FastAdaBelief: Improving Convergence Rate for Belief-based Adaptive Optimizers by Exploiting Strong Convexity

Yangfan Zhou, Kaizhu Huang, Cheng Cheng, Xuguang Wang, Amir Hussain, and Xin Liu

***Abstract*—AdaBelief, one of the current best optimizers, demonstrates superior generalization ability to the popular Adam algorithm by viewing the exponential moving average of observed gradients. AdaBelief is theoretically appealing in that it has a data-dependent $O(\sqrt{T})$ regret bound when objective functions are convex, where T is a time horizon. It remains however an open problem whether the convergence rate can be further improved without sacrificing its generalization ability. To this end, we make a first attempt in this work and design a novel optimization algorithm called FastAdaBelief that aims to exploit its strong convexity in order to achieve an even faster convergence rate. In particular, by adjusting the step size that better considers strong convexity and prevents fluctuation, our proposed FastAdaBelief demonstrates excellent generalization ability as well as superior convergence. As an important theoretical contribution, we prove that FastAdaBelief attains a data-dependant $O(\log T)$ regret bound, which is substantially lower than AdaBelief. On the empirical side, we validate our theoretical analysis with extensive experiments in both scenarios of strong and non-strong convexity on three popular baseline models. Experimental results are very encouraging: FastAdaBelief converges the quickest in comparison to all mainstream algorithms while maintaining an excellent generalization ability, in cases of both strong or non-strong convexity. FastAdaBelief is thus posited as a new benchmark model for the research community.**

***Index Terms*—Adaptive Learning Rate, Stochastic Gradient Descent, Online Learning, Optimization Algorithm, Strong Convexity**

I. Introduction

THE training process is a significant part in many fields of artificial neural networks such as deep learning [1], transfer learning [2] and meta learning [3]. From an optimization perspective, the purpose of the training process is to minimize (or maximize) the loss value (or reward value), and thus can be considered as an optimization process [4]. As a popular paradigm, the training process can be conducted in a supervised way that requires a large number of labeled samples in order to achieve satisfactory performance [5]. On one hand, it may however practically be very difficult due to the high cost involving in annotating samples manually (or even automatically) [6]; on the other hand, even in the case of sufficient label data, it is still a big challenge on how to design both fast and accurate training or optimization algorithms. To tackle this problem, many researchers have been paying great efforts in improving the convergence speed of optimization algorithms, so as to both reduce the need for labeled samples and speed up the training process with available data at hand [7], [8]. Specifically, online learning is often used to accomplish such training tasks because it does not require the information to be collected in batches at the same time [9].

One classic online optimization algorithm is online Stochastic Gradient Descent (SGD) [10]. SGD has been extensively applied over the last few decades in many training tasks of deep learning owing to its simple logic and good generalization ability [11], [12]. However, SGD has the limitation of slow convergence. This disadvantage hinders its application especially in large-scale problems which may take extremely long to converge. To address this issue, researchers have developed various methods trying to speed up the convergence rate for SGD. For example, one type of methods focus on exploring the first-order momentum to accelerate SGD; such methods include SGD with momentum [13] and Nesterov momentum [14]. Typically adopting a fixed step size, these methods may not be conducive to accelerate the convergence rate. To alleviate this problem, recent studies including the popular Adam [15] and AMSGrad [16] attempt to apply the second-order momentum and prefer an adaptive step size while maintaining the first-order momentum.

As one of the most successful adaptive online algorithms, Adam enjoys a fast convergence which is guaranteed with the regret bound of $O(\sqrt{T})$. Despite the outstanding performance, Reddi *et al.* indicated that Adam has the issue

Y. Zhou, C. Cheng, X. Wang, and X. Liu are with Suzhou Institute of Nano-Tech and Nano-Bionics (SINANO), Chinese Academy of Sciences, 398 Ruoshui Road, Suzhou Industrial Park, Suzhou City, Jiangsu Province, 215123 China.

Y.Zhou and X. Liu are also with School of Nano-Tech and Nano-Bionics, University of Science and Technology of China, 96 Jinzhai Road, Hefei City, Anhui Province, 230026 China.

K. Huang is with School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, 215123, China.

A. Hussain is with School of Computing, Edinburgh Napier University, Edinburgh, EH11 4BN, UK

E-mail addresses: yfzhou2020@sinano.ac.cn (Y. Zhou), kaizhu.huang@xjtlu.edu.cn (K. Huang), ccheng2017@sinano.ac.cn (C. Cheng), xgwang2009@sinano.ac.cn (X. Wang), A.Hussain@napier.ac.uk (A. Hussain), xliu2018@sinano.ac.cn (X. Liu).

Corresponding author: Xin Liu (email: xliu2018@sinano.ac.cn).

TABLE I
COMPARISON OF PERFORMANCE ON CONVERGENCE AND GENERALIZATION ABILITY OF FASTADABELIEF AND THE CURRENT MAINSTREAM OPTIMIZERS.

Optimizer	Loss Function	Regret Bound	Convergence	Generalization
SGD ([10])	*convex*	$O(\sqrt{T})$	*slow*	*excellent*
Adam([15])	*convex*	$O(\sqrt{T})$	*medium*	*poor*
SAdam ([23])	*strongly convex*	$O(\log T)$	*fast*	*poor*
AdaBelief ([22])	*convex*	$O(\sqrt{T})$	*medium*	*excellent*
FastAdaBelief (Ours)	*strongly convex*	$O(\log T)$	*fast*	*excellent*

of non-convergence [16], which is caused by not satisfying $\Gamma_t \succeq 0$ for all $t \in \{1, \ldots, T\}$, where $\Gamma_t = \frac{\sqrt{\mathbf{v}_t}}{\alpha_t} - \frac{\sqrt{\mathbf{v}_{t-1}}}{\alpha_{t-1}}$. Moreover, another limitation with Adam is that it could lead to a worse generalization ability than SGD. To tackle this issue, many variants of Adam have been further proposed. For instance, Luo *et al.* [17] proposed AdaBound with a dynamic bound of learning rate; Zaheer *et al.* considered the effect of increasing mini-batch size, and proposed Yogi [18]; Liu *et al.* developed RAdam [19] to rectify the variance of the learning rate; Balles and Hennig dissected Adam in the sign, magnitude, and variance of stochastic gradients, and proposed MSVAG [20]; Loshchilov and Hutter invented AdamW [21] to decouple the weight decay from the loss function.

Although these variants perform better than Adam in generalization ability, there is still a generalization gap compared with SGD on large-scale datasets. To fill this gap, Zhuang *et al.* proposed AdaBelief [22], which adapts the step size by the belief in observed gradients and leads to an superior generalization to Adam. Specifically, AdaBelief re-designs the second-order momentum into a novel form that more was closer to the ideal choice. Moreover, the regret bound of AdaBelief is proved to be $O(\sqrt{T})$ when loss functions are convex.

Albeit its success, it remains however as an open problem *if the convergence rate of AdaBelief can be further improved while not sacrificing its generalization ability*. To this end, in this work we design a novel optimization algorithm called FastAdaBelief that aims to exploiting the strong convexity in order to achieve an even faster convergence rate but maintaining an excellent generalization ability. Particularly, by adjusting the step size that better considers strong convexity, appropriately utilizes curvature information, and prevents fluctuation, our proposed FastAdaBelief attains a substantially lower data-dependant regret bound, which generally promotes AdaBelief from a sublinear level $O(\sqrt{T})$ to a logarithmic level $O\big(\sum_{i=1}^{n} \log\big(\|g_{1:T,i}\|^2\big)\big)$, and to $O(\log T)$ in the worst case. Despite that SGD and some other first-order optimization algorithms can achieve a data-independent bound $O(\log T)$ in online strongly convex optimization, the data-dependent regret bound $O\big(\sum_{i=1}^{n} \log\big(\|g_{1:T,i}\|^2\big)\big)$ can be much tighter than the data-independent bound whenever the gradients are sparse or small that because of $\|g_{1:T,i}\|^2 \ll TG_\infty^2$. To our best knowledge, FastAdaBelief presents a first attempt in designing a powerful optimizer that converges faster with a logarithmic regret bound while maintaining an excellent generalization ability simultaneously.

It is noted that Wang *et al.* proposed SAdam [23] to implement Adam into strong convexity, which is also able to accelerate the regret bound of Adam from $O(\sqrt{T})$ to $O(\log T)$. Unfortunately, SAdam generally has a poor generalization ability as rooted from Adam, which may hence limit its application in practice. In a closer examination, FastAdaBelief adopts the new second order form that is significantly different from the form of SAdam; this brings a brand new challenge for the convergence analysis of FastAdaBelief. Additionally, in order to fit strongly convex conditions, FastAdaBelief designs a tailored diagonal matrix of the second order momentum, which also lead to non-trivial challenge in the convergence analysis compared with Adam. For a simple summary, the performance on convergence and generalization ability of FastAdaBelief and the current mainstream optimizers can be seen in Table I.

Our major contributions are summarized below:

- We propose a fast variant of AdaBelief, named FastAdaBelief, to further improve the convergence rate under strongly convex conditions. We show that FastAdaBelief can lead to adaptive stepsize that is more in line with an ideal optimizer.
- We provide a convergence analysis for FastAdaBelief that presents a data-dependant $O\big(\sum_{i=1}^{n} \log\big(\|g_{1:T,i}\|^2\big)\big)$ guaranteed regret bound, which is substantially better than AdaBelief.
- We conduct extensive experiments to demonstrate that FastAdaBelief outperforms the other state-of-the-art main-stream optimization algorithms in a variety of tasks. Interestingly, even in case of non-strong convexity, FastAdaBelief shows superior performance to the other comparison algorithms consistently in all the datasets.

II. NOTATION AND PRELIMINARIES

A. Notation

In this paper, we use lower case bold letters to denote vectors, such as $\mathbf{x}$. Moreover, $\mathbf{x}_t$ denotes the value of vector $\mathbf{x}$ at time t. Furthermore, the i-th coordinate of vector $\mathbf{x}_t$ is denoted by vector $x_{t,i}$. In addition, matrices are represented in capital letters, such as M. Let $\mathcal{M}_+^n$ denote the set of n dimensional positive definite matrices. The ℓ_2-norm and the ℓ_∞-norm are denoted by $\|\cdot\|$ and $\|\cdot\|_\infty$ respectively. The M-weighted ℓ_2-norm is defined by $\|\mathbf{x}\|_M^2 = \mathbf{x}^\mathsf{T} M \mathbf{x}$. We denote the loss function by $f_t(\cdot)$ at time t, and its

gradient is represented by $\mathbf{g}_t$. Moreover, we use $\mathbf{g}_{1:T,i} = [g_{1,i}, \ldots, g_{T,i}]$ to denote the sequence consisting of the i-th element of the gradient sequence $\{\mathbf{g}_1, \ldots, \mathbf{g}_T\}$. The M-weighted projection operation of $\mathbf{x}$ on $\mathcal{F}$ is represented by $\prod_{\mathcal{F},M}(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathcal{F}} \|\mathbf{y}-\mathbf{x}\|_M^2$ for $\mathbf{x} \in \mathbb{R}^n$. Furthermore, we use $\mathbf{x}^2$ to denote element-wise square, $\frac{\mathbf{x}}{\mathbf{y}}$ for element-wise division, and $\sqrt{\mathbf{x}}$ to represent element-wise square root, where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. Moreover, we take $A = \text{diag}\{\mathbf{x}\}$ to denote the fact that A is a diagonal matrix composed of the elements of vector $\mathbf{x}$. We denote I_d as a $d \times d$ identity matrix. Furthermore, $\mathbf{x}^*$ is used to denote the best decision in hindsight, i.e., $\mathbf{x}^* = \min_{\mathbf{x}\in\mathcal{F}} \sum_{t=1}^{T} f_t(\mathbf{x})$.

