Article

Training Deep Neural Networks Using Conjugate Gradient-like Methods

Hideaki Iiduka [1,*] and Yu Kobayashi [2]

[1] Department of Computer Science, Meiji University, 1-1-1 Higashimita, Tama-ku, Kawasaki-shi, Kanagawa, 214-8571 Japan; iiduka@cs.meiji.ac.jp

[2] Department of Computer Science, Meiji University, 1-1-1 Higashimita, Tama-ku, Kawasaki-shi, Kanagawa, 214-8571 Japan; yuukbys@cs.meiji.ac.jp

* Correspondence: iiduka@cs.meiji.ac.jp

Version October 28, 2020 submitted to Journal Not Specified

Abstract: The goal of this article is to train deep neural networks that accelerate useful adaptive learning rate optimization algorithms such as AdaGrad, RMSProp, Adam, and AMSGrad. To reach this goal, we devise an iterative algorithm combining the existing adaptive learning rate optimization algorithms with conjugate gradient-like methods, which are useful for constrained optimization. Convergence analyses show that the proposed algorithm with a small constant learning rate approximates a stationary point of a nonconvex optimization problem in deep learning. Furthermore, it is shown that the proposed algorithm with diminishing learning rates converges to a stationary point of the nonconvex optimization problem. The convergence and performance of the algorithm are demonstrated through numerical comparisons with the existing adaptive learning rate optimization algorithms for image and text classification. The numerical results show that the proposed algorithm with a constant learning rate is superior for training neural networks.

Keywords: adaptive learning rate optimization algorithms; conjugate gradient-like method; deep neural network; nonconvex optimization

1. Introduction

Deep neural networks are used for many tasks, such as natural language processing, computer vision, and text and image classification (see also [1–3] for applications of neural networks), and a number of algorithms have been presented to tune the model parameters of such networks. The appropriate parameters are found by solving nonconvex stochastic optimization problems. In particular, the algorithms solve these problems in order to adapt the learning rates of the model parameters. Accordingly, they are called *adaptive learning rate optimization algorithms* [4, Subchapter 8.5], and they include AdaGrad [5], RMSProp [4, Algorithm 8.5], Adam [6], and AMSGrad [7].

Recently, reference [8] preformed convergence analyses on adaptive learning rate optimization algorithms for constant learning rates and diminishing learning rates. The convergence analyses indicated that the algorithms with sufficiently small constant learning rates approximate stationary points of the problems [8, Theorem 3.1]. This implies that useful algorithms, such as Adam and AMSGrad, can use constant learning rates to solve the nonconvex stochastic optimization problems in deep learning, in contrast to the results in [6] and [7] that presented only analyses assuming the convexity conditions of objective functions for diminishing learning rates. The analyses also indicated that the algorithms with diminishing learning rates converge to stationary points of the problems and achieve a certain convergence rate [8, Theorem 3.2]. Numerical comparisons showed that the algorithms with constant learning rates perform better than the ones with diminishing learning rates.

Meanwhile, conjugate gradient methods are useful for unconstrained nonconvex deterministic optimization (see [9] for details on conjugate gradient methods). These methods use the conjugate gradient direction (see also (2) for the definition of the conjugate gradient direction with the Fletcher-Reeves formula), and they accelerate the steepest descent method. Conjugate gradient methods converge globally and generate the descent direction. In particular, the Hager-Zhang, Polak-Ribière-Polyak, and Hestenes-Stiefel methods have efficient numerical performance [9]. It seems that conjugate gradient methods could be applied to constrained optimization, because they might accelerate the existing methods for constrained optimization. However, the inconvenient possibility that the conjugate gradient methods may not converge to solutions to constrained optimization problems [10, Proposition 3.2] means that we cannot apply them directly. Actually, the numerical results in [10] showed that the conjugate gradient methods with conventional formulas, such as the Fletcher-Reeves, Polak-Ribière-Polyak, and Hestenes-Stiefel formulas, do not always converge to solutions to constrained optimization problems.

The conjugate gradient direction has been modified so that it can be applied to constrained optimization. The modified direction is called the *conjugate gradient-like direction* [10–14], and it is obtained by replacing the formula used for finding the conventional conjugate gradient direction with a positive real sequence depending on the number of iterations (see (1) for the definition of the conjugate gradient-like direction). The *conjugate gradient-like method* with the conjugate gradient-like direction can be applied to constrained convex deterministic optimization. In particular, the conjugate gradient-like method converges to solutions to constrained convex deterministic optimization problems when the step sizes (which are called learning rates) are diminishing [10, Theorem 3.1]. Moreover, the numerical results in [10] showed that it converges faster than the existing steepest descent method.

Roughly speaking, the existing adaptive learning rate optimization algorithms [4, Subchapter 8.5] are first-order methods using the steepest descent direction of an observed function at each iteration. Accordingly, using the conjugate gradient-like method would be useful to accelerate these algorithms. Hence, in this article, we propose an iterative method combining the existing adaptive learning rate optimization algorithms [4, Subchapter 8.5] with the conjugate gradient-like method [10–14].

This article provides two convergence analyses. The first analysis shows that with a small constant learning rate, the proposed algorithm approximates a stationary point of a nonconvex optimization problem in deep learning (Theorem 1). The second analysis shows that with diminishing learning rates, it converges to a stationary point of the nonconvex optimization problem (Theorem 2). The convergence and performance of the proposed algorithm are examined through numerical comparisons with the existing adaptive learning rate optimization algorithms for image and text classification. The numerical results show that the proposed algorithm with a constant learning rate is superior for training neural networks, while the one with diminishing learning rates is not good for training neural networks.

This article is organized as follows. Section 2 gives the mathematical preliminaries and states the main problem. Section 3 presents the proposed algorithm for solving the main problem and analyzes its convergence. Section 4 numerically compares the behaviors of the proposed learning algorithms with those of the existing ones. Section 5 discusses the relationship between the previously reported results and the results in Sections 3 and 4. Section 6 concludes the paper with a brief summary.

2. Mathematical Preliminaries

2.1. Notation and definitions

$\mathbb{N}$ denotes the set of all positive integers and zero. $\mathbb{R}^d$ denotes a d-dimensional Euclidean space with inner product $\langle \cdot, \cdot \rangle$, which induces the norm $\| \cdot \|$. $\mathbb{S}^d$ denotes the set of $d \times d$ symmetric matrices, i.e., $\mathbb{S}^d = \{X \in \mathbb{R}^{d \times d} \colon X = X^\top\}$. $\mathbb{S}^d_{++}$ denotes the set of $d \times d$ symmetric positive-definite matrices, i.e., $\mathbb{S}^d_{++} = \{X \in \mathbb{S}^d \colon X \succ O\}$. $\mathbb{D}^d$ denotes the set of $d \times d$ diagonal matrices, i.e., $\mathbb{D}^d = \{X \in \mathbb{R}^{d \times d} \colon X = \mathsf{diag}(x_i),\ x_i \in \mathbb{R}\ (i = 1, 2, \ldots, d)\}$. $A \odot B$ denotes the Hadamard product of matrices A and B. For all $\boldsymbol{x} := (x_i) \in \mathbb{R}^d$, we have $\boldsymbol{x} \odot \boldsymbol{x} := (x_i^2) \in \mathbb{R}^d$.

Given $H \in \mathbb{S}^d_{++}$, the H-inner product of $\mathbb{R}^d$ and the H-norm are defined for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_H := \langle \boldsymbol{x}, H\boldsymbol{y} \rangle$ and $\|\boldsymbol{x}\|_H^2 := \langle \boldsymbol{x}, H\boldsymbol{x} \rangle$.

The *metric projection* [15, Subchapter 4.2, Chapter 28] onto a nonempty, closed convex set X ($\subset \mathbb{R}^d$), denoted by P_X, is defined for all $\boldsymbol{x} \in \mathbb{R}^d$ by $P_X(\boldsymbol{x}) \in X$ and $\|\boldsymbol{x} - P_X(\boldsymbol{x})\| = \inf_{\boldsymbol{y} \in X} \|\boldsymbol{x} - \boldsymbol{y}\|$. P_X satisfies the nonexpansivity condition, i.e., $\|P_X(\boldsymbol{x}) - P_X(\boldsymbol{y})\| \leq \|\boldsymbol{x} - \boldsymbol{y}\|$ $(\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d)$, and satisfies $\mathrm{Fix}(P_X) := \{\boldsymbol{x} \in \mathbb{R}^d \colon \boldsymbol{x} = P_X(\boldsymbol{x})\} = X$ [15, Proposition 4.8, (4.8)]. The metric projection onto X under the H-norm is denoted by $P_{X,H}$. When X is an affine subspace, a half-space, or a hyperslab, the projection onto X can be computed within a finite number of arithmetic operations [15, Chapter 28].

$\mathbb{E}[X]$ denotes the expectation of a random variable X. The history of the process $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots$ up to time n is denoted by $\boldsymbol{\xi}_{[n]} = (\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n)$. For a random process $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots$, $\mathbb{E}[X | \boldsymbol{\xi}_{[n]}]$ denotes the conditional expectation of X given $\boldsymbol{\xi}_{[n]} = (\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n)$. Unless stated otherwise, all relations between random variables hold almost surely.

