac{\mu}{2}||x - y||^2 \tag{3}$$

for some $L \geq 0$ and $\mu \geq 0$. $\nabla f(x)$ denotes the gradient of f at point x. By allowing $\mu = 0$ in Equation 3, we are also taking non-strongly convex functions into consideration. We denote the condition number of the function as $\kappa = \frac{L}{\mu}$. Another function in Objective 1, $\psi : \mathcal{X} \to \mathbb{R}$ is a simple convex function. It takes the form of a regularization term (i.e. ℓ_1 norm). It does not have to be strongly convex. We then define the Bregman Distance $V(x, y)$ with objective $f(x)$ as,

$$V(x, y) = f(x) - f(y) - \langle \nabla f(y), x - y \rangle$$

The Bregman distance is widely used in many algorithms to study the proximal mapping in dual averaging methods (see, e.g. [2], [4]). Next we define the noisy first order gradients. Despite the variety of definitions in different papers, there are two popular ways to define the noisy first order gradients. In recent papers, people more often use additive models (see, e.g. [1], [5]), where the true gradients are corrupted by an additive random noise. We use $\tilde{\nabla} f(x)$ to denote the corrupted gradients

$$\tilde{\nabla} f(x) = \nabla f(x) + \eta \tag{4}$$

where η is the noise from a predetermined distribution. Notice that this model is very easy to implement and people still use this expression in the implementation,even if models are defined in the other way. The other method is more general. It is written in formal probabilistic framework,

$$\mathbb{E}\Big[\tilde{\nabla} f(x, \eta)\Big] = \nabla f(x)$$

$$\mathbb{E}\Big[||\tilde{\nabla} f(x, \eta) - \nabla f(x)||_2^2\Big] \leq \sigma^2$$

where σ^2 is the bound on the noise term. This framework is more commonly considered in previous research (see, e.g. [2], [4]). Many algorithms try to use this framework to find a fast convergence rate while maintain a relatively low variance. Many of these algorithms characterize the performance using the expected error of the iterates, or $\mathbb{E}[f(x_n)] - f^*$, which admits a bound as a sum of two terms, the bias term and the variance term. While there are many different formulations for these two terms, the most widely accepted one is formed by Nemirovsky and Yudin in [6] and Raginsky and Rakhlim in [7]. Nemirovsky and Yudin showed that with the variance term $\sigma = 0$, the bound on the bias term is simply a decay from the initialization error $x_0 - x^*$, and is independent of the noise term.

$$\mathbb{E}[f(x_n)] - f^* \geq L||x_0 - x^*||_2^2 \exp\left(-\mathcal{O}(1)\right)\frac{n}{\sqrt{\kappa}} \tag{5}$$

Raginsky and Rakhlin proved the lower bound on the variance term,

$$\Omega(\frac{\sigma^2 d}{\mu n}) \tag{6}$$

Notice that these bounds are derived on μ-strongly convex cases and if we set $\mu = 0$, neither equation makes sense. In the next section, we will use this framework to discuss related works and see the motivation of each algorithms and how each of them works.

2 Related Works

Noisy gradients were first studied in 1951 by Robbins and Monro [8]. They introduced the Stochastic Approximation, or classical SA, method that uses gradient descents with noisy gradient information. After this initial approach, many algorithms used the SA idea in stochastic optimization problem (e.g. [9]). These SA algorithms mostly focus on finding the asymptotically optimal rate of convergence, but often have limited performance in practice. [10]. Now, most algorithms proposed with the noisy gradient framework are accelerated methods, modifications of Nesterov's accelerated gradient descent [11].

The first line of research to solve this type of problem is based on subgradient smoothing. In 2005, Juditsky et al. [12] modified Beck and Teboulle's second mirror descent algorithm [13] by keeping a smoothing gradient in the dual space. Based on the primal-dual subgradient methods [14], Xiao [15] proposed the regularized dual averaging (RDA) method, where he uses an average gradient smoothed out through iterations. Chen et al. [4] in 2010 modified Xiao's work and proposed the Optimal Regularized Dual Averaging (ORDA) and multistage ORDA (M-ORDA) method, by using two proximal mappings in each stage of the algorithm. ORDA could achieve optimal convergence rate for both convex and strongly convex cases.