B. Online Learning

Machine learning (ML) plays an important role in the field of artificial intelligence. Moreover, offline learning in ML is usually expected to enable a batch of tasks at the same time, but this situation is difficult to meet. In contrast, online learning based on regret considers a sequential setting in which tasks are revealed one by one. Online learning can better adapt to complex and changeable practical applications and has become a prominent paradigm for machine learning, which is attractive in both theory and practice [24]. Within this paradigm, a learner generates iteratively a decision $\mathbf{x}_t$ from a convex and compact domain $\mathcal{F} \subset \mathbb{R}^n$ in each round $t \in \{1, \ldots, T\}$. In response, an adversary produces a convex loss function $f_t(\cdot) : \mathcal{F} \to \mathbb{R}$ in round t, which causes the learner to suffer the loss $f_t(\mathbf{x}_t)$. The goal of the learner is to generate a decision $\mathbf{x}_t$ so that the regret can decrease quickly as T. Moreover, the regret is defined as follows:

$$R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{F}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{1}$$

To improve the generalization ability of the Adam optimizer family, AdaBelief fully considers the curvature information of loss functions, which will be introduced as one preliminary background in the next subsection.

C. AdaBelief

The algorithm design of AdaBelief is shown in Algorithm 1. Reviewing that Adam designs its second-order momentum as the following form:

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1-\beta_2)\mathbf{g}_t^2,$$

where $\mathbf{g}_t$ is the gradient, and $t \in \{1, \ldots, T\}$. Moreover, Algorithm 1 shows that AdaBelief's novel seconde-order momentum is designed as:

$$\mathbf{s}_t = \beta_2 \mathbf{s}_{t-1} + (1-\beta_2)(\mathbf{g}_t - \mathbf{m}_t)^2,$$

where $\mathbf{m}_t$ is the first-order momentum. Note that since the second order momentums of AdaBelief and Adam are quite different, thereby that of Adabelief is symbolized by $\mathbf{s}_t$, and that of Adam by $\mathbf{v}_t$ [15].

To better illustrate the difference of various optimizers, we exploit one illustrative example similarly utilized by

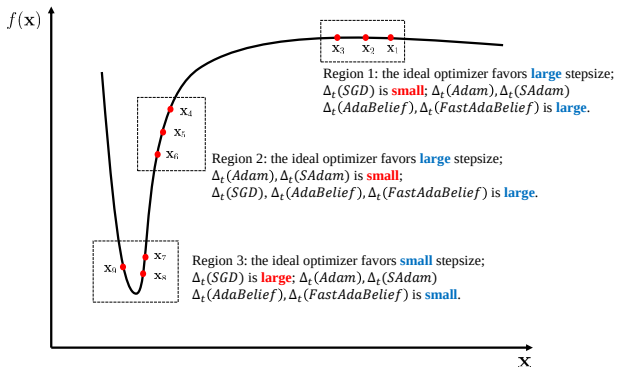


Fig. 1. An ideal optimizer considers the curvature of the loss function and prefers adaptive stepsize. Δ_t denotes the stepsize. FastAdaBelief selects a stepsize more in line with the ideal optimizer (see more details in Table II)(The figure is adapted from [22]).

Algorithm 1: AdaBelief

Input: β_1, β_2
Output: $\mathbf{x}_{t+1}$
Initialize: $\mathbf{x}_0, \mathbf{m}_0, \mathbf{s}_0$
for $t = 1 \ldots T$ **do**
  $t \leftarrow t+1$
  $\alpha_t \leftarrow \frac{\alpha}{\sqrt{t}}$
  $\mathbf{g}_t \leftarrow \nabla f_t(\mathbf{x}_t)$
  $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t$
  $\mathbf{s}_t \leftarrow \beta_2 \mathbf{s}_{t-1} + (1-\beta_2)(\mathbf{g}_t - \mathbf{m}_t)^2$
  $\hat{\mathbf{s}}_t \leftarrow \max\{\hat{\mathbf{s}}_{t-1}, \mathbf{s}_t\}$
  $\hat{S}_t \leftarrow \text{diag}\{\hat{\mathbf{s}}_t\}$
  $\mathbf{x}_{t+1} \leftarrow \prod_{\mathcal{F},\sqrt{\hat{S}_t}} \left(\mathbf{x}_t - \frac{\alpha_t \mathbf{m}_t}{\sqrt{\hat{S}_t}+\epsilon}\right)$
return $\mathbf{x}_{t+1}$

AdaBelief as shown in Figure 1. In region 1 where the loss function is flat, the gradient $\mathbf{g}_t$ and $|\mathbf{g}_t(\mathbf{x_1}) - \mathbf{g}_t(\mathbf{x_2})|$ are both very small. Therefore, a large stepsize should be taken in this case for the efficiency of the optimizer. In this case, AdaBelief and Adam both take large stepsizes, but SGD takes a small one.

In region 2 called the "large gradient, small curvature" case, the gradient $\mathbf{g}_t$ and $\mathbf{v}_t$ are both large while $|\mathbf{g}_t - \mathbf{g}_{t-1}|$ and $\mathbf{s}_t$ are both small. Therefore, the stepsize of an ideal optimizer should be increased. To this end, AdaBelief takes a large stepsize since that denominator $\sqrt{\mathbf{s}_t}$ is small, and SGD also takes a large stepsize. Instead, Adam takes a small stepsize because of the large denominator $\sqrt{\mathbf{v}_t}$.

In region 3, the loss function is "steep". Hence the gradient $\mathbf{g}_t$ and $|\mathbf{g}_t(\mathbf{x_8}) - \mathbf{g}_t(\mathbf{x_9})|$ are both very large. For this reason, an ideal optimizer should take a small stepsize. By the design of the second order momentums in AdaBelief and Adam, they take a small stepsize in this case while SGD exploits a large stepsize.

To sum up, AdaBelief fully considers all the above curvature situations, and adopts a good stepsize selection strategy in each situation. For this reason, AdaBelief has the same good generalization ability as the SGD optimizer

family. Though AdaBelief retains the same regret bound guarantee $O(\sqrt{T})$ as Adam, it is interesting to explore if it can be further sped up. To this end, we propose to utilize the strong convexity and develop a new model that is able to advance the regret bound of Adabelief to logarithmic convergence in this paper.

III. FastAdaBelief

In this section, we first present the detailed design of the proposed algorithm, and then analyze the theoretical guarantee of its regret bound.

A. Algorithm Design

Before presenting the proposed algorithm, we introduce some standard definitions and general assumptions, which follow the previous successful works including [15], [16], [22], [23], [25].

Definition 1: A function $f(\cdot) : \mathcal{F} \to \mathbb{R}$ is σ-strongly convex, where σ is a positive constant, if for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$ the following equation is satisfied

$$f(\mathbf{x}) - f(\mathbf{y}) \geq \nabla f(\mathbf{y})^{\mathsf{T}}(\mathbf{x} - \mathbf{y}) + \frac{\sigma}{2}\|\mathbf{x} - \mathbf{y}\|^2. \quad (2)$$

Assumption 1: The feasible region $\mathcal{F} \in \mathbb{R}^n$ is bounded, that is, for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, $\max_{\mathbf{x},\mathbf{y}\in\mathcal{F}} \|\mathbf{x} - \mathbf{y}\|_\infty \leq D_\infty$, where $D_\infty > 0$ is a constant.

Assumption 2: For all $t \in \{1, \ldots, T\}$, the gradients of all loss functions, $\{\nabla f_t(\mathbf{x})\}_{t=1}^T$, are bounded. Specially, there exists a constant $G_\infty > 0$ such that $\max_{\mathbf{x}\in\mathcal{F}} \|\nabla f_t(\mathbf{x})\|_\infty \leq G_\infty$.

Algorithm 2: FastAdaBelief

Input: $\{\beta_{1t}\}_{t=1}^T, \{\beta_{2t}\}_{t=1}^T, \delta$
Output: $\mathbf{x}_{t+1}$
Initialize: $\mathbf{x}_0, \mathbf{m}_0, \mathbf{s}_0$
for $t = 1 \ldots T$ **do**
  $t \leftarrow t + 1$
  $\alpha_t \leftarrow \frac{\alpha}{t}$
  $\mathbf{g}_t \leftarrow \nabla f_t(\mathbf{x}_t)$
  $\mathbf{m}_t \leftarrow \beta_{1t}\mathbf{m}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t$
  $\mathbf{s}_t \leftarrow \beta_{2t}\mathbf{s}_{t-1} + (1 - \beta_{2t})(\mathbf{g}_t - \mathbf{m}_t)^2$
  $\hat{\mathbf{s}}_t \leftarrow \max\{\hat{\mathbf{s}}_{t-1}, \mathbf{s}_t\}$
  $\hat{S}_t \leftarrow \mathrm{diag}\{\hat{\mathbf{s}}_t\} + \frac{\delta}{t} I_n$
  $\mathbf{x}_{t+1} \leftarrow \prod_{\mathcal{F},\hat{S}_t} \left(\mathbf{x}_t - \alpha_t \hat{S}_t^{-1}\mathbf{m}_t\right)$
return $\mathbf{x}_{t+1}$

Now we present an accelerated and accurate belief-based optimization algorithm for strongly convex functions based on the above standard definitions and assumptions, called FastAdaBelief. The detailed design of the proposed algorithm is shown in Algorithm 2, which follows the general design of [22]. In the proposed algorithm, β_{1t} and β_{2t} are time-variant non-increasing hyper-parameters, and δ is a positive constant. Moreover, the parameter of step size, α_t, is assigned as $\alpha_t = \frac{\alpha}{t}$, where α is a constant. Furthermore, the gradient of loss function at time t, $\mathbf{g}_t$, is calculated by $\mathbf{g}_t = \nabla f_t(\mathbf{x}_t)$. Next, the proposed algorithm computes the first-order momentum, $\mathbf{m}_t$, through Exponential Moving Average (EMA) of $\mathbf{g}_t$, which is shown as follows:

$$\mathbf{m}_t = \beta_{1t}\mathbf{m}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t. \quad (3)$$

Then, the second-order momentum, $\mathbf{s}_t$, in the proposed algorithm is calculated by EMA of the square of the observed gradient belief $(\mathbf{g}_t - \mathbf{m}_t)$, i.e.,

$$\mathbf{s}_t = \beta_{2t}\mathbf{s}_{t-1} + (1 - \beta_{2t})(\mathbf{g}_t - \mathbf{m}_t)^2. \quad (4)$$