2.2. Stationary point problem associated with nonconvex optimization problem

Let us consider the following problem [8] (see, e.g., Subchapter 1.3.1 in [16] for details on stationary point problems):

Problem 1. *Assume that*

(A1) $X \subset \mathbb{R}^d$ *is a nonempty, closed convex set onto which the projection can be easily computed;*
(A2) $f \colon \mathbb{R}^d \to \mathbb{R}$*, which is defined for all* $\boldsymbol{x} \in \mathbb{R}^d$ *by* $f(\boldsymbol{x}) := \mathbb{E}[F(\boldsymbol{x}, \boldsymbol{\xi})]$*, is well defined, where* $F(\cdot, \boldsymbol{\xi})$ *is continuously differentiable for almost every* $\boldsymbol{\xi} \in \Xi$*, where* $\boldsymbol{\xi} \in \Xi$ *is a random vector whose probability distribution* P *is supported on a set* $\Xi \subset \mathbb{R}^{d_1}$.

Then, we would like to find a stationary point $\boldsymbol{x}^\star$ *of the problem of minimizing* f *over* X*, i.e.,*

$$\boldsymbol{x}^\star \in X^\star := \{\boldsymbol{x}^\star \in X \colon \langle \boldsymbol{x} - \boldsymbol{x}^\star, \nabla f(\boldsymbol{x}^\star) \rangle \geq 0 \ (\boldsymbol{x} \in X)\},$$

where ∇f *denotes the gradient of* f.

We can see that, if $X = \mathbb{R}^d$, then $X^\star = \{\boldsymbol{x}^\star \in \mathbb{R}^d \colon \nabla f(\boldsymbol{x}^\star) = \boldsymbol{0}\}$ and that, if f is convex, then $\boldsymbol{x}^\star \in X^\star$ is a global minimizer of f over X [16, Subchapter 1.3.1].

Problem 1 is examined under the following conditions [8].

(C1) There is an independent and identically distributed sample $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots$ of realizations of the random vector $\boldsymbol{\xi}$;
(C2) There is an oracle which, for a given input point $(\boldsymbol{x}, \boldsymbol{\xi}) \in \mathbb{R}^d \times \Xi$, returns a stochastic gradient $\mathsf{G}(\boldsymbol{x}, \boldsymbol{\xi})$ such that $\mathbb{E}[\mathsf{G}(\boldsymbol{x}, \boldsymbol{\xi})] = \nabla f(\boldsymbol{x})$;
(C3) There exists a positive number M such that, for all $\boldsymbol{x} \in X$, $\mathbb{E}[\|\mathsf{G}(\boldsymbol{x}, \boldsymbol{\xi})\|^2] \leq M^2$.

3. Conjugate Gradient-like Method

Algorithm 1 is a method for solving Problem 1 under (C1)–(C3).

First, we would like to emphasize that Algorithm 1 uses a *conjugate gradient-like direction* [10–13] (see step 3 in Algorithm 1) defined by

$$\gamma_n = \gamma \in \left[0, \frac{1}{2}\right] \text{ or } \frac{1}{n}, \ \mathbf{G}_n = \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n \mathbf{G}_{n-1}. \tag{1}$$

The direction (1) differs from a conventional conjugate gradient direction using, for example, the Fletcher-Reeves formula,

$$\gamma_n^{\mathrm{FR}} = \frac{\|\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n)\|^2}{\|\mathsf{G}(\boldsymbol{x}_{n-1}, \boldsymbol{\xi}_{n-1})\|^2}, \ \mathbf{G}_n = \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n^{\mathrm{FR}} \mathbf{G}_{n-1}. \tag{2}$$

Algorithm 1 Conjugate gradient-like method for solving Problem 1

```
Require: $(\alpha_n)_{n\in\mathbb{N}} \subset (0,1), (\beta_n)_{n\in\mathbb{N}} \subset [0,1), (\gamma_n)_{n\in\mathbb{N}} \subset [0,1/2], \delta \in [0,1)$
$n \leftarrow 0, \boldsymbol{x}_0, \mathbf{G}_{-1}, \boldsymbol{m}_{-1} \in \mathbb{R}^d, \mathsf{H}_0 \in \mathbb{S}^d_{++} \cap \mathbb{D}^d$
loop
    $\mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n \mathbf{G}_{n-1}$
    $\boldsymbol{m}_n := \beta_n \boldsymbol{m}_{n-1} + (1-\beta_n)\mathbf{G}_n$
    $\hat{\boldsymbol{m}}_n := (1-\delta^{n+1})^{-1}\boldsymbol{m}_n$
    $\mathsf{H}_n \in \mathbb{S}^d_{++} \cap \mathbb{D}^d$
    Find $\mathbf{d}_n \in \mathbb{R}^d$ that solves $\mathsf{H}_n \mathbf{d} = -\hat{\boldsymbol{m}}_n$.
    $\boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n + \alpha_n \mathbf{d}_n)$
    $n \leftarrow n+1$
end loop
```

Although conventional conjugate gradient methods are powerful tools for solving unconstrained smooth nonconvex optimization (see, e.g., [9] for details on conjugate gradient methods), iterative methods with the conjugate gradient-like directions are useful for solving constrained smooth optimization problems [10–13] (see also Section 1 for details). Since Problem 1 is a constrained optimization problem, we will focus on using conjugate gradient-like directions.

We can see that Algorithm 1 with $\gamma_n = 0$ ($n \in \mathbb{N}$) coincides with the existing algorithm in [8] defined by

$$\begin{cases} \mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \boldsymbol{m}_n := \beta_n \boldsymbol{m}_{n-1} + (1-\beta_n)\mathbf{G}_n, \\ \hat{\boldsymbol{m}}_n := (1-\delta^{n+1})^{-1}\boldsymbol{m}_n, \\ \boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n - \alpha_n \mathsf{H}_n^{-1}\hat{\boldsymbol{m}}_n), \end{cases} \tag{3}$$

where $\mathsf{H}_n \in \mathbb{S}^d_{++} \cap \mathbb{D}^d$. We can also show that algorithm (3) (i.e., Algorithm 1 with $\gamma_n = 0$) includes AMSGrad [7] and Adam [6] by referring to [8, Section 3]. For example, consider H_n and $\boldsymbol{v}_n$ ($n \in \mathbb{N}$) defined for all $n \in \mathbb{N}$ by

$$\begin{aligned} \boldsymbol{v}_n &:= \zeta \boldsymbol{v}_{n-1} + (1-\zeta)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \hat{\boldsymbol{v}}_n &= (\hat{v}_{n,i}) := \left(\max\{\hat{v}_{n-1,i}, v_{n,i}\}\right), \\ \mathsf{H}_n &:= \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \end{aligned} \tag{4}$$

where $\boldsymbol{v}_{-1} = \hat{\boldsymbol{v}}_{-1} = \mathbf{0} \in \mathbb{R}^d$ and $\zeta \in [0,1)$. Then, algorithm (3) with (4) and $\delta = 0$ is the AMSGrad algorithm. When H_n and $\boldsymbol{v}_n$ ($n \in \mathbb{N}$) are defined for all $n \in \mathbb{N}$ by

$$\begin{aligned} \boldsymbol{v}_n &:= \zeta \boldsymbol{v}_{n-1} + (1-\zeta)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \bar{\boldsymbol{v}}_n &:= (1-\zeta^{n+1})^{-1}\boldsymbol{v}_n, \\ \hat{\boldsymbol{v}}_n &= (\hat{v}_{n,i}) := \left(\max\{\hat{v}_{n-1,i}, \bar{v}_{n,i}\}\right), \\ \mathsf{H}_n &:= \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \end{aligned} \tag{5}$$

algorithm (3) with (5) resembles the Adam algorithm.[1]

[1] The original Adam uses $\mathsf{H}_n := \mathsf{diag}(\sqrt{\bar{v}_{n,i}})$ and does not always converge [7, Theorems 1–3]. We use $\mathsf{H}_n := \mathsf{diag}(\sqrt{\hat{v}_{n,i}})$ to guarantee its convergence (see Theorems 1 and 2 for the convergence of Algorithm 1).

For example, let us consider Algorithm 1 with (4) and $\delta = 0$, i.e,

$$\begin{cases} \mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n \mathbf{G}_{n-1}, \\ \boldsymbol{m}_n := \beta_n \boldsymbol{m}_{n-1} + (1-\beta_n)\mathbf{G}_n, \\ \boldsymbol{v}_n := \zeta \boldsymbol{v}_{n-1} + (1-\zeta)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \hat{\boldsymbol{v}}_n = (\hat{v}_{n,i}) := \left(\max\{\hat{v}_{n-1,i}, v_{n,i}\}\right), \\ \mathsf{H}_n := \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \\ \boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n - \alpha_n \mathsf{H}_n^{-1} \boldsymbol{m}_n). \end{cases} \tag{6}$$

From the above discussion, algorithm (6) with $\gamma_n = 0$ coincides with AMSGrad. We can see that algorithm (6) uses a conjugate gradient-like direction $\mathbf{G}_n = \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n \mathbf{G}_{n-1}$, while AMSGrad (algorithm (3) with (4)) uses a gradient direction $\mathbf{G}_n = \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n)$.

The convergence analyses of Algorithm 1 assume the following conditions.

Assumption 1. *The sequence $(\mathsf{H}_n)_{n\in\mathbb{N}} \subset \mathbb{S}^d_{++} \cap \mathbb{D}^d$, denoted by $\mathsf{H}_n := \mathsf{diag}(h_{n,i})$, in Algorithm 1 satisfies the following conditions:*

(A3) *$h_{n+1,i} \geq h_{n,i}$ almost surely for all $n \in \mathbb{N}$ and all $i = 1, 2, \ldots, d$;*
(A4) *For all $i = 1, 2, \ldots, d$, a positive number B_i exists such that $\sup\{\mathbb{E}[h_{n,i}] \colon n \in \mathbb{N}\} \leq B_i$.*

Moreover,

(A5) *$D := \max_{i=1,2,\ldots,d} \sup\{(x_i - y_i)^2 \colon (x_i), (y_i) \in X\} < +\infty$.*

Assumption (A5) holds under the boundedness condition of X, which is assumed in [17, p.1574] and [7, p.2]. In [8, Section 3], it is shown that H_n and $\boldsymbol{v}_n$ defined by (4) or (5) satisfies (A3) and (A4).

