MSTGD: A Memory Stochastic sTratified Gradient Descend Method with an Exponential Convergence Rate

Aixiang(Andy) Chen\textsuperscript{a,b}, Jinting Zhang\textsuperscript{c}, Zanbo Zhang\textsuperscript{a,b}, Zhihong Li\textsuperscript{b}

\textsuperscript{a}Institute of Artificial Intelligence and Deep Learning, Guangdong University of Finance and Economics, Guangzhou 510320, China
\textsuperscript{b}School of Statistics and Mathematics, Guangdong University of Finance and Economics, Guangzhou 510320, China
\textsuperscript{c}Department of Statistics and Data Science, National University of Singapore, Singapore 117546

Abstract

The fluctuation effect of gradient expectation and variance caused by parameter update between consecutive iterations is neglected or confusing by current mainstream gradient optimization algorithms. Using this fluctuation effect, combined with the stratified sampling strategy, this paper designs a novel Memory Stochastic sTratified Gradient Descend (MSTGD) algorithm with an exponential convergence rate. Specifically, MSTGD uses two strategies for variance reduction: the first strategy is to perform variance reduction according to the proportion $p$ of used historical gradient, which is estimated from the mean and variance of sample gradients before and after iteration, and the other strategy is stratified sampling by category. The statistic $\bar{G}_{mst}$ designed under these two strategies can be adaptively unbiased, and its variance decays at a geometric rate. This enables MSTGD based on $\bar{G}_{mst}$ to obtain an exponential convergence rate of the form $\lambda^{2(k-k_0)} (\lambda \in (0, 1), k$ is the number of iteration steps, $\lambda$ is a variable related to proportion $p$). Unlike most other algorithms that claim to achieve an exponential convergence rate, the convergence rate is independent of parameters such as dataset size $N$, batch size $n$, etc., and can be achieved at

\textsuperscript{*}Research supported by Chinese National Office for Philosophy and Social Sciences(19FTJB003), National Bureau of Statistics of China(2016LY64), Natural Science Foundation of Guangdong Province (10451032001006140).
a constant step size. Theoretical and experimental results show the effectiveness of MSTGD.

**Keywords:** Memory Stochastic Gradient Descend, Exponential Convergence Rate, Smooth, Strongly Convex, Stratified Sampling

2010 MSC: 00-01, 99-00

1. Introduction

In this paper, we consider the problem of optimizing the objective function of the form (1) given a training set \((X, Y)\) with \(N\) records.

\[
\arg\min_{W \in \mathbb{R}^{\mathcal{P}}} J(W) = \frac{1}{N} \sum_{i=1}^{N} J(W; x^{(i)}, y^{(i)}) = \frac{1}{N} \sum_{i=1}^{N} J_i,
\]

where \(W = [w_1, w_2, \cdots, w_{\mathcal{P}}]^T\) is the optimization parameter, which can be a parameter set composed of connection weights in a deep network model. In different contexts, \(W\) can refer to model parameters or it can represent a deep network model.

The objective function represented by Equation (1) is in the form of a finite sum of the training sample set, and the term \(J_i = J(W; x^{(i)}, y^{(i)})\) in the summation is the loss function for sample \((x^{(i)}, y^{(i)})\). The loss function can be least squares, cross entropy, etc.

Problem of the form (1) are of broad interest, as they encompass a variety of problems in statistics, machine learning, and optimization. For example, many problems such as image processing and visual recognition [1][2][3][4][5], speech recognition [6][7][8], machine translation and natural language understanding [9] can be reduced to the optimization problem in the form of Equation (1).

Because of their wide applicability, it is important to carefully design and develop more efficient solver to such problems. Gradient descent method (GDM) which originated from the work of Robbins and Monro et al. [10] and its series of improved algorithms are the current mainstream effective optimization algorithms for such problems. However, Achieving the linear convergence rate of
the full gradient while keeping a low iteration cost as the stochastic gradient descend is still an open and challenging problem.

This paper proposes a new strategy called Memory Stochastic sTratified Gradient Descend (MSTGD). According to which a statistic called $\bar{G}_{mst}$ is designed, whose subscripts come from the first three letters of MSTGD. MSTGD iterates as follow

$$W_{k+1} = W_k - hG^k_{mst}(W_k) = W_k - h \sum_{j=1}^{C} w_j \cdot G^k_j(W_k),$$  \hspace{1cm} (2)

where $k$ represents the index of the iteration, $h$ is the step size of the algorithm, and the italicized $w_j$ represents the class weight, which can be determined according to the ratio of the total number of $j$th ($j \in \{1, 2, \cdots, C\}$) class samples to the total number of samples.

For the sake of brevity, this paper will $\bar{G}^k_{mst}(W_k), G^k_j(W_k)$ abbreviated as $\bar{G}^k_{mst}, G^k_j$ respectively. The value of $\bar{G}^k_{mst}$ in formulae (2) is obtained by averaging a $C$-dimensions auxiliary storage vector $G$ by components. The auxiliary vector $G$ tracks the gradient signal ever used, and this is why MSTGD being named as the memory algorithm. In the $k$-th iteration we set

$$G^k_j = p^k_j \cdot G^{k-1}_j + q^k_j \cdot g(W_k, \xi^k_j) \ for \ each \ j \in \{1, 2, \cdots, C\},$$  \hspace{1cm} (3)

where $\xi^k_j$ is a random index of sample in category $j$, the $g(W_k, \xi^k_j)$ is random gradient generated by network $W_k$ inputting this sample. The stochastic gradient $g(W_k, \xi^k_j)$ generated by the independent sample $\xi^k_j$ ensures that it is independent of the historical gradient $G^{k-1}_j$.

When $p^k_j = 0, q^k_j = 1$, Equation (3) is simplified into the memoryless form of $G^k_j = g(W_k, \xi^k_j)$. In this case, the mean value of traditional stratified sampling $\bar{G}^k_{st}$ is calculated according to $\sum_{j=1}^{C} \omega_j \cdot G^k_j$ in Equation (2). The subscript of $\bar{G}^k_{st}$ is the first two letters of the word stratification.

This paper has made the following three main contributions:

- A novel statistic $\bar{G}_{mst}$ is proposed, and gives the $p, q$ condition that makes $\bar{G}_{mst}$ adaptively unbiased ($p^k_j, q^k_j$ in Formulae (6) or (9)).
A new strategy called p-based variance reduction is proposed, which enables the variance of $G_{mst}$ to decay at a geometric rate. Prior to this, it was generally believed that in order to exponentially decay the variance of the mini-batch stochastic gradient, it was necessary to use a dynamic sample size strategy such as the exponential growth of the sample size.

This paper proves that MSTGD can achieve exponential convergence rate in the form of $\lambda^{2(k-k_0)} (0 < \lambda < 1, k$ is the number of iteration steps) under the condition of constant step size $h$ and constant sample size.

1.1. Related works

The gradient descent method (GDM) originated from the work of Robbins and Monro et al. [10] and its improved algorithms are the current mainstream optimization algorithms. GDM iterates as the form

$$W_{k+1} = W_k - h \cdot \bar{g} = W_k - \frac{h}{n} \sum_{i=1}^{n} g(W_k, i),$$

(4)

where $\bar{g}$ is sample gradient mean, $g(W_k, i)$ is gradient of sample $i$. The sample size is 1 or $N$ (population size), the corresponding algorithms are called Stochastic Gradient Descent (SGD) and Full Gradient Descent (FGD), respectively. If $1 < n < N$, the corresponding algorithm is called the mini-Batch stochastic gradient descent algorithm (Batch). In this paper, SGD, FGD and Batch are collectively referred to as GDM, which belongs to the category of SGD-like Algorithms.

As a kind of SGD-like Algorithms, one of the main features of MSTGD is to determine the next moving direction by weighting the current gradient and the historical gradient stored in the auxiliary vector $G$. The weight coefficient of the historical gradient plays a role in variance reduction while ensuring unbiased.

Similar to MSTGD, other SGD-like algorithms that use historical gradient information include Momentum [11] and its improved algorithms [12, 20, 21], Adam and AdaMax [20], Nadam [21], NAG [12], AMSGrad [26]. As far as we know, these algorithms are lacking in the discussion of the influence of the weight coefficient of the historical gradient on the unbiasedness and variance of the gradient direction. This paper will clarify how the weight coefficient affects the unbiasedness.
of the statistic $\tilde{G}_{mst}$, and give sufficient conditions to exponentially decay the variance of $\tilde{G}_{mst}$. It is theoretically proved that the algorithm MSTGD designed in this paper has a linear convergence rate that Momentum, Adam, AdaMax, Nadam, NAG, AMSGrad etc. have not claim.

The second feature of MSTGD is that when the objective function satisfies the assumption of continuous strong convexity, the theoretical linear convergence rate can be achieved under a constant step size. There is a lot of work to improve the algorithm around the step size. For example, most of the step size adaptive algorithms Adagrad\cite{17}, Adadelta\cite{27}, Adam and Adamax\cite{20}, Nadam\cite{21}, AMSGrad\cite{26}, etc., use the gradient second-order moment information to set the dynamic step size. To the best of the authors’ knowledge, there is little discussion of whether these improvements can improve the convergence order of the algorithm.

The third feature of MSTGD is to use a more efficient stratified sampling strategy to ensure a linear convergence rate without consuming too much storage space. Work on improving the performance of stochastic gradient optimization algorithms through more efficient stratified sampling strategies originally came from an algorithm named SSAG\cite{16}. The calculation of the gradient direction $\tilde{G}_{st}$ of SSAG also uses an auxiliary vector $G$ similar to this paper, but $\tilde{G}_{st}$ does not involve the preservation of historical gradients when updating the components in $G$, which is an essential difference with the memory-type stochastic stratified gradient $\tilde{G}_{mst}$ in this paper. As far as we know, the algorithms that claim to achieve linear convergence rate include SAG\cite{13}, SAGA\cite{14}, SVRG\cite{15}, DSSM(Dynamic Sample Size Methods)\cite{25}, SSAG\cite{16}, etc., but the linear convergence rate of these algorithms often needs to consume too much memory. For example, SAG must record the gradient for each sample, which is a large storage overhead in the case of massive training data. In contrast, MSTGD only needs to save gradient for each category of data, requiring much less storage space. For another example, in order to achieve a linear convergence rate, DSSM adopts a dynamic sample capacity strategy of exponential growth in the form of $\lceil \tau^{k-1} \rceil (\tau > 1)$. Correspondingly, the computing resources required for a single iteration also
increase exponentially, while the storage resources and computing resources required for each iteration of MSTGD are constant.

