Adaptive Random Walk Gradient Descent for Decentralized Optimization

Tao Sun [1] Dongsheng Li [1] Bao Wang [2]

Abstract

In this paper, we study the adaptive step size random walk gradient descent with momentum for decentralized optimization, in which the training samples are drawn dependently with each other. We establish theoretical convergence rates of the adaptive step size random walk gradient descent with momentum for both convex and nonconvex settings. In particular, we prove that adaptive random walk algorithms perform as well as the non-adaptive method for dependent data in general cases but achieve acceleration when the stochastic gradients are “sparse”. Moreover, we study the zeroth-order version of adaptive random walk gradient descent and provide corresponding convergence results. All assumptions used in this paper are mild and general, making our results applicable to many machine learning problems.

1. Introduction

Given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{1, 2, \cdots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ being the set of agents and the set of edges that connect agents, respectively. We consider the following decentralized minimization problem over the graph $\mathcal{G}$

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}),\ f_i(\boldsymbol{x}) = \mathbb{E}_{\xi\sim\mathcal{D}_i} F_i(\boldsymbol{x};\xi), \tag{1}$$

where $\mathcal{D}_i$ denotes the data distribution of the i-th client and $F_i(\boldsymbol{x};\xi)$ is the loss function associated with the training data ξ. In decentralized setting, the raw data $\mathcal{D}_i$ is only accessible to the i-th client and clients can share their locally updated $\boldsymbol{x}$ with other clients through the edges, a.k.a. communication channels. The problem (1) models many crucial application problems, including distributed adaptation, distributed training, and aircraft coordination (Sayed, 2014; Duchi et al., 2011b; Inalhan et al., 2002). Many interesting decentralized algorithms have been proposed to solve the problem in (1), and these algorithms can be classified into three categories:

- (i) **Local computing with all clients + communication.** These algorithms let all agents perform local computation and share the updated parameters with all or a subset of the neighboring agents, where the subset of neighbors can be selected either in a deterministic or a stochastic manner. Many decentralized algorithms belong to this category, including both deterministic, see e.g., (Nedic & Ozdaglar, 2009; Chen & Ozdaglar, 2012; Jakovetić et al., 2014; Matei & Baras, 2011; Yuan et al., 2016; Chang et al., 2015; Schizas et al., 2008; Shi et al., 2014; 2015; Zeng & Yin, 2018; Hosseini et al., 2016; McMahan & Streeter, 2014) and stochastic algorithms, see, e.g., (Ran et al., 2020; Xin et al., 2020; Sirb & Ye, 2016; Lan et al., 2017; Lian et al., 2017; 2018; Lu & De Sa, 2021).
- (ii) **Local computing with randomly selected clients + communication.** Algorithms in category (i) suffer from high computation and communication costs. To alleviate this problem, a class of algorithms activates randomly selected communication channels (edges) for bidirectional communication, and only the clients connected by these edges are involved in local computing, see e.g., (Boyd et al., 2005; Ram et al., 2010a; Srivastava & Nedic, 2011; Ram et al., 2010b; Hendrikx et al., 2019).
- (iii) **Random walk.** Another class of popular algorithms for solving (1) is using random walk gradient for decentralized optimization, see e.g., (Bertsekas, 1997; Ram et al., 2009; Johansson et al., 2010; Lopes & Sayed, 2007; Yin et al., 2018; Mao et al., 2020; Shah & Avrachenkov, 2018). These algorithms only involve one edge communication in each iteration, resulting in a minimum communication cost.

This paper focuses on studying the random walk gradient descent, a special kind of random walk, in which a variable $\boldsymbol{x}$ is employed and moves through a (random) succession of agents on the graph as follows: the agent receives $\boldsymbol{x}$ and updates it by using the gradient of f_i at $\boldsymbol{x}$; after the update,

*Equal contribution [1]College of Computer, National University of Defense Technology, Hunan, China. [2]Department of Mathematics and Scientific Computing and Imaging Institute, University of Utah. Correspondence to: Dongsheng Li <dsli@nudt.edu.cn>, Bao Wang < wangbaonj@gmail.com>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

$\boldsymbol{x}$ will be sent to a randomly selected neighbor. It has been known that the random walk over an undirected graph directly introduces a Markov chain [Chapter 11, (Levin & Peres, 2017)]. Thus, the theory of random walk gradient descent has been well-studied by Agarwal & Duchi (2012); Duchi et al. (2012); Johansson et al. (2007; 2010); Ram et al. (2009); Sun et al. (2018; 2020b). Moreover, it is well-known that the random walk gradient descent can converge as fast as SGD if the mixing time is not very long. As an efficient and celebrated optimization algorithm for large-scale machine learning, the adaptive gradient SGD — i.e., adaptive step size determined by the historical information — has been popular for solving many machine learning tasks (Bartlett et al., 2007; Duchi et al., 2011a; McMahan & Streeter, 2010; Li & Orabona, 2019; Ward et al., 2019; Kingma & Ba, 2015). Nevertheless, the adaptive random walk gradient descent remains unstudied. In particular, it is natural to ask:

Does adaptive random walk gradient descent output the desired minimizer? If it does, how fast? Can the adaptive algorithm be faster than the non-adaptive one?

We answer the above questions affirmatively and derive convergence rates for adaptive online learning for dependent data under mild assumptions.

1.1. Notation

Throughout this paper, we use bold face letters to denote vectors, e.g., $\boldsymbol{x} \in \mathbb{R}^d$, and we use $\mathbb{I} \in \mathbb{R}^d$ to denote the vector whose entries are all 1s. The j-th coordinate of a vector $\boldsymbol{x}$ is denoted by x_j. If $t > 0$ and $x_j \geq 0$, we define $(\boldsymbol{x})^t \in \mathbb{R}^d$ (or $[\boldsymbol{x}]^t \in \mathbb{R}^d$) in a coordinate-wise fashion. For another vector $\boldsymbol{y} \in \mathbb{R}^d$, $\boldsymbol{y}/\boldsymbol{x} \in \mathbb{R}^d$ is again defined coordinate-wisely as $(\boldsymbol{y}/\boldsymbol{x})_j := y_j/x_j$. If all elements of $\boldsymbol{y}$ are nonnegative, then we define $\|\boldsymbol{x}\|_{\boldsymbol{y}} := \sqrt{\sum_{j=1}^d y_j x_j^2}$. We denote $\mathbb{E}[\cdot]$ as the expectation with respect to the underlying probability measure. we use $\|\boldsymbol{x}\|_1$ and $\|\boldsymbol{x}\|$ to denote the L_1- and L_2-norm of $\boldsymbol{x}$, respectively. We denote the minimum value of the function f as $\min f$. We denote the sub-algebra as $\chi^k := \sigma(\xi^0, \xi^1, \ldots, \xi^k)$ with ξ^k being the data received in the k-th iteration. For two positive constants a, b, $a = \mathcal{O}(b)$ means that there exists $C > 0$ such that $a \leq Cb$. The notation $a = \Theta(b)$ means that $a = \mathcal{O}(b)$ and $b = \mathcal{O}(a)$. We use $a = \tilde{\mathcal{O}}(b)$ and $a = \tilde{\Theta}(b)$ to hide the logarithmic factor of b but still with the same order.

1.2. Adaptive random walk gradient descent

We assume that the random walk is $\{i_k\}_{k\geq 0}$, where $i_k \in \mathcal{V}$. We formulate the adaptive random walk gradient descent in Algorithm 1. Hyperparameter $\delta > 0$ is used for numerical stability, which can be set as a small number, e.g., 10^{-8}. In Algorithm 1, $\eta\mathbb{I}/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}$ is the coordinate-wise learning rate. Thus, Algorithm 1 can use much larger η than the previous SGD or random walk gradient descent whose η is selected in the same order of the desired error ϵ. The projection is used to guarantee the sequence to be bounded in the convex case since our convex analysis below requires the function values to be Lipschitz. We note that the projection is also used in previous works on random walk, see e.g., (Johansson et al., 2010; 2007; Ram et al., 2009; Duchi et al., 2012; Sun et al., 2018). If $\{i_k\}_{k\geq 0}$ reduces to i.i.d. samples and $\mathcal{K}$ is the full space, then Algorithm 1 reduces to the AdaFom (Li & Orabona, 2019) with $\theta = 0$. Here, we use $\theta \geq 0$, i.e., with momentum. Furthermore, if we denote $\hat{\boldsymbol{v}}^k := (\sum_{i=1}^k [\boldsymbol{g}^i]^2)/k = \boldsymbol{v}^k/k$, **step 3** and **step 4** can be reformulated as follows

$$\begin{cases} \textbf{step 3} \leftarrow \hat{\boldsymbol{v}}^k = (1-\frac{1}{k})\hat{\boldsymbol{v}}^{k-1} + \frac{1}{k}\hat{\boldsymbol{v}}^k, \\ \textbf{step 4} \leftarrow \boldsymbol{z}^{k+1} = \boldsymbol{x}^k - \frac{\eta}{\sqrt{k}}\boldsymbol{m}^k/(\hat{\boldsymbol{v}}^k + \frac{\delta}{k}\mathbb{I})^{\frac{1}{2}}. \end{cases}$$

Therefore, if $\{i_k\}_{k\geq 0}$ reduces to the i.i.d. data, Algorithm 1 is a modification of Adam (Kingma & Ba, 2015) that uses varying hyperparameters for the second moment $\boldsymbol{v}^k$. Compared with the vanilla random walk gradient descent, Algorithm 1 requires agents to send and receive extra information $(\boldsymbol{m}^k, \boldsymbol{v}^k)$ in each iteration.

Algorithm 1 Adaptive Random Walk Gradient Descent

```
Require: parameters $\eta > 0$, $0 \leq \theta < 1$, $\delta > 0$
  Initialization: $\boldsymbol{g}^0 = \mathbf{0}$, $\boldsymbol{m}^0 = \mathbf{0}$, $\boldsymbol{v}^0 = \mathbf{0}$
  for $k = 1, 2, \ldots$
    step 1: agent $i_k$ calculates $\boldsymbol{g}^k = \nabla f_{i_k}(\boldsymbol{x}^k)$
    step 2: $\boldsymbol{m}^k = \theta\boldsymbol{m}^{k-1} + (1-\theta)\boldsymbol{g}^k$
    step 3: $\boldsymbol{v}^k = \boldsymbol{v}^{k-1} + [\boldsymbol{g}^k]^2$
    step 4: $\boldsymbol{z}^{k+1} = \boldsymbol{x}^k - \eta\boldsymbol{m}^k/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}$
    step 5: $\boldsymbol{x}^{k+1} = \arg\min_{\boldsymbol{x}\in\mathcal{K}} \|\boldsymbol{z}^{k+1} - \boldsymbol{x}\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}$
    step 6: uses random walk to choose a neighbor $i_{k+1}$
      and sends $(\boldsymbol{x}^k, \boldsymbol{m}^k, \boldsymbol{v}^k)$ via edge $(i_k, i_{k+1})$ to $i_{k+1}$
  end for
```

1.3. Comparison with existing theoretical works

Previous closely related theoretical works can be classified into two categories: random walk gradient descent (Johansson et al., 2010; 2007; Ram et al., 2009; Duchi et al., 2012; Sun et al., 2018; 2020b) and adaptive SGD (Bartlett et al., 2007; Duchi et al., 2011a; McMahan & Streeter, 2010; Li & Orabona, 2019; Ward et al., 2019; Kingma & Ba, 2015; Ward et al., 2019; Reddi et al., 2018; Chen et al., 2019). The adaptive random walk gradient descent studied in this paper can be considered as an integration of random walk gradient descent and adaptive step size used in the adaptive SGD. We present a comparison of our work against the existing works in Table 1 to distinguish the novelty of this paper.