Moreover, to satisfy the condition of convergence, i.e., $\Gamma_t = \frac{\sqrt{\mathbf{s}_t}}{\alpha_t} - \frac{\sqrt{\mathbf{s}_{t-1}}}{\alpha_{t-1}} \succeq 0$, the proposed algorithm further provides the following operation on the second-order momentum:

$$\hat{\mathbf{s}}_t = \max\{\hat{\mathbf{s}}_{t-1}, \mathbf{s}_t\}. \quad (5)$$

Furthermore, to avoid step size explosion caused by too small gradients, the proposed algorithm adds a vanishing factor $\frac{\delta}{t}$ to the second-order momentum, and obtains the following diagonal matrix:

$$\hat{S}_t = \mathrm{diag}\{\hat{\mathbf{s}}_t\} + \frac{\delta}{t} I_n. \quad (6)$$

Finally, the proposed algorithm updates the decision point, $\mathbf{x}_{t+1}$, conditional on the projection to the feasible region, and attains the following:

$$\mathbf{x}_{t+1} = \prod_{\mathcal{F},\hat{S}_t} \left(\mathbf{x}_t - \alpha_t \hat{S}_t^{-1}\mathbf{m}_t\right). \quad (7)$$

In general, the proposed algorithm has two main modifications compared with AdaBelief. The first modification is about the step size, which is modified to $\frac{\alpha}{t}\hat{S}^{-1}$. The motivation behind this modification is to satisfy the property of strongly convex optimization. Moreover, the second modification is to change β_2 of AdaBelief to β_{2t}. Such time-varying parameter, β_{2t}, is set to constant β_2 in AdaBelief, which simplifies application and convergence proof but may lead to stepsize fluctuations. In this work, we apply β_{2t} in its original form which achieves good convergence [16].

The detailed design of the proposed algorithm has been introduced in this section. Next, we interpret why the proposed FastAdaBelief can choose a better stepsize and leads to faster convergence. After that, we theoretically prove that when the strong convexity of the loss functions holds, the proposed algorithm has a guaranteed regret bound, which is much better than AdaBelief.

B. Why FastAdaBelief can choose a better stepsize?

From the design o $\hat{S}_t$ in FastAdaBelief, our algorithm adds a vanishing factor to the stepsize, which is originally considered to meet the strongly convex condition, but unexpectedly brings significant benefits to the choice of step size. If we let Δ denote the stepsize, then stepsizes of SGD, Adam, SAdam, AdaBelief, and FastAdaBelief are shown in the following:

$$\Delta_t(SGD) = \alpha \mathbf{m}_t;$$
$$\Delta_t(Adam) = \alpha \mathbf{m}_t / \left(\sqrt{t\mathbf{v}_t}\right);$$
$$\Delta_t(SAdam) = \alpha \mathbf{m}_t / \left(t\mathbf{v}_t + \delta\right);$$
$$\Delta_t(AdaBelief) = \alpha \mathbf{m}_t / \left(\sqrt{t\mathbf{s}_t}\right);$$
$$\Delta_t(FastAdaBelief) = \alpha \mathbf{m}_t / \left(t\mathbf{s}_t + \delta\right).$$

If we look back into Figure 1, in regions 1 and 2, although the step size of FastAdaBelief is slightly smaller than that of AdaBelief, it is still consistent with the optimal choice. FastAdaBelief has large step sizes in both region 1 and 2, but SAdam takes small step sizes. Thereby FastAdaBelief outperforms SAdam on generalization ability. Importantly, the step size of FastAdaBelief decays in general on the order of $O(1/t)$ that allows the optimal solution to be approximated at a smaller step size later in the training process without unnecessary oscillations. In addition, in region 3, the ideal optimizer would prefer a small stepsize. Indeed, FastAdaBelief takes a smaller stepsize than that of AdaBelief in this region, which is due to the addition of vanishing factors.

In a brief summary, comparison of stepsize selection by FastAdaBelief, AdaBelief, SAdam, Adam and SGD can be seen in Table II.This analysis shows that FastAdaBelief, likes AdaBelief, is in line with the choice of the ideal optimizer and therefore can lead to better performance than the other mainstream optimizers. Moreover, FastAdaBelief has a smaller step size than that of AdaBelief in the later stage of training, which allows the optimizer approximates the optimal solution more steadily.

C. Theoretical Guarantee

In this section, we first review some convergence conditions as earlier developed by Reddi [16], which solves the convergence issue for Adam [15]. Let $\{\beta_{2t}\}$ satisfy the following conditions:

a) Condition 1: For some $\zeta > 0$ and all $t \in \{1, \ldots, T\}$, $j \in \{1, \ldots, n\}$, we have that

$$\frac{\sqrt{t}}{\alpha}\sqrt{\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{2(t-k+1)}(1-\beta_{2j})g_{j,i}^2} \geq \frac{1}{\zeta}\sqrt{\sum_{j=1}^{t} g_{j,i}^2}.$$

b) Condition 2.: For all $t \in \{1, \ldots, T\}$ and $i \in \{1, \ldots, n\}$, we have that

$$\frac{\sqrt{t}}{\alpha} s_{t,i}^{1/2} \geq \frac{\sqrt{t-1}}{\alpha} s_{t-1,i}^{1/2}.$$

As a matter of fact, Condition 1 is an important and standard condition for convergence analysis of adaptive momentum algorithms, such as Adam and AdaBelief. Furthermore, the intrinsic motivation for Condition 2 is to follow the key condition of SGD, where its step size $\frac{\alpha}{\sqrt{t}}$ satisfies that $\frac{\sqrt{t}}{\alpha} - \frac{\sqrt{t-1}}{\alpha} \geq 0, \forall t \in [T]$. For this reason, we also follow this motivation, and propose the following conditions with minor modifications:

c) Condition 3.: For some $\zeta > 0$ and all $t \in \{1, \ldots, T\}$, $j \in \{1, \ldots, n\}$, we have that

$$t\sum_{j=1}^{t}\prod_{k=1}^{t-j}\beta_{2(t-k+1)}(1-\beta_{2j})g_{j,i}^2 \geq \frac{1}{\zeta}\sum_{j=1}^{t} g_{j,i}^2. \tag{8}$$

d) Condition 4.: For all $t \in \{1, \ldots, T\}$ and $i \in \{1, \ldots, n\}$, we have that

$$0 \leq \frac{t}{\alpha} s_{t,i}^{1/2} - \frac{t-1}{\alpha} s_{t-1,i}^{1/2} \leq \sigma(1-\beta_1). \tag{9}$$

Now, we present the main results in the following for the convergence analysis when Conditions 3 and 4 are satisfied.

Theorem 1: Suppose that Assumptions 1 and 2 are satisfied, Conditions 3 and 4 hold, and loss functions $f_t(\cdot)$ are σ-strongly convex. Moreover, let parameter sequences $\{\beta_{1t}\}, \{\beta_{2t}\}$ and $\{\alpha_t\}$ are generated by the proposed algorithm, where $\beta_{1t} = \beta_1\lambda^t, \beta_1 \in [0,1), \lambda_1 \in [0,1), \beta_{2t} \in [0,1), \delta > 0, t \in \{1, \ldots, T\}$. For decision point $\mathbf{x}_t$ generated by the proposed algorithm, we have the following upper bound of the regret

$$R(T) \leq \frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{D_\infty^2(G_\infty+\delta)}{2\alpha}\sum_{i=1}^{n}\sum_{t=1}^{T}\frac{\beta_{1t}}{1-\beta_{1t}}t + \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3}\sum_{i=1}^{n}\log\left(\frac{1}{\zeta\delta}\|g_{1:T,i}\|^2 + 1\right).$$

The proof of Theorem 1 is provided in Appendix A. Accordingly, Theorem 1 implies that our proposed algorithm converges with $O\left(\sum_{i=1}^{n}\log\left(\|g_{1:T,i}\|^2\right)\right)$ regret bound in the case of strong convexity. Moreover, the regret bound of the worst case is $O(n\log T)$. In addition, the bound of the regret can be more tighter if the gradients are sparse or small such that $\|g_{1:T,i}\|^2 \ll TG_\infty^2$.

Corollary 1: Letting $\beta_{1t} = \beta_1\lambda^t$, where $\lambda \in (0,1)$ in Theorem 1, we have the following upper bound of the regret

$$R(T) \leq \frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{n\beta_1\lambda D_\infty^2(G_\infty+\delta)}{2\alpha(1-\beta_1)(1-\lambda)^2} + \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3}\sum_{i=1}^{n}\log\left(\frac{1}{\zeta\delta}\|g_{1:T,i}\|^2 + 1\right).$$

The above Corollary 1 also implies that our proposed algorithm has a convergence guarantee $O(n\log T)$ for condition $\beta_{1t} = \beta_1\lambda^t, \lambda \in (0,1), t \in \{1, \ldots, T\}$. Then, our proposed algorithm executes with $\lim_{T\to+\infty}\frac{R(T)}{T} = 0$. Therefore, our proposed algorithm converges when loss functions are strongly convex, and its theoretical proof is provided in Appendix A. In order to verify the performance of our algorithm in specific applications, we present a series of experiments on benchmark public datasets in the following section.

IV. Experiments

In this section, we conduct two groups of experiments to verify that our proposed algorithm works excellently for standard optimization problems in both cases of strong and non-strong convexity. In the first group, we consider

TABLE II
STEPSIZE SELECTED BY THE IDEAL OPTIMIZER, FASTADABELIEF, ADABELIEF, SADAM, ADAM AND SGD IN THREE REGIONS OF FIGURE 1. FASTADABELIEF IS MORE IN LINE WITH THE IDEA OPTIMIZER.

Stepsize	**Region 1**	**Region 2**	**Region 3**	**Later Period**
$\Delta_t(ideal)$	Large value	Large value	Small value	steady
$\Delta_t(SGD)$	S	L	L	oscillating
$\Delta_t(Adam)$	L	S	S	oscillating
$\Delta_t(SAdam)$	L	S	S	steady
$\Delta_t(AdaBelief)$	L	L	S	oscillating
$\Delta_t(FastAdaBelief)$	L	L	S	steady

a strongly convex optimization problem of mini-batch ℓ_2-regularized softmax regression; in the second group, we apply our algorithm into non strongly convex cases of deep training tasks with the traditional softmax function. To be specific, the traditional softmax function is generally convex but not strongly convex when the data is uniformly distributed. In some extreme cases such as sparse or imbalanced data [27], [28], for example, when a certain category does not appear in the training set, softmax may even not satisfy convexity [29]. To examine the effectiveness of FastAdaBelief in real scenarios, we intentionally conduct the second group of experiments on image classification with CNN and language modeling with LSTM respectively, both of which are commonly seen in practice. In all of our experiments, the source codes are implemented in the torch 1.1.0 module of python 3.6 and executed on $4\times$1080ti GPUs. Furthermore, we compare FastAdaBelief with other algorithms in both experiments, including SGD [13], Adam [15], Yogi [18], AdaBound [17], AdaBelief [22], SAdam [23].