3.1. Constant learning rate rule

The following is the convergence analysis of Algorithm 1 with a constant learning rate. Theorem 1 can be inferred by referring to the proof of Theorem 3.1 in [8]. The proof of Theorem 1 is given in Appendix A.

Theorem 1. *Suppose that (A1)–(A5) and (C1)–(C3) hold and $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is the sequence generated by Algorithm 1 with $\alpha_n := \alpha$, $\beta_n := \beta$, and $\gamma_n := \gamma$ ($n \in \mathbb{N}$). Then, for all $\boldsymbol{x} \in X$,*

$$\limsup_{n\to+\infty} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_n, \nabla f(\boldsymbol{x}_n)\rangle\right] \geq -\frac{\tilde{B}^2 \tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha - \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta - \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma,$$

where $\tilde{\delta} := 1-\delta$, $\tilde{b} := 1-\beta$, M is defined as in (C3), $\hat{M}^2 := \max\{M^2, \|\mathbf{G}_{-1}\|^2\}$, $\tilde{M}^2 := \max\{\|\boldsymbol{m}_{-1}\|^2, 4\hat{M}^2\}$, D is defined as in (A5), and $\tilde{B} := \sup\{\max_{i=1,2,\ldots,d} h_{n,i}^{-1/2} \colon n \in \mathbb{N}\} < +\infty$.

Theorem 1 shows that using a small constant learning rate approximates a solution to Problem 1. The result for $\gamma := 0$ coincides with Theorem 3.1 in [8].

We have the following proposition for convex stochastic optimization.

Proposition 1. *Suppose that (A1)–(A5) and (C1)–(C3) hold, $F(\cdot, \boldsymbol{\xi})$ is convex for almost every $\boldsymbol{\xi} \in \Xi$, and $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is the sequence generated by Algorithm 1 with $\alpha_n := \alpha$, $\beta_n := \beta$, and $\gamma_n := \gamma$ ($n \in \mathbb{N}$). Then,*

$$\liminf_{n\to+\infty} \mathbb{E}\left[f(\boldsymbol{x}_n) - f^\star\right] \leq \frac{\tilde{B}^2 \tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma,$$

where $f^\star$ denotes the optimal value of the problem of minimizing f over X, and $\tilde{\delta}$, $\tilde{b}$, M, $\hat{M}$, $\tilde{M}$, D, and $\tilde{B}$ are defined as in Theorem 1.

The previously reported results in [7] showed that AMSGrad, which is an example of Algorithm 1 (see algorithm (3) with (4) and $\delta = 0$), ensures that there exists a positive real number B such that

$$\frac{R(T)}{T} = \frac{1}{T}\left(\sum_{t=1}^{T} F(\boldsymbol{x}_t, \boldsymbol{\xi}_t) - f^\star\right) \leq B\sqrt{\frac{1+\ln T}{T}}, \tag{7}$$

where T is the number of training examples and $F(\cdot, \boldsymbol{\xi})$ is convex for almost every $\boldsymbol{\xi} \in \Xi$. Inequality (7) indicates that the value $R(T)/T$ generated by AMSGrad has an upper bound; however, it is not guaranteed that AMSGrad solves Problem 1. Meanwhile, Proposition 1 shows that Algorithm 1, which includes Adam and AMSGrad, can approximate a global minimizer of f by using a small constant learning rate.

3.2. Diminishing learning rate rule

The following is the convergence analysis of Algorithm 1 with diminishing learning rates. Theorem 2 can be proven by referring to the proof of Theorem 3.2 in [8]. The proof of Theorem 2 is given in Appendix A.

Theorem 2. *Suppose that (A1)–(A5) and (C1)–(C3) hold and $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is the sequence generated by Algorithm 1 with α_n, β_n, and γ_n $(n \in \mathbb{N})$*[2] *satisfying $\sum_{n=0}^{+\infty} \alpha_n = +\infty$, $\sum_{n=0}^{+\infty} \alpha_n^2 < +\infty$, $\sum_{n=0}^{+\infty} \alpha_n\beta_n < +\infty$, and $\sum_{n=0}^{+\infty} \alpha_n\gamma_n < +\infty$. Then, for all $\boldsymbol{x} \in X$,*

$$\limsup_{n\to+\infty} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_n, \nabla f(\boldsymbol{x}_n)\rangle\right] \geq 0. \tag{8}$$

Moreover, suppose that $\alpha_n := 1/n^\eta$, $\beta_n := \beta^n$, $\gamma_n := \gamma^n$ or $1/n^\kappa$, where $\eta \in [1/2, 1)$, $\kappa > 1 - \eta$, and $\beta, \gamma \in (0,1)$. Then, Algorithm 1 achieves the following convergence rate:

$$\frac{1}{n}\sum_{k=1}^{n} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_k, \nabla f(\boldsymbol{x}_k)\rangle\right] \geq \begin{cases} -\mathcal{O}\left(\sqrt{\dfrac{1+\ln n}{n}}\right) & \text{if } \eta = \frac{1}{2}, \\ -\mathcal{O}\left(\dfrac{1}{n^{1-\eta}}\right) & \text{if } \eta \in \left(\frac{1}{2}, 1\right). \end{cases}$$

Inequality (8) implies that there exists a subsequence $(\boldsymbol{x}_{n_j})_{j\in\mathbb{N}}$ of $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ such that $(\boldsymbol{x}_{n_j})_{j\in\mathbb{N}}$ converges to $\boldsymbol{x}_\star$ and, for all $\boldsymbol{x} \in X$,

$$\lim_{j\to+\infty} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_{n_j}, \nabla f(\boldsymbol{x}_{n_j})\rangle\right] = \limsup_{n\to+\infty} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_n, \nabla f(\boldsymbol{x}_n)\rangle\right] \geq 0,$$

which implies that $\boldsymbol{x}_\star$ satisfies $\langle \boldsymbol{x} - \boldsymbol{x}_\star, \nabla f(\boldsymbol{x}_\star)\rangle \geq 0$ $(\boldsymbol{x} \in X)$; i.e., $\boldsymbol{x}_\star$ is a solution to Problem 1.

Theorem 2 leads to the following proposition, which indicates that Algorithm 1 converges to a global minimizer of f when $F(\cdot, \boldsymbol{\xi})$ is convex for almost every $\boldsymbol{\xi} \in \Xi$.

Proposition 2. *Suppose that (A1)–(A5) and (C1)–(C3) hold, $F(\cdot, \boldsymbol{\xi})$ is convex for almost every $\boldsymbol{\xi} \in \Xi$, and $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is the sequence generated by Algorithm 1 with α_n, β_n, and γ_n satisfying $\sum_{n=0}^{+\infty} \alpha_n = +\infty$, $\sum_{n=0}^{+\infty} \alpha_n^2 < +\infty$, $\sum_{n=0}^{+\infty} \alpha_n\beta_n < +\infty$, and $\sum_{n=0}^{+\infty} \alpha_n\gamma_n < +\infty$. Then,*

$$\liminf_{n\to+\infty} \mathbb{E}\left[f(\boldsymbol{x}_n) - f^\star\right] = 0,$$

[2] Let $\alpha_n := 1/n^\eta$, $\beta_n := \beta^n$, $\gamma_n := \gamma^n$ or $1/n^\kappa$, where $\eta \in (1/2, 1]$, $\kappa > 1 - \eta$, and $\beta, \gamma \in (0,1)$. Then, $\sum_{n=1}^{+\infty} \alpha_n = +\infty$, $\sum_{n=1}^{+\infty} \alpha_n^2 < +\infty$, $\sum_{n=1}^{+\infty} \alpha_n\beta_n < +\infty$, and $\sum_{n=1}^{+\infty} \alpha_n\gamma_n < +\infty$ hold. Since $(\gamma_n)_{n\in\mathbb{N}}$ converges to 0, there exists $k_0 \in \mathbb{N}$ such that, for all $n \geq k_0$, $\gamma_n \leq 1/2$.

where $f^\star$ denotes the optimal value of the problem of minimizing f over X. Moreover, suppose that $\alpha_n := 1/n^\eta$, $\beta_n := \beta^n$, $\gamma_n := \gamma^n$ or $1/n^\kappa$, where $\eta \in [1/2, 1)$, $\kappa > 1 - \eta$, and $\beta, \gamma \in (0, 1)$. Then, any accumulation point of $(\tilde{x}_n)_{n\in\mathbb{N}}$ defined by $\tilde{x}_n := (1/n)\sum_{k=1}^n x_k$ almost surely belongs to the solution set $X^\star$, and Algorithm 1 achieves the following convergence rate:

$$\mathbb{E}\left[f(\tilde{x}_n) - f^\star\right] = \begin{cases} \mathcal{O}\left(\sqrt{\dfrac{1+\ln n}{n}}\right) & \text{if } \eta = \frac{1}{2}, \\ \mathcal{O}\left(\dfrac{1}{n^{1-\eta}}\right) & \text{if } \eta \in \left(\frac{1}{2}, 1\right). \end{cases}$$

4. Numerical Experiments

The experiments used a fast scalar computation server [3] at Meiji University. The environment has two Intel(R) Xeon(R) Gold 6148 (2.4 GHz, 20 cores) CPUs, an NVIDIA Tesla V100 (16GB, 900Gbps) GPU, and a Red Hat Enterprise Linux 7.6 operating system. The experimental code was written in Python 3.8.2, and we used the NumPy 1.19.1 package and PyTorch 1.5.0 package.