Another line of work originated from the mirror descent Stochastic Approximation (SA) algorithm by Nemirovski et al. [16] in 2009. In this paper, they showed that the mirror descent SA has better performance than previously widely-accepted sample average approximation approach (see, e.g., [17]). This motivated the ACcelerated Stochastic Approximation method (AC-SA) by Ghadimi and Lan in [2]. AC-SA uses three intertwined sequences to track during the convergence. The method is proved to have optimal convergence rate for smooth and non-strongly convex functions but sub-optimal for smooth and strongly convex setting. In 2013, Ghadimi and Lan [18] published the multistage ACSA method where they use a domain-shrinking procedure to improve convergence bound.

There are several problems remaining in this area. While almost all accelerated methods are modifications from Nesterov's AGD, the lack of intuition in the convergence analysis still attracts researchers to study on this topic. Another problem is the robustness of the algorithm when it comes to large noises. To gain convergence rate (i.e. to decrease the bias), most algorithms sacrifice the stability of convergence. Instead of only focusing on optimal convergence as a whole, people start to pay attention to the acceleration and noise trade-off or the stability and robustness problem during convergence (see, e.g., [1], [5], [19]).

Recently, many theoretical improvements have provided alternative proof or intuitive explanations on the accelerated methods, which solves the first problem in the area. For

example, in 2017, Allen-Zhu and Orecchia [20] interpret the AGD algorithm as the combination of mirror descent and gradient descent steps. Krichene et al. [21] provided an ODE interpretation on AGD method. These new improvements have motivated new accelerated methods.

The Accelerated Extra-Gradient Descent (AXGD) [22] from Diakonikolas and Orecchia et al was greatly motivated by the ODE interpretation [21]. In this paper, they proposed a new accelerated method that relies on predictor-corrector approach, which is a new accelerated method different from Nesterov's AGD methods. Cohen et al. [5] then came up with a similar AGD+ algorithm that generalized Nesterov's AGD [11] and AXGD to reduce mean and variance of the error from gradient noises. The algorithm switches the step size during the iteration and this process is called `Restart + SlowDown`. The idea is to switch to a smaller step size, when the accumulative gradients are smaller than the weighted expectation of noisy gradients (i.e. in this case the noisy gradient dominates). In this process, the fast convergence brought by large step sizes decreases bias in the first stage. In the second stage with smaller step sizes, despite the slow convergence rate, the algorithm is more robust to noisy gradients.

This idea to have flexible choices of step sizes to decrease bias in the first stage and then stabilize variance in the second stage also shows up in later works. For example, The Optimal Multistage Accelerated Stochastic Gradient (MASG) algorithm (Aybat et al. [1])has achieved optimal convergence rate without the knowledge of the noisy term and the initial optimality gap. MASG is a modified version of Nesterov Accelerated Gradient Descent method [5]. The idea of MASG is to use a constant step size in the first stage, to decrease the bias before the noisy term accumulates. Then the algorithm switch to decaying step sizes such that the noise term would also decay. This idea requires a very specific criteria on how many steps to perform for the constant step size. They prove that using $n_1 = \lceil \sqrt{\kappa} \log(\kappa) \rceil$, they can achieve the optimal bound on variance without knowing the initial optimality gap Δ or the bound on variance σ. However, the paper does not prove any result on non strongly convex cases. They only showed how the algorithm will work on logistic regression problem in the numerical experiment session.

While different papers focus on different aspect of convergence, e.g.([4] on strongly convex and non-strongly convex functions and [1] on bias and variance tradeoff). The framework of the convergence analysis for these papers are the generalization of the bounds provided in Equation 5 and 6. Chen et al. (Table 1 in [4]) studied in detail the convergence of algorithms on a optimal or uniformly optimal basis. Uniformly optimality means that the algorithm could achieve optimal results in both stochastic and deterministic optimization, which is proposed in [23]. We combine Table 1 in [4] and Table 1 in [1], and shown in Table 1 the convergence analysis of the models of our interest. We also list and compare the features of various algorithms related in this project.