The theoretical analysis of adaptive random walk gradient descent is different from any existing framework. In

References	C	NC	ASZ	Moment	**RW**
(Johansson et al., 2010; 2007; Ram et al., 2009; Duchi et al., 2012)	√	×	×	×	√
(Sun et al., 2018)	√	√	×	×	√
(Bartlett et al., 2007)	√	×	√	×	×
(Duchi et al., 2011a; McMahan & Streeter, 2010)	√	×	√	×	×
(Li & Orabona, 2019)	√	√	√	×	×
(Ward et al., 2019)	×	√	√	×	×
(Kingma & Ba, 2015)	√	×	√	×	×
(Reddi et al., 2018; Chen et al., 2019)	×	√	√	×	×
This paper	√	√	√	√	√

Table 1. Comparisons with previous closely related works under different settings. "C" stands for "Convexity", "NC" stands for "NonConvexity", "ASZ" is short for " Adaptive Step siZe", "Moment" means the "Momentum", and "RW" is "Random Walk".

particular, the traditional analysis of adaptive stochastic algorithms requires the variance of the stochastic gradient to be bounded, but this assumption usually does not hold for random walk. Moreover, the bias cannot be treated as noise in the random walk scenario; otherwise, the algorithm will fail to enjoy theoretical convergence.

The dependent samples make the analysis of random walk much more challenging than the analysis of SGD. Previous works (Johansson et al., 2010; 2007; Ram et al., 2009; Duchi et al., 2012; Sun et al., 2018; 2020b) on random walk gradient descent leverage the delayed expectation technique, i.e., by investigating the conditional expectation $\mathbb{E}[\nabla f_{i_k}(\boldsymbol{x}^{k-\tau}) \mid \chi^{k-\tau}]$, which is sufficiently close to $\nabla f(\boldsymbol{x}^{k-\tau})$ if τ is larger than the mixing time. Another important technique is to bound the term $\|\nabla f(\boldsymbol{x}^{k-\tau}) - \nabla f(\boldsymbol{x}^{k})\|$ by the successive difference sum $L\sum_{t=1}^{\tau}\|\boldsymbol{x}^{k-t+1} - \boldsymbol{x}^{k-t}\|$ using the Lipschitz property of ∇f. These techniques pose more challenges for random walk than SGD. We note that Johansson et al. (2010; 2007); Ram et al. (2009); Duchi et al. (2012) only consider random walk gradient descent in the convex setting; nonconvex analysis is first established in (Sun et al., 2018).

Agarwal & Duchi (2012) present a framework for analyzing online learning for dependent data (without adaptive stepsize). However, we cannot use the results presented in (Agarwal & Duchi, 2012) for three reasons: 1) The step size in adaptive online learning is not summable. Theoretically, the k-th step size is greater than $1/\sqrt{k}$. 2) The convexity is not satisfied. Our problem considers the case that the objective may be nonconvex. 3) We use the adaptive step size determined by the historical information rather than preset or constants.

Existing works on adaptive SGD (Bartlett et al., 2007; Duchi et al., 2011a; McMahan & Streeter, 2010; Li & Orabona, 2019; Ward et al., 2019; Kingma & Ba, 2015; Ward et al., 2019; Reddi et al., 2018; Chen et al., 2019) consider the unbiased stochastic gradient. This work is the *first* one that studies the performance of adaptive random walk gradient descent. We also leverage the delayed expectation technique, which indicates that our proof needs to deal with the term $L\sum_{t=1}^{\tau}\|\boldsymbol{x}^{k-t+1} - \boldsymbol{x}^{k-t}\|^2$. The adaptive SGD is quite complicated, let alone with extra time-varying items. To this end, a novel line of analysis is developed in this paper. The high-level idea for overcoming these difficulties is summarized as follows: In the convex case, we need to modify the Lyapunov function $\{\xi_k\}_{k\geq 1}$ used in the adaptive random walk as $\{\xi_k + \sum_{t=0}^{\tau} A_{k-t}\}$, where A_k is a composition of $\|\boldsymbol{g}^0\|, \|\boldsymbol{g}^1\|, \ldots, \|\boldsymbol{g}^k\|$. In this way, we can derive that $\mathbb{E}\sum_{t=0}^{\tau} A_{k+1-t} - \mathbb{E}\sum_{t=0}^{\tau} A_{k-t} \geq \sum_{t=1}^{\tau}\mathbb{E}\|\boldsymbol{x}^{k-t+1} - \boldsymbol{x}^{k-t}\|^2$. The nonconvex case is even more complicated, and we need to design another Lyapunov function.

1.4. More related works

Random walk gradient descent. Bertsekas (1997) uses a cyclic way to access the data with a special weighting, which is indeed a random walk on a ring graph, for least squares problems. Lopes & Sayed (2007) apply random walks to the adaptive networks to address the problem of linear estimation in a cooperative fashion. Johansson et al. (2010; 2007) study the performance of a more general case of random walk gradient descent, i.e., SGD with Markov chain samplings. In a later work, Ram et al. (2009) alleviate the Markov chain's time-homogeneous assumption but present stronger results. Duchi et al. (2012) prove that random walk gradient descent can be as fast as SGD even with time non-homogeneous Markov chains. In the works mentioned above, the Markov chain is required to be reversible, and the functions are assumed to be convex. The non-reversible Markov chains online algorithms are studied in (Sun et al., 2018). An interesting result is that Sun et al. (2018) prove the convexity can be removed for an expected minimization problem. The variance reduction random walk gradient descent is further studied in (Sun et al., 2020b). The generalization bound is proved in (Agarwal & Duchi, 2012) for general SGD for dependent data, and Lei et al. (2015; 2019) present the data-dependent generalization bounds for multi-class classification. In (Mao et al., 2020), the ADMM combined with random walk is proposed for solving convex decentralized optimization problems, in which the authors prove that ADMM with random walk can cost less communications than many other decentralized algorithms to reach the same desired error in some cases.

Adaptive stochastic algorithms. The first adaptive online gradient descent is developed in (Bartlett et al., 2007), and their regret rates are established under the strong convexity assumption. The adaptive stochastic gradient (AdaGrad) is proposed by Duchi et al. (2011a); McMahan & Streeter (2010), whose convergence is proved under the convex assumption. AdaGrad can achieve faster convergence than SGD when the gradients are sparse. The convergence of AdaGrad, in terms of gradient norm, for nonconvex problems is proved by Li & Orabona (2019). A sharp analysis of AdaGrad is given in (Ward et al., 2019). In (Zou et al., 2018), the authors present the convergence of a unified variant for AdaGrad. Momentum has also been integrated into the adaptive gradient algorithms to accelerate their convergence and results in Adam and NAdam (Kingma & Ba, 2015; Dozat, 2016). The nonergodic convergence result of the gradient's norm for adaptive gradient descent is presented in (Li & Orabona, 2019). The authors of (Reddi et al., 2018) provide a convex stochastic optimization problem for which Adam fails to converge to the optimal solution. To rectify the possible divergence of Adam, a maximum way modification to the weights is proposed in (Reddi et al., 2018). In (Chen et al., 2019), a theoretical framework for analyzing general adaptive stochastic algorithms is established. In (Zou et al., 2019), the authors provide easy-to-check sufficient conditions for the convergence of adaptive SGD, helping to use the algorithms in applications. The non-ergodic convergence results of the adaptive SGD is proved in (Li & Orabona, 2019; Sun et al., 2020a). The quantized Adam is proposed by (Chen et al., 2021), which significantly improves the communication efficiency.

1.5. Our contributions

In this paper, we consider the adaptive random walk gradient descent — a decentralized method — for solving the finite-sum minimization problem in Equation (1). We summarize our contributions below.

- In convex cases, we prove the convergence of adaptive random walk gradient descent in function values, showing adaptive random walk gradient descent is as fast as the counterpart algorithm without using adaptive step size.
- In nonconvex cases, we establish the convergence rate, in the gradient norm, for adaptive random walk gradient descent.
- We further study the zeroth-order adaptive random walk gradient descent; again we prove its convergence in both convex and nonconvex settings.
- We show that when the stochastic gradients are sparse, the adaptive step size schemes considered in this paper all enjoy theoretical acceleration guarantees.

2. Assumptions

In this section, we list several necessary and commonly used assumptions for the subsequent analysis.

In the random walk gradient descent, i_k calculates local gradient and selects a neighbor $i_{k+1} \in \mathcal{N}(i_k)$ randomly. Then i_k sends the gradient information to i_{k+1} via edge (i_k, i_{k+1}). We assume that the random walk satisfies the following assumption.

Assumption 1: *Let $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ be the transition probability matrix of the irreducible and aperiodic Markov chain induced by the random walk, where $\boldsymbol{P}_{i,j} = \mathbb{P}(i_{k+1} = j \mid i_k = i)$ and the stationary distribution is $\pi^* := \mathbb{I}^\top/n$.*

For all $i \in \{1, 2, \ldots, n\}$, the mixing time of the Markov chain is defined as follows

$$\tau_{\text{mix}}(\varepsilon) := \arg\min_t\{t \mid \|[\boldsymbol{P}^t]_{i,:} - \pi^*\| \leq \varepsilon\}. \qquad (2)$$

Mixing time is an important notion for Markov process, which tells us how long a stochastic process evolves until its state has a distribution that is close to its stationary distribution. Several kinds of mixing times have been well studied, see, e.g., (Montenegro & Tetali, 2006). For random walk gradient descent, Assumption 1 has been widely used in (Johansson et al., 2010; 2007; Ram et al., 2009; Duchi et al., 2012; Sun et al., 2018; 2020b). In [Footnote 1, (Mao et al., 2020)], the authors show that

$$\tau_{\text{mix}}(\varepsilon) = \Theta\Big(\frac{\ln(1/\epsilon)}{\ln(1/\sigma(\boldsymbol{P}))}\Big),^{1}$$

where $0 \leq \sigma(\boldsymbol{P}) := \max\{\|\boldsymbol{P}^\top \boldsymbol{y}\|/\|\boldsymbol{y}\| \mid \boldsymbol{y}^\top \mathbb{I} = 0, \boldsymbol{y} \in \mathbb{R}^n\} < 1$ [2]. We can see that $\sigma(\boldsymbol{P})$ represents the speed of the Markov chain converges to the stationary state: the smaller $\sigma(\boldsymbol{P})$ is, the faster the Markov chain converges. Note that $\sigma(\boldsymbol{P}) = 0$ corresponds the complete graph and i.i.d. sampling case, i.e., $\mathbb{P}(i_{k+1} = j \mid i_k = i) = 1/n$. Assumption 1 can also be satisfied for many other kinds of Markov processes. For instance, Ram et al. (2009) prove that the time non-homogeneous Markov process with several extra assumptions [Assumptions 4 and 5 in Section 4 of (Ram et al., 2009)] also satisfies Assumption 1. Moreover,

[1] Compared with (Mao et al., 2020), we do not use the Taylor series version. Instead, we use the convention $\frac{1}{0} = +\infty$.

[2] If the transition matrix $\boldsymbol{P}$ is real and symmetric, it holds that $\sigma(\boldsymbol{P}) = \lambda_2(\boldsymbol{P}) := \max\{|\lambda_i(\boldsymbol{P})| \mid \lambda_i(\boldsymbol{P}) \neq 1\}$.

Assumption 1 also holds for the finite-state non-reversible Markov chains, according to [Lemma 1, (Sun et al., 2018)].

Assumption 2: *The function $f_i(\boldsymbol{x})$ is convex in the full space with respect to $\boldsymbol{x}$, where $i \in \{1, 2, \ldots, n\}$. The set $\mathcal{K}$ is convex and compact, which is bounded by a constant $R > 0$, i.e., $\max_{\boldsymbol{x}\in\mathcal{K}} \|\boldsymbol{x}\| \leq R$.*

This paper assumes the constrained set is bounded in the convex case. It is worth mentioning that the constrained set could be unbounded for stochastic methods in the convex case (McMahan & Abernethy, 2013; McMahan & Orabona, 2014; Orabona et al., 2015; Joulani et al., 2020); we leave how to get rid of the boundedness assumption as future work. Notice that the subgradient is bounded if the constrained set is bounded [Theorem 10.4, (Rockafellar, 1970)]. Thus, according to Assumption 2, it holds that $\max_{i\in\{1,2,\ldots,n\},\boldsymbol{x}\in\mathcal{K}}\{\|\nabla f_i(\boldsymbol{x})\|\} \leq L$, where L is a positive constant. The Lipschitz continuity of the function value also follows directly from Assumption 2. In particular, if Assumptions 2 holds, we have

$$|f_i(\boldsymbol{x}) - f_i(\boldsymbol{y})| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|, \forall i \in \{1, \ldots, n\}. \tag{3}$$

If f_i is nonconvex, several contraction properties fail to hold. In this case, we use the full space assumption to simplify the algorithm and analysis. In particular, we use the following Assumption 2' as a surrogate of Assumption 2.