A. Hyperparameter Tuning

We perform the following hyperparameter tuning in experiments of image classification and language modeling. To be fair, we initialize the decision variables and momentums of each algorithm to $\mathbf{x}_0 = \mathbf{0}, \mathbf{m}_0 = \mathbf{0}, \mathbf{v}_0 = \mathbf{0}$, and $\mathbf{s}_0 = \mathbf{0}$.

SGD: We follow the standard settings of ResNet [30] and DenseNet [31], and set the momentum as 0.9. We choose the learning rate from $\{10.0, 1.0, 0.1, 0.01, 0.001\}$.
Adam: We adopt the same parameter setting as the original article [15] where the first-order momentum β_1 is set to 0.9, and the second-order momentum β_2 is set to 0.999. Moreover, the step size α_t is set to $\alpha/\sqrt{t}$, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.
Yogi: Following [18], we set the first-order momentum β_1 to 0.9 and set the second-order momentum β_2 to 0.999. In addition, the step size α_t is set to $\alpha/\sqrt{t}$, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.
AdaBound: We directly apply the default hyper-parameters following [17] for AdaBound (i.e., $\beta_1 = 0.9$ and $\beta_2 = 0.999$). Moreover, the step size α_t is set to $\alpha/\sqrt{t}$, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.
AdaBelief: We use the default hyperparameters as suggested in [22], i.e., $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. In addition, the step size α_t is set to $\alpha/\sqrt{t}$, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.
SAdam: We set the hyperparameters by following [23], i.e., $\beta_1 = 0.9$, $\beta_{2t} = 1 - \frac{0.9}{t}$. The step size α_t is set to α/t, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.
FastAdaBelief: We adopt the same hyperparameters as SAdam: $\beta_1 = 0.9$, $\beta_{2t} = 1 - \frac{0.9}{t}$. Moreover, the step size is set to α/t, where α is chosen from $\{0.1, 0.01, 0.001, 0.0001\}$.

It can be seen that, for fair comparison, all the above algorithms basically follow a similar parameter setting.

B. Datasets

In the experiments of CNN based image classification, we perform evaluations on the benchmark CIFAR-10 dataset. Moreover, we apply the algorithms on three standard baseline models, i.e., DenseNet-121, ResNet-34, and VGG-11. DenseNet-121 is a dense convolutional network, which connects each layer to all other layers feed-forwardly; ResNet-34 is a residual learning framework; VGG-11 is a deep network using an architecture with small convolution filters. In the LSTM based language modeling experiments, we test the various algorithms on Penn Treebank dataset. Furthermore, we compare the algorithms in 1,2,3-layer LSTM models. For clarity, we show the summary of datasets and architectures used in our experiments in Table III.

C. Optimization with Strong Convexity

In this group of experiments, we consider a mini-batch task. In round t of this task, the optimizer receives a mini-batch of training samples denoted by $\{\mathbf{x}_m, y_m\}_{i=1}^m$, where m is the batch size, K is the number of classes, and $y_i \in [K]$ and $\forall i \in [m]$. Then the optimizer generates decision vectors denoted by $\{\mathbf{w}_i, b_i\}_{i=1}^K$. Finally, the generated result suffers a loss. The loss function is then given as

$$J(\mathbf{w}) = -\frac{1}{m}\sum_{i=1}^{m}\log\left(\frac{e^{\mathbf{w}_{y_i}^\top \mathbf{x}_i + b_{y_i}}}{\sum_{j=1}^{K} e^{\mathbf{w}_j^\top \mathbf{x}_i + b_j}}\right) + \sigma_1 \sum_{k=1}^{K}\|\mathbf{w}_k\|^2 + \sigma_2 \sum_{k=1}^{K} b_k^2. \tag{10}$$

TABLE III
DATASETS AND ARCHITECTURES USED IN OUR EXPERIMENTS.

Task	Dataset	Architecture
Image Classification	MNIST	4-layers CNN
Image Classification	CIFAR-10	4-layers CNN,DenseNet-121, ResNet-34, VGG-11
Image Classification	CIFAR-100	4-layers CNN
Language Modeling	Penn Treebank	1,2,3-Layer LSTM.

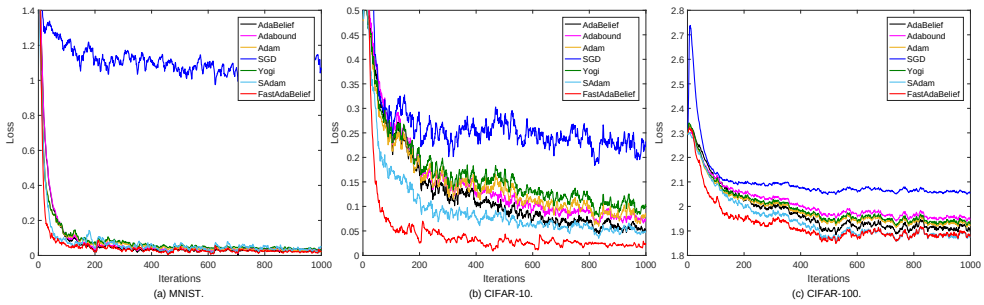


Fig. 2. Loss *v.s.* iterations for mini-batch ℓ_2-regularized softmax regression (**strongly convex**). FastAdaBelief converges the quickest.

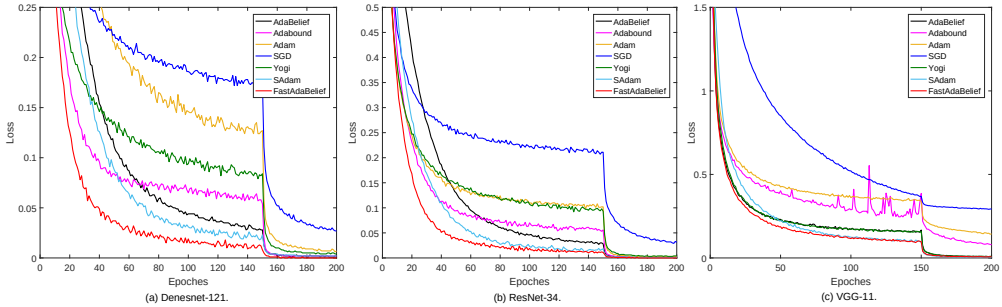


Fig. 3. Comparison of loss of SGD, Adam, AdaBound, Yogi, AdaBelief, SAdam and FastAdaBelief on CIFAR-10. FastAdaBelief converges the quickest.

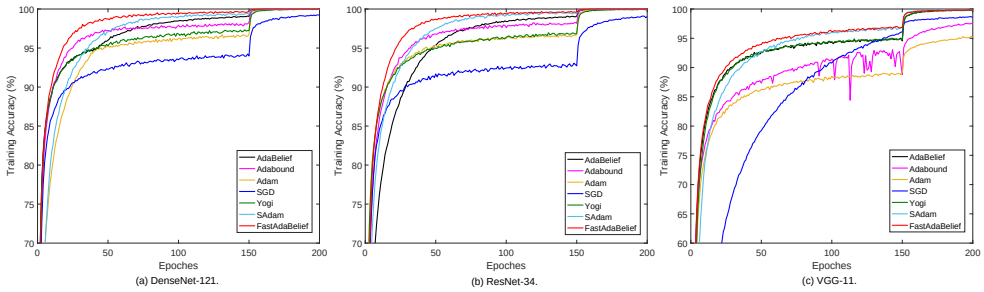


Fig. 4. Comparison of training accuracy of SGD, Adam, AdaBound, Yogi, AdaBelief, SAdam and FastAdaBelief on CIFAR-10. FastAdaBelief achieves the highest training accuracy.

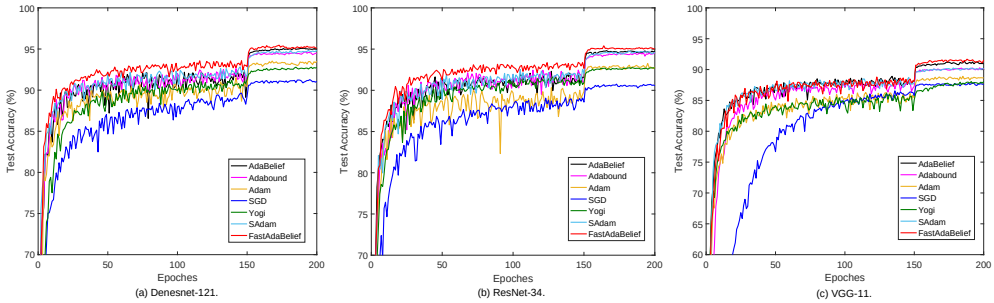


Fig. 5. Comparison of test accuracy of SGD, Adam, AdaBound, Yogi, AdaBelief, SAdam and FastAdaBelief on CIFAR-10. FastAdaBelief achieves the highest test accuracy.

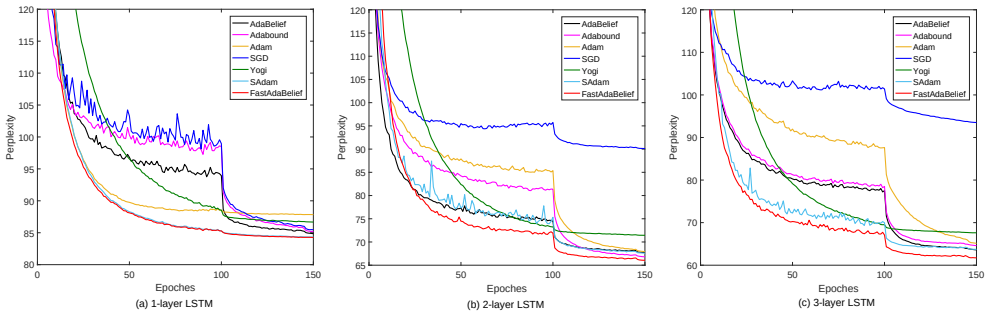


Fig. 6. Comparison of perplexity of SGD, Adam, AdaBound, Yogi, AdaBelief, SAdam and FastAdaBelief (**Lower** is better) on Penn Treebank. FastAdaBelief converges the quickest.

TABLE IV
TEST PERPLEXITY (**lower** IS BETTER) OF 1,2,3-LAYER LSTM ON PENN TREEBANK. NOTE THAT THE TWO BEST PERFORMING ALGORITHMS ARE MARKED IN BOLD.

Model	SGD	Adam	AdaBound	Yogi	AdaBelief	SAdam	FastAdaBelief
1-layer LSTM	85.07	84.28	84.78	86.59	84.21	**84.19**	**84.18**
2-layer LSTM	67.42	67.27	67.53	71.33	**66.29**	68.11	**66.08**
3-layer LSTM	63.58	64.28	63.58	67.51	**61.23**	64.71	**61.21**

In our experiments, we set parameters σ_1 and σ_2 both to 0.01. In addition, we conduct the experiment on the loss *v.s.* iterations. The results of this experiment are shown in Figure 2. As clearly observed, the loss of our proposed algorithm decreases the quickest and FastAdaBelief leads to the best convergence in all the mainstream algorithms. As guaranteed theoretically, the strongly convex optimization algorithms (such as FastAdaBelief and SAdam) outperform the convex optimization algorithms (like Adam, AdaBelief, etc.) in the strongly convex case. When we inspect the difference between the two strongly convex optimization algorithms, FastAdaBelief generates much lower losses (particularly on CIFAR10 and CIFAR100) than SAdam, which echos the advantages of FastAdaBelief than SAdam regarding the generalization ability.