1.2. Notation and basic definition

The following is notation and basic definition used in this paper.

- $\xi^k_j : N_j \rightarrow n^k_j$ indicates that $n^k_j$ samples are drawn from the $j^{th}$ subpopulation at the $k^{th}$ iteration. $f = \frac{n}{N}, f^k = \frac{n^k_j}{N_j}$ is the corresponding sampling ratio. $n^k_j = 1$ by default.

- $g(W_k, \xi^k_j)$: the stochastic gradient generated when a sample is randomly drawn from the $j^{th}$ subpopulation and fed into the network $W_k$. $g(W_k, i)$: the gradient generated when the $i^{th}$ sample in the set is input into the network $W_k$.

- $\nabla J(W_k) = \nabla J_N(W_k) = \frac{1}{N} \sum_{i=1}^{N} g(W_k, i)$: population mean of gradient at iteration $k$. $\nabla J_n(W_k) = \bar{g} = \frac{1}{n} \sum_{i=1}^{n} g(W_k, i)$: sample mean of gradient at iteration $k$.

- $\sigma^2_k = \frac{1}{N} \sum_{i=1}^{N} (g(W_k, i) - \nabla J_N(W_k))^2$: population variance of gradient at iteration $k$.

- $V(\cdot)$: the variance of the variable in parentheses.

- $s^2_k := V(\bar{g}) = \frac{(1-f_k)n^k}{n^k}$: the variance of the sample gradient mean at the $k^{th}$ iteration.

2. Properties of the statistic $\bar{G}_{\text{mst}}$

2.1. Conditions on $p, q$ to ensure the unbiasedness of $\bar{G}_{\text{mst}}$

The basic idea to ensure $\bar{G}_{\text{mst}}$ an unbiased estimation is to transfer the gradient signal of the previous iteration to the current iteration in an appropriate proportion. This result is given in the below theorem.
Theorem 1. Let \( g(W_{k-1}, \xi_j^{k-1}), g(W_k, \xi_j^k), j = 1, \cdots, C \) denote the random gradients of the networks \( W_{k-1}, W_k \) with random sample \( \xi_j^{k-1}, \xi_j^k \) from the \( j \)-th class as input (that is, the random gradients are produced in two consecutive iterations), and \( E_j^{k-1}, E_j^k \) be their expectations respectively. If \( \frac{p_j^k}{q_j^k} = \frac{E_j^k}{E_j^{k-1}} \), then \( \bar{G}_{mst} \) is an unbiased estimation of the population mean \( \bar{G} \), i.e., \( E(\bar{G}_{mst}) = \bar{G} \), or \( E(\bar{G}_{mst}) = \bar{G} \) for simplification.

In Theorem 1, the expectations \( E_j^{k-1}, E_j^k \) under two consecutive iterations are theoretical values and cannot be known exactly in general. In practice, they are generally estimated with the gradient mean of random mini-batch samples.

2.2. Variance of \( \bar{G}_{mst} \)

In this section, we firstly give a lemma, which we use as a basis for the discussion of the variance of \( \bar{G}_{mst} \).

Lemma 1. Let \( E_j^{k-1} \) and \( E_j^k \) be the gradient mean of the \( j \)-th sub-population of the networks \( W_{k-1} \) and \( W_k \). \( V_j^{k-1} = V(G_j^{k-1}) \) and \( V_j^k = V(G_j^k) \) be the gradient variance of the \( j \)-th sub-population of the networks \( W_{k-1} \) and \( W_k \) respectively. Then, \( \bar{G}_{mst} \) has a stationary point variance called \( V_{sp}(\bar{G}_{mst}) \) as follow

\[
V_{sp}(\bar{G}_{mst}) = \sum_j C_j \left[ w_j^2 \frac{(E_j^k)^2 V_j^{k-1} - (E_j^{k-1})^2 V_j^k}{(E_j^k)^2 V_j^{k-1} + (E_j^{k-1})^2 V_j^k} \right] \tag{5}
\]

if we let

\[
p_j^k = \frac{E_j^k E_j^{k-1} V_j^k}{(E_j^k)^2 V_j^{k-1} + (E_j^{k-1})^2 V_j^k}, q_j^k = \frac{(E_j^k)^2 V_j^{k-1}}{(E_j^k)^2 V_j^{k-1} + (E_j^{k-1})^2 V_j^k} \tag{6}
\]

in (3).

The subscript of the left-hand term of the above formulae (5) comes from the first letter of Stationary Point, not \( \text{min} \) as expected. That is because the function of (22) in appendix 7.2 has only stationary point and no minimum point.

Likewise, in Lemma 1 the exact values of gradient expectations \( E_j^{k-1}, E_j^k \) and gradient variances \( V_j^{k-1}, V_j^k \) for the previous and subsequent iterations are
generally not known. In practice, these variables are estimated by the gradient mean and variance of a random mini-batch samples.

Since the parameter $W_{k-1}$ becomes another different parameter $W_k$ after iteration, generally $E^{k-1} \neq E^k, V^{k-1} \neq V^k$. We call this phenomenon the fluctuation effect of gradient mean and variance. Many existing algorithms, such as the momentum method \[11\], incremental average gradient method \[33\], Adam \[20\], etc., do not fully consider the influence of this fluctuation effect, so the gradient direction of these algorithms are usually biased estimators.

According to (6), $q^k_j < 1$ since $V^k_j \neq 0$. However, $p^k_j$ can be great than 1 if no further restriction. Theorem 2 in the next subsection gives a sufficient condition to ensure $p^k_j < 1$.

2.3. Design effect of statistics $\bar{G}_{mst}$

The aforementioned general results are not easy to see the effect of the new statistic $\bar{G}_{mst}^k$ on variance reduction. In fact, the variance of the layer (category) samples is left in each component of the vector of memory $G$ by $\bar{G}_{mst}^k$ in different proportions $p^k_j (j \in \{1, \cdots, C\}, k \in \{1, 2, \cdots, \})$, and is rapidly attenuated as the iteration proceeds. Therefore, the variance of the statistic $\bar{G}_{mst}^k$ (memory type) is smaller than that of the traditional stratified sampling statistic $\bar{G}_{st}^k$ (memoryless type), and the variance remaining in the memory part will be rapidly attenuated as the iteration progresses.

Since any $w_i, w_j$ in the vector $W = [w_1, w_2, \cdots, w_\varphi]^T$ represent different connection weights in the network, the variance matrices $V_{sp}(\bar{G}_{mst}), V(\bar{G}_{st})$ of $\bar{G}_{mst}(W), \bar{G}_{st}(W)$ will be diagonal square matrices of order $\varphi$.

In order to be able to compare $V_{sp}(\bar{G}_{mst}), V(\bar{G}_{st})$, the definition of the comparison of the same type of diagonal square matrix is given below.
**Definition 1.** Given diagonal square matrix of order $\varphi$

\[
\bar{\lambda}_1 = \begin{pmatrix} \sigma_1^2 & & \\ & \sigma_2^2 & \\ & & \ddots & \sigma_\varphi^2 \end{pmatrix}, \quad \bar{\lambda}_2 = \begin{pmatrix} \Sigma_1^2 & & \\ & \Sigma_2^2 & \\ & & \ddots & \Sigma_\varphi^2 \end{pmatrix},
\]

if for any $i (i = 1, 2, \cdots, \varphi)$, $\sigma^2_i \leq \Sigma^2_i$, then $\bar{\lambda}_1 \leq \bar{\lambda}_2$

**Definition 2.** $\bar{\lambda}_1, \bar{\lambda}_2$ are defined as [1]. If there is at least one $i (i = 1, 2, \cdots, \varphi)$, such that $\sigma^2_i < \Sigma^2_i$, the rest $j (j \neq i, j = 1, 2, \cdots, \varphi) \sigma^2_i \leq \Sigma^2_i$, then $\bar{\lambda}_1 < \bar{\lambda}_2$

With the above definition of the comparison of diagonal square matrices, we can get the following Corollary [1]

**Corollary 1.** $V(\cdot)$ represents the variance of the variable in parentheses. The statistic $\bar{G}_{\text{mst}}^k$ has the following properties.

1. $V_{sp}(\bar{G}_{\text{mst}}^k) < V(\bar{G}_{\text{st}}^k)$
2. $V_{sp}(\bar{G}_{\text{mst}}^{k+i}) \leq p^{2i}V(\bar{G}_{\text{mst}}^k) + \sum_{i=1}^{t} p^{2(t-i)} q^2 V(\bar{G}_{\text{st}}^{k+i}), 0 < q < 1$

Property (1) in Corollary [1] shows that the memory statistic $\bar{G}_{\text{mst}}^k$ has a smaller design effect (smaller variance) than $\bar{G}_{\text{st}}^k$.

Property (2) in Corollary [1] shows that this property holds when $k$ takes any value, regardless of the starting point. Therefore, for the sake of brevity, the inequality of property (2) can be reduced to a simpler form

\[
V_{sp}(\bar{G}_{\text{mst}}^k) \leq p^{2k}V(\bar{G}_{\text{mst}}) + \sum_{i=1}^{k} p^{2(k-i)} q^2 V(\bar{G}_{\text{st}}^i), 0 < q < 1. \tag{7}
\]

The expansion or contraction of $V_{sp}(\bar{G}_{\text{mst}}^k)$ mainly depends on the value of $p$. Next, we first introduce a lemma, which is used to obtain a sufficient condition to ensure that $V_{sp}(\bar{G}_{\text{mst}}^k)$ decays rapidly with the number of iteration steps (Theorem [2])

9
Lemma 2. If \( \eta \in (0, 1) \), then there exists \( \gamma \in (0, 1) \) such that
\[
(1 + \frac{1}{1 - \eta})\eta^{2k} \leq \gamma^{2(k-k_0)}
\] holds for a sufficiently large positive integer \( k, k_0, k > k_0 \).

The closer \( \eta \) in Lemma 2 is to 1, the larger the \( k \) is required to achieve the geometric decay rate on the right-hand side of (8).