Assumption 2': *The set $\mathcal{K}$ is the full space and $\max_{i\in\{1,\ldots,n\}}\{\|\nabla f(\boldsymbol{x})\|\} \leq L$.*

The continuity of $f_i(\boldsymbol{x})$ with respect to $\boldsymbol{x}$ indicates that $f_i(\boldsymbol{x})$ enjoys the finite maximum over a constrained set. We formulate this in Assumption 3 below.

Assumption 3: *It holds that $\hat{R} := \max_{\boldsymbol{x}\in\mathcal{K}, i\in\{1,\ldots,n\}}\{|f_i(\boldsymbol{x})|\} < +\infty$.*

In the nonconvex case, we do not need the Lipschitz property (3) because the convergence results for the nonconvex case are described by the gradient norm rather than the function value. Thus, nonconvex proofs mainly deal with the (stochastic) gradients instead of the (stochastic) function values, which are controlled by (3) in the convex case. Alternatively, we need something else to bound the gradients. Thus, the last assumption is about the differentiability and gradient Lipschitz property.

Assumption 4: *$f_i(\cdot)$ is differentiable with Lipschitz gradient, i.e., for $i \in \{1, \ldots, n\}$, we have*

$$\|\nabla f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{y})\| \leq L_H\|\boldsymbol{x} - \boldsymbol{y}\|,\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d. \tag{4}$$

3. Main Theory

3.1. Convex case

In this part, we consider function $f_i(\boldsymbol{x})$ to be both convex and smooth, and we show that Algorithm 1 converges as fast as the random walk gradient descent. Furthermore, if the historical gradients are "sparse", we show that Algorithm 1 enjoys a faster rate than the non-adaptive scheme. Notice that $\boldsymbol{v}^K$ collects the historic gradients up to the K-th iteration, thus, the sparsity is mathematically described as

$$\mathbb{E}\|(\boldsymbol{v}^K + \delta\mathbb{I})^{\frac{1}{2}}\|_1 \leq CK^\alpha, \tag{5}$$

where $C > 0$ is a constant and $0 < \alpha < 1/2$. Due to the boundedness of stochastic gradients $\{\boldsymbol{g}^k\}_{k\geq 0}$, $\alpha = 1/2$ can hold in (5) without any extra assumption. Condition (5) is very standard for analyzing adaptive stochastic optimization (Liao et al., 2021; Duchi et al., 2011a; Reddi et al., 2018; Chen et al., 2018; 2019; Liu et al., 2019).

Theorem 3.1. *Let Assumptions 1, 2, 3, 4, and condition (5) hold. Assume $\{\boldsymbol{x}^k\}_{k\geq 1}$ is generated by Algorithm 1. By setting $\eta = \min\{\frac{\ln(1/\sigma(\boldsymbol{P}))}{\ln(1/\epsilon)}, 1\}$, then*

$$\mathbb{E}\left[f\Big(\frac{\sum_{k=1}^K \boldsymbol{x}^k}{K}\Big) - \min f\right] = \mathcal{O}(\epsilon), \tag{6}$$

with $K = \widetilde{\mathcal{O}}\Big(\max\Big\{\frac{1}{\epsilon^{\frac{1}{1-\alpha}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{1}{1-\alpha}}}, \frac{1}{\epsilon^{\frac{1}{1-\alpha}}}\Big\}\Big)$.

From Theorem 3.1, η is set as $\tilde{\mathcal{O}}(1)$ in Algorithm 1, which is much larger than $\tilde{\mathcal{O}}(\epsilon)$ that is used in SGD or random walk. In the adaptive random walk, the historic gradients affect the convergence rate. We see that the speed of the Markov chain converges to the stationary state has an impact on the speed of the adaptive random walk: the faster the Markov chain is, i.e., a smaller $\sigma(\boldsymbol{P})$, the faster the adaptive random walk gradient descent converges. This phenomenon is similar to the convergence of the Markov gradient descent proved by Ram et al. (2009); Duchi et al. (2012); Sun et al. (2018). In the general case ($\alpha = 1/2$), if $\sigma(\boldsymbol{P})$ is not too closed to 1, the adaptive random walk gradient descent converges almost as fast as random walk and SGD ($\tilde{\mathcal{O}}(1/\epsilon^2)$); while when stochastic gradients are "sparse", adaptive random walk can achieve a better iteration complexity, given by

$$\tilde{\mathcal{O}}\Big(\frac{1}{\epsilon^{\frac{1}{1-\alpha}}}\Big),\ \ 0 < \alpha < \frac{1}{2}.$$

When $\sigma(\boldsymbol{P}) = 0$ (i.e., the i.i.d case), the convergence result of Theorem 3.1 matches the existing rates of adaptive SGD.

3.2. Nonconvex case

In this part, we consider nonconvex function $f_i(\boldsymbol{x})$, and we assume $\mathcal{K}$ is the full space. Then, the **step 5** of Algorithm 1 reduces to $\boldsymbol{x}^{k+1} = \boldsymbol{z}^{k+1}$, and the update becomes

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \eta \boldsymbol{m}^k / (\boldsymbol{v}^k + \delta \mathbb{I})^{\frac{1}{2}}.$$

In the nonconvex case, we consider the following condition

$$\|\boldsymbol{v}^k\|_1 \leq C \cdot k^{\nu}, \text{ where constants } C > 0, 0 < \nu \leq 1. \quad (7)$$

We can see that $\nu = 1$ holds for the bounded stochastic gradients; while $\nu < 1$ means the stochastic gradients are "sparse". Compared with (5), the interval of ν changes since (7) gets rid of the square root.

Theorem 3.2. *Let Assumptions 1, 2', 4, and condition* (7) *hold. Assume* $\{\boldsymbol{x}^k\}_{k\geq 1}$ *is generated by Algorithm 1. By setting* $\eta = \min\{\frac{\ln(1/\sigma(\boldsymbol{P}))}{\ln(1/\epsilon)}, 1\}$, *we have*

$$\min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\} = \mathcal{O}(\epsilon) \quad (8)$$

with $K = \widetilde{\mathcal{O}}\Big(\max\Big\{\frac{1}{\epsilon^{\frac{2}{2-\nu}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{2}{2-\nu}}}, \frac{1}{\epsilon^{\frac{2}{2-\nu}}}\Big\}\Big)$.

Theorem 3.2 shows that the convergence rate of nonconvex adaptive random walk gradient descent is almost as fast as the nonconvex random walk even without sparse gradients assumption. If $\sigma(\boldsymbol{P}) = 0$, i.e., i.i.d. sampling, we obtain the same convergence rate for adaptive SGD as the existing results (Duchi et al., 2011a; Chen et al., 2018) ($\nu = 1$ can be easily satisfied for the bounded stochastic gradients). Similar to the convex case, when $0 < \nu < 1$, adaptive random walk gradient descent enjoys a faster rate than the non-adaptive scheme.

4. Extension to Zeroth-Order Oracle

In this section, we consider the zeroth-order adaptive random walk gradient descent; the gradient may not be available in this case. We assume the function $f_i(\boldsymbol{x})$ is differentiable. The first-order information is usually obtained by using a two-points feedback strategy (Duchi et al., 2015; Ghadimi & Lan, 2013; Agarwal et al., 2010; Shamir, 2017), and we employ the method given in (Duchi et al., 2015; Ghadimi & Lan, 2013). In particular, we use the estimator of the gradient of a smooth function G by querying at $\boldsymbol{x} + r\boldsymbol{h}$ and $\boldsymbol{x}$ with returning $\frac{d(G(\boldsymbol{x}+r\boldsymbol{h})-G(\boldsymbol{x}))}{r}\boldsymbol{h}$, where $\boldsymbol{h}$ is a random unit vector, and $r > 0$ is a small parameter, and d is the number of the dimension. When ∇G is uniformly bounded, [Theorem 3.1, (Ghadimi & Lan, 2013)] gives

$$\left\|\mathbb{E}_{\boldsymbol{h}}\left[\frac{d(G(\boldsymbol{x}+r\boldsymbol{h})-G(\boldsymbol{x}))}{r}\boldsymbol{h}\right] - \nabla G(\boldsymbol{x})\right\| \leq \frac{rL(d+3)^{\frac{3}{2}}}{2}.$$

Note that $d + 3 \leq 4d$, $\frac{rL(d+3)^{\frac{3}{2}}}{2} \leq 4Lrd^{\frac{3}{2}}$, which implies that

$$\left\|\mathbb{E}_{\boldsymbol{h}}\left(\frac{d(G(\boldsymbol{x}+r\boldsymbol{h})-G(\boldsymbol{x}))}{r}\boldsymbol{h}\right) - \nabla G(\boldsymbol{x})\right\| = \mathcal{O}(r\cdot d^{\frac{3}{2}}).$$

In the zeroth-order version, we use the following estimate of $\nabla f_{i_k}(\boldsymbol{x})$,

$$\boldsymbol{g}^k = \frac{d(f_{i_k}(\boldsymbol{x}+r\boldsymbol{h}) - f_{i_k}(\boldsymbol{x}))}{r}\boldsymbol{h}. \quad (9)$$

We summarize the zeroth-order adaptive gradient online learning algorithm in Algorithm 2. By the mean value theorem, it holds that $\|\boldsymbol{g}^k\| \leq d \cdot \hat{R}$. Thus, we consider replacing C in (5) and (7) with Cd^{α} and Cd^{ν}, respectively.

Algorithm 2 Zeroth-Order Adaptive Gradient Online Learning

```
Require: parameters $\eta > 0$, $0 \leq \theta < 1$, $\epsilon > 0$, $\tau > 0$, $d > 0$
  Initialization: $\boldsymbol{g}^0 = \boldsymbol{0}$, $\boldsymbol{m}^0 = \boldsymbol{0}$, $\boldsymbol{v}^0 = \boldsymbol{0}$
  for $k = 1, 2, \ldots$
    step 1: agent $i_k$ calculates $\boldsymbol{g}^k$ by (9)
    step 2∼6: same as Algorithm 1
  end for
```

The convergence of Algorithm 2 with convex and nonconvex settings is presented in the following proposition.

Proposition 4.1. *Assume* $\{\boldsymbol{x}^k\}_{k\geq 1}$ *is generated by Algorithm 2 with* $\eta = \min\Big\{\frac{\ln(1/\sigma(\boldsymbol{P}))}{\ln(1/\epsilon)}, 1\Big\}$.

- *(Convex) Let Assumptions 1, 2, 3, 4, and condition* (5) *hold with* $0 < \alpha \leq \frac{1}{2}$ *and* $C = \mathcal{O}(d^{\alpha})$. *To reach* ϵ *error as* (6), *we need to set* $r = \mathcal{O}(\epsilon/d^{\frac{3}{2}})$ *and* $K = \tilde{\mathcal{O}}\Big(\max\Big\{\frac{d^{\frac{\alpha}{1-\alpha}}}{\epsilon^{\frac{1}{1-\alpha}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{1}{1-\alpha}}}, \frac{1}{\epsilon^{\frac{1}{1-\alpha}}}\Big\}\Big)$ *in the worst case.*
- *(Nonconvex) Let Assumptions 1, 2', 4, and condition* (7) *hold with* $0 < \nu \leq 1$ *and* $C = \mathcal{O}(d^{\nu})$. *To reach* ϵ *error as* (8), *we need to set* $r = \mathcal{O}(\epsilon^{\frac{2}{2-\nu}}/d^{\frac{3}{2}})$ *and* $K = \widetilde{\mathcal{O}}\Big(\max\Big\{\frac{d^{\frac{2\nu}{2-\nu}}}{\epsilon^{\frac{2}{2-\nu}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{2}{2-\nu}}}, \frac{1}{\epsilon^{\frac{2}{2-\nu}}}\Big\}\Big)$ *in the worst case.*

Proposition 4.1 shows that in zeroth versions, large dimension d deteriorates the rate of adaptive algorithms. When r is small enough, Algorithm 2 can also enjoy the same convergence rate as the fist-order version, which is unsurprising due to (9). When $r \to 0$, it follows $\lim_{r\to 0}\mathbb{E}(\frac{d(f_{i_k}(\boldsymbol{x}+r\boldsymbol{h})-f_{i_k}(\boldsymbol{x}))}{r}\boldsymbol{h}\big|\boldsymbol{h}) = \nabla f_{i_k}(\boldsymbol{x})$. Compared to the convex case, the nonconvex case requires a much smaller r.