We compared the existing algorithms, such as the momentum method [18, (9)], [19, Section 2], AdaGrad [5], RMSProp [4, Algorithm 8.5], Adam [6], and AMSGrad [7] in `torch.optim`[4] using the default values and learning rate 10^{-3}, with Algorithm 1 defined as follows:

Algorithm 1 with a constant learning rate (Algorithm 1 with $\gamma_n = 0$, such as Momentum-Ci, Adam-Ci, and AMSGrad-Ci ($i = 1, 2, 3$), is Algorithm 1 in [8]):

- Momentum-C1: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, $\alpha_n = \beta_n = 10^{-1}$, and $\gamma_n = 0$.
- Momentum-C2: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, $\alpha_n = \beta_n = 10^{-2}$, and $\gamma_n = 0$.
- Momentum-C3: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, $\alpha_n = \beta_n = 10^{-3}$, and $\gamma_n = 0$.
- MomentumCG-C1: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\alpha_n = \beta_n = \gamma_n = 10^{-1}$.
- MomentumCG-C2: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\alpha_n = \beta_n = \gamma_n = 10^{-2}$.
- MomentumCG-C3: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\alpha_n = \beta_n = \gamma_n = 10^{-3}$.
- Adam-C1: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), $\alpha_n = \beta_n = 10^{-1}$, and $\gamma_n = 0$.
- Adam-C2: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), $\alpha_n = \beta_n = 10^{-2}$, and $\gamma_n = 0$.
- Adam-C3: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), $\alpha_n = \beta_n = 10^{-3}$, and $\gamma_n = 0$.
- AdamCG-C1: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\alpha_n = \beta_n = \gamma_n = 10^{-1}$.
- AdamCG-C2: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\alpha_n = \beta_n = \gamma_n = 10^{-2}$.
- AdamCG-C3: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\alpha_n = \beta_n = \gamma_n = 10^{-3}$.
- AMSGrad-C1: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), $\alpha_n = \beta_n = 10^{-1}$, and $\gamma_n = 0$.
- AMSGrad-C2: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), $\alpha_n = \beta_n = 10^{-2}$, and $\gamma_n = 0$.
- AMSGrad-C3: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), $\alpha_n = \beta_n = 10^{-3}$, and $\gamma_n = 0$.
- AMSGradCG-C1: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\alpha_n = \beta_n = \gamma_n = 10^{-1}$.
- AMSGradCG-C2: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\alpha_n = \beta_n = \gamma_n = 10^{-2}$.
- AMSGradCG-C3: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\alpha_n = \beta_n = \gamma_n = 10^{-3}$.

Algorithm 1 with diminishing learning rates $\alpha_n = 1/\sqrt{n}$ and $\beta_n = 1/2^n$ based on [7, Theorem 4 and Corollary 1] (Algorithm 1 with $\gamma_n = 0$, such as Momentum-D1, Adam-D1, and AMSGrad-D1, is Algorithm 1 in [8]):

- Momentum-D1: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\gamma_n = 0$.
- MomentumCG-D1: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\gamma_n = 1/2^n$.
- MomentumCG-D2: Algorithm 1 with $\delta = 0$, $\mathsf{H}_n = \mathsf{diag}(1)$, and $\gamma_n = 1/n$.
- Adam-D1: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\gamma_n = 0$.
- AdamCG-D1: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\gamma_n = 1/2^n$.

[3] https://www.meiji.ac.jp/isys/hpc/ia.html

[4] https://pytorch.org/docs/stable/optim.html

- AdamCG-D2: Algorithm 1 with $\delta = 0.9$, $\zeta = 0.999$, H_n defined by (5), and $\gamma_n = 1/n$.
- AMSGrad-D1: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\gamma_n = 0$.
- AMSGradCG-D1: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\gamma_n = 1/2^n$.
- AMSGradCG-D2: Algorithm 1 with $\delta = 0$, $\zeta = 0.999$, H_n defined by (4), and $\gamma_n = 1/n$.

Python implementations of the algorithms are available at https://github.com/iiduka-researches/202008-cg-like.

4.1. Image classification

This experiment used the CIFAR10 dataset[5], a benchmark for image classification. The dataset consists of 60,000 color images (32 × 32) in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images. The test batch contained exactly 1,000 randomly selected images from each class. We trained a 44-layer ResNet (ResNet-44) [20] organized into 43 convolutional layers which had 3 × 3 filters and a 1,000-way-fully-connected layer with a softmax function. We used the cross entropy as the loss function for fitting ResNet in accordance with the commonly used strategy in image classification.

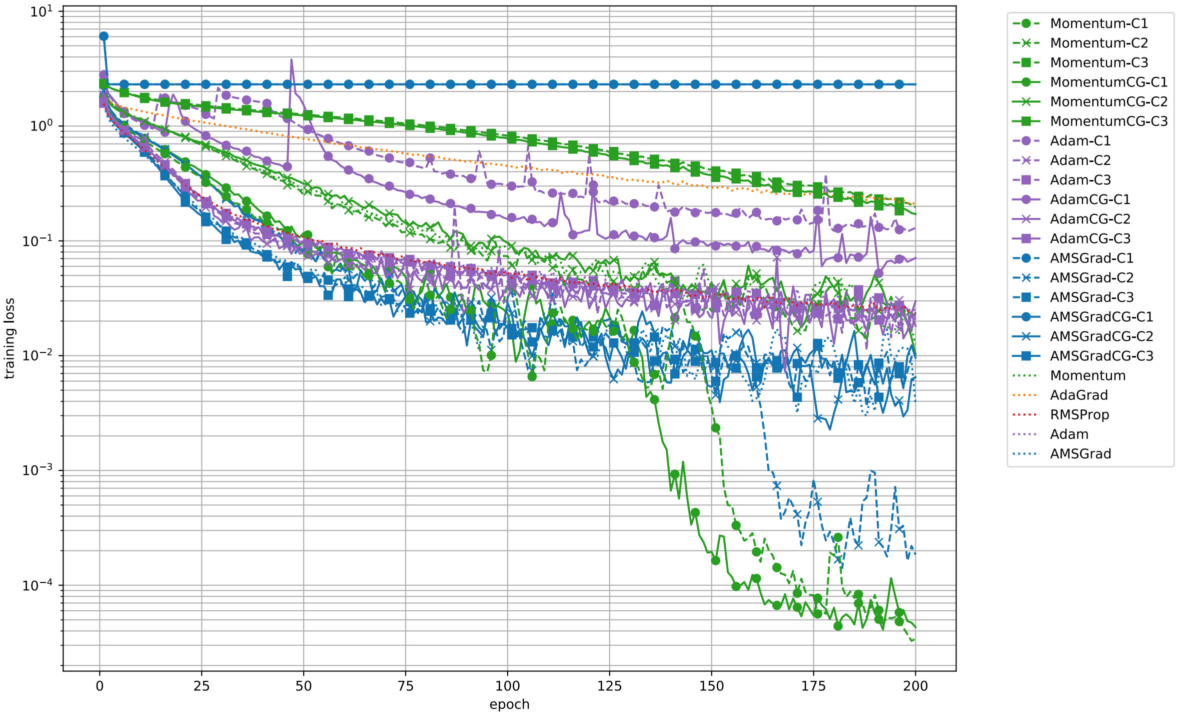

Figure 1. Loss function value versus number of epochs on the CIFAR-10 dataset for training (constant).

[5] https://www.cs.toronto.edu/~kriz/cifar.html

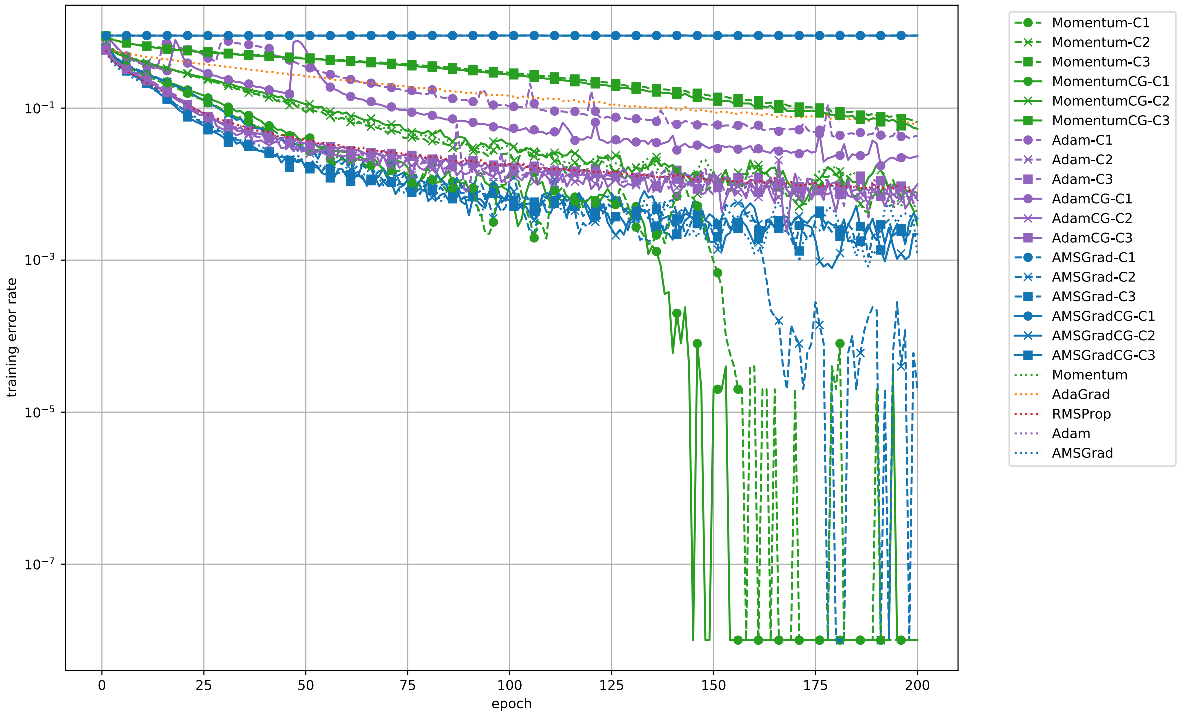

Figure 2. Classification error rate versus number of epochs on the CIFAR-10 dataset for training (constant).