Theorem 2. \( E_j^k, E_j^{k-1}, V_j^k, V_j^{k-1} \) are defined as Lemma 1. If \( E_j^k = E_j^{k-1} \), then (8) can be simplified as
\[
p_j^k = \frac{V_j^k}{V_j^{k-1} + V_j^k}, \quad q_j^k = \frac{V_j^{k-1}}{V_j^{k-1} + V_j^k}.
\] In this case, there exist \( \gamma \in (0, 1) \) and a sufficiently large \( k_0 \) such that
\[
V_{sp}(\bar{G}_{mst}^k) \leq \gamma^{2(k-k_0)} \Phi \cdot I,
\] where \( \Phi \) is a positive constant, \( I \) is an identity matrix of order \( \wp \).

Theorem 2 shows that, as long as we try to ensure that \( E_j^k = E_j^{k-1} \), that is, the gradients of two adjacent iterations are expected to be equal, after a sufficiently large \( k_0 \) iterations, the variance of \( \bar{G}_{mst}^k \) can decay at the geometric rate on the right side of (10).

Theorem 2 can lead to the following theorem 3.

Theorem 3. Let \( \text{Var}(\bar{G}_{mst}^k) := E(||\bar{G}_{mst}^k||_2^2) - ||E(\bar{G}_{mst}^k)||_2^2, \Phi = \wp M \), then
\[
\text{Var}(\bar{G}_{mst}^k) \leq \gamma^{2(k-k_0)} \wp M
\] = \[
\gamma^{2(k-k_0)} \wp \Phi.
\] (11)

\( V_{sp}(\bar{G}_{mst}^k) \) can decay at the geometric rate of (11), mainly because of the effect of the parameter \( p_j^k \) (the proportion of the historical gradient being reused). This can be seen from the proof process of Theorem 2 in Appendix 7.5. Therefore, the variance reduction strategy of \( V_{sp}(\bar{G}_{mst}^k) \) is named as p-based variance reduction strategy, where the letter p of p-based refer to the maximal proportion of the historical gradient being reused, that is \( p = \max p_j^k \).

Based on the result that \( V_{sp}(\bar{G}_{mst}^k) \) decays at a geometric rate, we can design the corresponding MSTGD algorithm.
3. MSTGD algorithm based on $\bar{G}_{mst}$

The update direction of MSTGD is determined by calculating the mean value of $G$ by component, which means it needs to maintain a $C$-dimension vector $G$ during iterations. At each iteration MSTGD calculates a mini-batch gradient mean $G_j$ of samples of the $j^{th}$ class, and then the $j^{th}$ component in $G$ is updated by the new $G_j$ for each class $j \in \{1, \cdots, C\}$.

The pseudo code of MSTGD is described in Algorithm 1 which is designed according to Theorem 2 to ensure convergence.

\begin{algorithm}
\caption{Memory-type Stochastic Stratified Gradient Descent (MSTGD)}
\begin{algorithmic}[1]
\Function{MSTGD}{for minimizing $J(W) = \frac{1}{N} \sum_{i=1}^{N} J_i$ with step size $h$}
\State Parameters: Step size $h$, Batch size $B$, Training data size $N$, the total number of class $C$, the class weight $w_j$, Iteration bound $\text{max\_iter}$
\State Inputs: training data $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(N)}, y^{(N)})$
\State set $G_j = 0$ for $j \in \{1, 2, \cdots, C\}$, $k=0$
\While{$k \leq \text{max\_iter}$}
\For{$j = 1, 2, \cdots, C$}
\State calculate sample mean $E_j$ and variance $V_j$ of $\lceil \frac{B}{C} \rceil$ samples from $j^{th}$ class
\State calculate $p_j^k, q_j^k$ according to equation 9
\State update $j^{th}$ component of $G$, $G_j \leftarrow p_j \times G_j + q_j \times (g(W_k, \xi_j^k) - E_j)$, where $\xi_j^k$ is a random sample from $j^{th}$ class
\EndFor
\State $W_{k+1} = W_k - h \sum_{j=1}^{C} w_j \cdot (G_j + E_j)$
\State $k++$
\EndWhile
\EndFunction
\end{algorithmic}
\end{algorithm}

In order to satisfy the convergence condition of $E_j^k = E_j^{k-1}$ in Theorem 2, MSTGD does not directly store the stochastic gradient $g(W_k, \xi_j^k)$ in the auxiliary variable $G$ as in [3], but first perform the mean-zeroing operation $g(W_k, \xi_j^k) - E_j$, and then store it in the corresponding component of $G$ (Line 8 in Algorithm 1).
Our experiments show that such a mean-zeroing operation can ensure the convergence of the algorithm.

4. Linear convergence rate of MSTGD

This section discusses the convergence rate of MSTGD. Section 4.1 firstly give some basic assumptions. The theorem in Section 4.2 shows how the variance of the gradient in (4) specifically affects the convergence rate of the algorithm, also reveals that, if without reusing the historical gradients, due to the existence of gradient variance, the algorithms based on iteration (4) can only reach the sub-linear convergence rate. Section 4.3 is the linear convergence theorem of MSTGD. This theorem and its related proofs reveal the mainly reason of linear convergence of MSTGD is that the reused historical gradient greatly reduces the variance, which is called p-based variance reduction.

4.1. Background and assumptions

To build the general convergent result we need the following assumptions.

- **A1** The Cost function $J(W)$ is continuously differentiable and first order Lipschitz continuous with Lipschitz constant $L > 0$, i.e.,

  $$ \| \nabla J(W) - \nabla J(W') \| \leq L \| W - W' \|. \quad (12) $$

- **A2** Cost function $J(W)$ is strongly convex, i.e.,

  $$ J(W') \geq J(W) + \nabla J(W)(W' - W) + \frac{1}{2} c \| W' - W \|^2. \quad (13) $$

  This assumption leads to a useful fact (proved in Appendix 7.7)

  $$ 2c(J(W) - J_*) \leq ||\nabla J(W)||^2. \quad (14) $$

- **A3** The objective function $J$ and stochastic gradient $g(w_k, \xi^k_j)$ satisfy the following conditions:

  1. The sequence of iterates $\{W_k\}$ is contained in an open set over which $J$ is bounded below by a scalar $J_{\inf}$
2. There exist scalars $\mu_G \geq \mu > 0$ such that, for all $k \in \{1, 2, \cdots\}$,
\[
\nabla J(W_k)^T E[\bar{G}^k_{mst}] \geq \mu \|\nabla J(W_k)\|^2_2,
\]
and
\[
\|E[\bar{G}^k_{mst}]\|_2 \leq \mu_G \|\nabla J(W_k)\|_2.
\]

4.2. General convergent result of gradient descent

Before analyzing the convergence rate of MSSG, we present a general convergent result of GDM (Gradient Descent Methods) in this section, where GDM refers specifically to FGD, SGD and mini-batch SGD. From this general result, we can see how gradient variance impacts the convergence rate of an algorithm.

Theorem 4 formalizes the relationship between gradient variance and convergence rate. With smaller gradient variance, GDM gets closer to the optimal solution, and if gradient variance is reduced to zero, GDM can achieve linear convergence rate.

**Theorem 4 (Convergence-Variance Inequality, CVI).** Let $W_k$ be a network obtained by a method of GDM, $\sigma^2_k(s^2_k)$ is the gradient variance on population(samples) at the $k^{th}$ iteration, $\sigma^2_{\min} = \min\{\sigma^2_k\}$, $\sigma^2_{\max} = \max\{\sigma^2_k\}$. If assumptions $A1$ and $A2$ hold, then under the condition of step size $h_k < \frac{2}{L}$ and the ratio of sample size $\frac{n_{k+1}}{n_k} \geq \frac{\sigma^2_{\min} h_{k+1}}{\sigma^2_{\max} h_k} \frac{2-h_k L}{2-h_{k+1} L}$, the following inequality holds for all $k \in \{1, 2, \cdots\}$:
\[
E[J(W_{k+1}) - J_*] \leq \Lambda_1 + \rho^k (E[J(W_1) - J_*] - \Lambda_1),
\]
where $J_*$ is the optimal value, $\Lambda_k = \frac{h_k L s^2_k}{2\mu(2-h_k L)} \rho_i := 1 - \mu h_i(2 - L h_i), \rho = \max\{\rho_i\}$.

CVI theorem is a general result of FGD, SGD, and Batch. In the case of FGD, the sample size $n$ is equal to data size (population size) $N$; the sampling ratio $f$ equals 1, so $s^2_k = 0$, also $\Lambda_1 = 0$. This leads to linear convergence rate of FGD. In the case of SGD, the sample size $n$ is equal to one, the number $\Lambda_1$ ceases
In this case, SGD cannot achieve a linear convergence rate, it can only be a slower sub-linear convergence rate. As for Batch, the sample size $n$ is a random number between 1 and $N$. Therefore, if a strategy of cleverly setting the size of $n$, such as a dynamic sample size $n$ that satisfies certain conditions, the number $\Lambda$ can be reduced infinitely close to zero, Batch still retains the possibility of linear convergence.

This paper presents Theorem 4 and its proof independently (see Appendix 7.8), but Theorem 4 is not our first creation, similar works and related results can be found in earlier literature\[34, 28\].

### 4.3. Convergence results of MSTGD

Theorem 4 shows that for the GDM using the iteration of (4), since no historical gradient is used, even if a strategy such as dynamic step size is employed, the linear convergence rate cannot generally be achieved. Only in special cases, such as using a dynamic sample size strategy, can a linear rate of convergence be achieved.

Theorem 5 pioneered in this paper and its corresponding proof (see Appendix 7.9) show that the MSTGD using the iteration of (2) and (3) can achieve linear convergence rate with a constant stepsize and constant sample size due to the efficient reuse of historical gradient.

**Theorem 5 (Linear Convergence of MSTGD).** Suppose that Assumptions $A_1, A_2, A_3$ (with $J_{inj} = J_*$) hold, In addition, MSTGD is run with a fix stepsize $h_k = \bar{h}, \forall k \in \{1, 2, \cdots\}$, and satisfying

$$0 \leq \bar{h} \leq \min\{\frac{\mu}{L\mu_G}, \frac{1}{c\mu}\},$$

then for all $k \in \{1, 2, \cdots\}$, the expected optimality gap satisfies

$$E[J(W_k) - J_*] \leq \Omega \lambda^2(k - k_0),$$

where

$$\Omega := \max\{\frac{\bar{h}L\Phi}{c\mu}, J(W_1) - J_*\}.$$
and

\[ \lambda := \max \{ 1 - \frac{\bar{h}c\mu}{2}, \gamma \} < 1. \]  

(20)

Theorem 5 shows that after a sufficiently large \( k_0 \) iterations, MSTGD can linearly converge to the optimal solution at the rate of the right-hand side of (18).