Algorithm	Opt. Bias	Opt. Var.	Final $\hat{x}$	Breg.	$\bar{\nabla}$
AC-SA	✗	✓	Avg.	✓	✗
M-AC-SA	✓	✓	Avg.	✓	✗
ORDA	✗	✓	Prox.	✓	✓
M-ORDA	✓	✓	Prox.	✓	✓
μAGD+	✗	✓	Step	✓	✓
M-ASG (Coro 3.9)	✓	✓	Step	✗	✗

Table 1: Comparison of algorithms in strongly convex case (an extension of Table 1. in [1]). `Final` $\hat{x}$ shows the way for the final update on x. Avg. means the final x is an average of this step and previous step. Prox. means the step is completed with a proximal mapping. Step means a single update with noisy gradient. `Breg.` shows whether the algorithm uses Bregman distance and proximal mapping. $\bar{\nabla}$ shows whether the algorithm uses gradient smoothing method.

3 Strongly Convex Case: Logistic Regression with ℓ_2 norm

3.1 Setup

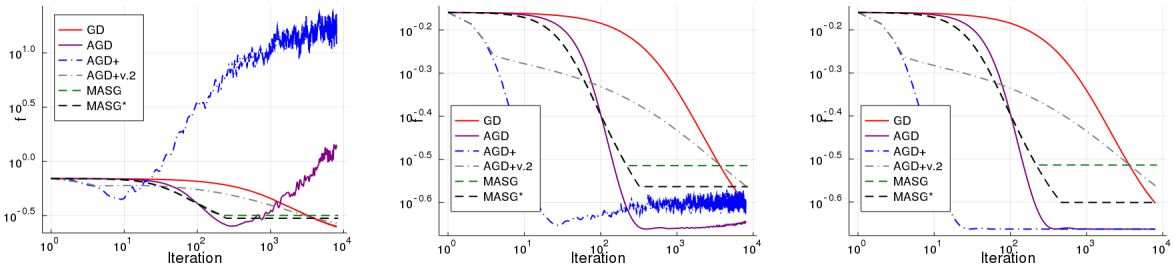
In this section, we compare Accelerated Gradient Descent+ (AGD+ [5]), Multi-stage Accelerated Stochastic Gradient Descent (MASG [1]), Gradient Descent and Accelerated Gradient Descent on a penelized logistic regression problem. Logistic regression is a useful classification algorithm, in the form of a generalized linear model with logit link. For $i \in [1, n]$ observations, $x_i \in \mathbb{R}^d$ denotes the data features, and $y_i \in \{-1, 1\}$ is the label. The problem is stated as follows,

$$\begin{aligned} \min_{\beta} \quad & f(\beta) = \frac{1}{n}\sum_{i=1}^{n} \ln(1 + \exp(-y_i \beta^T x_i)) + \frac{\lambda}{2}||\beta||_2^2 \\ \text{s.t.} \quad & \beta \in \mathbb{R}^p \end{aligned} \tag{7}$$

Logistic loss is a well-known non-strongly convex function, but when we add a ℓ_2 penalty to the loss, the resulting objective is actually strongly-convex [24].

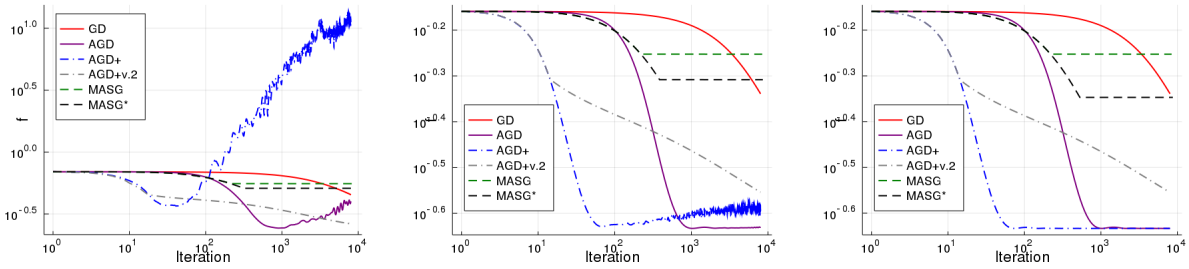
In the computational experiment, we generate an artificial dataset, with the number of observations $n = 1000, 10000$ and the dimension $d = 200$. We first generate the design matrix $X \in \mathbb{R}^{n \times d}$ from standard normal distribution. We then generate a weight vector $\beta \in \mathbb{R}^d$, also from a standard normal distribution. To generate the observation y, we simply multiply X and β and take the sign. i.e. $y = \text{sign}(X\beta)$. This setup is similar to [1]. We always set the penalty hyperparameter $\lambda = 0.01$.