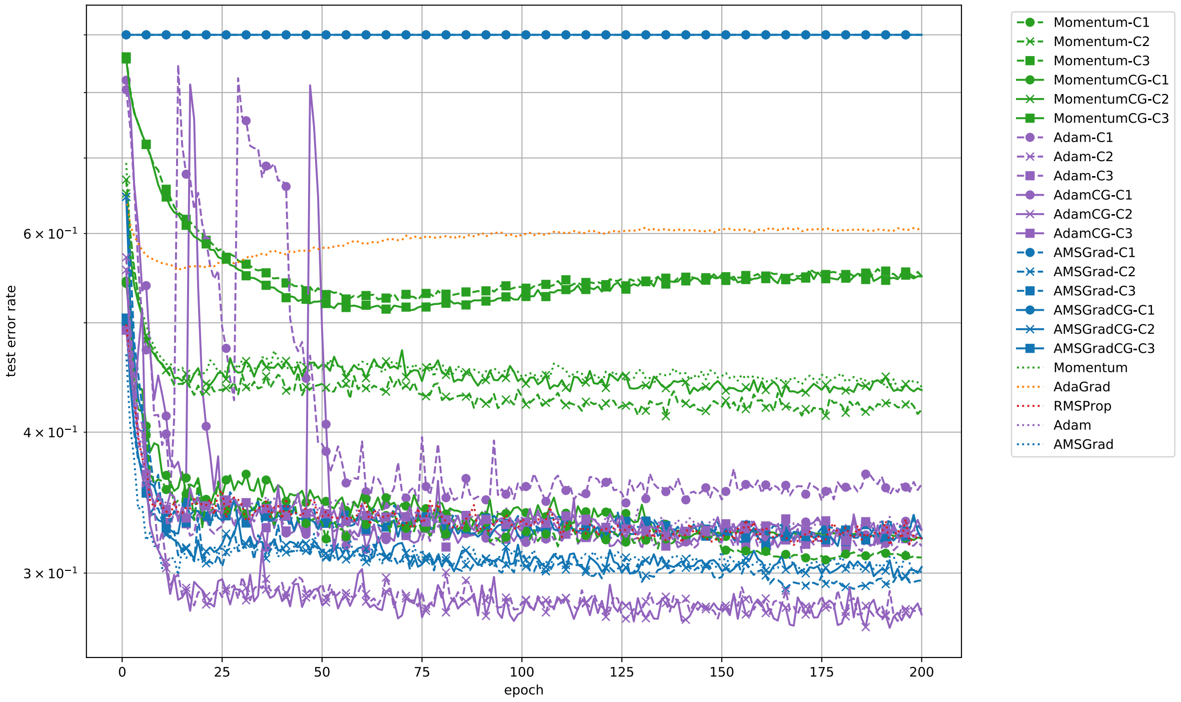

Figure 3. Classification error rate versus number of epochs on the CIFAR-10 dataset for testing (constant).

Figures 1–3 compare the behaviors of the proposed algorithm with a constant learning rate with those of Momentum, AdaGrad, RMSProp, Adam, and AMSGrad using the default values in `torch.optim` (i.e., $\alpha_n = 10^{-3}, \beta_n = 0.9$). Figure 1 shows that Momentum-C1, MomentumCG-C1, and AMSGrad-C2 minimized the training loss function faster than the existing algorithms, and Figure 2 shows that they decreased the training error rate faster as well. Moreover, AdamCG-C*i* (resp.

207 AMSGradCG-Ci) ($i = 2, 3$) outperformed AdamCG-C1 (resp. AMSGradCG-C1); this implies AdamCG
208 and AMSGradCG require fewer iterations at smaller learning rates. Figure 3 shows that Adam-C2,
209 AdamCG-C2, AMSGrad-C2, AMSGradCG-C2 decreased the test error rate faster than other algorithms.
210 A similar trend was observed in the numerical results in [21].

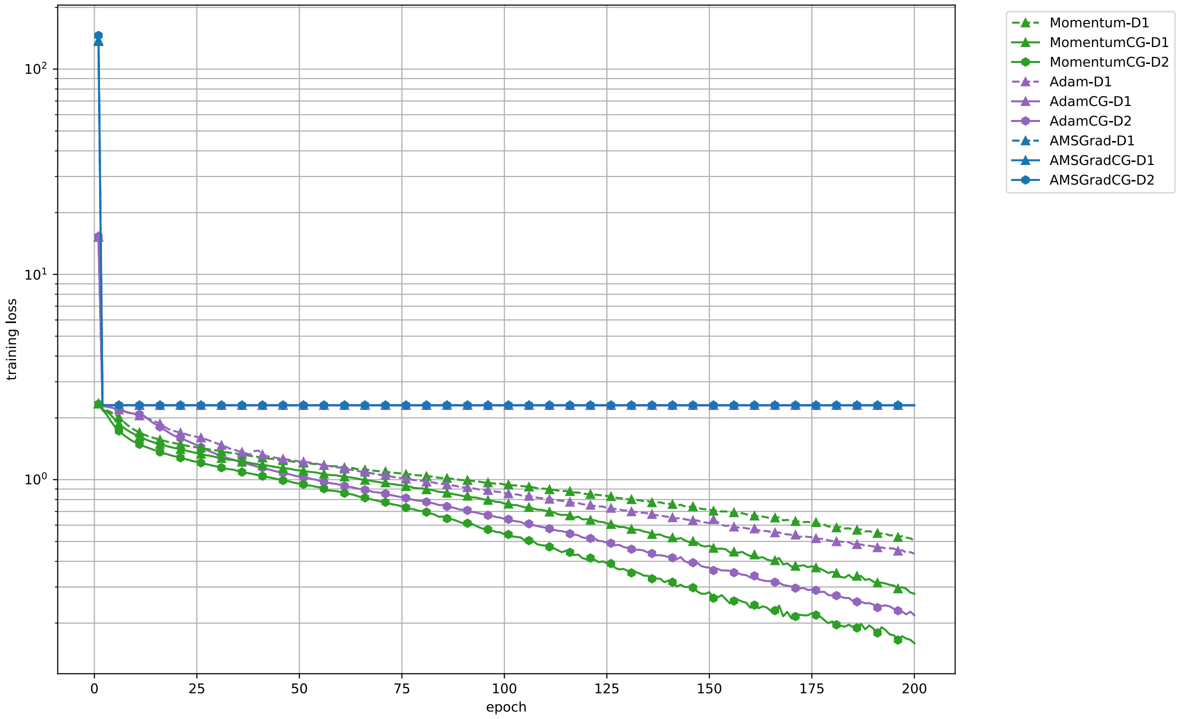

Figure 4. Loss function value versus number of epochs on the CIFAR-10 dataset for training (diminishing).

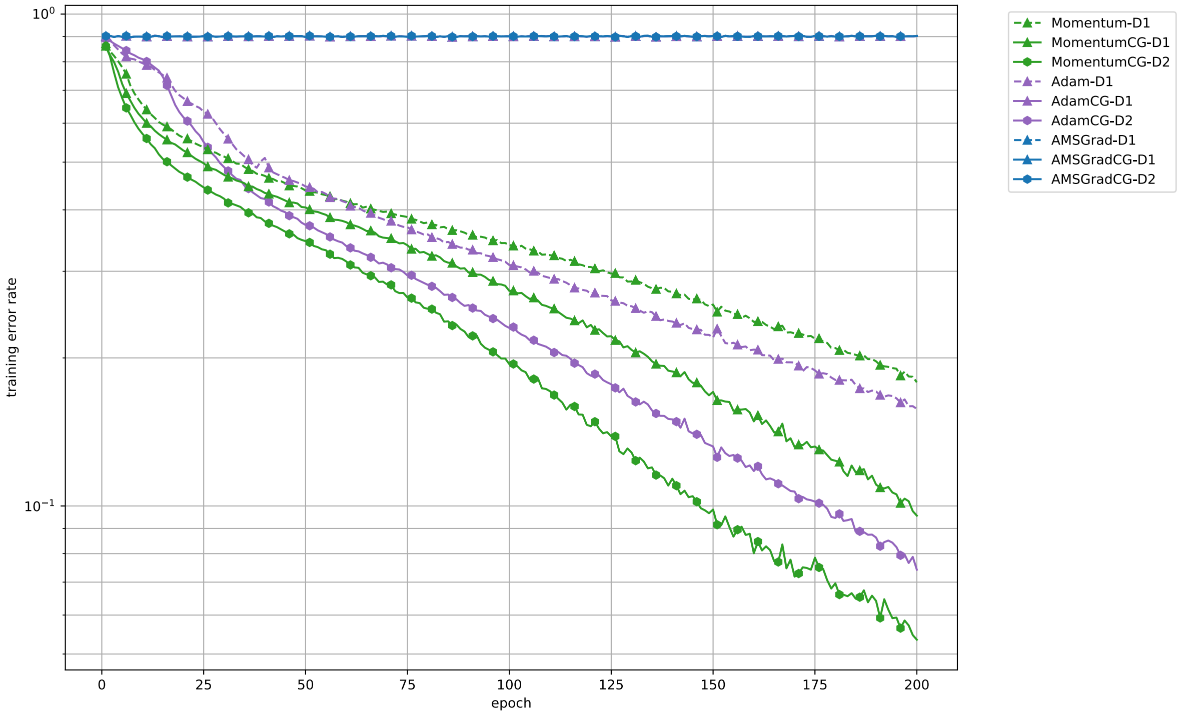

Figure 5. Classification error rate versus number of epochs on the CIFAR-10 dataset for training (diminishing).