5. Experimental results

We make comparison among \( \bar{G}_{mst}, \bar{G}_{st}, SGD \) and Batch, by testing them on artificial data as well as the MNIST data set. It turns out that \( \bar{G}_{mst} \) has the higher estimation accuracy than the other methods.

The artificial data is generated in the form of a 40 \( \times \) 10 random matrix (that is, 40 random numbers in each round, and 10 rounds successively, each random number is a simulation of a random gradient \( g(W^k, \xi^k) \)). Also, various kinds of data set are generated so that they are with increasing mean, decreasing mean, increasing variance, and decreasing variance in successive rounds, respectively. In each round, we divide the 40 numbers into 4 sub-populations as a layered simulation. In the experiment, we take the square of the deviation between the estimated value generated by the estimator and the overall mean (true value) as an evaluation of the accuracy of the search direction provided by the estimator.

For fair and comparable consideration, the number of samples for each method is set to 4, except for SGD which uses a single sample. \( \bar{G}_{mst} \) and \( \bar{G}_{st} \) randomly select a sample from each of the 4 sub-populations, and Batch randomly select 4 samples from the whole population.

The parameters \( p^k_j \) and \( q^k_j \) in \( \bar{G}_{mst} \) are calculated according to formulae (6), where \( E_j \) and \( V_j \) respectively takes the value of the mean and variance of the sub-populations in the random data set.
5.1. Results on a uniformly random data set

In the first experiment, we uniformly sample from each of the following 10 intervals

\[ [8, 12], [8, 10], [6, 9], [5, 8], [4, 7], [3, 6], [3, 5], [2, 4], [2, 3], [0, 3] \]

(with decreasing mean) to form a random data set with a specification of 40 × 10 as the data population.

The error in this paper is uniformly expressed by the square of the deviation between the estimator and the true value. Figure 1 (a) and (b) show the error curve of each estimator and the corresponding error descriptive statistics. As can be seen in Figure 1 (a), the blue curve corresponding to \( \bar{G}_{\text{mst}} \) is located at the bottom, under all other curves, indicating that the estimated value provided by \( \bar{G}_{\text{mst}} \) statistic is the closest to the true value. Figure 1 (b) shows that the error of \( \bar{G}_{\text{mst}} \) has the smallest mean and standard deviation, indicating that the estimated value provided by \( \bar{G}_{\text{mst}} \) is more accurate and more stable.

![Error curve and descriptive statistics](image)

| OPTIMIZER | Dec._udata |
|-----------|------------|
| \( \bar{G}_{\text{mst}} \) | 1.11 ± 0.97 |
| Batch     | 2.91 ± 0.59 |
| SGD       | 9.14 ± 3.26 |
| \( \bar{G}_{\text{st}} \)  | 1.9 ± 0.68  |

(a) The error curve (Dec._udata)  
(b) The mean and standard deviation of error(Dec._udata)

Figure 1: the error curve and its description statistics(Dec._udata)

The second random data set is uniformly generated from ten intervals

\[ [0, 3], [2, 3], [2, 4], [3, 5], [3, 6], [4, 7], [5, 8], [6, 9], [8, 10], [8, 12], \]

with increasing mean, which is also formatted into a 40 × 10 matrix as the data population.
It can be seen from Figure 2 that on the mean increasing data set, the search direction provided by $\bar{G}_{mst}$ is still satisfactory. For the increasing mean data set, in Figure 2 (a), the blue curve corresponding to $\bar{G}_{mst}$ is also located at the bottom, under all other curves. In Figure 2 (b), the error of $\bar{G}_{mst}$ also has the smallest mean and standard deviation, so the estimated value provided by $\bar{G}_{mst}$ is more accurate and more stable.

5.2. Results on normal random data set

To further investigate the performance of the algorithms on other kind of data sets, we generate sets of $40 \times 10$ random numbers from the normal population $N(\mu, \sigma)$ as experimental data. According to the conventional way of experimental data design, the following five different types of data sets are generated.

1. Random data set where $\mu, \sigma$ are both uniformly distributed random numbers in the interval $[1, 20]$, denoted by randomndata
2. Random data set with decreasing mean $\mu$, denoted by MeanDndata
3. Random data set with increasing mean $\mu$, denoted by MeanIndata
4. Random data set with decreasing variance, denoted by VarDndata
5. Random data set with increasing variance, denoted by VarIndata

The experimental results are summarized in Figure 3. It can be seen that, in experiment on each type of data, the gradient estimation value generated by

| OPTIMIZER | Inc._udata |
|-----------|------------|
| $\bar{G}_{mst}$ | 0.8 ± 0.44 |
| $\bar{G}_{st}$ | 2.09 ± 1.0 |
| SGD | 11.49 ± 3.23 |
| Batch | 2.12 ± 0.74 |
the statistic $\bar{G}_{mst}$ always has the smallest error and smallest standard deviation. This shows that the statistic $\bar{G}_{mst}$ can approximate the gradient mean (true value) of population more accurately and more stably.

| OPTIMIZER | randomndata |
|-----------|-------------|
| $\bar{G}_{mst}$ | 11.92 ± 16.45 |
| Batch | 33.02 ± 45.15 |
| SGD | 208.79 ± 384.43 |
| $\bar{G}_{st}$ | 24.93 ± 29.49 |

Figure 3: the error curve and its description statistics(randomndata)

| OPTIMIZER | MeanDndata |
|-----------|-------------|
| $\bar{G}_{mst}$ | 5.65 ± 11.28 |
| Batch | 35.25 ± 49.47 |
| SGD | 187.38 ± 404.46 |
| $\bar{G}_{st}$ | 24 ± 33.25 |

Figure 4: the error curve and its description statistics(MeanDndata)

5.3. Results on the MNIST dataset

Next we compare the performance of different algorithms on the MNIST data set. We use the full gradient method to train a 5-layer forward network with a structure of [784, 500, 500, 200, 10], a training set of 60,000 scale, step size $h = 0.2$, and weight decay coefficient $\lambda = 0.001$. After 60 iterations, the network achieves 87.73% accuracy on the 10,000-scale test set. We record the weight
Figure 5: the error curve and its description statistics (Mean\_ndata)

(a) The error curve (Mean\_ndata)  

(b) The mean and standard deviation of error (Mean\_ndata)

| OPTIMIZER | Mean\_ndata |
|-----------|-------------|
| $\tilde{G}_{mast}$ | **18.62 ± 39.1** |
| Batch | 47.55 ± 57.71 |
| SGD | 228.21 ± 475.61 |
| $\tilde{G}_{st}$ | 37.56 ± 52.93 |

Figure 6: the error curve and its description statistics (Var\_ndata)

(a) The error curve (Var\_ndata)  

(b) The mean and standard deviation of error (Var\_ndata)

| OPTIMIZER | Var\_ndata |
|-----------|-------------|
| $\tilde{G}_{mast}$ | **27.57 ± 36.7** |
| Batch | 82.77 ± 123.53 |
| SGD | 183.26 ± 271.74 |
| $\tilde{G}_{st}$ | 78.51 ± 97.83 |

Figure 7: the error curve and its description statistics (Var\_ndata)

(a) The error curve (Var\_ndata)  

(b) The mean and standard deviation of error (Var\_ndata)

| OPTIMIZER | Var\_ndata |
|-----------|-------------|
| $\tilde{G}_{mast}$ | **18.26 ± 32.05** |
| Batch | 74.04 ± 108.15 |
| SGD | 307.69 ± 611.96 |
| $\tilde{G}_{st}$ | 59.73 ± 90.98 |
between the first neuron in the output layer and the first neuron in the penultimate layer as the gradient information about 60,000 training samples in 60 iterations, forming a gradient matrix with a scale of 60,000 × 60. The average value of each column of the gradient matrix is the true gradient direction of the parameter update.

On the 60,000 × 60 gradient matrix, we calculate the expected $E_j$ and variance $V_j$ of category $j$, then we calculate $p_j, q_j$ according to formulae (6), where $j$ runs from 0 to 9. Finally, the value of the $\bar{G}_{mst}$ can be calculated using these parameters.

For the sake of comparability, except that SGD uses a single sample, the sample sizes of Batch, $\bar{G}_{mst}, \bar{G}_{st}$ are all set to 10. Among them, $\bar{G}_{mst}$ and $\bar{G}_{st}$ randomly select a single sample from each of the 10 categories and Batch randomly selects 10 samples from the 60,000 samples.

The blue curve in Figure 8 represents the true gradient direction (denoted by “Pop” in Figure 8), which is the average gradient sequence generated by the full gradient method after 60 iterations. The red curve in each part stands for the curve formed by the gradient sequence generated after 60 iterations of different algorithms. The four graphics show the subtle differences in tracking the direction of the blue curve by the gradient curves generated by different methods.

In order to further investigate the accuracy difference of the four methods of Batch, SGD, $\bar{G}_{mst}, \bar{G}_{st}$, we calculate and record the deviation square of the random gradient direction generated by the four methods and the true gradient direction. Thus, each method obtains 60 such deviation squares, and the average and standard deviation of these 60 deviation squares are used to measure the performance of the algorithm.

Each algorithm is run repeatedly for 10 times, and the deviation square values generated are recorded. It can be seen from Figure 9 that even in the case of a small sampling ratio $f = \frac{10}{60,000}$, the search direction provided by $\bar{G}_{mst}$ is closer to the true value than the other methods (the mean and standard deviation of the deviation square are the smallest).
Figure 8: Tracing curves of four different estimators to true values

Figure 9: the error curve and its description statistics (mnist)
we further compare the performance of the four algorithms Batch, SGD, MSTGD, and $\tilde{G}_{st}$ in optimizing a 5-layer forward network $[784, 500, 500, 200, 10]$. The data used is still the 60,000-scale MNIST training set and the 10,000-scale MNIST test set. But to be fair, we first implement a grid search procedure for the optimal hyperparameters in the range of $h=[0.01,1,0.001]$, $\lambda=[0.001,0.0001]$, then examine the training and test accuracy differences of the four algorithms after 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10 thousand iterations under their respective optimal hyperparameters.

Table 1 shows the training and testing accuracy achieved by the four algorithms Batch, SGD, MSTGD, $\tilde{G}_{st}$ under their respective optimal hyperparameters. As can be seen from Table 1, except for 7 and 9 thousand iterations, the training and the test accuracy of $\tilde{G}_{mst}$ are the best, outperform than the other four algorithms.