To simulate noisy gradients, we use additive model that adds a noisy term with normal distribution $\mathcal{N}(0, I_d\sigma)$ to the gradient, as shown in Equation (4). Also notice that the ℓ_2 norm of this noise follows a χ_d^2 distribution. Therefore, $\mathbb{E}||\eta||_2^2 = d$, where η is the additive noise term. We use this result in AGD+ algorithm with flexible step sizes.



(a) $\sigma^2 = 10^{-1}$ (b) $\sigma^2 = 10^{-2}$ (c) $\sigma^2 = 10^{-4}$

Figure 1: Computational Result on $n = 1000, d = 100$



(a) $\sigma^2 = 10^{-1}$ (b) $\sigma^2 = 10^{-2}$ (c) $\sigma^2 = 10^{-4}$

Figure 2: Computational Result on n=10000, d=100

Most algorithm requires a specific setup, for example step size, to achieve optimal results. To ensure a fair competition, we use the optimal step sizes from each paper. For vanilla Gradient Descent method and Nesterov's Accelerated Gradient Descent method, we use $1/L$ as step size. For AGD+ algorithm, we run two experiments one with `Restart + SlowDown` process (denote as AGD+v.2) and the other without (denote as AGD). The initial step size a_k is chosen to be $\frac{1}{L}$ and with `Restart + SlowDown`, the step size changes into a slower $1/(\kappa\sqrt{k})$. For MASG, we run the algorithm under two step size choices. Without the knowledge of the initial optimality gap, we first run our algorithm on $n_1 = \lceil \kappa \log(\kappa) \rceil$, and then with the gap known, we run $n_1 = \lceil \sqrt{\kappa} \log(\frac{2L\Delta}{\sigma^2\sqrt{\kappa}}) \rceil$. These are also the two settings that [1] uses to conduct the numerical experiment.

3.2 Results

From our results in Figure 1 and Figure 2, the AGD+ algorithm (in blue dotdash) has fastest convergence rate, but it tends to be very unstable with large σ^2. AGD+ does not even converge when the noisty term is large. An improvement to it uses `Restart and SlowDown` algorithm (in grey dotdash), which has slower convergence rate, but is more robust to noisy gradients. It timely changed the best step size after some iterations, so that it continues to converge.

The MASG has similar convergence rate as AGD, but does not converge to the optimal points eventually. The important thing to notice is that MASG is very robust to large noise terms, but they do not perform well when n is large. We are very curious why MASG

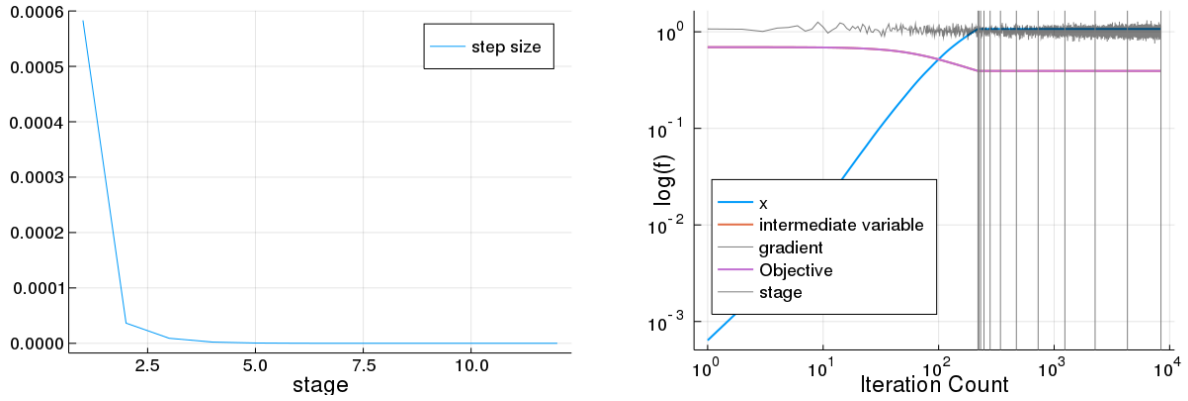


Figure 3: An illustration on the convergence of MASG. The left figure is the step size computed in theorem 3.4 in [1]. The right figure is the updates during the convergence. The vertical line is the stages in MASG. After the first stage, the default step size gets so small that the update almost stops.