5. Experimental Results

We contrast the performance of adaptive and non-adaptive random walk algorithms for training machine learning models, including logistic regression (LR), multi-layer perceptron (MLP), and convolutional neural networks (CNNs). We evaluate the performance of the models on the benchmark MNIST and CIFAR10 image classification tasks, where MNIST/CIFAR10 contains 60K/50K and 10K/10K images for training and test, respectively. For decentralized optimization, we consider two classical undirected graphs that connect all clients, namely, the ring and 3-regularly expander graphs (Hoory et al., 2006), as illustrated in Figure 1. Again, each node represents a client, and each edge represents a potential communication channel.

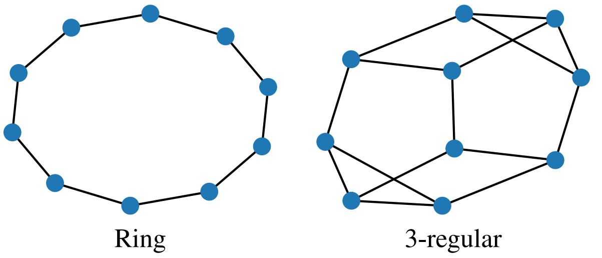


Figure 1. An illustration of the ring and 3-regular expander graphs with 10 nodes.

We aim to validate our theoretical results numerically. In particular, when the gradients are sparse, adaptive random walk gradient descent converges faster than the non-adaptive counterpart. We simulate the sparse gradient in training machine learning models in a decentralized fashion by using the following randomized scheme: Given a probability vector $\boldsymbol{p} \in \mathbb{R}^d$ ($p_i > 0$ for $i \in \{1, 2, \cdots, d\}$), let $\boldsymbol{z} \in \mathbb{R}^d$ be a binary-valued random vector, in which $\mathbb{P}(z_i = 1) = p_i$ and $\mathbb{P}(z_i = 0) = 1 - p_i$. Then, we define the sparsification operation $T_{\boldsymbol{p}}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ as follows

$$[T_{\boldsymbol{p}}(\boldsymbol{x})]_i := \frac{x_i z_i}{p_i}, \quad \text{for } i = 1, 2, \cdots, d.$$

It is evident that $\mathbb{E}[T_{\boldsymbol{p}}(\boldsymbol{x})] = \boldsymbol{x}$, making the sparsified gradient an unbiased estimate of the exact gradient. Additionally, the expected sparsity of $T_{\boldsymbol{p}}(\boldsymbol{x})$ is $\|\boldsymbol{p}\|_1$, which makes it easy to control the sparsity rate. We denote $\|\boldsymbol{p}\|_1/d$ as p.

5.1. Training LR for MNIST classification

We consider decentralized training of the logistic regression model for MNIST classification in this subsection. We randomly partition the training data into ten even groups in an i.i.d. fashion. Each client keeps one group of training data without sharing it with other clients. In training, we set the batch size to be 128. We fine-tune the step size for both adaptive and non-adaptive random walk gradient descent, and we use the initial learning rate of 0.003 and 0.1 for adaptive and non-adaptive algorithms, respectively [3]. The momentum hyperparameter is set to 0.9 for both solvers. Moreover, we set the weight decay for both adaptive and non-adaptive algorithms to be 5×10^{-4}.

Figures 2 (clients are connected by a ring graph) and 3 (clients are connected by a 3-regular expander graph) plot the training and test loss of training logistic regression model for MNIST classification using adaptive and non-adaptive gradient descent, respectively. In these experiments, we consider gradient with different sparsity levels, which is controlled by the parameter p, i.e., we set all components of the vector $\boldsymbol{s}$ to be p. We see that when $p = 1$ (non-sparse gradient) adaptive algorithm achieves the smaller loss in both training and test. As the sparsity of gradients increases (p decreases), the performance of the non-adaptive algorithm becomes worse and worse. However, the adaptive algorithm performs quite consistently under different sparsity levels. These results confirm that when the gradient is sparse, the adaptive algorithm is much faster than the non-adaptive counterpart. Another interesting result is that when the gradient is very sparse, e.g., when $p = 0.1$ or 0.2, both training and test loss of non-adaptive algorithm increases after a certain number of iterations, indicating that non-adaptive algorithm can be unstable for stochastic decentralized optimization with very sparse gradient.

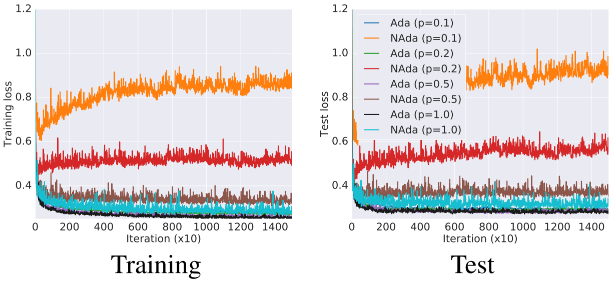


Figure 2. Contrasting adaptive (Ada) and non-adaptive (NAda) random walk algorithms for decentralized optimization with both sparse ($p < 1$) and non-sparse gradients ($p = 1$). There are ten clients connected by a ring graph, and they are training a logistic regression model for MNIST classification.

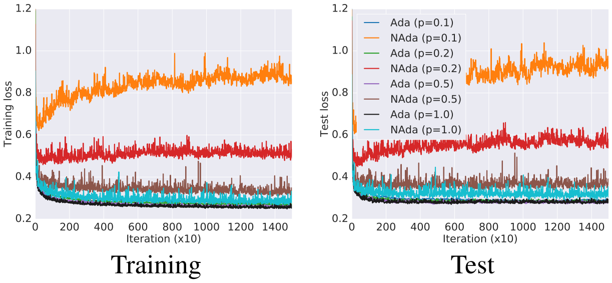
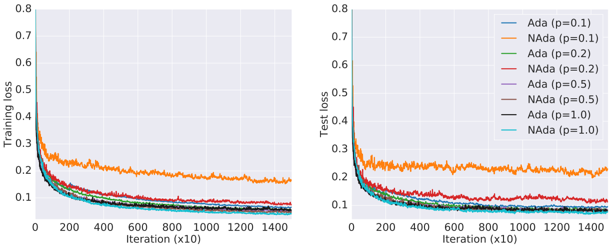


Figure 3. Contrasting adaptive and non-adaptive random walk gradient descent, for training logistic regression model for MNIST classification, with both sparse ($p < 1$) and non-sparse gradients ($p = 1$). There are ten clients connected by a 3-regular expander graph.

[3] These step sizes are consistent with the widely used ones for Adam and SGD, respectively. Indeed, we tune the step sizes leveraging Adam and SGD experiences.

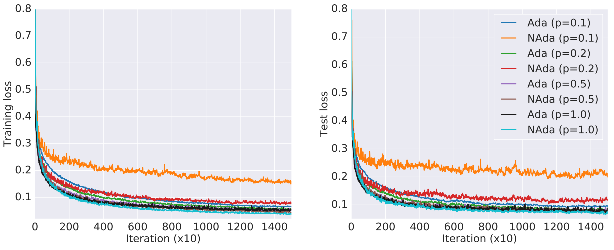
5.2. Training MLP for MNIST classification

In this part, we consider decentralized training of a simple MLP model for MNIST classification. The model contains two hidden layers with 200 units, each using ReLU activation (199,210 parameters). Again, we consider i.i.d. partitioning of the data into ten clients and using the same setting as Section 5.1 for both adaptive and non-adaptive stochastic gradient algorithms. Figures 4 and 5 show the training and test loss of both solvers when the clients are connected by the ring and 3-regular expander graphs, respectively. These results indicate that both adaptive and non-adaptive algorithms converge with different gradient sparsity. However, when the gradient is sparse, say $p = 0.1$ or 0.2, the non-adaptive solver converges much slower in both training and test loss no matter the clients are connected by a ring or expander graph. In contrast, the adaptive solver performs consistently well under different gradient sparsity rates.



Training Test

Figure 4. Comparison of adaptive and non-adaptive random walk algorithms, with both sparse ($p < 1$) and non-sparse gradients ($p = 1$), for training an MLP model for MNIST classification. In this experiment, we have ten clients connected by a ring graph.

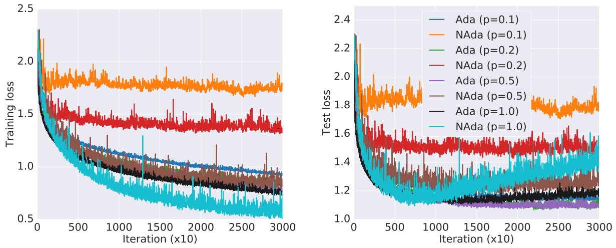


Training Test

Figure 5. Comparison of adaptive and non-adaptive random walk algorithms, with both sparse ($p < 1$) and non-sparse gradients ($p = 1$), for training an MLP for MNIST classification. Here we consider ten clients connected by a 3-regular expander graph.

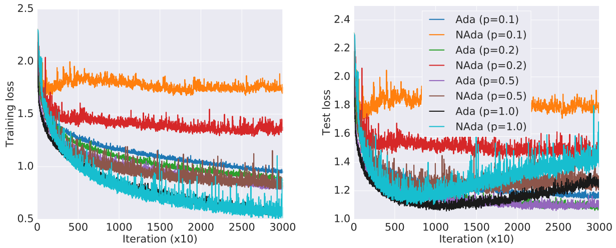
5.3. Training CNN for CIFAR10 classification

We further use the adaptive and non-adaptive random walk to train a small convolutional neural network for CIFAR10 classification. We use the same network architecture, experimental setting, data augmentation, and i.i.d. data partition over clients as used in the paper (McMahan et al., 2017). Figures 6 and 7 depict the training and test loss of different solvers at different gradient sparsity rates when the ring and 3-regular expander graphs are used to connect clients. These results resonate with the results in Sections 5.1 and 5.2 and further confirm that adaptive random walk is significantly faster than the non-adaptive counterpart when the stochastic gradient is sparse; the gain becomes more significant when the gradient becomes more sparse. Another interesting observation is that both adaptive and non-adaptive random walks can easily overfit when the gradient is not sparse. Using a sparse gradient can overcome overfitting. We leave the study of the generalization of the adaptive random walk as future work.



Training Test

Figure 6. Contrasting adaptive and non-adaptive random walk gradient descent for training a CNN model for CIFAR10 classification in a decentralized fashion. In this experiment, we consider the setting where ten clients are connected by a ring graph and consider both sparse ($p < 1$) and non-sparse gradients ($p = 1$).



Training Test

Figure 7. Contrasting adaptive and non-adaptive random walk gradient descent for training a CNN for CIFAR10 classification in a decentralized fashion. In this experiment, we consider the setting where ten clients are connected by a 3-regular expander graph and study both sparse ($p < 1$) and non-sparse gradients ($p = 1$).

6. Concluding Remarks

In this paper, we investigate the adaptive random walk gradient descent and establish its theoretical performance bounds in both convex and nonconvex settings. Our results reveal that the convergence speed of adaptive random walk gradient descent outperforms the one without adaptive step sizes for sparse gradients. We also propose the zeroth-order surrogate algorithms with performance guarantees when the gradient is unavailable. There are numerous avenues for future work: 1) How to relax the constrained set to the unbounded scenario in the convex case? 2) Can we establish the lower bound for adaptive random walk gradient descent? And 3) Can we integrate the idea of federated learning (McMahan et al., 2017), i.e., using multiple local iterations before communication, with the adaptive random walk?

Acknowledgements

This work is sponsored in part by National Key R&D Program of China (2021YFB0301200), Hunan Provincial Natural Science Foundation of China (2022JJ10065), and the National Science Foundation of China (62025208 and 61906200).

References

Agarwal, A. and Duchi, J. C. The generalization ability of online algorithms for dependent data. *IEEE Transactions on Information Theory*, 59(1):573–587, 2012.

Agarwal, A., Dekel, O., and Xiao, L. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *The 23rd Conference on Learning Theory*, pp. 28–40, 2010.

Bartlett, P. L., Hazan, E., and Rakhlin, A. Adaptive online gradient descent. *Proceedings of the 20th International Conference on Neural Information Processing Systems*, pp. 65–72, 2007.