Table 1: the training and testing accuracy achieved by Batch, SGD, MSTGD, $\tilde{G}_{st}$

| Iterations $(10^3)$ | SGD(%) | MSTGD(%) | Batch(%) | $\tilde{G}_{st}$(%) |
|--------------------|--------|----------|----------|------------------|
|                    | test accu | train accu | test accu | train accu | test accu | train accu | test accu | train accu |
| 1(SGD: x20)        | 91.48   | 91.39    | 94.46     | 94.56       | 93.31     | 93.38      | 94.12     | 94.28      |
| 2(SGD: x20)        | 92.87   | 92.94    | 96.05     | 96.37       | 95.52     | 95.88      | 95.92     | 96.18      |
| 3(SGD: x20)        | 93.83   | 94.08    | 96.53     | 96.93       | 96.35     | 96.86      | 96.17     | 96.42      |
| 4(SGD: x20)        | 94.57   | 94.76    | 96.74     | 97.39       | 96.61     | 97.25      | 96.44     | 96.78      |
| 5(SGD: x20)        | 94.70   | 94.79    | 97.17     | 97.98       | 97.12     | 97.81      | 96.65     | 97.18      |
| 6(SGD: x20)        | 95.26   | 95.8     | 97.3      | 98.11       | 97.16     | 97.91      | 96.94     | 97.39      |
| 7(SGD: x20)        | 95.05   | 95.7     | 97.13     | 98.19       | 97.23     | 97.89      | 96.64     | 97.22      |
| 8(SGD: x20)        | 95.55   | 95.82    | 97.65     | 98.4        | 97.43     | 98.33      | 96.62     | 97.09      |
| 9(SGD: x20)        | 95.66   | 96.24    | 97.41     | 98.41       | 97.72     | 98.49      | 96.92     | 97.42      |
| 10(SGD: x20)       | 95.44   | 95.9     | 97.63     | 98.81       | 97.4      | 98.45      | 97.31     | 97.81      |

5.4. Results on Convergence rate

An author of this paper independently tested the convergence performance of MSTGD on BP feedforward networks and convolutional networks using a four-layer feedforward network of $[784, 64, 32, 10]$, and the form of Fig. 10 Convolutional Neural Networks.
To be fair, whether it is a BP forward network or a CNN convolutional network, during training, except for SGD, the Batchsize of the other three algorithms is 20. Also to be fair, except for SGD, the other three algorithms are the test accuracy recorded after every 1000 iterations, while SGD is the test accuracy after every $20 \times 1000 = 20,000$ iterations. For each algorithm, the network was trained 14 times with the MNIST dataset, and the average test accuracy of the 14 times was recorded.

Figure 10: The convolutional neural network structure used in this paper to test the convergence performance

Figure 11 is the change curve of the average test accuracy of the four algorithms MSTGD, SGD, Batch, $\tilde{G}_{st}$ when training the BP forward network for 14 times. As can be seen from Figure 11 compared with other algorithms, MSTGD can obtain the best test accuracy under the same number of iterations. This shows that MSTGD has a faster convergence rate when training the forward network with MNIST data.

Figure 12 is the change curve of the average test accuracy of the four algorithms MSTGD, SGD, Batch, $\tilde{G}_{st}$ when training the convolutional network for 14 times. It can also be seen that, the accuracy improvement of MSTGD significantly requires fewer iterations than other algorithms. This shows that MSTGD also exhibits a faster convergence rate when training convolutional
networks with MNIST data.

From the above experimental results, MSTGD indeed exhibits a fast convergence rate, which is consistent with the theoretical results of Theorem 5 on the linear convergence of MSTGD in this paper.

6. Comparison of related works

The algorithm MSTGD in this paper uses historical gradient and stratified sampling for variance reduction, so that the algorithm can achieve a linear convergence rate under a constant step size. Section 6.1 compares MSTGD with other algorithms such as SAG, SVRG, SAGA which use historical gradient and claim to achieve a linear convergence rate. Section 6.2 compares MSTGD with other algorithms using different sampling strategies. Section 6.3 compares MSTGD with other classic algorithms using historical gradient strategies.
6.1. Comparison of Related Algorithms for Variance Reduction Based on Historical Gradients

SAG, SAGA use a same storage structure $G$ to record historical gradient information. Each time a random sample $i (i \in \{1, 2, \cdots, N\})$ is selected, the current gradient $J'_i(W_k)$ is calculated and replaced with the old gradient value $J'_i(W_{old})$ in the $i^{th}$ component in $G$. This means that SAG, SAGA both need to save $N$ stochastic gradient vectors, which is a lot of storage overhead in the case of massive training samples. MSTGD only needs to save $C$ ($C$ is the number of categories, $C \ll N$) stochastic gradient vectors.

SAG, SAGA have slightly different variance reduction strategies designed around the historical gradient in $G$. This difference results in that the gradient update direction of SAG is not an unbiased estimate of the full gradient direction, while the gradient update direction of SAGA and MSTGD are both unbiased estimates of the full gradient direction.

Zhang Tong et al. proposed a progressively stochastic gradient variance reduction algorithm SVRG\cite{15} and its extension Prox-SVRG\cite{29}. The core
idea of their algorithm is to calculate and record the full gradient $\nabla J(\tilde{W}) = \frac{1}{N} \sum_{i=1}^{N} J_i'(\tilde{W})$ on a reference network $\tilde{W}$ of the outer loop, and then use this $\nabla J(\tilde{W})$ to reduce the variance of the stochastic gradient in the inner loop that performs parameter update. Based on this idea, SVRG, Prox-SVRG can achieve effective variance reduction while maintaining unbiasedness, and have been theoretically proved to obtain a linear convergence rate.

The variance reduction strategies of SAG, SAGA, SVRG are similar. They all calculate and save a full gradient $\nabla J = \frac{1}{N} \sum_{i=1}^{N} J_i'$ in advance, and then use the recorded full gradient $\nabla J$ for variance reduction during the iterative parameter update process. This is essentially different from p-based variance reduction strategy of MSTGD.

6.2. Comparison of Related Algorithms for Variance Reduction Based on Sampling Strategy

In order to achieve a better stochastic gradient variance reduction and accelerate algorithm Convergence, Zhang Tong et al. proposed an importance scoring scheme that assigns weight $Pr^k_i$ to sample $i$ in iteration $k$, and then performs importance sampling according to distribution $Pr^k_i$, replacing the original uniform sampling when selecting a random sample[31, 32]. From their experimental results, compared with uniform sampling, although the importance sampling strategy shows obvious advantages in the decline rate of the objective function value, there is no obvious improvement in the test accuracy of the trained model.

Fartash Faghri et al. studied the gradient distribution of deep learning. They proposed an idea of gradient clustering and stratified sampling according to the categories obtained by the clustering[30]. The method developed by Fartash Faghri et al. can find the optimal clustering scheme that minimizes the variance of the stratified mean of gradients. However, the work of Fartash Faghri et al. give neither further information on training the deep network based on the obtained optimal stratified mean of gradients, nor final conclusions about the effect of their variance scheme on the test accuracy of the network.
Different from the above, MSTGD directly uses the category information in the supervision signal \( Y \) to perform stratified sampling without any additional knowledge.

6.3. Comparison of other classical algorithms

From the update formula \( v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta) \) and \( \theta = \theta - v_t \) of the Momentum optimization\[11\], the gradient direction required for parameter update in the Momentum optimization is obtained as the weighted sum of the current gradient \( \nabla_\theta J(\theta) \) and the historical gradient information stored in \( v_{t-1} \) with weights \( \eta \) and \( \gamma \). Here, \( \eta \) and \( \gamma \) are similar to the parameters \( q \) and \( p \), and \( v_t \) corresponds to the auxiliary variable \( G \) in our model. However, the unbiasedness of \( v_t \) has not been discussed. The results of this paper show that if \( \eta \) and \( \gamma \) satisfy the conditions in formula 6 as \( q \) and \( p \), the search direction provided by Momentum optimization satisfies unbiasedness.

Xie et al.\[35\] propose a Positive-Negative Momentum (PNM) approach. Similar to the traditional Momentum method, their work does not discuss the influence of the choice of Momentum coefficients on unbiasedness, and their work mainly focuses on how to simulate the noise in the stochastic gradient to enhance the generalization ability of the network.

The popular algorithm of Adam \[20\] for training deep models is of gradient update formula \( m_t = \beta_1 m_{t-1} + (1 - \beta_1)g_t \) and \( \hat{m}_t = \frac{m_t}{1 - \beta_1} \). The parameters \( \beta_1 \) and \( (1 - \beta_1) \) therein are respectively equivalent to \( p \) and \( q \). According to the results of the current paper, one of the prerequisites for the effectiveness of Adam’s coefficients \( \beta_1 \) and \( (1 - \beta_1) \) of \( m_{t-1}, g_t \) is to ensure that the mean value of the gradients before and after the iterations are equal. Generally, the equal-mean properties of the gradient before and after the iteration are generally not satisfied, unless additional restrictions are introduced. Therefore, the author of the Adam algorithm posed a so-called unbiased correction \( \hat{m}_t = \frac{m_t}{1 - \beta_1} \) on \( m_t \). Obviously, this is an empirical correction formulae, the theoretical basis behind the correction is not fully understood. In fact, the revised estimator must be a biased estimator, which is contrary to the original intention of the proponent.
Other variance reduction methods\cite{33} that perform \(k\)-step averaging on historical trajectories are equivalent to the method in this paper with \(p = \frac{k-1}{k}\) and \(q = \frac{1}{k}\). However, without an additional strategy to ensure equal gradient mean between different iterations, the unbiasedness of this approach cannot be satisfied, and the effect of the algorithm will be difficult to guarantee.

7. Appendix

In this Appendix, we present the proofs of main conclusions in this paper.

7.1. Proof of Theorem 1

Proof. In order to ensure that the starting point \(\bar{G}_{mst}^1\) of the sequence \(\{\bar{G}_{mst}^k\}\) is an unbiased estimate of \(\bar{G}^1\), without loss of generality, we randomly select a sample from each category \(j \in \{1, 2, \cdots C\}\), calculate its gradient, fill the \(G\) vector and calculate \(\bar{G}_{mst}^1\) accordingly. Now, at the beginning, we have \(\bar{G}_{mst}^1 = \bar{G}_{st}^1 = \frac{1}{C} \sum_{j=1}^{C} w_j G_j^1\).