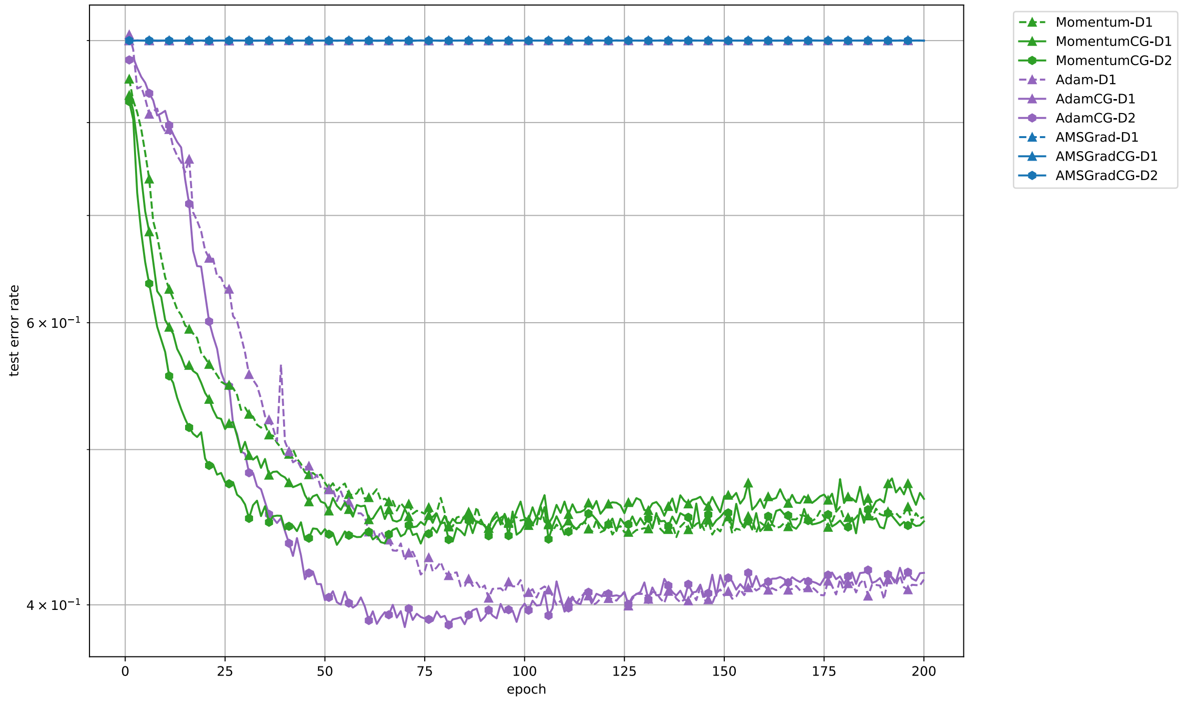

Figure 6. Classification error rate versus number of epochs on the CIFAR-10 dataset for testing (diminishing).

211 Figures 4–6 plot the behaviors of the proposed algorithms with diminishing learning rates. These 212 algorithms did not work, and thus, it clear that using diminishing learning rates is not good for training 213 neural networks (see Section 5 for the details). A similar problem was observed in the numerical 214 results in [8].

Table 1. Mean and variance of elapsed time per epoch for the existing algorithms and Algorithm 1 on the CIFAR-10 dataset

		Existing	C1	C2	C3	CG-C1	CG-C2	CG-C3
Momentum	mean	14.815106	14.766352	14.643343	14.191675	14.370240	14.536258	13.732973
	variance	0.268979	1.144346	0.268576	0.363746	0.180754	0.872769	0.314055
Adam	mean	17.621361	17.388947	18.511805	18.084771	18.106918	18.108820	17.127479
	variance	0.149553	0.044539	1.392942	0.056606	0.213341	0.063594	1.317213
AMSGrad	mean	18.122551	17.650377	17.796328	19.335775	18.855297	18.272888	16.328777
	variance	1.245563	0.313088	0.289944	4.738541	2.820650	1.671705	1.754373

Table 2. Results of t-test on the training error rates of the existing algorithms (Momentum, Adam, and AMSGrad) and Algorithm 1 (Ci and CG-Ci ($i = 1, 2, 3$)) on the CIFAR-10 dataset (significance level is 5%; the p-values for the proposed algorithms with significantly low error rates are indicated in bold)

		C1	C2	C3	CG-C1	CG-C2	CG-C3
Momentum	t-statistic	3.70879	0.23783	-13.65314	3.34063	-0.17214	-12.77890
(Existing)	p-value	**2.38E-04**	8.12E-01	4.44E-35	**9.15E-04**	8.63E-01	1.43E-31
Adam	t-statistic	-10.46006	0.03599	0.20774	-6.70248	0.37493	0.04342
(Existing)	p-value	8.73E-23	9.71E-01	8.36E-01	7.03E-11	7.08E-01	9.65E-01
AMSGrad	t-statistic	-157.96917	-1.59278	-0.16230	-157.97057	-1.59440	-0.00869
(Existing)	p-value	0.00E+00	1.12E-01	8.71E-01	0.00E+00	1.12E-01	9.93E-01

Table 1 shows the mean and variance of elapsed time per epoch for the existing algorithms and Algorithm 1 with a constant learning rate. This table indicates that the elapsed time of Momentum was almost the same as those of the proposed algorithms, e.g., Momentum-Ci and MomentumCG-Ci ($i = 1, 2, 3$). Adam and AMSGrad also had such a trend.

Table 2 compares the training error rates of the existing algorithms with those of Algorithm 1 by using the `scipy.stats.ttest_ind` function in Python. The p-value is the probability associated with a t-test, and the significance level is set at 5 %. If the value is less than 0.05, then there is a significant difference between the existing algorithm and the proposed algorithms. Table 2 and Figure 2 indicate that Momentum-C1 and MomentumCG-C1 outperformed Momentum and the performance of the existing algorithm (Momentum) was significantly different from the performances of the proposed algorithms (Momentum-C1 and MomentumCG-C1). Adam-Ci and AdamCG-Ci ($i = 1, 2, 3$) had almost the same performance as Adam, while the performance of AMSGrad was not significantly different from that of AMSGrad-Ci and AMSGradCG-Ci ($i = 1, 2, 3$).

4.2. Text classification

This experiment used the IMDb dataset[6] for text classification tasks. The dataset contains 50,000 movie reviews along with their associated binary sentiment polarity labels. The dataset is split into 25,000 training and 25,000 test sets. We used an embedding layer that generated 50-dimensional embedding vectors and two bidirectional long short-term memory (LSTM) with an affine layer and a sigmoid function as an activation function for the output. To train it, we used the binary cross entropy (BCE) as a loss function minimized by the existing and proposed algorithms.

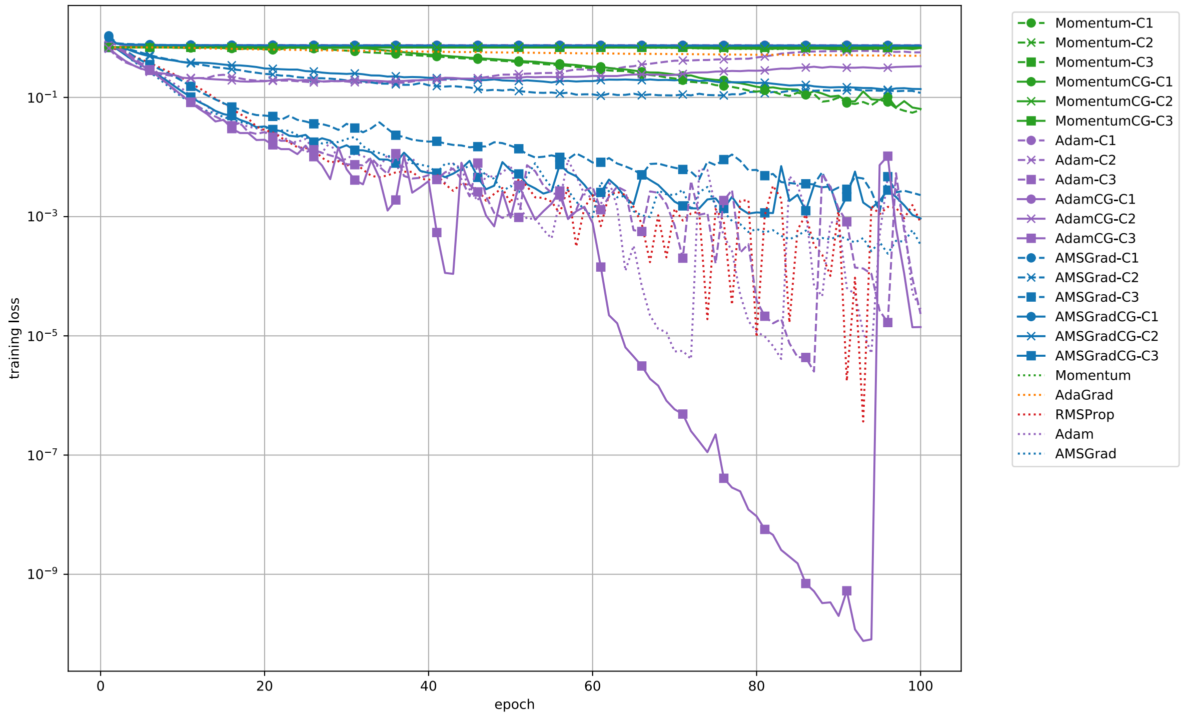

Figure 7. Loss function value versus number of epochs on the IMDb dataset for training (constant).