Bertsekas, D. P. A new class of incremental gradient methods for least squares problems. *SIAM Journal on Optimization*, 7(4):913–926, 1997.

Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. Gossip algorithms: Design, analysis and applications. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pp. 1653–1664. IEEE, 2005.

Chang, T.-H., Hong, M., and Wang, X. Multi-agent distributed optimization via inexact consensus admm. *IEEE Trans. Signal Processing*, 63(2):482–497, 2015.

Chen, A. I. and Ozdaglar, A. A fast distributed proximal-gradient method. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pp. 601–608. IEEE, 2012.

Chen, C., Shen, L., Huang, H., and Liu, W. Quantized Adam with error feedback. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 12(5):1–26, 2021.

Chen, X., Liu, S., Sun, R., and Hong, M. On the convergence of a class of Adam-type algorithms for non-convex optimization. In *ICLR (Poster)*, 2019. URL https://openreview.net/forum?id=H1x-x309tm.

Chen, Z., Yuan, Z., Yi, J., Zhou, B., Chen, E., and Yang, T. Universal stagewise learning for non-convex problems with convergence on averaged solutions. In *International Conference on Learning Representations*, 2018.

Dozat, T. Incorporating nesterov momentum into Adam. In *ICLR Workshop*, 2016. URL https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011a.

Duchi, J. C., Agarwal, A., and Wainwright, M. J. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2011b.

Duchi, J. C., Agarwal, A., Johansson, M., and Jordan, M. I. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012.

Duchi, J. C., Jordan, M. I., Wainwright, M. J., and Wibisono, A. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Hendrikx, H., Bach, F., and Massoulie, L. Accelerated decentralized optimization with local updates for smooth and strongly convex objectives. In Chaudhuri, K. and Sugiyama, M. (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 897–906. PMLR, 16–18 Apr 2019.

Hoory, S., Linial, N., and Wigderson, A. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

Hosseini, S., Chapman, A., and Mesbahi, M. Online distributed convex optimization on dynamic networks. *IEEE Trans. Automat. Contr.*, 61(11):3545–3550, 2016.

Inalhan, G., Stipanovic, D. M., and Tomlin, C. J. Decentralized optimization, with application to multiple aircraft coordination. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, volume 1, pp. 1147–1155. IEEE, 2002.

Jakovetić, D., Xavier, J., and Moura, J. M. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.

Johansson, B., Rabi, M., and Johansson, M. A simple peer-to-peer algorithm for distributed optimization in sensor networks. In *46th Conference on Decision and Control*, pp. 4705–4710. IEEE, 2007.

Johansson, B., Rabi, M., and Johansson, M. A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization*, 20(3):1157–1170, 2010.

Joulani, P., György, A., and Szepesvári, C. A modular analysis of adaptive (non-) convex optimization: Optimism, composite objectives, variance reduction, and variational bounds. *Theoretical Computer Science*, 808:108–138, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

Lan, G., Lee, S., and Zhou, Y. Communication-efficient algorithms for decentralized and stochastic optimization. *arXiv preprint arXiv:1701.03961*, 2017.

Lei, Y., Dogan, U., Binder, A., and Kloft, M. Multi-class SVMs: From tighter data-dependent generalization bounds to novel algorithms. In *Advances in Neural Information Processing Systems*, pp. 2035–2043, 2015.

Lei, Y., Dogan, Ü., Zhou, D.-X., and Kloft, M. Data-dependent generalization bounds for multi-class classification. *IEEE Transactions on Information Theory*, 65(5): 2995–3021, 2019.

Levin, D. A. and Peres, Y. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

Li, X. and Orabona, F. On the convergence of stochastic gradient descent with adaptive stepsizes. *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 983–992, 2019.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 5330–5340, 2017.

Lian, X., Zhang, W., Zhang, C., and Liu, J. Asynchronous decentralized parallel stochastic gradient descent. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3043–3052, 2018.

Liao, L., Shen, L., Duan, J., Kolar, M., and Tao, D. Local adagrad-type algorithm for stochastic convex-concave minimax problems. *arXiv preprint arXiv:2106.10022*, 2021.

Liu, M., Mroueh, Y., Ross, J., Zhang, W., Cui, X., Das, P., and Yang, T. Towards better understanding of adaptive gradient algorithms in generative adversarial nets. In *International Conference on Learning Representations*, 2019.

Lopes, C. G. and Sayed, A. H. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing*, 55(8):4064–4077, 2007.

Lu, Y. and De Sa, C. Optimal complexity in decentralized training. In *International Conference on Machine Learning*, pp. 7111–7123. PMLR, 2021.

Mao, X., Yuan, K., Hu, Y., Gu, Y., Sayed, A. H., and Yin, W. Walkman: A communication-efficient random-walk algorithm for decentralized optimization. *IEEE Transactions on Signal Processing*, 68:2513–2528, 2020.

Matei, I. and Baras, J. S. Performance evaluation of the consensus-based distributed subgradient method under random communication topologies. *IEEE Journal of Selected Topics in Signal Processing*, 5(4):754–771, 2011.

McMahan, B. and Streeter, M. Delay-tolerant algorithms for asynchronous distributed online learning. In *Advances in Neural Information Processing Systems*, pp. 2915–2923, 2014.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

McMahan, H. B. and Abernethy, J. Minimax optimal algorithms for unconstrained linear optimization. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pp. 2724–2732, 2013.

McMahan, H. B. and Orabona, F. Unconstrained online linear learning in hilbert spaces: Minimax algorithms and normal approximations. In *Conference on Learning Theory*, pp. 1020–1039. PMLR, 2014.

McMahan, H. B. and Streeter, M. Adaptive bound optimization for online convex optimization. In *The 23rd Conference on Learning Theory*, pp. 244–256, 2010.

Montenegro, R. R. and Tetali, P. *Mathematical aspects of mixing times in Markov chains*. Now Publishers Inc, 2006.

Nedic, A. and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

Orabona, F., Crammer, K., and Cesa-Bianchi, N. A generalized online mirror descent with applications to classification and regression. *Machine Learning*, 99(3):411–435, 2015.

Ram, S. S., Nedić, A., and Veeravalli, V. V. Incremental stochastic subgradient algorithms for convex optimization. *SIAM Journal on Optimization*, 20(2):691–717, 2009.

Ram, S. S., Nedić, A., and Veeravalli, V. V. Asynchronous gossip algorithm for stochastic optimization: Constant stepsize analysis. In *Recent Advances in Optimization and its Applications in Engineering*, pp. 51–60. Springer, 2010a.

Ram, S. S., Nedić, A., and Veeravalli, V. V. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications*, 147(3):516–545, 2010b.

Ran, X., Shi, P., Nedi, A., and Khan, U. A. A general framework for decentralized optimization with first-order methods. *Proceedings of the IEEE*, 108(11):1869–1889, 2020.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryQu7f-RZ.

Rockafellar, R. T. *Convex analysis*. Number 28. Princeton university press, 1970.

Sayed, A. H. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7(ARTICLE):311–801, 2014.

Schizas, I. D., Ribeiro, A., and Giannakis, G. B. Consensus in ad hoc wsns with noisy links part i: Distributed estimation of deterministic signals. *IEEE Transactions on Signal Processing*, 56(1):350–364, 2008.

Shah, S. M. and Avrachenkov, K. E. Linearly convergent asynchronous distributed admm via markov sampling. *arXiv preprint arXiv:1810.05067*, 2018.

Shamir, O. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *Journal of Machine Learning Research*, 18(52):1–11, 2017.

Shi, W., Ling, Q., Yuan, K., Wu, G., and Yin, W. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Trans. Signal Processing*, 62(7): 1750–1761, 2014.

Shi, W., Ling, Q., Wu, G., and Yin, W. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Sirb, B. and Ye, X. Consensus optimization with delayed and stochastic gradients on decentralized networks. In *Big Data (Big Data), 2016 IEEE International Conference on*, pp. 76–85. IEEE, 2016.

Srivastava, K. and Nedic, A. Distributed asynchronous constrained stochastic optimization. *IEEE Journal of Selected Topics in Signal Processing*, 5(4):772–790, 2011.

Sun, T., Sun, Y., and Yin, W. On markov chain gradient descent. *Advances in neural information processing systems*, 2018.

Sun, T., Qiao, L., Liao, Q., and Li, D. Novel convergence results of adaptive stochastic gradient descents. *IEEE Transactions on Image Processing*, 30:1044–1056, 2020a.

Sun, T., Sun, Y., Xu, Y., and Yin, W. Markov chain block coordinate descent. *Computational Optimization and Applications*, 75(1):35–61, 2020b.

Ward, R., Wu, X., and Bottou, L. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, pp. 6677–6686. PMLR, 2019.

Xin, R., Kar, S., and Khan, U. A. Decentralized stochastic optimization and machine learning: A unified variance-reduction framework for robust performance and fast convergence. *IEEE Signal Processing Magazine*, 37(3): 102–113, 2020.

Yin, W., Mao, X., Yuan, K., Gu, Y., and Sayed, A. H. A communication-efficient random-walk algorithm for decentralized optimization. *arXiv preprint arXiv:1804.06568*, 2018.

Yuan, K., Ling, Q., and Yin, W. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Zeng, J. and Yin, W. On nonconvex decentralized gradient descent. *IEEE Transactions on Signal Processing*, 66 (11):2834–2848, 2018.

Zou, F., Shen, L., Jie, Z., Sun, J., and Liu, W. Weighted AdaGrad with unified momentum. *arXiv preprint arXiv:1808.03408*, 2018.

Zou, F., Shen, L., Jie, Z., Zhang, W., and Liu, W. A sufficient condition for convergences of adam and rmsprop. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11127–11135, 2019.

Supplementary materials for

Adaptive Random Walk Gradient Descent for Decentralized Optimization

We denote

$$\beta(T) := \Big\{ \Big(\sum_{i=1}^{n} \|[\boldsymbol{P}^T]_{i,:} - \pi^*\|^2 \Big)^{1/2} \Big\}.$$

With Assumption 1 and [Footnote 1, (Mao et al., 2020)], it holds $\beta(T) = \mathcal{O}\Big(\sigma(\boldsymbol{P})^T\Big)$.

A. Proofs of Results in the Convex Scenario

A.1. Technical lemmas

Lemma A.1. *Let $\{\boldsymbol{x}^k\}_{k\geq 1}$ be generated by the adaptive online learning with dependent data (Algorithm 1) and let the set $\mathcal{K}$ be bounded. Then, it follows that*

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\| \leq \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|_{(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}} / \delta^{\frac{1}{4}} \leq \eta \|\boldsymbol{m}^k / (\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{4}}\| / \delta^{\frac{1}{4}}.$$

Given any $\boldsymbol{x}^* \in \mathcal{K}$, $\alpha > 0$, and $T \in \mathbb{Z}^+$, we first introduce the following shorthand notation

$$\begin{cases} A_k := \mathbb{E}\|[\boldsymbol{m}^k]^2/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}\|_1, \\ B_k := \mathbb{E}\left(\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{m}^k \rangle\right), \\ C_k := \eta\theta A_{k-1} + 2(1-\theta)\eta^2 T L_H \sum_{h=1}^{T} A_{k-h} + 2(1-\theta)(RL + \hat{R})\beta(T). \end{cases} \tag{10}$$

We have the following lemmas.