Taking the expectation of \(\bar{G}_{mst}^1\), we have \(E(\bar{G}_{mst}^1) = E(\bar{G}_{st}^1) = E\left(\frac{1}{C} \sum_{j=1}^{C} w_j G_j^1\right) = \frac{1}{C} \sum_{j=1}^{C} w_j E(G_j^1) = \bar{G}^1\). Thus the unbiasedness of the starting point of the gradient sequence is established.

Similarly, for the \(k\)th iteration, we have

\[
E(\bar{G}_{mst}^k) = E\left(\sum_{j=1}^{C} w_j G_j^k\right) = \sum_{j=1}^{C} w_j E(G_j^k) = \sum_{j=1}^{C} w_j [p_j^k G_j^{k-1} + q_j^k g(W_k, \xi_j^k)]
\]

\[
= \sum_{j=1}^{C} w_j [p_j^k E^{k-1}_j + q_j^k E^{k}_j]
\]

\[
= \sum_{j=1}^{C} w_j [(1 - q_j^k)E^{k}_j + q_j^k E^{k}_j]
\]

\[
= \sum_{j=1}^{C} w_j E^k_j = G^k,
\]

and this concludes the proof. \(\square\)
7.2. Proof of Lemma 1

Proof. We already know that \( \bar{G}_{k \text{ mst}} = \sum_j w_j G_k^j = \sum_j [w_j (p_k^j G_k^{j-1} + q_k^j g(W_k^j, \xi_k^j))] \), and the stochastic gradient \( g(W_k^j, \xi_k^j) \) generated by independent sample and the historical gradient \( G_k^{j-1} \) in the auxiliary storage are independent of each other. Therefore, the variance of \( \bar{G}_{k \text{ mst}} \) is:

\[
V(\bar{G}_{k \text{ mst}}) = \sum_j [w_j^2 ((p_k^j)^2 V_k^{j-1} + (q_k^j)^2 V(g(W_k^j, \xi_k^j)))]
\]

Since \( \bar{G}_{k \text{ mst}} \) is an unbiased estimator, substituting the unbiased condition \( \frac{p_j}{1-q_j} = \frac{E_{k}^j}{E_{k-1}^j} \) in Theorem into the variable \( z \) in formulae (21) leads to:

\[
z = (1 - q_k^j)^2 \left( \frac{E_k^j}{E_{k-1}^j} \right)^2 V_k^{j-1} + (q_k^j)^2 V_k^j.
\]

When \( p_k^j, q_k^j \) take the values according to formulae (6), \( z = \frac{(E_k^j)^2 V_k^{j-1} - V_k^j}{(E_k^j)^2 V_k^{j-1} + (E_k^j)^2 V_k^j} \). Substituting it back into Equation (21) we get the final variance expression in the form of Equation (5). We conclude the proof.

7.3. Proof of Corollary 1

Proof. We deduce (1) from the equivalent transformation of Equation (5). Setting \( \rho_j = \frac{p_j}{1-q_j} = \frac{E_k^j}{E_{k-1}^j} < 1 \) and taking \( \rho = \max_{j} \rho_j, j \in \{1, \cdots, C\} \), we have:

\[
V_{sp}(\bar{G}_{k \text{ mst}}) = \sum_j [w_j^2 \left( \frac{(E_k^j)^2 V_k^{j-1} - V_k^j}{(E_k^j)^2 V_k^{j-1} + (E_k^j)^2 V_k^j} \right)] = \sum_j [w_j^2 \left( \frac{(E_k^j)^2 V_k^{j-1} - V_k^j}{(E_k^j)^2 V_k^{j-1} + (E_k^j)^2 V_k^j} \right)]
\]

Therefore, property (1) holds.

Next we prove (2) by induction.
Let

\[ p = \max_{j \in \{1, \ldots, C\}} p_j^k, \quad \text{and} \quad q = \max_{j \in \{1, \ldots, C\}} q_j^k. \]

When \( k = 1 \), \( \bar{G}^1_{\text{mst}} = \bar{G}^1_{st} \) by Theorem 1. And according to formulae (2), we have

\[ \bar{G}^2_{\text{mst}} = \sum_{j=1}^{C} w_j (p_j^2 \cdot G^1_j + q_j^2 \cdot g(W_2, \xi^2_j)) \]

\[ = \sum_{j=1}^{C} w_j (p_j^2 \cdot G^1_j) + \sum_{j=1}^{C} w_j (q_j^2 \cdot g(W_2, \xi^2_j)) \]

\[ \leq p \cdot \sum_{j=1}^{C} w_j G^1_j + q \cdot \sum_{j=1}^{C} w_j g(W_2, \xi^2_j) \]

\[ = p \cdot \bar{G}^1_{\text{mst}} + q \cdot \bar{G}^2_{st}. \]

Let \( p_j^2 \) and \( q_j^2 \) be given by formulae (6), we get

\[ V_{sp}(\bar{G}^2_{\text{mst}}) = V[\sum_{j=1}^{C} w_j (p_j^2 \cdot G^1_j)] + V[\sum_{j=1}^{C} w_j (q_j^2 \cdot g(W_2, \xi^2_j))] \]

\[ \leq p^2 \cdot V[\sum_{j=1}^{C} w_j G^1_j] + q^2 \cdot V[\sum_{j=1}^{C} w_j g(W_2, \xi^2_j)] \]

\[ = p^2 \cdot V(\bar{G}^1_{\text{mst}}) + q^2 \cdot V(\bar{G}^2_{st}), \]

which means (2) holds when \( k = 1, t = 1 \).

Similar to (23) and (24) we can verify that \( V(\bar{G}^{k+1}_{\text{mst}}) \leq p^2 V(\bar{G}^k_{\text{mst}}) + q^2 V(\bar{G}^{k+1}_{st}) \), that is, (2) is valid when \( t = 1 \) and \( k \) takes any value.

Now assume that when \( t = n \) and \( k \) takes an arbitrary value, (2) holds. Below we show that when \( t = n + 1 \) and \( k \) remains unchanged, (2) still holds.

\[ V_{sp}(\bar{G}^{k+n+1}_{\text{mst}}) \leq p^2 \cdot V(\bar{G}^{k+n}_{\text{mst}}) + q^2 \cdot V(\bar{G}^{k+n+1}_{st}) \]

\[ \leq p^2 \cdot [p^{2n} V(\bar{G}^k_{\text{mst}}) + \sum_{i=1}^{n} p^{2(n-i)} q^2 V(\bar{G}^{k+i}_{st})] + q^2 \cdot V(\bar{G}^{k+n+1}_{st}) \]

\[ = p^{2(n+1)} V(\bar{G}^k_{\text{mst}}) + p^2 \sum_{i=1}^{n} p^{2(n-i)} q^2 V(\bar{G}^{k+i}_{st}) + q^2 \cdot V(\bar{G}^{k+n+1}_{st}) \]

\[ = p^{2(n+1)} V(\bar{G}^k_{\text{mst}}) + \sum_{i=1}^{n+1} p^{2(n+1-i)} q^2 V(\bar{G}^{k+i}_{st}). \]

(25)

In summary, (2) holds for any \( k \) and \( t \), and the proof is complete. \( \square \)
7.4. Proof of Lemma 2

Proof. Divide both sides of inequality (2) by $\eta^2t$ and multiply by $\gamma^{2k_0}$ to get a new equivalent inequality

$$\left(1 + \frac{1}{1-\eta}\right)\gamma^{2k_0} \leq \left(\frac{\gamma}{\eta}\right)^{2k}. \quad (26)$$

Let $0 < \eta < \gamma < 1$, then $\frac{\gamma}{\eta} > 1$. Given the values of $\gamma, \eta, k_0$, the left side of inequality (26) is a constant, and the right side is a function about the exponential growth of $2k$. In the case of sufficiently large $k$, inequality (26) is obviously established, so inequality (2) is also established.

7.5. Proof of Theorem 2

Proof. Let $p = \max_{j \in \{1, \cdots, C\}, k \in \{1,2,\cdots\}} p^k_j$. In the following two case, $p \leq 1$ holds.

When $E^k_j = E^k_{j-1}$, according to Equation (6) and the fact $(E^k_j)^2V^{k-1}_j \geq 0$, we have $p^k_j \leq 1$. So $p \leq 1$.

Further let $0 = 1$. When $E^k_j = E^k_{j-1} = 0$, according to (9) and the fact $V^{k-1}_j \geq 0$, we have $p^k_j = \frac{e^{\frac{E^k_j}{V_j^k}}}{\frac{e^{\frac{E^k_j}{V_j^k}}}{V_j^k} + 1} = \frac{V_j^k}{V_j^k + 1} \leq 1$. Also $p \leq 1$ holds. We know

$$V_{sp}(\tilde{G}_{mst}) = \left(\begin{array}{cccc}
\sigma_1^2 & \sigma_2^2 & \cdots & \\
\sigma_1^2 & \cdots & \\
\sigma_2^2 & \cdots & \\
\sigma_{p}^2 & \cdots & 
\end{array}\right), \quad V(\tilde{G}_{st}) = \left(\begin{array}{cccc}
x_{1i}^2 & x_{2i}^2 & \cdots & \\
x_{1i}^2 & \cdots & \\
x_{2i}^2 & \cdots & \\
x_{pi}^2 & \cdots & 
\end{array}\right)$$

Let $\rho = \max(p, q), M = \max\{\sigma_1^2, \sigma_2^2, \cdots, \sigma_{p}^2, \Sigma_{1i}, \Sigma_{2i}, \cdots, \Sigma_{pi}^2 \}$. When
$p \leq 1$, according to property (2) in Corollary(1), we have
\begin{align*}
V_{sp}(\bar{G}_{mst}^k) &\leq p^{2k}V(G_{mst}) + \sum_{i=1}^{t} p^{2(k-i)}q^{2}V(\bar{G}_{i}^k) \\
&\leq p^{2k}M \cdot I + \sum_{i=1}^{t} p^{2(k-i)}q^{2}M \cdot I \\
&= p^{2k}M \cdot I + \frac{1}{1-pq}q^{2}M \cdot I \\
&\leq (1 + \frac{1}{1-pq})p^{2k}M \cdot I \\
&\leq (1 + \frac{1}{1-pq})p^{2k}M \cdot I \\
&\leq (1 + \frac{1}{1-pq})p^{2k}M \cdot I \\
&\leq \gamma^{2(k-k_0)}M \cdot I.
\end{align*}
(27)

The equation of step 3 in formulae (27) is established because the proportional sequence in the previous formula is summed, and the last inequality is established because of Lemma (2). The proof is concluded.