D. Training DNN with Non-strong Convexity

FastAdaBelief enjoys the theoretical superiority to the other mainstream optimizers when strong convexity holds. On the empirical side, the current DNNs may however adopt loss functions that are typically not strongly convex (e.g. only convex). Thus it is both interesting and important to investigate if the proposed fast algorithm can still work well. For this purpose, we conduct a series of empirical study on the tasks of image classification and language modeling in the following.

1) Image Classification: In the experiments of image classification, we take the CIFAR-10 as one typical example and compare the various algorithms with DenseNet-121, ResNet-34 and VGG-11. First, we compare the convergence rate for all the algorithms used in our experiment. Such results are reported in Figure 3. As clearly observed, though the strong convexity may not hold, FastAdaBelief still leads to remarkable convergence, which is consistently faster than all the other algorithms. In comparison, SAdam also converges well, which empirically demonstrates the power of the strongly convex algorithms. Furthermore, SGD converges the slowest; Adam and Adabelief are also much slower than both SAdam and FastAdaBelief. All these empirical results are consistent with the theoretical analysis as we discussed earlier in Section II and Section III though the loss functions are not strongly convex.

Second, we record the training and test accuracy curves of all the algorithms executed in our experiments, which are shown in Figure 4 and Figure 5. We can see that FastAdaBelief outperforms the other comparison algorithms in 200 epochs on DenseNet-121, ResNet-34 and VGG-11. In more details, FastAdaBelief demonstrates much faster convergence as well as the highest accuracy within 200 epochs than all the other comparison algorithms on the three baseline DNN models. Additionally, it is also evident that AdaBelief and FastAdaBelief generally lead to the best accuracy in the 200 epoch, which verifies the excellent generalization ability of the belief-based adaptive algorithms. It is noted that SGD did not converge actually due to its slow convergence rate though it could still catch up with the accuracy of AdaBelief and FastAdaBelief in the long run.

To sum up, the experiments of image classification with DenseNet-121, ResNet-34 and VGG-11 on CIFAR-10 validate the fast convergence rate and excellent accuracy performance of FastAdaBelief even when strong convexity does not hold in the loss functions.

2) Language Modeling: We also conduct a group of experiments on the language modeling task. In this group of experiments, we use a classic recurrent network (i.e., LSTM) and an open dataset (i.e., Penn Treebank). Same as the previous works [22], [23], [32], [33], we take the perplexity to measure the performance of all the comparison algorithms.

Note that the lower perplexity is better.

The perplexity curves of all the algorithms are shown in Figure 6. In the figure, once again we can see that FastAdaBelief performs similar to SAdam on 1-layer LSTM, and these two algorithms both perform better than the rest of the algorithms. However, on the 2,3-layer LSTM, FastAdaBelief performs better than SAdam and the other algorithms. Moreover, the perplexity of FastAdaBelief decreases the fastest among all the algorithms, which further validates the convergence analysis of FastAdaBelief.

We also summarize the test perplexities of all the algorithms performed in this group of experiments, which are shown in Table IV. From the table, we can see that FastAdaBelief attains superior performance to the other algorithms on all three models (i.e., 1,2,3-layer LSTM). In summary, FastAdaBelief also keeps both excellent generalization ability and fast convergence rate for language modeling tasks even when loss functions are not strongly convex. These empirical results are very encouraging, suggesting that FastAdaBelief has a high potential to be widely applied in real scenarios.

V. Conclusion and Future Work

In this paper, we made a first attempt and presented an affirmative answer to the question whether AdaBelief can be further improved on its convergence rate under the strongly convex condition. Specifically, we exploited strong convexity and proposed a new algorithm named FastAdaBelief, which exhibits an even faster data-dependent regret bound of $O(\log T)$ while maintaining excellent generalization ability. In light of our theoretical findings, we carried out a series of empirical studies which validated the superiority of our proposed algorithm. Importantly, we showed that FastAdaBelief converged the fastest in not only strong convexity, but also non-strong convexity, hence demonstrating its high potential as a new benchmark model that can be widely utilised in various scenarios.

In our current work, by exploiting its strong convexity, FastAdaBelief has empirically demonstrated excellent generalization as well as fast convergence even when the loss functions are not strongly convex. We believe that such phenomenon may be partially due to the vanishing factor δ/t as engaged in the second order moment that enables a closer approximation to an ideal step size. However, it remains unclear why this may happen strictly in theory. We will leave this investigation as future work. Besides, research on the sparsity of samples can improve the convergence rate of SGD, such as [34], [35], [36]. However, it remains unclear whether sparse samples will further improve FastAdaBelief's convergence rate. Hence, we will leave this investigation as another future work.

Acknowledgment

This work was partially supported by Chinese Academy of Sciences (No. Y9BEJ11001).

References

[1] Huang, K., Zhang, S., Zhang, R., and Hussain, A. "Pattern Field Classification Using Deep Neural Networks," *Neural Networks*, vol. 127, pp. 82-95, 2020.

[2] Jiang, Y., Wu, D., Deng, Z., etc. "Seizure Classification From EEG Signals Using Transfer Learning, Semi-Supervised Learning and TSK Fuzzy System," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 12, pp. 2270-2284, 2017 Dec.

[3] Wang, J., Hu, J., Min, G., Zomaya, A. Y., and Georgalas, N. "Fast Adaptive Task Offloading in Edge Computing Based on Meta Reinforcement Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 242-253, 2021 Jan.

[4] Jin, X., Zhang, X., Huang, K., Geng, G. "Stochastic Conjugate Gradient Algorithm with Variance Reduction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1360-1369, 2019.

[5] Jia, X., Li, Z., Zheng, X., Li, W., and Huang, S. J. "Label Distribution Learning with Label Correlations on Local Samples," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 4, pp. 1619-1631, 2021 April.

[6] Niu, S., Li, B., Wang, X., and Lin, H. "Defect Image Sample Generation With GAN for Improving Defect Recognition," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1611-1622, 2020 July.

[7] Khan, M., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y. "Fast and Scalable Bayesian Deep Learning by Weight-Perturbation in Adam," in *International Conference on Machine Learning*, 2018, vol. 80, pp. 2611-2620.

[8] Mukkamala, M. C. and Hein, M. "Variants of RMSProp and Adagrad with Logarithmic Regret Bounds," in International Conference on Machine Learning, 2017, vol. 70, pp. 2545-2553.

[9] Zhou, Y., Zhang, M., Zhu, J., Zheng, R., Wu, Q. "A Randomized Block-Coordinate Adam online learning optimization algorithm," *Neural Computing and Applications*, vol. 32, pp. 12671-1268, 2020 Aug.

[10] Zinkevich, M. "Online convex programming and generalized infinitesimal gradient ascent," in *International Conference on Machine Learning*, 2003, pp. 928-936.

[11] Lei, Y., Hu, T., Li, G., and Tang, K. "Stochastic Gradient Descent for Nonconvex Learning Without Bounded Gradient Assumptions," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 4394-4400, 2020 Oct.

[12] Gu, B., Shan, Y., Quan, X., and Zheng, G. "Accelerating Sequential Minimal Optimization via Stochastic Subgradient Descent," *IEEE Transactions on Cybernetics*, vol. 51, no. 4, pp. 2215-2223, 2021 April.

[13] Sutskever, I., Martens, J., Dahl, G., and Hinton, G. "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, 2013, pp. 1139-1147.

[14] Nesterov, Y. "A method of solving a convex programming problem with convergence rate $o(1/k^2)$," in *Soviet Mathematics Doklady*, 1983, vol. 27.

[15] Kingma, D.P. and Ba, J.L. "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[16] Reddi, S. J., Kale, S., and Kumar, S. "On the convergence of adam and beyond," in *International Conference on Learning Representations*, 2018.

[17] Luo, L., Xiong, Y., Liu, Y., Sun, X. "Adaptive gradient methods with dynamic bound of learning rate," in *International Conference on Learning Representations*, 2019.

[18] Zaheer, M., Reddi, S., Sachan, D., Kale, S., and Kumar, S. "Adaptive methods for nonconvex optimization," in *Neural Information Processing Systems*, 2018, pp. 9793-9803.

[19] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. "On the variance of the adaptive learning rate and beyond," in *International Conference on Learning Representations*, 2020.

[20] Balles, L. and Hennig, P. "Dissecting adam: The sign, magnitude and variance of stochastic gradients," in *International Conference on Machine Learning*, 2018.

[21] Loshchilov, I. and Hutter, F. "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[22] Zhuang, J., Tang, T., Ding, Y., Takiconda, S., Dvornek, N., Papademetris, X., and Duncan, J. "AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients," in *Neural Information Processing Systems*, 2020.
[23] Wang, G., Lu, S., Tu, W., and Zhang, L. "SAdam: A Variant of Adam for Strongly Convex Functions," in *International Conference on Learning Representations*, 2020.
[24] Shalev-Shwartz, S. "Online Learning and Online Convex Optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107-194, 2011.
[25] Boyd, S. and Vandenberghe, L. "Convex optimization," *Cambridge university press*, 2004.
[26] McMahan, B. H. and Streeter, M. "Adaptive bound optimization for online convex optimization," arXiv preprint arXiv:1002.4908, 2010.
[27] Song, Y., Li, M., Luo, X., Yang, G., and Wang, C. "Improved Symmetric and Nonnegative Matrix Factorization Models for Undirected, Sparse and Large-Scaled Networks: A Triple Factorization-Based Approach," in *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3006-3017, May 2020.
[28] Luo, X., Zhou, M., Li, S., Wu, D., Liu, Z., and Shang, M. "Algorithms of Unconstrained Non-Negative Latent Factor Analysis for Recommender Systems," in *IEEE Transactions on Big Data*, vol. 7, no. 1, pp. 227-240, 1 March 2021.
[29] Liu, W., Wen, Y., Yu, Z., and Yang, M. "Large-Margin Softmax Loss for Convolutional Neural Networks," in *Proceedings of The 33rd International Conference on Machine Learning*, PMLR, vol. 48, pp. 507-516, 2016.
[30] He, K., Zhang, X., Ren, S., and Sun, J. "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.
[31] Huang, G., Liu, Z., Maaten, L. V. D., and Weinberger, K., Q. "Densely connected convolutional networks," in *IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700-4708.
[32] Shuang, K., Li, R.,Gu, M.,Loo, J., and Su, S. "Major-Minor Long Short-Term Memory for Word-Level Language Model," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 3932-3946, Oct. 2020.
[33] Huang, S. and Renals, S. "Hierarchical Bayesian Language Models for Conversational Speech Recognition," in *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 1941-1954, Nov. 2010.
[34] Wu, D., Luo, X., Shang, M., et al. "A Deep Latent Factor Model for High-Dimensional and Sparse Matrices in Recommender Systems," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 7, pp. 4285-4296, July 2021.
[35] Luo, X., Liu, Z., Li, S., et al. "A Fast Non-Negative Latent Factor Model Based on Generalized Momentum Method," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 610-620, Jan. 2021.
[36] Luo, X., Wang, D., Zhou, M. and Yuan, H. "Latent Factor-Based Recommenders Relying on Extended Stochastic Gradient Descent Algorithms," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 2, pp. 916-926, Feb. 2021.