[6] https://datasets.imdbws.com/

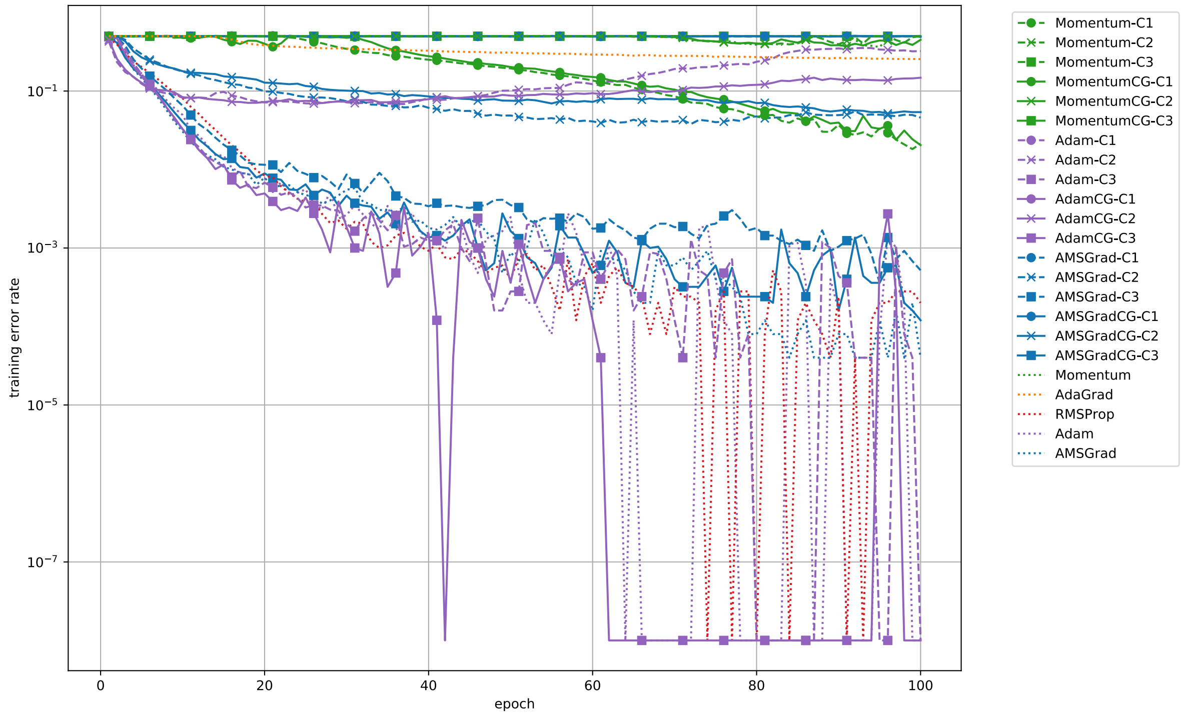

Figure 8. Classification error rate versus number of epochs on the IMDb dataset for training (constant).

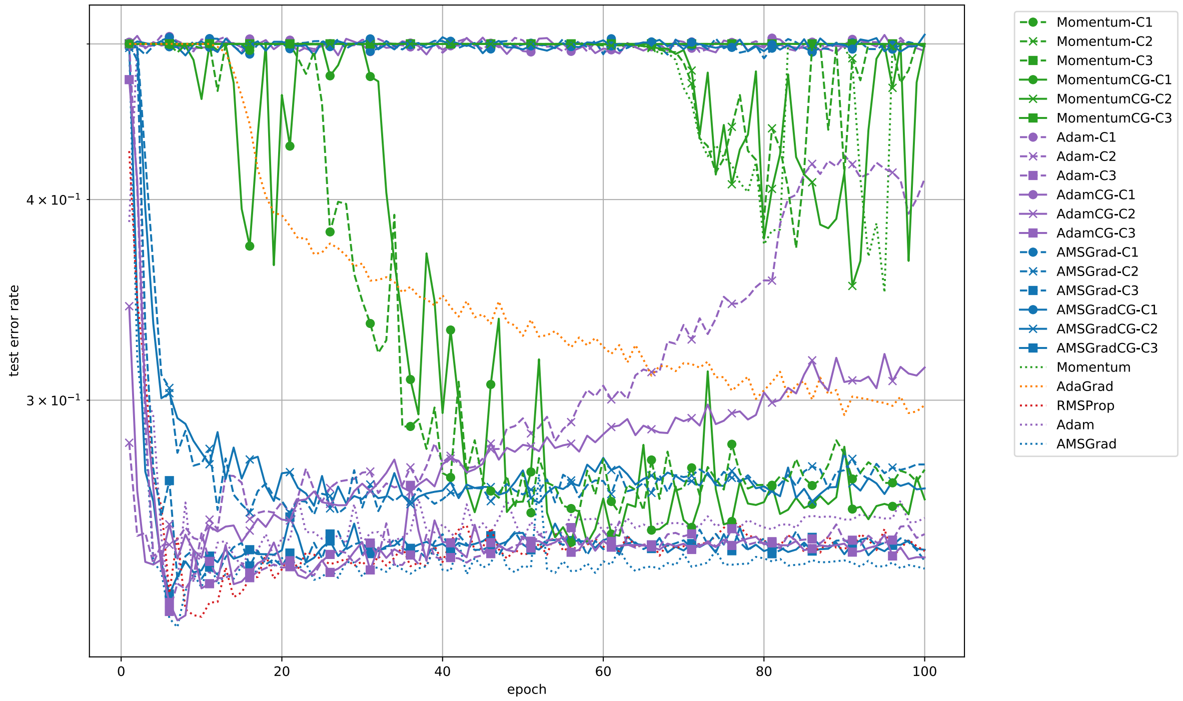

Figure 9. Classification error rate versus number of epochs on the IMDb dataset for testing (constant).

Figures 7–9 compare the behaviors of the proposed algorithm with a constant learning rate with those of Momentum, AdaGrad, RMSProp, Adam, and AMSGrad, using the default values in `torch.optim` (i.e., $\alpha_n = 10^{-3}, \beta_n = 0.9$). These figures show that Adam-C3, AdamCG-C3, AMSGrad-C3, RMSProp, Adam, and AMSGrad all performed well. In particular, Figure 8 shows that AdamCG-C3 (resp. AMSGradCG-C3) performed better than Adam-C3 (resp. AMSGrad-C3), which implies that using conjugate gradient-like directions would be good for training neural networks.

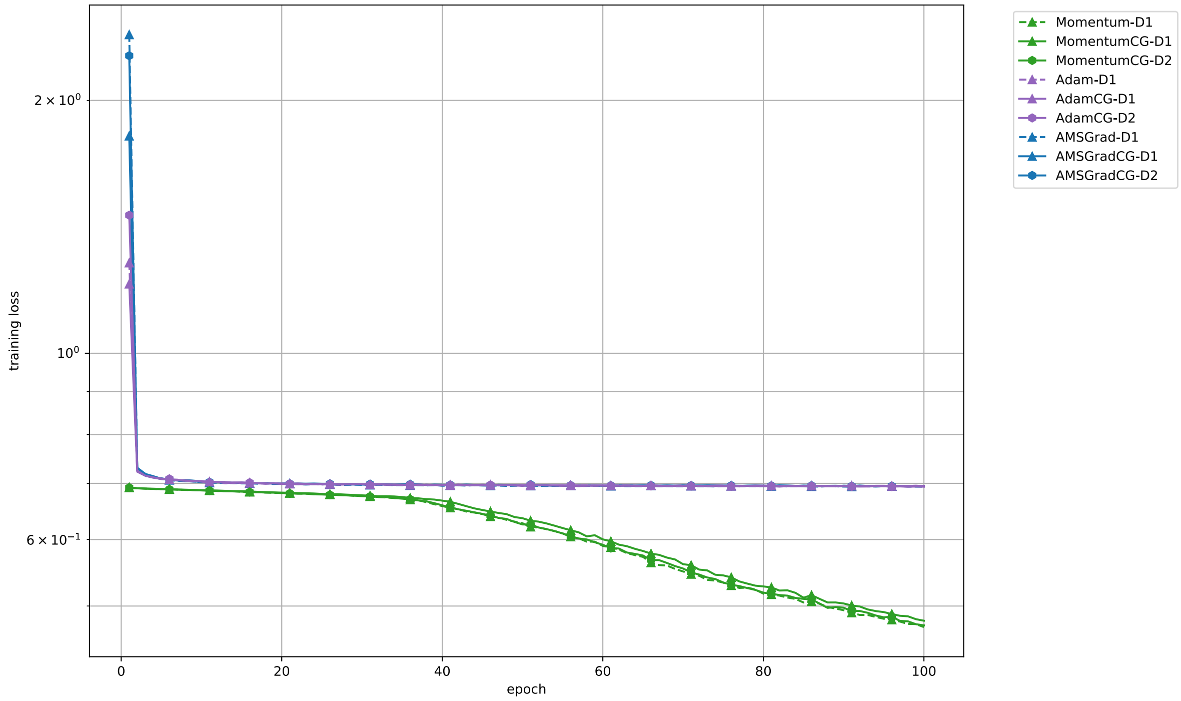

Figure 10. Loss function value versus number of epochs on the IMDb dataset for training (diminishing).