7.6. Proof of Theorem 3

Proof. Let $\sigma_i^2 = E[\bar{G}_{mst}^k(w_i)]^2 - [E(\bar{G}_{mst}^k(w_i))]^2$. According to the definitions of $V_{sp}(\bar{G}_{mst}^k), Var(\bar{G}_{mst}^k)$, we know
\begin{align*}
V_{sp}(\bar{G}_{mst}) &= \begin{pmatrix}
\sigma_1^2 & \ \ & \ \ \\
\sigma_2^2 & \ \ & \ \ \\
\sigma_\phi^2 & \ \ & \ \ \\
\end{pmatrix},
Var(\bar{G}_{mst}^k) = \sum_{i=1}^{\ell} \sigma_i^2,
\end{align*}
i.e. $Var(\bar{G}_{mst}^k)$ is the trace of $V_{sp}(\bar{G}_{mst})$. Applying Theorem (2), we have
\begin{align*}
Var(\bar{G}_{mst}^k) &= \sum_{i=1}^{\ell} \sigma_i^2 \\
&= tr(V_{sp}(\bar{G}_{mst})) \\
&\leq \gamma^{2(k-k_0)}F \Phi \\
&= \gamma^{2(k-k_0)}F.
\end{align*}
(28)

This conclude the proof. □
7.7. Proof of Inequality \([14]\)

Proof. For a quadratic model

\[ f(\bar{W}) = J(W) + \nabla J(W)^T (\bar{W} - W) + \frac{1}{2} ||\bar{W} - W||_2^2 \]

with respect to \(W \in \mathbb{R}^p\), its unique minima is \(\bar{W}_* = W - \frac{1}{\varepsilon} \nabla J(W)\) and minimal value \(f(\bar{W}_*) = J(W) - \frac{1}{2\varepsilon} ||\nabla J(W)||_2^2\).

Taking \(W' = W_*\) for \([13]\) will have \(J_* \leq J(W) + \nabla J(W)^T (W_* - W) + \frac{1}{2} \varepsilon ||W_* - W||_2^2 \leq J(W) - \frac{1}{2} \varepsilon ||\nabla J(W)||_2^2\), so inequality \([14]\) holds.

\[ \Box \]

7.8. Proof of Theorem \([4]\)

Proof. According to assumption \(A_1\), we have

\[
J(W_{k+1}) = J(W_k) + \int_0^1 \frac{\partial J(W_k + t(W_{k+1} - W_k))}{\partial t} dt \\
= J(W_k) + \int_0^1 \nabla J(W_k + t(W_{k+1} - W_k))^T (W_{k+1} - W_k) dt \\
= J(W_k) + \nabla J(W_k)^T (W_{k+1} - W_k) + \int_0^1 \nabla J(W_k + t(W_{k+1} - W_k)) - \nabla J(W_k)^T (W_{k+1} - W_k) dt \\
\leq J(W_k) + \nabla J(W_k)^T (W_{k+1} - W_k) + \int_0^1 L ||t(W_{k+1} - W_k)||_2 ||W_{k+1} - W_k||_2 dt \\
= J(W_k) + \nabla J(W_k)^T (W_{k+1} - W_k) + \frac{1}{2} ||W_{k+1} - W_k||_2^2.
\]

This leads to

\[
J(W_{k+1}) - J(W_k) \leq \nabla J(W_k)^T (W_{k+1} - W_k) + \frac{1}{2} L ||W_{k+1} - W_k||_2^2. \tag{29}
\]

Substituting formulae \([4]\) into the above inequality and taking expectation on both sides, we have

\[
E[J(W^{k+1})] - J(W_k) \leq -h_k \nabla J_N(W_k)^T E[\bar{g}] + \frac{1}{2} L E[||\bar{g}||_2^2] h_k^2 \\
= -h_k \nabla J_N(W_k)^T E[\bar{g}] + \frac{1}{2} L (s_k^2 + E[||\bar{g}||_2^2]) h_k^2 \\
= -h_k \nabla J_N(W_k)^T \nabla J_N(W_k) + \frac{1}{2} L (s_k^2 + \nabla J_N(W_k)^2) h_k^2 \\
= \left(\frac{h_k L}{2} - h_k\right) ||\nabla J_N(W_k)||_2^2 + \frac{h_k^2 L s_k^2}{2}.
\tag{30}
\]

The first equation in the above formula holds because of the known conclusion that \(s_k^2 = Var(\bar{g}) = E[||\bar{g}||_2^2] - ||E(\bar{g})||_2^2\), and the second equation holds because the expectation of the sample mean is equal to the population mean, that is, \(E[\bar{g}] = \nabla J_N(W_k)\).
With the condition \( h_k < \frac{2}{L} \) and the fact \( (14) \), formula \( (30) \) will be changed as

\[
E[J(W^{k+1})] - J(W^k) \leq \left( \frac{h_k^2 L}{2} - h_k \right) \| \nabla J_N(W^k) \|_2^2 + \frac{h_k^2 L s_k^2}{2} \\
\leq \left( \frac{h_k L}{2} - 1 \right) 2h_k c (J(W^k) - J^*) + \frac{h_k^2 L s_k^2}{2} .
\]

Subtracting \( J^* \) on both sides of the above inequality, taking expectation and reordering it, we have

\[
E[J(W^{k+1})] - J^* \leq (h_k^2 c L - 2h_k c + 1) E[J(W^k) - J^*] + \frac{h_k^2 L s_k^2}{2} .
\]

Let \( e_k = E[J(W^k) - J^*] \), \( \Lambda_k = \frac{h_k L s_k^2}{2c(2-h_k L)} (c \neq 0, h_k L \neq 2) \), \( \rho_i = (h_i^2 c L - 2h_i c + 1) < 1 \), \( \rho = \max_{i=1,\cdots,k} \{ \rho_i \} \), then

\[
E[J(W^{k+1})] - J_k - \Lambda_k \leq (h_k^2 c L - 2h_k c + 1) (E[J(W^k) - J^*] - \Lambda_k) , \quad \quad (31)
\]

\[
e_{k+1} - \Lambda_k \leq \rho_k (e_k - \Lambda_k) .
\]

In addition, let \( \sigma_{\min}^2 = \min\{ \sigma_1^2, \sigma_2^2, \cdots, \sigma_k^2, \cdots \} \), \( \sigma_{\max}^2 = \max\{ \sigma_1^2, \sigma_2^2, \cdots, \sigma_k^2, \cdots \} \), we can carefully select the ratio of sample size \( \frac{n_{k+1}}{n_k} \) in adjacent iterations such that it satisfies the following inequality

\[
\frac{n_{k+1}}{n_k} \geq r^2 \frac{h_{k+1}}{h_k} \frac{2-h_k L}{2-h_{k+1} L} , \quad (32)
\]

where \( r^2 = \frac{\sigma_{\min}^2}{\sigma_{\max}^2} \leq 1 \). When \( h_k \to 0, \forall k = 1,2,\cdots \), formulae \( (32) \) can be simplified as \( \frac{n_{k+1}}{n_k} \geq r^2 \), this is a weak and easily satisfied condition.

We can verify formulae \( (32) \) leads to \( \Lambda_{k-1} - \Lambda_k > 0 \).

Applying the inequality \( e_{k+1} - \Lambda_k \leq \rho_k (e_k - \Lambda_k) \) recursively, with the guar-
antee of formulae (32), formulae (31) will be changed as 

\[ e_{k+1} - \Lambda_k \leq \rho_k (e_k - \Lambda_{k-1} + \Lambda_k) \]

\[ \leq \rho_k [\rho_{k-1} (e_{k-1} - \Lambda_{k-1}) + \Lambda_{k-1} - \Lambda_k] = \rho_k \rho_{k-1} (e_{k-1} - \Lambda_{k-1}) + \rho_k (\Lambda_{k-1} - \Lambda_k) \]

\[ \leq \rho_k \rho_{k-1} \rho_{k-2} (e_{k-2} - \Lambda_{k-2}) + \rho_k \rho_{k-1} (\Lambda_{k-2} - \Lambda_{k-1}) + \rho_k (\Lambda_{k-1} - \Lambda_k) \]

\[ \ldots \]

\[ \leq (e_1 - \Lambda_1) \prod_{i=1}^k \rho_i + (\Lambda_1 - \Lambda_2) \prod_{i=2}^k \rho_i + (\Lambda_2 - \Lambda_3) \prod_{i=3}^k \rho_i + \cdots + (\Lambda_{k-2} - \Lambda_{k-1}) \prod_{i=k-1}^k \rho_i + (\Lambda_{k-1} - \Lambda_k) \rho_k \]

\[ \leq (e_1 - \Lambda_1) \prod_{i=1}^k \rho_i + \sum_{j=2}^k [(\Lambda_{j-1} - \Lambda_j) \prod_{i=j}^k \rho_i]. \]  

(33)

Formulae (33) shows that the difference of adjacent iterations \( \Lambda_{j-1} - \Lambda_j \) decays at a rate of \( \prod_{i=j}^k \rho_i \) during the iteration.