stops so early, as the straight lines also appear in their paper [1]. In Figure 3, we show all the updates in the convergence. From the figure, we can see that the gradients performs correctly, but with the step size decreases suddenly, the algorithm seems to cease to update anymore. This step size is derived from Theorem 3.4 in their paper, which uses a $O(\frac{1}{2^{2k}})$ decaying step size. This is the reason the algorithm does not update from a certain iteration. In practice, it might be a good idea to change this theoretical step size with a larger step.

4 Non Strongly Convex Case: Sparse Linear Regression with ℓ_1 regularization

4.1 Setup

After fitting a strongly convex objective, we are also interested in how these algorithms would perform in non-strongly convex cases. Recent papers (e.g. [1], [5]) do not really talk about the non-strongly convex case. The setup of this section is a simple sparse linear regression problem, where we want to

$$\min_{\beta \in \mathbb{R}^n} \quad f(\beta) = \frac{1}{2n}||X\beta - y||_2^2 + \lambda||\beta||_1$$

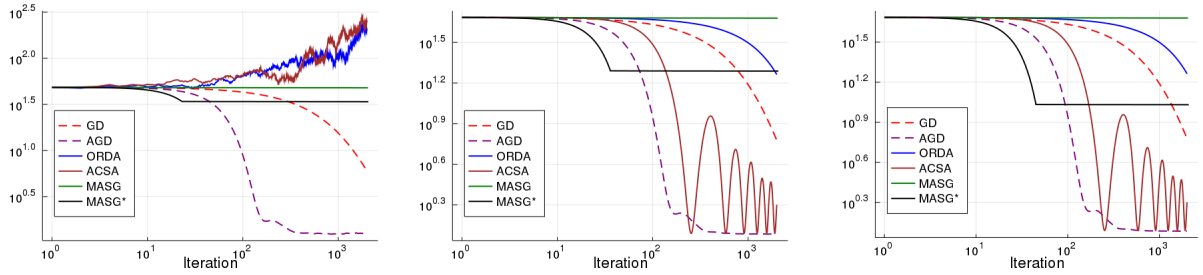
This setup was also similar to [4]. We apply this function to the algorithms above in non-strongly convex case. With this simpler experiment, we can derive a close form for ORDA and AC-SA problems.

We compare the MACG algorithm with GD, AGD, ACSA and ORDA algorithms. AGD+ methods simply cannot be applied to non-strongly convex functions, since it is in the assumption that f and the penalty term must be a strongly convex function. So in this setting we cannot compare the AGD+ algorithm.

To generate the fake data, we first sample from standard normal distribution the $X \in \mathbb{R}^{n \times d}$, We then generate a vector $\beta \in \mathbb{R}^d$ also from standard normal distribution. We multiply

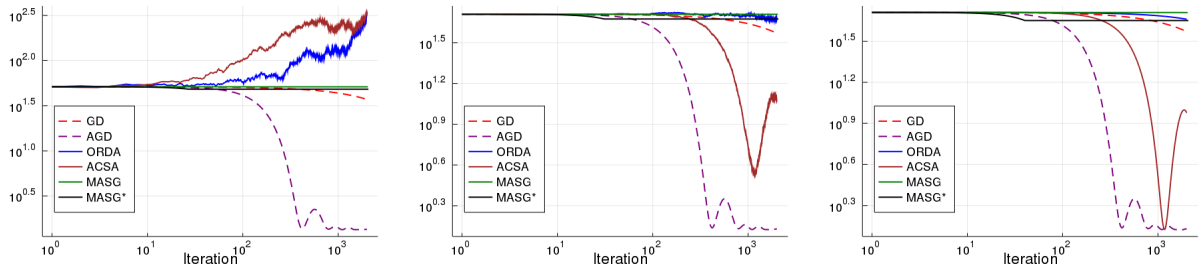
them and add random perturbations $y = X\beta + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 1)$. For fair competition, all parameters are chosen according to the paper and all other setups are the same as in Section 3.