Lemma A.2. *Assume* $\{\boldsymbol{x}^k\}_{k\geq 1}$ *is generated by Algorithm 1, then we have*

$$\sum_{k=1}^{K} A_k \leq \sum_{k=1}^{K} \mathbb{E}\|(\boldsymbol{g}^k)^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1.$$

Lemma A.3. *Assume* $\{\boldsymbol{x}^k\}_{k\geq 1}$ *is generated by Algorithm 1, and Assumption 1, 2, 3, 4 hold, then the following result holds*

$$B_k + (1-\theta)\mathbb{E}(f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)) \leq \theta B_{k-1} + C_k.$$

A.2. Proof of Theorem 3.1

Given $K \in \mathbb{Z}^+$, Lemma A.3 indicates the following inequalities

$$\begin{aligned}
B_K + (1-\theta)\mathbb{E}(f(\boldsymbol{x}^K) - f(\boldsymbol{x}^*)) &\leq \theta B_{K-1} + C_K,\\
B_{K-1} + (1-\theta)\mathbb{E}(f(\boldsymbol{x}^{K-1}) - f(\boldsymbol{x}^*)) &\leq \theta B_{K-2} + C_{K-1},\\
B_{K-2} + (1-\theta)\mathbb{E}(f(\boldsymbol{x}^{K-2}) - f(\boldsymbol{x}^*)) &\leq \theta B_{K-3} + C_{K-2},\\
&\vdots\\
B_1 + (1-\theta)\mathbb{E}(f(\boldsymbol{x}^1) - f(\boldsymbol{x}^*)) &\leq \theta B_0 + C_1.
\end{aligned}$$

Summing the inequalities above, we then get

$$\begin{aligned}
(1-\theta)\sum_{k=1}^{K}\mathbb{E}(f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)) &\leq -B_K + (\theta-1)\sum_{k=1}^{K-1} B_k + \sum_{k=1}^{K} C_k\\
&\leq (\theta-1)\sum_{k=1}^{K-1} B_k + \sum_{k=1}^{K} C_k + 2RL, \qquad (11)
\end{aligned}$$

where we used the following inequality

$$-B_K = \mathbb{E}\left(\langle \boldsymbol{x}^K - \boldsymbol{x}^*, \boldsymbol{m}^K\rangle\right) \leq \mathbb{E}\|\boldsymbol{x}^K - \boldsymbol{x}^*\| \cdot \|\boldsymbol{m}^K\| \leq 2RL,$$

due to Assumption 2.

The definition of $\boldsymbol{z}^k$ indicates that

$$\boldsymbol{z}^{k+1} - \boldsymbol{x}^* = \boldsymbol{x}^k - \boldsymbol{x}^* - \eta\boldsymbol{m}^k/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}.$$

On the other hand, we have

$$\text{Diag}\big((\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\big)\cdot(\boldsymbol{z}^{k+1} - \boldsymbol{x}^*) = \text{Diag}\big((\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\big)\cdot(\boldsymbol{x}^k - \boldsymbol{x}^*) - \eta\boldsymbol{m}^k.$$

We are then led to

$$\|\boldsymbol{z}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} = \|\boldsymbol{x}^* - \boldsymbol{x}^k\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} + 2\eta\langle\boldsymbol{m}^k, \boldsymbol{x}^* - \boldsymbol{x}^k\rangle + \eta^2\|[\boldsymbol{m}^k]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1.$$

Let

$$\textbf{Proj}(\boldsymbol{x}) := \arg\min_{\boldsymbol{y}\in\mathcal{K}} \|\boldsymbol{x} - \boldsymbol{y}\|_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}.$$

The convexity of $\mathcal{K}$ indicates that **Proj**$(\cdot)$ is contractive. For any $\boldsymbol{x}^* \in \mathcal{K}$, **Proj**$(\boldsymbol{x}^*) = \boldsymbol{x}^*$, we then have

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} = \|\textbf{Proj}(\boldsymbol{z}^{k+1}) - \textbf{Proj}(\boldsymbol{x}^*)\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} \leq \|\boldsymbol{z}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}.$$

Thus, we can get

$$\begin{aligned}
\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}}} &= \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} + \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}}-(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}\\
&\leq \|\boldsymbol{x}^* - \boldsymbol{x}^k\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} + \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}}-(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}\\
&\quad + 2\eta\langle\boldsymbol{m}^k, \boldsymbol{x}^* - \boldsymbol{x}^k\rangle + \eta^2\|[\boldsymbol{m}^k]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1.
\end{aligned}$$

Taking total expectation gives us

$$
\begin{aligned}
-2\eta B_k \leq \mathbb{E}\|\boldsymbol{x}^* - \boldsymbol{x}^k\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} - \mathbb{E}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}}} \\
+ \eta^2 A_k + \mathbb{E}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}} - (\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}.
\end{aligned} \tag{12}
$$

With Assumption 2, we are then led to

$$
\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2_{(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}} - (\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} \leq 4R^2(\|(\boldsymbol{v}^{k+1}+\delta\mathbb{I})^{\frac{1}{2}}\|_1 - \|(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1). \tag{13}
$$

With (13), summation of (12) from $k = 1$ to $K - 1$ yields

$$
\sum_{k=1}^{K-1}(-B_k) \leq \frac{1}{2\eta}\|\boldsymbol{x}^* - \boldsymbol{x}^1\|^2_{(\boldsymbol{v}^1+\delta\mathbb{I})^{\frac{1}{2}}} + \frac{1}{2}\eta\sum_{k=1}^{K-1} A_k + \frac{2R^2}{\eta}\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1.
$$

Together with (11), we then have

$$
\begin{aligned}
(1-\theta)\sum_{k=1}^{K}\mathbb{E}(f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)) \leq (1-\theta)\Big(\frac{1}{2\eta}\mathbb{E}\|\boldsymbol{x}^* - \boldsymbol{x}^1\|^2_{(\boldsymbol{v}^1+\delta\mathbb{I})^{\frac{1}{2}}} + \frac{1}{2}\eta\sum_{k=1}^{K-1} A_k \\
+ \frac{2R^2}{\eta}\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) + \sum_{k=1}^{K} C_k + 2RL.
\end{aligned} \tag{14}
$$

Now, we turn to bound the right side of (14). Obviously, we have $\frac{1}{2\eta}\mathbb{E}\|\boldsymbol{x}^* - \boldsymbol{x}^1\|^2_{(\boldsymbol{v}^1+\delta\mathbb{I})^{\frac{1}{2}}} \leq \frac{2R^2}{\eta}\sqrt{L+\delta}$. Using [Lemma 9 in the appendix, (Li & Orabona, 2019)] and Lemma A.2, we have

$$
\frac{1}{2}\eta\sum_{k=1}^{K-1} A_k \leq \frac{\eta}{2}\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1. \tag{15}
$$

Also, we can get

$$
\sum_{k=1}^{K-1} C_k \leq \eta\theta\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1 + 2K(1-\theta)RL\beta(T) + 2T^2\eta^2 L_H(1-\theta)\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1. \tag{16}
$$

Substituting the bounds (15) and (16) into (14), we then get

$$
\mathbb{E}\Big[f\Big(\frac{\sum_{k=1}^{K}\boldsymbol{x}^k}{K}\Big) - \min f\Big] \leq \frac{c_1 + c_2(K) + c_3(T,K)}{K},
$$

where $c_1 := \frac{2RL}{1-\theta} + \frac{2R^2}{\eta}\sqrt{L+\delta}$, $c_2(K) := [\frac{2R^2}{\eta} + \frac{\eta}{2} + \frac{\eta\theta}{1-\theta}] \cdot \mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1$, and $c_3(T,K) := 2K(RL+\hat{R})\beta(T) + 2T^2\eta^2 L_H\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1$. By setting $\eta = \min\{1/T, 1\}$, we then get

$$
\mathbb{E}\Big[f\Big(\frac{\sum_{k=1}^{K}\boldsymbol{x}^k}{K}\Big) - \min f\Big] = \mathcal{O}\left(\frac{(T+1)\mathbb{E}\|(\boldsymbol{v}^K+\delta\mathbb{I})^{\frac{1}{2}}\|_1}{K} + \frac{T}{K} + \beta(T)\right) = \mathcal{O}\left(\frac{T+1}{K^{1-\alpha}} + \frac{T}{K} + \beta(T)\right). \tag{17}
$$

By setting $\beta(T) = \mathcal{O}(\epsilon)$, we get $T = \mathcal{O}\Big(\frac{\ln(1/\epsilon)}{\ln(1/\sigma(\boldsymbol{P}))}\Big)$ and $K = \mathcal{O}\Big(\max\Big\{\frac{\ln(1/\epsilon)^{\frac{1}{1-\alpha}}}{\epsilon^{\frac{1}{1-\alpha}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{1}{1-\alpha}}}, \frac{\ln(1/\epsilon)^{\frac{1}{1-\alpha}}}{\epsilon^{\frac{1}{1-\alpha}}}\Big\}\Big)$.

B. Proofs of Results in the Nonconvex Scenario

B.1. Technical lemmas

Due to the fact that the scheme is changed in the nonconvex case (we get rid of the projections), we define new notation for the subsequent analysis. Given any $T \in \mathbb{Z}^+$, we denote the following items to simplify the presentations of the following lemmas

$$\left\{\begin{array}{c} \hat{A}_k := \mathbb{E}\|[\boldsymbol{m}^k]^2/(\boldsymbol{v}^k+\delta\mathbb{I})\|_1, \\ \hat{B}_k := \mathbb{E}\left(\langle -\nabla f(\boldsymbol{x}^k), \boldsymbol{m}^k/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\rangle\right), \\ \hat{C}_k := \theta L_H \eta \hat{A}_{k-1} + \frac{(1-\theta)\eta^2 L_H^2 T}{2\delta}\sum_{h=1}^{T}\hat{A}_{k-h} + \theta L^2(\mathbb{E}\|\mathbb{I}/(\boldsymbol{v}^{k-1}+\delta\mathbb{I})^{\frac{1}{2}}\|_1 - \mathbb{E}\|\mathbb{I}/(\boldsymbol{v}^{k}+\delta\mathbb{I})^{\frac{1}{2}}\|_1) + (1-\theta)L^2\beta(T). \end{array}\right. \tag{18}$$

Lemma B.1. *Assume $\{\boldsymbol{x}^k\}_{k\geq 1}$ is generated by Algorithm 1, then we have*

$$\sum_{k=1}^{K}\hat{A}_k \leq \sum_{k=1}^{K}\mathbb{E}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^k+\delta\mathbb{I})\|_1.$$

Lemma B.2. *Assume $\{\boldsymbol{x}^k\}_{k\geq 1}$ is generated by Algorithm 1 and the functions are nonconvex. Let $\mathcal{K}$ be the full space and Assumptions 2' and 4 hold, then the following result holds*

$$\hat{B}_k + \frac{(1-\theta)}{2}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) \leq \theta\hat{B}_{k-1} + \hat{C}_k.$$

B.2. Proof of Theorem 3.2

According to Lemma B.2, we have

$$\begin{aligned} \frac{(1-\theta)}{2}\sum_{k=1}^{K}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) &\leq -\hat{B}_K + (\theta-1)\sum_{k=1}^{K-1}\hat{B}_k + \sum_{k=1}^{K}\hat{C}_k \\ &\leq (\theta-1)\sum_{k=1}^{K-1}\hat{B}_k + \sum_{k=1}^{K}\hat{C}_k + \frac{L^2}{\sqrt{\delta}}. \end{aligned} \tag{19}$$

The Lipschitz property of the gradients gives

$$\begin{aligned} \mathbb{E}f(\boldsymbol{x}^{k+1}) - \mathbb{E}f(\boldsymbol{x}^k) &\leq \mathbb{E}\langle\nabla f(\boldsymbol{x}^k), \boldsymbol{x}^{k+1}-\boldsymbol{x}^k\rangle + \frac{L_H\mathbb{E}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2}{2} \\ &= \eta\hat{B}_k + \frac{L_H\eta^2}{2}\hat{A}_k. \end{aligned} \tag{20}$$

Combine with (20), we get the following estimate

$$\sum_{k=1}^{K-1} -\hat{B}_k \leq L_H\eta\sum_{k=1}^{K-1}\hat{A}_k + \frac{2f(\boldsymbol{x}^1)}{\eta}. \tag{21}$$

On the other hand, we have the following bound

$$\frac{2}{1-\theta}\sum_{k=1}^{K}\hat{C}_k \leq (\frac{2\theta}{1-\theta}L_H\eta + \frac{\eta^2 L_H^2 T^2}{\delta})\sum_{k=1}^{K}\hat{A}_k + \frac{2\theta L^2}{(1-\theta)\sqrt{\delta}} + 2KL^2\beta(T). \tag{22}$$

Using [Lemma 2, (Li & Orabona, 2019)] and Lemma B.1, we have

$$\sum_{k=1}^{K}\hat{A}_k \leq \ln(\frac{KL^2+\delta}{\delta}). \tag{23}$$

Substituting (23), (22) and (21) into (19), then we get

$$\begin{aligned} \sum_{k=1}^{K}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) &\leq \Big(\frac{2\theta}{1-\theta}L_H\eta + \frac{\eta^2L_H^2T^2}{\delta}\Big)\ln(\frac{KL^2+\delta}{\delta}) \\ &+ \frac{(1+\theta)L^2}{(1-\theta)\sqrt{\delta}} + 2KL^2\beta(T) + \frac{2f(\boldsymbol{x}^1)}{\eta}. \end{aligned} \tag{24}$$