Considering an extreme case, the difference of adjacent iterations \( \Lambda_{j-1} - \Lambda_j \) will cease decaying by removing the coefficient \( \prod_{i=j}^k \rho_i \) from the term \( (\Lambda_{j-1} - \Lambda_j) \prod_{i=j}^k \rho_i \), and let \( \rho = \max \rho_i \), we have

\[ e_{k+1} \leq (e_1 - \Lambda_1) \prod_{i=1}^k \rho_i + (\Lambda_1 - \Lambda_2) + \cdots + (\Lambda_{k-1} - \Lambda_k) + \Lambda_k \]

\[ < \Lambda_1 + (e_1 - \Lambda_1) \prod_{i=1}^k \rho_i \]

\[ \leq \Lambda_1 + \rho^k (e_1 - \Lambda_1). \]  

(34)

Taking \( e_k = E[J(W) - J_*] \) back into above inequality, the final result of Theorem 4 is derived. \( \square \)

### 7.9. Proof of Theorem 5

**Proof.** The step size in this proof is a constant step size \( \bar{h} \). Substituting \( W_{k+1} = W_k - \bar{h} G^k_{\text{mst}} \) (formulae 2) into inequality (29), we get

\[ J(W_{k+1}) - J(W_k) \leq \nabla J(W_k)^T (W_{k+1} - W_k) + \frac{1}{2} L ||W_{k+1} - W_k||^2 \]

\[ \leq -\bar{h} \nabla J(W_k)^T \bar{G}^k_{\text{mst}} + \frac{1}{2} \bar{h}^2 L ||\bar{G}^k_{\text{mst}}||^2. \]
Since $E(G_{mst}^k) = G^k = \nabla J(W_k)$, substituting \[2\] into the above inequality, and taking the expectation on both sides, the above inequality can be transformed into the following form

$$E(J(W_{k+1}) - J(W_k) \leq -h\nabla J(W_k)^T E(G_{mst}^k) + \frac{1}{2}h^2 LE||G_{mst}^k||^2 \]

$$= -h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 LE||G_{mst}^k||^2.$$

Assuming $\mu_G \geq \mu \geq 0$, such that for all $k \in N$ , $\nabla J(W_k)^T E(G_{mst}^k) \geq \mu||\nabla J(W_k)||^2$, and $||E(G_{mst}^k)||_2 \leq \mu_G||\nabla J(W_k)||_2$, the above inequality can be further transformed into the following form

$$E(J(W_{k+1}) - J(W_k) \leq -\mu h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 LE||G_{mst}^k||^2.$$

Let $Var(G_{mst}^k) := E(||G_{mst}^k||^2) - ||E(G_{mst}^k)||^2$. Applying $||E(G_{mst}^k)||_2 \leq \mu_G||\nabla J(W_k)||_2$, $Var(G_{mst}^k) \leq \gamma^{2(k-k_0)} \Phi$(Theorem 3), $0 < h \leq \min\{\frac{\mu}{\mu_G}, \frac{1}{2\gamma}\}$, we have

$$E(J(W_{k+1})) - J(W_k) \leq -\mu h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 L(\mu_G^2 ||\nabla J(W_k)||^2 + \Phi \gamma^{2(k-k_0)})

\leq -(\mu - \frac{1}{2}L(\mu_G^2))h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 L\Phi \gamma^{2(k-k_0)}

\leq -\frac{1}{2}\mu h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 L\Phi \gamma^{2(k-k_0)}.$$

Under the strong convex assumption and the fact \[14\], we further have

$$E(J(W_{k+1})) - J(W_k) \leq \frac{1}{2}\mu h||\nabla J(W_k)||^2 + \frac{1}{2}h^2 L\Phi \gamma^{2(k-k_0)}

\leq -\mu c h (J(W_k) - J_*) + \frac{1}{2}h^2 L\Phi \gamma^{2(k-k_0)}.$$

Subtracting $J_*$ and adding $J(W_k)$ from both sides of the above inequality, rearranging and taking expectation on both sides, we get

$$E(J(W_{k+1})) - J_* \leq (1 - \mu c h)E[J(W_k) - J_*] + \frac{1}{2}h^2 L\Phi \gamma^{2(k-k_0)}. \tag{35}$$

Let

$$\Omega := \max\{\frac{h L \Phi}{c \mu}, J(W_{k_0}) - J_*\}

\lambda := \max\{1 - \frac{hc \mu}{2\gamma}, \gamma\} < 1$$

then \[35\] can be reformed as

$$E[J(W_k) - J_*] \leq \Omega \lambda^{2(k-k_0)},$$
which is what we expected as (18). The following is an inductive proof for the above inequality. When \( k = k_0 \), we have

\[
E(J(W_{k_0}) - J_*) \leq \Omega = \Omega \lambda^{2(k_0-k_0)}.
\]

This shows that inequality (18) holds for \( k = k_0 \).

Now suppose that when \( k \geq k_0 \), inequality (18) holds, the following process

\[
E[J(W_{k+1}) - J_*] \leq (1 - \hbar \mu)E[J(W_k) - J_*] + \frac{\hbar}{2} L \Phi \gamma^{2(k-k_0)}
\]

\[
\leq (1 - \hbar \mu)\Omega \lambda^{2(k-k_0)} + \frac{\hbar}{2} L \Phi \gamma^{2(k-k_0)}
\]

\[
= \Omega \lambda^{2(k-k_0)} (1 - \hbar \mu + \frac{\hbar^2 L \Phi}{2 \mu} (\frac{\lambda}{\gamma})^{2(k-k_0)})
\]

\[
\leq \Omega \lambda^{2(k-k_0)} (1 - \hbar \mu + \frac{\hbar^2 L \Phi}{2 \mu} (\frac{\lambda}{\gamma})^{2(k-k_0)})
\]

\[
= \Omega \lambda^{2(k-k_0)} (1 - \hbar \mu + \frac{\hbar^2 \mu}{2} \frac{\hbar \mu}{2})
\]

\[
= \Omega \lambda^{2(k-k_0)} (1 - \frac{\hbar \mu}{2})^2
\]

\[
\leq \Omega \lambda^{2(k+1-k_0)},
\]

shows that inequality (18) also holds for \( k + 1 \). The proof is concluded. \( \square \)

References

[1] Aixiang chen. Deep learning. Tsinghua University press.2020

[2] LeCun Y, Bottou L, Bengio Y and Haffner P. Gradient-Based Learning Applied to Document Recognition[J]. Proceedings of IEEE, 1998, 86(11):2278-2324.

[3] Ciresan D C, Meier U, Gambardella L M, Schmidhuber J. Deep, big, simple neural nets for handwritten digit recognition[J]. Neural Computation, 2010, 22(12): 3207-3220

[4] Ciresan D C, Meier U, Masci J, Schmidhuber J. Multi-column deep neural network for traffic sign classification[J]. Neural Networks, 2012. 32:333-338.
[5] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks [C]. Proceedings of International Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, United States, December 3-6, 2012: 1106–1114.

[6] Alex Graves, Santiago Fernandez, Faustino Gomez, Jurgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks [C]. Proceedings of the 23rd international conference on Machine learning, Pittsburgh Pennsylvania USA June 25 - 29, 2006: 369-376

[7] Graves A, Mohamed A, Hinton G E. Speech recognition with deep recurrent neural networks [C]. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, May 26-31, 2013: 6645-6649.

[8] Shillingford B, Assael Y, Hoffman M W, et al. Large-scale visual speech recognition [J]. arXiv:1807.05162

[9] Yonghui W, Schuster M, Zhifeng C, et al. Google’s neural machine translation system: bridging the gap between human and machine translation [J]. arXiv:1609.08144

[10] Herbert Robbins, Sutton Monro. A Stochastic Approximation Method. Annals of Mathematical Statistics, 22(3): 400-407, 1951.

[11] Ning Qian. On the momentum term in gradient descent learning algorithms. Neural networks. The official journal of the International Neural Network Society, 12(1): 145-151, 1999.

[12] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence o(1/k^2). Doklady ANSSR (translated as Soviet. Math. Dokl.), 269: 543-547
[13] N. Le Roux, M. Schmidt, and F. Bach. A Stochastic Gradient Method with an Exponential Convergence Rate for Finite Training Sets. Advances in Neural Information Processing System 25, pages 2672-2680, 2012.

[14] Aaron Defazio, Francis Bach, Simon Lacoste-Julien. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. NIPS, 2014.

[15] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. Advances in Neural Information Processing System 26, pages 315-323, 2013.

[16] Aixiang (Andy) Chen, Xiaolong Chai, Bingchuan Chen, Rui Bian, Qingliang Chen. A novel stochastic stratified average gradient method: Convergence rate and its complexity. in Proceedings of International Joint Conference of Neural Networks (IJCNN), July 2018 [arXiv:1710.07783V3].

[17] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research, 12:2121-2159, 2011.

[18] Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. arXiv preprint arXiv:1212.5701, 2012.

[19] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSE-ERA: Neural networks for machine learning, 4(2):26-31.

[20] Diederik P. Kingma and Jimmy Lei Ba. Adam: a Method for Stochastic Optimization. International Conference on Learning Representations, pages 1-13, 2015.

[21] Timothy Dozat. Incorporating Nesterov Momentum into Adam. ICLR Workshop, (1):2013-2016, 2016.
[22] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In Proceedings of International Conference on Learning Representations, 2018.

[23] B. T. Polyak and A. Juditsky. Acceleration of stochastic approximation by averaging. SIAM Journal on Control and Optimization, 30:838-855, 1992.

[24] Yu. Nesterov. Primal-dual subgradient methods for convex problems. Mathematical Programming, 120(1):221-259, 2009. Appeared early as CORE discussion paper 2005/67, Catholic University of Louvain, Center for Operations Research and Econometrics.

[25] Hashemi, Fatemeh & Ghosh, Soumyadip & Pasupathy, Raghu. On adaptive sampling rules for stochastic recursions. Proceedings - Winter Simulation Conference. 2015. 3959-3970. 10.1109/WSC.2014.7020221.

[26] Sashank J. Reddi and Satyen Kale and Sanjiv Kumar. On the Convergence of Adam and Beyond. Proceedings of 6th International Conference on Learning Representations, ICLR, Vancouver, BC, Canada, April 30 - May 3, 2018.

[27] Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. arXiv preprint arXiv:1212.5701, 2012.

[28] Y. Nesterov, Introductory lectures on convex optimization: A basic course, Vol. 87, Springer Science & Business Media, 2013.

[29] L. Xiao and T. Zhang, A proximal stochastic gradient method with progressive variance reduction, SIAM J. Optim., 24 (2014), pp. 2057–2075.

[30] Fartash Faghri, David Duvenaud, David J. Fleet, Jimmy Ba. A Study of Gradient Variance in Deep Learning. arXiv preprint arXiv:2007.04532, 2020.

[31] Peilin Zhao, Tong Zhang. Stochastic Optimization with Importance Sampling. arXiv preprint arXiv:1401.2753, 2014.
[32] Peilin Zhao, Tong Zhang. Stochastic Optimization with Importance Sampling for Regularized Loss Minimization. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, pp. 1-9. ACM, 2015.

[33] D. Blatt, A. O. Hero, and H. Gauchman. A convergent incremental gradient method with a constant step size. SIAM J. Optim., 18(2007), pp. 29-51, https://doi.org/10.1137/040615961.

[34] L. Bottou, F. Curtis, J. Nocedal. Optimization methods for large-scale machine learning. SIAM Rev., 6 (2) (2018), pp. 223-311, https://doi.org/10.1137/16M1080173

[35] Zeke Xie, Li Yuan, Zhanxing Zhu, Masashi Sugiyama. Positive-Negative Momentum: Manipulating Stochastic Gradient Noise to Improve Generalization. Proceedings of the 38th International Conference on Machine Learning, PMLR 139:11448-11458, 2021