Appendix A
Convergence Analysis in Strongly Convex Online Optimization

Before presenting the proof of Theorem 1, we first review the following Lemma 1.

Lemma 1: [26] For all $M \in \mathcal{M}_+^n$ and convex feasible region $\mathcal{F} \in \mathbb{R}^n$, let

$$\mathbf{y}_1 = \min_{\mathbf{x}\in\mathcal{F}} \left\| M^{1/2}(\mathbf{x} - \mathbf{z}_1) \right\|,$$

and

$$\mathbf{y}_2 = \min_{\mathbf{x}\in\mathcal{F}} \left\| M^{1/2}(\mathbf{x} - \mathbf{z}_2) \right\|,$$

then we obtain the following:

$$\left\| M^{1/2}(\mathbf{y}_1 - \mathbf{y}_2) \right\| \leq \left\| M^{1/2}(\mathbf{z}_1 - \mathbf{z}_2) \right\|.$$

Theorem 1 *Suppose that Assumptions 1 and 2 are satisfied, Conditions 3 and 4 hold, and loss functions $f_t(\cdot)$ are σ-strongly convex. Moreover, let parameter sequences $\{\beta_{1t}\}, \{\beta_{2t}\}$ and $\{\alpha_t\}$ are generated by the proposed algorithm, where $\beta_{1t} = \beta_1\lambda^t, \beta_1 \in [0,1), \lambda_1 \in [0,1), \beta_{2t} \in [0,1), \delta > 0, t \in \{1,\ldots,T\}$. For decision point $\mathbf{x}_t$ generated by the proposed algorithm, we have the following upper bound of the regret*

$$\begin{aligned} R(T) \leq & \frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{D_\infty^2(G_\infty + \delta)}{2\alpha} \sum_{i=1}^{n}\sum_{t=1}^{T} \frac{\beta_{1t}}{1-\beta_{1t}} t \\ & + \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3} \sum_{i=1}^{n} \log\left(\frac{1}{\zeta\delta}\|g_{1:T,i}\|^2 + 1 \right). \end{aligned}$$

a) Proof*:* By the updating method of decision variable, i.e., Equation (7), we have:

$$\begin{aligned} \mathbf{x}_{t+1} &= \prod_{\mathcal{F},\hat{S}_t} \left(\mathbf{x}_t - \alpha_t \hat{S}_t^{-1}\mathbf{m}_t \right) \\ &= \min_{\mathbf{x}\in\mathcal{F}} \left\| \hat{S}_t^{1/2} \left[\mathbf{x} - (\mathbf{x}_t - \alpha_t \hat{S}_t^{-1}\mathbf{m}_t) \right] \right\|. \end{aligned} \tag{11}$$

From the definitions of $\mathbf{x}^*$ and projection $\prod(\cdot)$, we have that $\mathbf{x}^* = \prod_{\mathcal{F},\hat{S}_t}(\mathbf{x}^*) = \min_{\mathbf{x}\in\mathcal{F}}(\mathbf{x} - \mathbf{x}^*)$. In addition, if we apply Lemma 1, let $\mathbf{y}_1 = \mathbf{x}_{t+1}, \mathbf{y}_2 = \mathbf{x}^*$, by the update rules of $\mathbf{x}_t$ and $\mathbf{m}_t$, we obtain the following:

$$\begin{aligned} & \left\| \hat{S}_t^{1/2}(\mathbf{x}_{t+1} - \mathbf{x}^*) \right\|^2 \\ & \leq \left\| \hat{S}_t^{1/2} \left(\mathbf{x}_t - \alpha_t \hat{S}_t^{-1}\mathbf{m}_t - \mathbf{x}^* \right) \right\|^2 \\ & = \left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2 - \hat{S}_t \left\langle \mathbf{x}_t - \mathbf{x}^*, \alpha_t \hat{S}_t^{-1}\mathbf{m}_t \right\rangle \\ & + \left\| \alpha_t \hat{S}_t^{-1/2}\mathbf{m}_t \right\|^2 \\ & = \left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2 + \alpha_t^2 \left\| \hat{S}_t^{-1/2}\mathbf{m}_t \right\|^2 \\ & - 2\alpha_t \langle \mathbf{x}_t - \mathbf{x}^*, \mathbf{m}_t \rangle \\ & = \left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2 + \alpha_t^2 \left\| \hat{S}_t^{-1/2}\mathbf{m}_t \right\|^2 \\ & - 2\alpha_t \langle \mathbf{x}_t - \mathbf{x}^*, \beta_{1t}\mathbf{m}_{t-1} + (1-\beta_{1t})\mathbf{g}_t \rangle. \end{aligned} \tag{12}$$

Next, rearranging equation (12), we have that

$$\begin{aligned} & \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{x}^* \rangle \\ & \leq \frac{\left[\left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2 - \left\| \hat{S}_t^{1/2}(\mathbf{x}_{t+1} - \mathbf{x}^*) \right\|^2 \right]}{2\alpha_t(1-\beta_{1t})} \\ & + \frac{\alpha_t}{2(1-\beta_{1t})} \left\| \hat{S}_t^{-1/2}\mathbf{m}_t \right\|^2 \underbrace{- \frac{\beta_{1t}}{1-\beta_{1t}} \langle \mathbf{m}_{t-1}, \mathbf{x}_t - \mathbf{x}^* \rangle}_{(a)}. \end{aligned} \tag{13}$$

Applying Young's inequality (i.e., $\langle a, b\rangle \leq \frac{a^2\epsilon}{2} + \frac{b^2}{2\epsilon}, \forall \epsilon > 0$) into the term (a) of Equation (13), and considering $\alpha_t > 0, \beta_{1t} \in [0,1)$, we can attain

$$\begin{aligned} (a) &= -\frac{\beta_{1t}}{1-\beta_{1t}} \langle \mathbf{m}_{t-1}, \mathbf{x}_t - \mathbf{x}^* \rangle \\ &\leq \frac{\alpha_t\beta_{1t}}{2(1-\beta_{1t})} \left(\hat{S}_t^{-1/2}\mathbf{m}_{t-1} \right)^2 \\ &+ \frac{\beta_{1t}}{2\alpha_t(1-\beta_{1t})} \left(\hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right)^2. \end{aligned} \tag{14}$$

Furthermore, applying Cauchy-Schwartz inequality into Equation (14), we have

$$\begin{aligned} (a) &\leq \frac{\alpha_t\beta_{1t}}{2(1-\beta_{1t})} \left\| \hat{S}_t^{-1/2}\mathbf{m}_{t-1} \right\|^2 \\ &+ \frac{\beta_{1t}}{2\alpha_t(1-\beta_{1t})} \left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2. \end{aligned} \tag{15}$$

Then, plugging Equation (15) into Equation (13), we obtain the following

$$\begin{aligned} & \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{x}^* \rangle \\ & \leq \frac{\left[\left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2 - \left\| \hat{S}_t^{1/2}(\mathbf{x}_{t+1} - \mathbf{x}^*) \right\|^2 \right]}{2\alpha_t(1-\beta_{1t})} \\ & + \frac{\alpha_t}{2(1-\beta_{1t})} \left\| \hat{S}_t^{-1/2}\mathbf{m}_t \right\|^2 + \frac{\alpha_t\beta_{1t}}{2(1-\beta_{1t})} \left\| \hat{S}_t^{-1/2}\mathbf{m}_{t-1} \right\|^2 \\ & + \frac{\beta_{1t}}{2\alpha_t(1-\beta_{1t})} \left\| \hat{S}_t^{1/2}(\mathbf{x}_t - \mathbf{x}^*) \right\|^2. \end{aligned} \tag{16}$$

On the other hand, let $\mathbf{x} = \mathbf{x}^*, \mathbf{y} = \mathbf{x}_t$ in Equation (2). With the strong convexity of $f_t(\cdot)$, we attain

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \leq \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{x}^* \rangle - \frac{\sigma}{2}\|\mathbf{x}_t - \mathbf{x}^*\|^2. \tag{17}$$

Therefore, from definition of the regret (i.e., Equation (1)), and Equation (17), we obtain the following

$$\begin{aligned} R(T) &= \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{F}} \sum_{t=1}^{T} f_t(\mathbf{x}) \\ &= \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}^*) \\ &\leq \sum_{t=1}^{T} \langle \mathbf{g}_t, \mathbf{x}_t - \mathbf{x}^* \rangle - \frac{\sigma}{2} \sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{x}^*\|^2. \end{aligned} \tag{18}$$

In addition, plugging Equation (16) into Equation (18), we can attain the following

$$\begin{aligned}R(T) \leq & \sum_{t=1}^{T} \frac{\left[\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2-\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_{t+1}-\mathbf{x}^*\right)\right\|^2\right]}{2\alpha_t(1-\beta_{1t})} \\ & +\sum_{t=1}^{T} \frac{\alpha_t}{2(1-\beta_{1t})}\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2 \\ & +\sum_{t=1}^{T} \frac{\alpha_t\beta_{1t}}{2(1-\beta_{1t})}\left\|\hat{S}_t^{-1/2}\mathbf{m}_{t-1}\right\|^2 \\ & +\sum_{t=1}^{T} \frac{\beta_{1t}}{2\alpha_t(1-\beta_{1t})}\left\|\hat{S}_t^{1/2}(\mathbf{x}_t-\mathbf{x}^*)\right\|^2 \\ & -\frac{\sigma}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}^*\|^2. \end{aligned} \tag{19}$$

Furthermore, since $0 \leq s_{t-1,i} \leq s_{t,i}, 0 \leq \alpha_t \leq \alpha_{t-1}, 0 \leq \beta_{1t} \leq \beta_1 < 1$, by Equation (19), we have

$$\begin{aligned}R(T) \leq & \underbrace{\sum_{t=1}^{T} \frac{\left[\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2-\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_{t+1}-\mathbf{x}^*\right)\right\|^2\right]}{2\alpha_t(1-\beta_{1t})}}_{} \\ & \underbrace{-\frac{\sigma}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}^*\|^2}_{E_1} \\ & \underbrace{+\sum_{t=1}^{T} \frac{\alpha_t}{2(1-\beta_{1t})}\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2}_{} \\ & \underbrace{+\sum_{t=2}^{T} \frac{\beta_1\alpha_{t-1}}{2(1-\beta_1)}\left\|\hat{S}_{t-1}^{-1/2}\mathbf{m}_{t-1}\right\|^2}_{E_2} \\ & \underbrace{+\sum_{t=1}^{T} \frac{\beta_{1t}}{2\alpha_t(1-\beta_{1t})}\left\|\hat{S}_t^{1/2}(\mathbf{x}_t-\mathbf{x}^*)\right\|^2}_{E_3}. \end{aligned} \tag{20}$$