Noticing that

$$\sum_{k=1}^{K}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) \geq (\sum_{k=1}^{K}\frac{1}{k^{\frac{\nu}{2}}\sqrt{C}})\cdot \min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\},$$

we then complete the proof by the fact that $\frac{2}{2-\nu}\leq 2$ when $0<\nu\leq 1$. Thus, we can get

$$\min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\} \leq \frac{c_3+c_4(K)+c_5(T,K)}{K^{1-\frac{\nu}{2}}},$$

where $c_3 := \frac{8\theta L^2\sqrt{C}}{(1-\theta)\sqrt{\delta}}+\frac{8\sqrt{C}f(\boldsymbol{x}^1)}{\eta}$, $c_4(K) := \frac{4(1+\theta)L_H\sqrt{C}\eta}{(1-\theta)}\ln(\frac{KL^2+\delta}{\delta})$, and $c_5(T,K) := 8KL^2\sqrt{C}\beta(T)+\frac{4\eta^2L_H^2\sqrt{C}T^2}{\delta}\ln(\frac{KL^2+\delta}{\delta})$. With $\eta=\min\{1/T,1\}$, it is easy to see that

$$\min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\} = \widetilde{\mathcal{O}}\Big(\frac{T+1}{K^{1-\frac{\nu}{2}}}+K^{\frac{\nu}{2}}\beta(T)\Big).$$

If the finite mixing time assumption holds, to get the ϵ error for $\min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\}$, we need to set

$$\begin{cases} K^{\frac{\nu}{2}}\beta(T)=\widetilde{\mathcal{O}}(\epsilon), \\ \frac{T+1}{K^{1-\frac{\nu}{2}}}=\widetilde{\mathcal{O}}(\epsilon). \end{cases} \Rightarrow \begin{cases} T=\widetilde{\Theta}(\tau_{\text{mix}}(\epsilon^{\frac{2}{2-\nu}}))=\widetilde{\Theta}(\frac{\ln(1/\epsilon)}{\ln(1/\sigma(\boldsymbol{P}))}), \\ K=\widetilde{\Theta}\Big(\max\{\frac{[\ln(1/\epsilon)]^{\frac{2}{2-\nu}}}{\epsilon^{\frac{2}{2-\nu}}[\ln(1/\sigma(\boldsymbol{P}))]^{\frac{2}{2-\nu}}},\frac{[\ln(1/\epsilon)]^{\frac{2}{2-\nu}}}{\epsilon^{\frac{2}{2-\nu}}}\}\Big). \end{cases}$$

C. Proofs of Results of the Zeroth Version

C.1. Technical lemmas

Lemma C.1. *Assume $\{\boldsymbol{x}^k\}_{k\geq 1}$ is generated by Algorithm 2 and Assumption 1 2, 3, 4 hold, then the following result holds*

$$B_k+(1-\theta)\mathbb{E}(f(\boldsymbol{x}^k)-f(\boldsymbol{x}^*))\leq \theta B_{k-1}+C_k+F_k,$$

where A_k, B_k and C_k are defined in (10) and $F_k=\mathcal{O}(r)$.

Lemma C.2. *Assume $\{\boldsymbol{x}^k\}_{k\geq 1}$ is generated by Algorithm 2 with nonconvex functions, $\mathcal{K}$ is the full space, and Assumptions 1, 2' and 4 hold. Then the following result holds*

$$\hat{B}_k+\frac{(1-\theta)}{2}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big)\leq \theta\hat{B}_{k-1}+\hat{C}_k+\hat{F}_k,$$

where $\hat{A}_k$, $\hat{B}_k$ and $\hat{C}_k$ are defined in (18) and $\hat{F}_k=\mathcal{O}(r)$

C.2. Proof of Proposition 4.1

Convex part: Note that Lemma A.2 also holds and $\{\boldsymbol{g}^k\}_{k\geq 1}$ is uniformly bounded by the mean-value theorem. The rest of proof is similar to the proof of Theorem 3.1 using Lemma C.1. By setting $\eta=\min\{1/T,1\}$, we can get

$$\mathbb{E}\Big[f\Big(\frac{\sum_{k=1}^{K}\boldsymbol{x}^k}{K}\Big)-\min f\Big]=\mathcal{O}\left(\frac{d^\alpha}{K}+\frac{Td^\alpha}{K^{1-\alpha}}+\beta(T)+r\cdot d^{\frac{3}{2}}\right).$$

Nonconvex part: The proof is similar to the proof of Theorem 3.1 using Lemma C.2. By setting $\eta=\min\{1/T,1\}$, we can get

$$\min_{1\leq k\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{x}^k)\|^2\}=\widetilde{\mathcal{O}}\left(\frac{d^\nu}{K^{1-\frac{\nu}{2}}}+\frac{Td^\nu}{K^{1-\frac{\nu}{2}}}+K^{\frac{\nu}{2}}\beta(T)+K^{\frac{\nu}{2}}r\cdot d^{\frac{3}{2}}\right).$$

D. Proofs of the Technical Lemmas

D.1. Proof of Lemma A.1

Let $\textbf{Proj}(\boldsymbol{x}) := \arg\min_{\boldsymbol{y}\in\mathcal{K}} \|\boldsymbol{x}-\boldsymbol{y}\|_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}$. Note that $\textbf{Proj}(\boldsymbol{x}^k)=\boldsymbol{x}^k$, then we have

$$\begin{aligned}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} &= \|\textbf{Proj}(\boldsymbol{z}^{k+1})-\textbf{Proj}(\boldsymbol{x}^k)\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}\\ &\leq \|\boldsymbol{z}^{k+1}-\boldsymbol{x}^k\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} = \eta^2\|\boldsymbol{m}^k/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|^2_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}} = \eta^2\|\boldsymbol{m}^k/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{4}}\|^2.\end{aligned}$$

Combing the fact that $\delta^{\frac{1}{4}}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\| \leq \|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|_{(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}}$, we then get the desired result.

D.2. Proof of Lemma A.2

Notice the fact that $\boldsymbol{m}^k=(1-\theta)\sum_{j=1}^k \theta^{k-j}\boldsymbol{g}^j$ when $k\geq 1$, then we have

$$\begin{aligned}\|[\boldsymbol{m}^k]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|^1 &= \sum_{i=1}^d |m_i^k/(v_i^k+\delta)^{\frac{1}{4}}|^2 \leq \sum_{i=1}^d (1-\theta)^2|\sum_{j=1}^k \theta^{k-j} g_i^j/(v_i^k+\delta)^{\frac{1}{4}}|^2\\ &\overset{a)}{\leq} \sum_{i=1}^d (1-\theta)^2(\sum_{j=1}^k \theta^{k-j}(v_i^k+\delta)^{\frac{1}{2}}) \times \sum_{j=1}^k \theta^{k-j}(g_i^j)^2/(v_i^k+\delta)\\ &\leq \sum_{i=1}^d (1-\theta)^2\cdot\frac{(v_i^k+\delta)^{\frac{1}{2}}}{1-\theta}\cdot\sum_{j=1}^k \theta^{k-j}(g_i^j)^2/(v_i^k+\delta)\\ &= (1-\theta)\cdot\sum_{j=1}^k \theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1 \overset{b)}{\leq} (1-\theta)\cdot\sum_{j=1}^k \theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})^{\frac{1}{2}}\|_1\end{aligned}$$

where $a)$ uses the fact that $(\sum_{j=1}^k a_j b_j)^2 \leq (\sum_{j=1}^k a_j^2)\cdot(\sum_{j=1}^k b_j^2)$ with $a_j=\theta^{\frac{k-j}{2}}(v_i^k+\delta)^{\frac{1}{4}}$ and $b_j=\theta^{\frac{k-j}{2}}g_i^j/(v_i^k+\delta)^{\frac{1}{2}}$; $b)$ is due to $v_i^j\leq v_i^k$ when $j\leq k$ and $1\leq i\leq d$.

Direct calculations yield

$$\begin{aligned}\sum_{k=1}^K\sum_{j=1}^k \theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})^{\frac{1}{2}}\|_1 &= \sum_{j=1}^K\sum_{k=j}^K \theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})^{\frac{1}{2}}\|_1\\ &= \sum_{j=1}^K\sum_{k=j}^K \theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})^{\frac{1}{2}}\|_1 \leq \frac{1}{1-\theta}\sum_{j=1}^K\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})^{\frac{1}{2}}\|_1.\end{aligned}$$

Combining the inequalities above and replace j with k, we then get the desired result.

D.3. Proof of Lemma A.3

The convexity of $f_{i_k}(\boldsymbol{x})$ with respect to $\boldsymbol{x}$ and the fact that $\boldsymbol{g}^k=\nabla f_{i_k}(\boldsymbol{x}^k)$ indicate that

$$\begin{aligned}\mathbb{E}\langle\boldsymbol{x}^*-\boldsymbol{x}^k,\boldsymbol{g}^k\rangle &\leq \mathbb{E}[f_{i_k}(\boldsymbol{x}^*)-f_{i_k}(\boldsymbol{x}^k)]\\ &= \mathbb{E}[f_{i_k}(\boldsymbol{x}^{k-T})-f_{i_k}(\boldsymbol{x}^k)+f_{i_k}(\boldsymbol{x}^*)-f_{i_k}(\boldsymbol{x}^{k-T})].\end{aligned} \tag{25}$$

The Lipchitz gradient continuity of $f_{i_k}(\boldsymbol{x})$ with respect to $\boldsymbol{x}$ gives us

$$\mathbb{E}[f_{i_k}(\boldsymbol{x}^{k-T})-f_{i_k}(\boldsymbol{x}^k)] \leq \frac{L_H}{2}\mathbb{E}\|\boldsymbol{x}^k-\boldsymbol{x}^{k-T}\|^2+\mathbb{E}\langle\boldsymbol{x}^{k-T}-\boldsymbol{x}^k,\nabla f_{i_k}(\boldsymbol{x}^k)\rangle. \tag{26}$$

We also have the following result

$$\begin{aligned}
&\mathbb{E}(f_{i_k}(\boldsymbol{x}^*) - f_{i_k}(\boldsymbol{x}^{k-T}) \mid \chi^{k-T}) \\
&= \sum_{i=1}^{n}(f_i(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^{k-T}))\mathbb{P}(i_k = i \mid \chi^{k-T}) \\
&= \sum_{i=1}^{n}(f_i(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^{k-T}))\mathbb{P}(i_k = i \mid i_{k-T}) \\
&= \sum_{i=1}^{n}(f_i(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^{k-T}))[\boldsymbol{P}^T]_{i_{k-T},i} \\
&= \sum_{i=1}^{n}(f_i(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^{k-T}))/n + \sum_{i=1}^{n}(f_i(\boldsymbol{x}^*) - f_i(\boldsymbol{x}^{k-T}))([\boldsymbol{P}^T]_{i_{k-T},i} - 1/n) \\
&\leq f(\boldsymbol{x}^*) - f(\boldsymbol{x}^{k-T}) + 2\hat{R}\beta(T) \\
&= f(\boldsymbol{x}^k) - f(\boldsymbol{x}^{k-T}) + f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k) + 2\hat{R}\beta(T).
\end{aligned} \tag{27}$$

and

$$\begin{aligned}
&\mathbb{E}(\langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \nabla f_{i_k}(\boldsymbol{x}^{k-T}) - \nabla f(\boldsymbol{x}^{k-T})\rangle \mid \sigma(\mathbf{x}^k, \chi^{k-T})) \\
&= (\langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \sum_{i=1}^{n} \nabla f_i(\boldsymbol{x}^{k-T})\mathbb{P}(i_k = i \mid i_{k-T}) - \nabla f(\boldsymbol{x}^{k-T})\rangle) \\
&= (\langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \sum_{i=1}^{n} \nabla f_i(\boldsymbol{x}^{k-T})[\boldsymbol{P}^T]_{i_{k-T},i} - \nabla f(\boldsymbol{x}^{k-T})\rangle) \\
&= \langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \sum_{i=1}^{n} \nabla f_i(\boldsymbol{x}^{k-T})([\boldsymbol{P}^T]_{i_{k-T},i} - 1/n))\rangle \\
&\leq 2RL\beta(T).
\end{aligned} \tag{28}$$