4.2 Results



(a) $\sigma^2 = 10^{-1}$ (b) $\sigma^2 = 10^{-2}$ (c) $\sigma^2 = 10^{-4}$

Figure 4: Computational Result on n=1000, d=100



(a) $\sigma^2 = 10^{-1}$ (b) $\sigma^2 = 10^{-2}$ (c) $\sigma^2 = 10^{-4}$

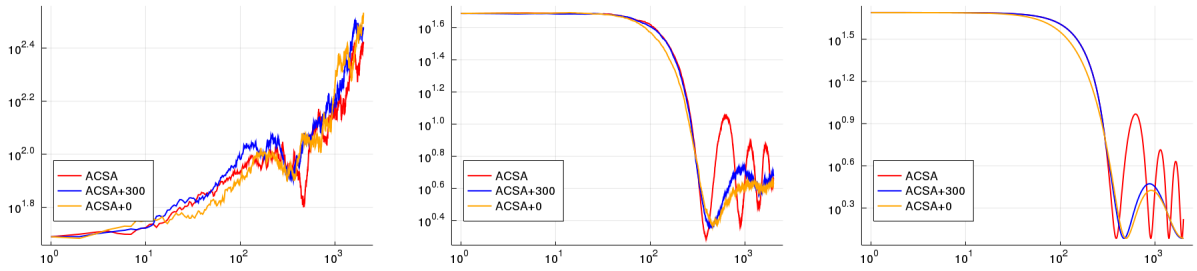
Figure 5: Computational Result on n=10000, d=100

From Figure 4 and Figure 5, none of the new models perform well when the noise term is large, while AGD still performs quite robust. For smaller noises, ACSA appears to have oscillating gradients that bouncing towards the optimal point. On the other hand ORDA is very stable but the convergence rate is so slow that it's even slower than the vanilla gradient descent algorithm. Also notice that the MASG method does not really update, since one of the update in MASG is based on μ, which is set to 0 in non-strongly convex case.

In fact, if you change these hyperparameters and step sizes, the results changes a lot, especially for the variance term. For reproducibility, all codes are uploaded at a Github repository, lest anything suspicious happen that you might want to check the codes and reproduce the results.

5 Apply the long-short step sizes to AC-SA algorithm

In this section, we are curious about how this changing-step-size method could help other algorithms converge. We apply the flexible step setup in μAGD+ and MASG algorithms



(a) $\sigma^2 = 10^{-1}$ (b) $\sigma^2 = 10^{-2}$ (c) $\sigma^2 = 10^{-4}$

Figure 6: Flexible Step sizes applied to AC-SA algorithm. The red line is the vanilla AC-SA with long step sizes. The blue line is applying short step sizes after 300 steps. The yellow line is applying the short step sizes from the first iteration.

into ACSA. The basic idea is very intuitive. During the first stages, we use large a step size to ensure fast convergence. When the error accumulates (i.e. in AGD+) or for a certain step size (i.e. in MASG), we decrease step sizes to stabilize the convergence.

The reason we choose ACSA is that in Figure 4, the gradient of ACSA varies the most. Therefore, we wonder if the steps applied in [1], [5] would make the convergence better. It should also be noticed that in ACSA, the gradients are already computed on smoothed intermediate variables and the final update on x is also averaged out.

The experiment setup is the same as in Section 5, where we apply a non-strongly convex function to ACSA algorithm. Figure 6 shows the result. The same idea seems not applicable in AC-SA algorithm. The result of applying a short step size in later iterations does seem to be better, but if we use this short step size in the first place, it actually has even faster speed than the long step size iteration.

This result is out of our expectations. Due to time limit, we are not able to derive the mathematical optimal step sizes, which might be one of the reason causing this problem. The idea of first using a long step size and later a shorter one requires a well-defined optimal switching point that in this case we did not fully derive. The small step size we use is the proposition 8 in [2].