Next, we consider the upper bounds of three parts (E_1, E_2 and E_3) in Equation (20) respectively. For part E_1, we attain the following

$$\begin{aligned}E_1 = & \frac{1}{2\alpha_1(1-\beta_1)}\left\|\hat{S}_1^{1/2}\left(\mathbf{x}_1-\mathbf{x}^*\right)\right\|^2 \\ & +\sum_{t=2}^{T} \frac{1}{2\alpha_t(1-\beta_{1t})}\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2 \\ & -\sum_{t=2}^{T} \frac{1}{2\alpha_{t-1}(1-\beta_{1(t-1)})}\left\|\hat{S}_{t-1}^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2 \\ & -\frac{1}{2\alpha_T(1-\beta_{1T})}\left\|\hat{S}_T^{1/2}\left(\mathbf{x}_{T+1}-\mathbf{x}^*\right)\right\|^2 \\ & -\frac{\sigma}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}^*\|^2 \\ & \leq \sum_{t=2}^{T} \frac{1}{1-\beta_{1t}}\left[\frac{\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2}{2\alpha_t}\right. \\ & \left.-\frac{\left\|\hat{S}_{t-1}^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2}{2\alpha_{t-1}}\right] \\ & +\frac{1}{2\alpha_1(1-\beta_1)}\left\|\hat{S}_1^{1/2}\left(\mathbf{x}_1-\mathbf{x}^*\right)\right\|^2-\frac{\sigma}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}^*\|^2. \end{aligned} \tag{21}$$

Since $\alpha_t = \frac{\alpha}{t}$ and from Equation (21), we have the following

$$\begin{aligned}E_1 \leq & \sum_{t=2}^{T} \frac{1}{1-\beta_{1t}}\left[\frac{t\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2}{2\alpha}\right. \\ & \left.-\frac{(t-1)\left\|\hat{S}_{t-1}^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2}{2\alpha}\right] \\ & +\frac{1}{2\alpha_1(1-\beta_1)}\left\|\hat{S}_1^{1/2}\left(\mathbf{x}_1-\mathbf{x}^*\right)\right\|^2 \\ & -\frac{\sigma}{2}\sum_{t=2}^{T}\|\mathbf{x}_t-\mathbf{x}^*\|^2-\frac{\sigma}{2}\|\mathbf{x}_1-\mathbf{x}^*\|^2 \\ & =\sum_{t=2}^{T} \frac{1}{2\alpha(1-\beta_{1t})}\underbrace{\left[t\left\|\hat{S}_t^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2\right.}_{} \\ & \underbrace{\left.-(t-1)\left\|\hat{S}_{t-1}^{1/2}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|^2-\sigma\alpha(1-\beta_{1t})\|\mathbf{x}_t-\mathbf{x}^*\|^2\right]}_{E_1'} \\ & \underbrace{+\frac{1}{2\alpha_1(1-\beta_1)}\left\|\hat{S}_1^{1/2}\left(\mathbf{x}_1-\mathbf{x}^*\right)\right\|^2-\frac{\sigma}{2}\|\mathbf{x}_1-\mathbf{x}^*\|^2}_{E_1''}. \end{aligned} \tag{22}$$

In addition, for term E_1' of Equation (22), and from

Equation (6), we can obtain

$$E_1^{'} = \sum_{i=1}^{n}(x_{t,i} - x_{,i}^*)^2 \Bigg[t\left(\hat{s}_{t,i} + \frac{\delta}{t}\right)^{1/2} - (t-1)\left(\hat{s}_{t-1,i} + \frac{\delta}{t-1}\right)^{1/2} - \sigma\alpha(1-\beta_{1t})\Bigg], \quad (23)$$

where $i \in \{1, \ldots, n\}$, and n is the dimension of decision vectors. Moreover, since $(a+b)^{1/2} \leq a^{1/2}+b^{1/2}$ and $1-\beta_1 \leq 1-\beta_{1t}$, by Equation (9), we can have the following

$$\begin{aligned} E_1^{'} &\leq \sum_{i=1}^{n}(x_{t,i} - x_{,i}^*)^2 \Bigg[t\hat{s}_{t,i}^{1/2} - (t-1)\hat{s}_{t-1,i}^{1/2} - \sqrt{\delta(t-1)} - \sigma\alpha(1-\beta_{1t})\Bigg] \\ &\leq \sum_{i=1}^{n}(x_{t,i} - x_{,i}^*)^2 \Bigg[ts_{t,i}^{1/2} - (t-1)s_{t-1,i}^{1/2} - \sqrt{\delta(t-1)} - \sigma\alpha(1-\beta_{1t})\Bigg] \\ &\leq \sum_{i=1}^{n}(x_{t,i} - x_{,i}^*)^2 \Bigg[\sigma\alpha(1-\beta_1) - \sqrt{\delta(t-1)} - \sigma\alpha(1-\beta_{1t})\Bigg] \\ &\leq 0. \end{aligned} \quad (24)$$

Next, for term $E_1^{''}$ of Equation (22), we have

$$\begin{aligned} E_1^{''} &= \frac{1}{2\alpha_1(1-\beta_1)}\left\|\hat{S}_1^{1/2}(\mathbf{x}_1 - \mathbf{x}^*)\right\|^2 - \frac{\sigma}{2}\|\mathbf{x}_1 - \mathbf{x}^*\|^2 \\ &= \sum_{i=1}^{n}\left[\frac{s_1 + \delta - \sigma\alpha_1(1-\beta_1)}{2\alpha_1(1-\beta_1)}\right](x_{1,i} - x_{,i}^*)^2. \end{aligned} \quad (25)$$

By Equation (9), $s_1 - \sigma\alpha_1(1-\beta_1) \leq 0$ when $t=1$, and from Equation (25), we obtain the following

$$E_1^{''} \leq \sum_{i=1}^{n}\frac{\delta}{2\alpha_1(1-\beta_1)}(x_{1,i} - x_{,i}^*)^2. \quad (26)$$

Moreover, from Assumption 1, we have that $x_{1,i} - x_{,i}^* \leq D_\infty$. Then, combining Equations (22), (24) and (26), we finally attain

$$E_1 \leq \sum_{i=1}^{n}\frac{\delta D_\infty^2}{2\alpha_1(1-\beta_1)} = \frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)}. \quad (27)$$

Therefore, we have obtained the upper bound of part E_1. Then we consider part E_2 of the remaining two parts in Equation (20). Since $\beta_{1t} \leq \beta_1$, we have the following

$$E_2 \leq \sum_{t=1}^{T}\frac{\alpha_t}{2(1-\beta_1)}\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2 + \sum_{t=2}^{T}\frac{\beta_1\alpha_{t-1}}{2(1-\beta_1)}\left\|\hat{S}_{t-1}^{-1/2}\mathbf{m}_{t-1}\right\|^2. \quad (28)$$

For Equation (28), we first consider the term $\sum_{t=1}^{T}\alpha_t\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2$, we obtain

$$\begin{aligned} &\sum_{t=1}^{T}\alpha_t\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2 \\ &= \sum_{t=1}^{T-1}\alpha_t\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2 + \alpha_T\left\|\hat{S}_T^{-1/2}\mathbf{m}_T\right\|^2 \\ &\leq \sum_{t=1}^{T-1}\alpha_t\left\|\hat{S}_t^{-1/2}\mathbf{m}_t\right\|^2 + \underbrace{\alpha_T\sum_{i=1}^{n}\frac{m_{T,i}}{s_{T,i} + \frac{\delta}{T}}}_{E_2^{'}}. \end{aligned} \quad (29)$$

Moreover, assuming $\frac{g_{t,i} - m_{t,i}}{g_{t,i}} = \varpi_t$ where $\varpi_t \in (0,1)$ and $i \in \{1, \ldots, n\}$, and let $\varpi = \min\{\varpi_1, \ldots, \varpi_t\}$. Furthermore, applying the recursive algorithm to (3) and (4), we have that

$$\begin{aligned} E_2^{'} &= \alpha\sum_{i=1}^{n}\frac{\left(\sum_{j=1}^{T}(1-\beta_{1j})\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}\right)^2}{T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}(g_{j,i} - m_{j,i})^2 + \delta} \\ &\leq \alpha\sum_{i=1}^{n}\frac{\left(\sum_{j=1}^{T}(1-\beta_{1j})\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}\right)^2}{\varpi^2 T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta}. \end{aligned} \quad (30)$$

From Equation (30) and $\beta_{1t} \in (0,1]$, $E_2^{'}$ can be further bounded as

$$E_2^{'} \leq \alpha\sum_{i=1}^{n}\frac{\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}\right)^2}{\varpi^2 T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta}. \quad (31)$$

Furthermore, according to Cauchy-Schwarz inequality, i.e., $\left(\sum_{k=1}^{n}\langle a_k, b_k\rangle\right)^2 \leq \left(\sum_{k=1}^{n}a_k^2\right)\left(\sum_{k=1}^{n}b_k^2\right)$, and from Equation (31), we attain the following

$$\begin{aligned} E_2^{'} &\leq \\ &\alpha\sum_{i=1}^{n}\frac{\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}\right)\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}^2\right)}{\varpi^2 T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta} \\ &\leq \alpha\sum_{i=1}^{n}\frac{\left(\sum_{j=1}^{T}\beta_1^{T-j}\right)\left(\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}^2\right)}{\varpi^2 T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta}. \end{aligned} \quad (32)$$

Since $\beta_{1t} \leq \beta_1$ and from Equation (32), we can further attain the following bound for $E_2^{'}$:

$$\begin{aligned} E_2^{'} &\leq \frac{\alpha}{\varpi^2(1-\beta_1)}\sum_{i=1}^{n}\frac{\sum_{j=1}^{T}\prod_{k=1}^{T-j}\beta_{1(T-k+1)}g_{j,i}^2}{T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta} \\ &\leq \frac{\alpha}{\varpi^2(1-\beta_1)}\sum_{i=1}^{n}\frac{\sum_{j=1}^{T}\beta_1^{T-j}g_{j,i}^2}{T\sum_{j=1}^{T}(1-\beta_{2j})\prod_{k=1}^{T-j}\beta_{2(T-k+1)}g_{j,i}^2 + \delta}. \end{aligned} \quad (33)$$

By Equation (8), we have the following

$$E_2^{'} \leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \frac{\sum_{j=1}^{T} \beta_1^{T-j} g_{j,i}^2}{\sum_{j=1}^{T} g_{j,i}^2 + \zeta\delta} \leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \sum_{j=1}^{T} \beta_1^{T-j} \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta}. \quad (34)$$

Moreover, plugging Equation (34) into Equation (29), and applying recursive algorithm, we attain

$$\begin{aligned}
\sum_{t=1}^{T} \alpha_t \left\| \hat{S}_t^{-1/2} \mathbf{m}_t \right\|^2 &\leq \sum_{t=1}^{T-1} \alpha_t \left\| \hat{S}_t^{-1/2} \mathbf{m}_t \right\|^2 + \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \sum_{j=1}^{T} \beta_1^{T-j} \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{j,i}^2 + \zeta\delta} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \sum_{t=1}^{T} \sum_{j=1}^{t} \beta_1^{t-j} \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \sum_{j=1}^{T} \sum_{l=0}^{T-j} \beta_1^{l} \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)} \sum_{i=1}^{n} \sum_{j=1}^{T} \frac{\sum_{l=0}^{T-j} \beta_1^{l} g_{j,i}^2}{\sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^2} \sum_{i=1}^{n} \sum_{j=1}^{T} \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta}. \quad (35)
\end{aligned}$$