Note that

$$f(\boldsymbol{x}^k) - f(\boldsymbol{x}^{k-T}) \leq \langle \nabla f(\boldsymbol{x}^{k-T}), \boldsymbol{x}^k - \boldsymbol{x}^{k-T}\rangle + \frac{L_H}{2}\|\boldsymbol{x}^k - \boldsymbol{x}^{k-T}\|^2. \tag{29}$$

Substituting (26), (27), (29) and (28) into (25), we then get

$$\begin{aligned}
&\mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k\rangle \leq L_H\|\boldsymbol{x}^k - \boldsymbol{x}^{k-T}\|^2 + \mathbb{E}\langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \nabla f_{i_k}(\boldsymbol{x}^{k-T}) - \nabla f(\boldsymbol{x}^{k-T})\rangle \\
&\quad + \mathbb{E}\langle \boldsymbol{x}^{k-T} - \boldsymbol{x}^k, \nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{x}^{k-T})\rangle + f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k) + 2\hat{R}\beta(T) \\
&\quad \leq 2L_H\mathbb{E}\|\boldsymbol{x}^k - \boldsymbol{x}^{k-T}\|^2 + 2RL\beta(T) + f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k) + 2\hat{R}\beta(T) \\
&\quad \leq 2(RL + \hat{R})\beta(T) + 2TL_H\eta^2\sum_{h=1}^{T}\mathbb{E}\|[\boldsymbol{m}^{k-h}]^2/(\boldsymbol{v}^{k-h} + \delta\mathbb{I})^{\frac{1}{2}}\|_1 + f(\boldsymbol{x}^*) - f(\boldsymbol{x}^k).
\end{aligned} \tag{30}$$

According to our algorithm and we denote $\Lambda := \mathbb{E}(\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k\rangle \mid \chi^k)$, then we have

$$\begin{aligned}
&\mathbb{E}\left(\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{m}^k\rangle \mid \chi^k\right) = \mathbb{E}\left(\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \theta\boldsymbol{m}^{k-1} + (1-\theta)\boldsymbol{g}^k\rangle \mid \chi^k\right) \\
&\quad = (1-\theta)\cdot\Lambda + \theta\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{m}^{k-1}\rangle \\
&\quad = (1-\theta)\cdot\Lambda + \theta\langle \boldsymbol{x}^* - \boldsymbol{x}^{k-1}, \boldsymbol{m}^{k-1}\rangle + \theta\langle \boldsymbol{x}^k - \boldsymbol{x}^{k-1}, \boldsymbol{m}^{k-1}\rangle \\
&\quad \overset{b)}{\leq} (1-\theta)\cdot\Lambda + \theta\langle \boldsymbol{x}^* - \boldsymbol{x}^{k-1}, \boldsymbol{m}^{k-1}\rangle + \theta\|\boldsymbol{x}^{k-1} - \boldsymbol{x}^k\|_{(\boldsymbol{v}^{k-1}+\delta\mathbb{I})^{\frac{1}{2}}} \cdot \|\boldsymbol{m}^{k-1}/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{4}}\| \\
&\quad \overset{c)}{\leq} (1-\theta)\cdot\Lambda + \theta\langle \boldsymbol{x}^* - \boldsymbol{x}^{k-1}, \boldsymbol{m}^{k-1}\rangle + \eta\theta\|\boldsymbol{m}^{k-1}/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{4}}\|^2
\end{aligned}$$

where $b)$ uses the Cauchy-Schwarz inequality $\langle \boldsymbol{a}, \boldsymbol{b} \rangle \leq \|\boldsymbol{a}\|_{\boldsymbol{c}} \cdot \|\boldsymbol{b}/\boldsymbol{c}^{1/2}\|$, and $c)$ depends on the iteration of our algorithm. Taking total expectations on both sides of I and using $\mathbb{E}(\mathbb{E}(\cdot \mid \chi^k)) = \mathbb{E}(\cdot)$, we get

$$B_k \leq (1-\theta)\mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k \rangle + \theta B_{k-1} + \eta\theta A_{k-1}. \tag{31}$$

Substituting (30) into (31), we then proved the desired result.

D.4. Proof of Lemma B.1

Similar to the proof of Lemma A.2, we have

$$\begin{aligned}
\|\boldsymbol{m}^k/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}\|^2 &= \sum_{i=1}^{d} |m_i^k/(v_i^k+\delta)^{\frac{1}{2}}|^2 \leq \sum_{i=1}^{d}(1-\theta)^2 |\sum_{j=1}^{k} \theta^{k-j} g_i^j/(v_i^k+\delta)^{\frac{1}{2}}|^2 \\
&\overset{a)}{\leq} \sum_{i=1}^{d}(1-\theta)^2 (\sum_{j=1}^{k}\theta^{k-j}) \cdot \sum_{j=1}^{k}\theta^{k-j}\frac{(g_i^j)^2}{(v_i^k+\delta)} \\
&\leq \sum_{i=1}^{d}(1-\theta)^2 \cdot \frac{1}{1-\theta} \cdot \sum_{j=1}^{k-1}\theta^{k-j}(g_i^j)^2/(v_i^k+\delta) \\
&= (1-\theta)\cdot\sum_{j=1}^{k}\theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^k+\delta\mathbb{I})\|_1 \overset{b)}{=} (1-\theta)\cdot\sum_{j=1}^{k}\theta^{k-j}\|(\boldsymbol{g}^j)^2/(\boldsymbol{v}^j+\delta\mathbb{I})\|_1
\end{aligned}$$

where $a)$ uses the fact that $(\sum_{j=1}^k a_j b_j)^2 \leq \sum_{j=1}^k a_j^2 \sum_{j=1}^k b_j^2$ with $a_j = \theta^{\frac{k-j}{2}}$ and $b_j = \theta^{\frac{k-j}{2}} g_i^j/(v_i^k+\delta)^{\frac{1}{2}}$, and $b)$ is because of $v_i^j \leq v_i^k$ when $j \leq k$ and $1 \leq i \leq d$. The rest of this proof is identical to the proof of Lemma A.2.

D.5. Proof of Lemma B.2

We first consider to bound

$$\begin{aligned}
\mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k\rangle &= -\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) \\
&+ \mathbb{E}\langle \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}, \nabla f(\boldsymbol{x}^k) - \nabla f(\boldsymbol{x}^{k-T})\rangle \\
&+ \mathbb{E}\langle \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}, \nabla f(\boldsymbol{x}^{k-T}) - \nabla f_{i_k}(\boldsymbol{x}^{k-T})\rangle \\
&+ \mathbb{E}\langle \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}, \nabla f_{i_k}(\boldsymbol{x}^{k-T}) - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle.
\end{aligned} \tag{32}$$

Direct calculation together with Assumption 2 give us

$$|\mathbb{E}\langle \nabla f(\boldsymbol{x}^k), \nabla f_{i_k}(\boldsymbol{x}^{k-T}) - \nabla f(\boldsymbol{x}^{k-T})\rangle| \leq L^2\beta(T). \tag{33}$$

The Lipschitz property of the gradient yields

$$\begin{aligned}
&|\mathbb{E}\langle \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}, \nabla f_{i_k}(\boldsymbol{x}^{k-T}) - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle| \\
&\leq \frac{L_H}{\delta^{\frac{1}{4}}}\sum_{h=1}^{T}\mathbb{E}\|\boldsymbol{x}^{k-h+1} - \boldsymbol{x}^{k-h}\|\|\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{4}}\| \\
&\leq \frac{\eta L_H}{\delta^{\frac{1}{2}}}\sum_{h=1}^{T}\mathbb{E}\|\boldsymbol{m}^{k-h}/(\boldsymbol{v}^{k-h}+\delta\mathbb{I})^{\frac{1}{4}}\| \times \|\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{4}}\| \\
&\overset{a)}{\leq} \frac{\eta L_H}{2\alpha\delta^{\frac{1}{2}}}\sum_{h=1}^{T}\mathbb{E}\|[\boldsymbol{m}^{k-h}]^2/(\boldsymbol{v}^{k-h}+\delta\mathbb{I})^{\frac{1}{2}}\|_1 + \frac{\eta L_H T\alpha\mathbb{E}\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1}{2\delta^{\frac{1}{2}}} \\
&\overset{b)}{\leq} \frac{\eta^2 L_H^2 T}{2\delta}\sum_{h=1}^{T}\mathbb{E}\|[\boldsymbol{m}^{k-h}]^2/(\boldsymbol{v}^{k-h}+\delta\mathbb{I})^{\frac{1}{2}}\|_1 + \frac{\mathbb{E}\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k+\delta\mathbb{I})^{\frac{1}{2}}\|_1}{2},
\end{aligned} \tag{34}$$

where $a)$ uses the Cauchy inequality and $\alpha > 0$, and $b)$ is obtained by setting $\alpha = \frac{\delta^{\frac{1}{2}}}{\eta L_H T}$. Substituting (34) and (33) into (32), we get

$$\mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k\rangle \leq -\frac{1}{2}\mathbb{E}\Big(\|[\nabla f(\boldsymbol{x}^k)]^2/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}\|_1\Big) + \frac{\eta^2 L_H^2 T}{2\delta}\sum_{h=1}^{T}\hat{A}_{k-h} + L^2\beta(T). \tag{35}$$

We also use a shorthand notation $\Lambda := \mathbb{E}(\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k\rangle \mid \chi^k)$ and then we have

$$\begin{aligned}
&\mathbb{E}\left(\langle -\nabla f(\boldsymbol{x}^k), \boldsymbol{m}^k/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}\rangle \mid \chi^k\right) \\
&= \mathbb{E}\left(\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \theta\boldsymbol{m}^{k-1} + (1-\theta)\boldsymbol{g}^k\rangle \mid \chi^k\right) \\
&= (1-\theta)\cdot\Lambda + \theta\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{m}^{k-1}\rangle \\
&= (1-\theta)\cdot\Lambda + \theta\langle -\nabla f(\boldsymbol{x}^{k-1})/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{m}^{k-1}\rangle \\
&+ \theta\langle[\nabla f(\boldsymbol{x}^{k-1}) - \nabla f(\boldsymbol{x}^k)]/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{m}^{k-1}\rangle \\
&+ \theta\langle \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{2}} - \nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{m}^{k-1}\rangle \\
&\overset{a)}{\leq} (1-\theta)\cdot\Lambda + \theta\hat{B}_{k-1} + \theta L_H\eta\hat{A}_{k-1} \\
&+ \theta L^2(\|\mathbb{I}/(\boldsymbol{v}^{k-1} + \delta\mathbb{I})^{\frac{1}{2}}\|_1 - \|\mathbb{I}/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}\|_1),
\end{aligned} \tag{36}$$

where $a)$ uses the Cauchy-Schwarz inequality and the Lipschitz property of f. Substituting (36) into (35), we then proved the result.

D.6. Proof of Lemma C.1

Direct calculation gives us

$$\mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k\rangle = \mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \nabla f_{i_k}(\boldsymbol{x}^k)\rangle + \mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle.$$

[Theorem 3.1, (Ghadimi & Lan, 2013)] gives the following bound

$$\|\mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \boldsymbol{g}^k - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle\| = \mathcal{O}(r\cdot d^{3/2}).$$

Note that if $\boldsymbol{g}^k \leftarrow \nabla f_{i_k}(\boldsymbol{x}^k)$ in (30), $\mathbb{E}\langle \boldsymbol{x}^* - \boldsymbol{x}^k, \nabla f_{i_k}(\boldsymbol{x}^k)\rangle$ can be bounded in the same way as (30), we then complete the proof.

D.7. Proof of Lemma C.2

For the nonconvex zeroth-order version, we have

$$\begin{aligned}
\mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k\rangle &= \mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \nabla f_{i_k}(\boldsymbol{x}^k)\rangle \\
&+ \mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle.
\end{aligned}$$

Leveraging [Theorem 3.1, (Ghadimi & Lan, 2013)] and using the bound

$$\mathbb{E}\langle -\nabla f(\boldsymbol{x}^k)/(\boldsymbol{v}^k + \delta\mathbb{I})^{\frac{1}{2}}, \boldsymbol{g}^k - \nabla f_{i_k}(\boldsymbol{x}^k)\rangle = \mathcal{O}(r\cdot d^{\frac{3}{2}}),$$

we then complete the proof.