6 Conclusion and Future Work

In this project, we reviewed some of the papers on the topic of stochastic accelerated optimization problem with noisy first order gradients. While most algorithms (e.g. ACSA[2] and ORDA [4]) try to approximate the optimal bound in both strongly convex and non-strongly convex case, recent papers (e.g. [5], [1]) adopt a flexible or switchable step sizes to interpret the bias and variance trade off in the convergence with noisy first order gradients. We also applied this idea into previous algorithm ACSA, but the result does not meet our expectation. The variance does decrease but still perform worse than if we use the small step size in the first place. We believe that the reason might be the new switching step size method requires a very specific and well-derived constrain on when to switch to the small step size. Future works might include a more general algorithm to find this switch points and a universal bias and variance trade off interpretation.

References

[1] N. S. Aybat, A. Fallah, M. Gurbuzbalaban, and A. Ozdaglar, "A universally optimal multistage accelerated stochastic gradient method", *ArXiv preprint, arXiv:1901.08022*, 2019.

[2] G. Lan and S. Ghadimi, "Optimal stochastic approximation algorithms for strongly convex stochastic compsite optimization, part i: A generic algorithmic framework", *Technical report, University of Florida*, 2010.

[3] R. Bassily, A. Smith, and A. Thakurta, "Private empirical risk minimization: Efficient algorithms and tight error bounds", *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium*, pp. 464–473, 2014.

[4] X. Chen, Q. Lin, and J. Pena, "Optimal regularized dual averaging methods for stochastic optimization", in *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 2012, pp. 395–403.

[5] M. B. Cohen, J. Diakonikolas, and L. Orecchia, "On acceleration with noise-corrupted gradients", *ArXiv e-print, arXiv:1805.12591*, 2018.

[6] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization*. Wiley, 1983.

[7] M. Raginsky, f. Alexander Rakhlin. Information-based complexity, and dynamics in convex programming, *IEEE Transactions on Information Theory*, 57(10):7036–7056, 2011.

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

[9] H. J. Kushner and G. Yin, "Stochastic approximation and recursive algorithms and applications", *Applications of Mathematics*, 2003.

[10] J. Spall, "Introduction to stochastic search and optimization: Estimation, simulation, and control", 2003.

[11] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o (1/k2)", *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.

[12] A. Juditsky, A. Nazin, A. Tsybakov, and N. Vayatis, "Recursive aggregation of estimators by mirror descent algorithm with averaging", *Problems of Information Transmission*, 41:n.4, 2005.

[13] A. Nemirovski and D. B. Yudin, "Problem complexity and method efficiency in optimization", *Wiley Interscience Series in Discrete Mathematics*, 1983.

[14] Y. Nesterov, "Primal-dual subgradient methods for convex problems", *Mathematical Programming*, 120:221–259, 2009.

[15] L. Xiao, "Dual averaging methods for regularized stochastic learning and online optimization", *Journal of Machine Learning Research*, 11:2543–2596, 2010.

[16] A. Juditsky, G. Lan, A. Nemirovski, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming", *SIAM Journal on Optimization*, 2009.

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

[18] S. Ghadimi and G. Lan, "Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, ii: Shrinking procedures and optimal algorithms", *SIAM J. Optimiz*, 23(4):2061–2089, 2013.

[19] M. Hardt. (2014). Robustness versus acceleration, [Online]. Available: `http://blog.mrtz.org/2014/08/18/robustness-versus-acceleration.html`.

[20] Z. Allen-Zhu and L. Orecchia, "Linear coupling: An ultimate unification of gradient and mirror descent", *ArXiv e-print: 1407.1537*, 2014.

[21] W. Krichene, A. Bayen, and P. L. Bartlett, "Accelerated mirror descent in continuous and discrete time", in *Advances in Neural Information Processing Systems 28*, Curran Associates, Inc., 2015, pp. 2845–2853.

[22] L. O. Jelena Diakonikolas, "Accelerated extra-gradient descent: A novel, accelerated first-roder method", *Proc. ITCS'18*, 2018.

[23] A. Nemirovski and D. B. Yudin, "Problem complexity and method efficiency in optimization", *John Wiley New York*, 1983.

[24] C. Guestrin. (2013). L2 regularization for logistic regression, [Online]. Available: `https://courses.cs.washington.edu/courses/cse599c1/13wi/slides/l2-regularization-online-perceptron.pdf`.