If let $\Psi = \frac{g_{j,i}^2}{\sum_{k=1}^{j} g_{j,i}^2 + \zeta\delta}$, $\omega_j = \sum_{k=1}^{j} g_{k,i}^2 + \zeta\delta$, and $\omega_0 = \zeta\delta$, we obtain

$$\Psi = \frac{\omega_j - \omega_{j-1}}{\omega_j}. \quad (36)$$

In addition, for any $a \geq b > 0$, the inequality $1 + x \leq e^x$ implies that

$$\frac{a-b}{a} \leq \log \frac{a}{b}. \quad (37)$$

Therefore, by Equation (37), Equation (36) has the following bound:

$$\Psi \leq \log \frac{\omega_j}{\omega_{j-1}}. \quad (38)$$

Plugging Equation (38) into Equation (35), we have

$$\begin{aligned}
&\sum_{t=1}^{T} \alpha_t \left\| \hat{S}_t^{-1/2} \mathbf{m}_t \right\|^2 \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^2} \sum_{i=1}^{n} \sum_{j=1}^{T} \log \frac{\omega_j}{\omega_{j-1}} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^2} \sum_{i=1}^{n} \log \frac{\omega_T}{\omega_0} \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^2} \sum_{i=1}^{n} \log \left(\frac{\sum_{k=1}^{T} g_{k,i}^2}{\zeta\delta} + 1 \right). \quad (39)
\end{aligned}$$

From Equations (28) and (39), E_2 can be further bounded as

$$\begin{aligned}
E_2 &\leq \frac{1}{1-\beta_1} \sum_{t=1}^{T} \alpha_t \left\| \hat{S}_t^{-1/2} \mathbf{m}_t \right\|^2 \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3} \sum_{i=1}^{n} \log \left(\frac{\sum_{k=1}^{T} g_{k,i}^2}{\zeta\delta} + 1 \right) \\
&\leq \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3} \sum_{i=1}^{n} \log \left(\frac{1}{\zeta\delta} \|g_{1:T,i}\|^2 + 1 \right). \quad (40)
\end{aligned}$$

Next, we consider the last term E_3 in Equation (20). From the definition of $\hat{S}_t$ and Assumption 2, we obtain the following

$$\begin{aligned}
E_3 &\leq \sum_{t=1}^{T} \frac{t\beta_{1t}}{2\alpha(1-\beta_{1t})} \left\| \left(\mathbf{s}_t + \frac{\delta}{t} \right)^{1/2} (\mathbf{x}_t - \mathbf{x}^*) \right\|^2 \\
&\leq \sum_{i=1}^{n} \sum_{t=1}^{T} \frac{t\beta_{1t}}{2\alpha(1-\beta_{1t})} \left\| \left(s_{t,i} + \frac{\delta}{t} \right)^{1/2} (x_{t,i} - x_{,i}^*) \right\|^2 \\
&\leq \frac{D_\infty^2 (G_\infty + \delta)}{2\alpha} \sum_{i=1}^{n} \sum_{t=1}^{T} \frac{\beta_{1t}}{1-\beta_{1t}} t. \quad (41)
\end{aligned}$$

Finally, combining Equations (20), (27), (40) and (41), we obtain the upper bound of $R(T)$ as follows

$$\begin{aligned}
R(T) \leq &\frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{D_\infty^2 (G_\infty + \delta)}{2\alpha} \sum_{i=1}^{n} \sum_{t=1}^{T} \frac{\beta_{1t}}{1-\beta_{1t}} t \\
&+ \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3} \sum_{i=1}^{n} \log \left(\frac{1}{\zeta\delta} \|g_{1:T,i}\|^2 + 1 \right). \quad (42)
\end{aligned}$$

Therefore, the proof of Theorem 1 is completed. ■

Corollary 1 Let $\beta_{1t} = \beta_1 \lambda^t$, where $\lambda \in (0,1)$ in Theorem 1. Then we have the following upper bound of the regret

$$\begin{aligned}
R(T) \leq &\frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{n\beta_1\lambda D_\infty^2 (G_\infty + \delta)}{2\alpha(1-\beta_1)(1-\lambda)^2} \\
&+ \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3} \sum_{i=1}^{n} \log \left(\frac{1}{\zeta\delta} \|g_{1:T,i}\|^2 + 1 \right).
\end{aligned}$$

b) Proof.*:* Since $\beta_{1t} = \beta_1 \lambda^t$, Equation (41) can be further bounded as follows

$$\begin{aligned}
E_3 &\leq \frac{\beta_1 D_\infty^2 (G_\infty + \delta)}{2\alpha(1-\beta_1)} \sum_{i=1}^{n} \sum_{t=1}^{T} t\lambda^t \\
&\leq \frac{\beta_1 D_\infty^2 (G_\infty + \delta)}{2\alpha(1-\beta_1)} \sum_{i=1}^{n} \left[\frac{\lambda(1-\lambda^T)}{(1-\lambda)^2} - \frac{T\lambda^{T+1}}{1-\lambda} \right] \\
&\leq \frac{\beta_1 D_\infty^2 (G_\infty + \delta)}{2\alpha(1-\beta_1)} \sum_{i=1}^{n} \frac{\lambda}{(1-\lambda)^2} \\
&= \frac{n\beta_1\lambda D_\infty^2 (G_\infty + \delta)}{2\alpha(1-\beta_1)(1-\lambda)^2}. \quad (43)
\end{aligned}$$

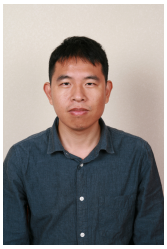
In addition, plugging Equation (43) into Equation (42), we further attain the upper bound of $R(T)$ as follows

$$R(T) \leq \frac{n\delta D_\infty^2}{2\alpha(1-\beta_1)} + \frac{n\beta_1\lambda D_\infty^2(G_\infty+\delta)}{2\alpha(1-\beta_1)(1-\lambda)^2} + \frac{\alpha\zeta}{\varpi^2(1-\beta_1)^3}\sum_{i=1}^{n}\log\left(\frac{1}{\zeta\delta}\|g_{1:T,i}\|^2+1\right). \quad (44)$$

Therefore, the proof of Corollary 1 is completed. ■

Yangfan Zhou is currently pursuing the Ph.D. degree in the School of Nano-Tech and Nano-Bionics, University of Science and Technology of China. His current research interests are focused theoretical and algorithmic issues related to on large-scale optimization, stochastic optimization, convex online optimization, and their applications to deep learning, meta learning, and networking. He is currently a reviewer for many well-known journals, such as the IEEE Transactions on Neural Networks and Learning Systems.

Kaizhu Huang is currently a Professor at Department of Intelligent Science, Xi'an Jiaotong-Liverpool University (XJTLU), China. He acts as associate dean of research in School of Advanced Technology, XJTLU and is also the founding director of Suzhou Municipal Key Laboratory of Cognitive Computation and Applied Technology. Prof. Huang obtained his PhD degree from Chinese University of Hong Kong (CUHK) in 2004. He worked in Fujitsu Research Centre, CUHK, University of Bristol, National Laboratory of Pattern Recognition, Chinese Academy of Sciences from 2004 to 2012. Prof. Huang has been working in pattern recognition, machine learning, and neural information processing. He was the recipient of 2011 Asia Pacific Neural Network Society Young Researcher Award. He received best paper or book award six times. Until October 2020, he has published 9 books and over 200 international research papers (80+ international journals) e.g., in journals (JMLR, Neural Computation, IEEE T-PAMI, IEEE T-NNLS, IEEE T-BME, IEEE T-Cybernetics) and conferences (NeurIPS, IJCAI, SIGIR, UAI, CIKM, ICDM, ICML, ECML, CVPR). He serves as associated editors/advisory board members in a number of journals and book series. He was invited as keynote speaker in more than 30 international conferences or workshops.



Cheng Cheng is currently an associate professor. He received the B.S. degree and M.S. degree in Computer Science and Technology from Guizhou University, Guiyang, China, in 2004 and 2009, respectively, and the Ph.D. degree in Information Engineering from Tokyo University of Agriculture and Technology (TUAT), Japan, in 2013. His current research interests focus on 3D vision, particularly on 3D feature learning, 3D modeling, 3D object recognition, and 3D shape measurement, etc.

Xuguang Wang received the Ph.D. degree from the Department of Electronic Engineering, University of Texas, Austin, USA, with a master's degree from the Department of Electronic Engineering, Rice University, USA, and a bachelor's degree from the Department of Materials, Tsinghua University, Beijing. Dr. Wang has been engaged in the research of semiconductor storage technology for 10 years, and has undertaken the research of semiconductor memory in many scientific research institutions such as National Natural Science Foundation of the United States, MARCO, SRC, etc. Dr. Wang has published many important papers as the first author in the top two journals and conferences of the international semiconductor device category, all of which belong to SCI and have been cited more than 100 times.

Amir Hussain received his B.Eng (highest 1st Class Honours with distinction) and Ph.D degrees, from the University of Strathclyde, Glasgow, U.K., in 1992 and 1997, respectively. He is a member of the member of the UK Computing Research Committee (UKCRC) - the Expert Panel of the Institution of Engineering and Technology (IET) and the BCS, The Chartered Institute for IT, for computing research in the UK. He has been invited Advisor/Consultant for various international Governments and organisations, including at: Kuwait Institute for Scientific Research (KISR), Kuwait Government; and the National Centre of Big Data & Cloud Computing (NCBC), Higher Education Commission, Pakistan Government. He acts as a Consultant for various global companies and is co-founder/Advisor for a number of successful spin-out/start-up companies, including SenticNet, Smart Big Data Solutions Ltd. and AiGenics. He has been appointed invited Associate Editor/Editorial Board member for a number of prestigious journals, including: the IEEE Transactions on Artificial Intelligence (AI), the IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Systems, Man, and Cybernetics: Systems, Information Fusion, AI Review, IEEE Computational Intelligence Magazine, and the IEEE Transactions on Emerging Topics in Computational Intelligence.

Xin Liu received the Ph.D. degree from the Department of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, in 2007. Dr. Liu worked as a research scientist, principal researcher and director of intelligent Computing Chips at the Singapore Institute of Microelectronics, Singapore Technology Development Board, from 2007 to 2018. Dr. Liu joined the Suzhou Institute of Nano-Tech and Nano-Bionics (SINANO), Chinese Academy of Sciences in 2018. His research interests include artificial intelligence, wireless communications, signal processing algorithms and chip design, high-performance large-scale parallel processing chip architecture design, ultra-low power digital processor design, embedded non-volatile memory circuit design, and so on. As the person in charge, he has presided over and completed more than 10 large and national semiconductor chip research and industrial projects, with a total research fund of about 200 million RMB. He has more than 10 international invention patents authorized in relevant technical fields, published more than 60 articles in international core journals and conferences, and been cited nearly a thousand times.