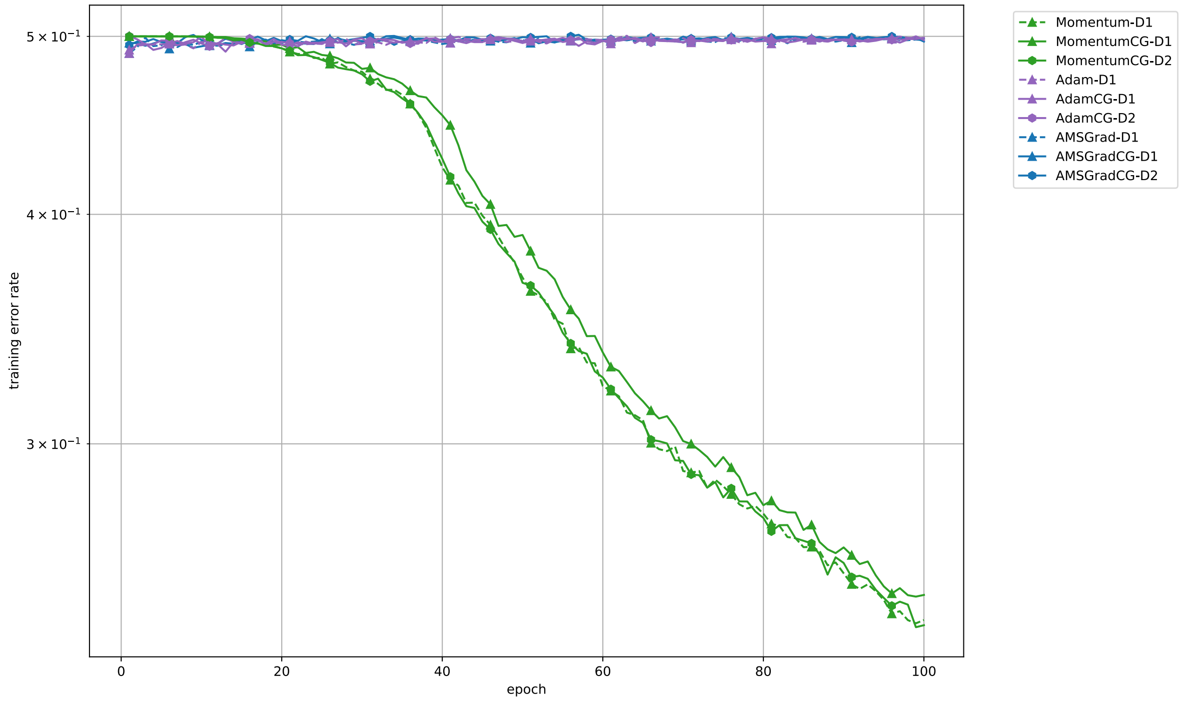

Figure 11. Classification error rate versus number of epochs on the IMDb dataset for training (diminishing).

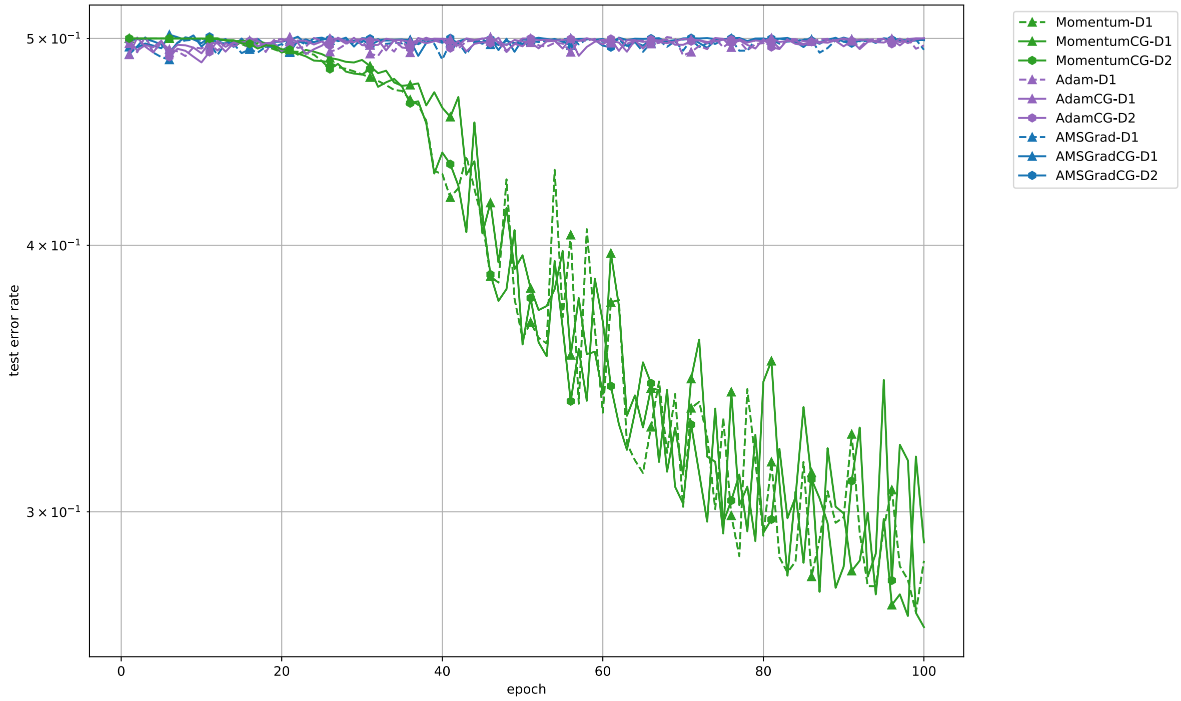

Figure 12. Classification error rate versus number of epochs on the IMDb dataset for testing (diminishing).

Figures 10–12 indicate the behaviors of the proposed algorithms with diminishing learning rates. These figures show that the algorithms did not work, as was the case in Figures 4–6 (see Section 5 for the details).

Table 3. Mean and variance of elapsed time per epoch for the existing algorithms and Algorithm 1 on the IMDb dataset

		Existing	C1	C2	C3	CG-C1	CG-C2	CG-C3
Momentum	mean	19.029660	18.999186	18.957496	19.098836	19.241769	19.286854	18.671163
	variance	0.095132	0.074935	0.107259	0.196841	0.035649	0.058319	0.003906
Adam	mean	20.256827	20.194220	20.193260	20.260705	20.231550	20.388470	19.536741
	variance	0.061552	0.023485	0.041777	0.060461	0.039103	0.174818	0.165803
AMSGrad	mean	20.109489	20.092463	20.102763	20.025613	20.146646	20.136673	19.335856
	variance	0.075432	0.059149	0.059561	0.089540	0.113563	0.098914	0.003543

Table 4. Results of t-test on the training error rates of the existing algorithms (Momentum, Adam, and AMSGrad) and Algorithm 1 (Ci and CG-Ci ($i = 1, 2, 3$)) on the IMDb dataset (significance level is 5%; the p-values for the proposed algorithms with significantly low error rates are indicated in bold)

		C1	C2	C3	CG-C1	CG-C2	CG-C3
Momentum	t-statistic	13.87142	0.63115	-4.59306	13.22951	1.71477	-4.59306
(Existing)	p-value	5.17E-31	5.29E-01	7.76E-06	4.82E-29	8.80E-02	7.76E-06
Adam	t-statistic	-63.39972	-11.01275	-0.00287	-63.33435	-9.41552	0.11707
(Existing)	p-value	1.79E-133	2.61E-22	**9.98E-01**	2.17E-133	1.24E-17	**9.07E-01**
AMSGrad	t-statistic	-63.53084	-5.68706	-0.63240	-63.42279	-7.93451	-0.06863
(Existing)	p-value	1.21E-133	4.59E-08	5.28E-01	1.67E-133	1.53E-13	9.45E-01

Table 3 indicates that the elapsed time for the existing algorithm was almost the same as the one for the proposed algorithms, as seen in Table 1. Table 4 and Figure 8 show that, although Momentum,

Momentum-Ci, and MomentumCG-Ci did not perform better than the existing algorithms such as Adam and AMSGrad, the performance of Momentum was significantly different from that of almost all of proposed algorithms. It can be seen that Adam, Adam-C3, and AdamCG-C3 performed well and that, although AMSGrad, AMSGrad-C3, and AMSGradCG-C3 did not perform better than Adam, AMSGrad-C3 and AMSGradCG-C3 had almost the same performance as AMSGrad.

5. Discussion

Let us first discuss the relationship between the momentum method [18, (9)], [19, Section 2] with MomentumCG used in Section 4. The momentum method [18, (9)], [19, Section 2] is defined by

$$\boldsymbol{m}_n := -\epsilon \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) + \mu \boldsymbol{m}_{n-1}, \ \boldsymbol{x}_{n+1} := P_X(\boldsymbol{x}_n + \boldsymbol{m}_n), \text{ i.e.,} \tag{9a}$$

$$\boldsymbol{x}_{n+1} := P_X\left(\boldsymbol{x}_n - \epsilon \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) + \mu \boldsymbol{m}_{n-1}\right), \tag{9b}$$

where $\epsilon > 0$ is the learning rate and $\mu \in [0, 1]$ is the momentum coefficient. We can see that $\boldsymbol{m}_n$ defined by (9) is the conjugate gradient-like direction of $\epsilon \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n)$. Meanwhile, MomentumCG used in Section 4 is as follows:

$$\mathbf{G}_n = \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - \gamma_n \mathbf{G}_{n-1}, \tag{10a}$$

$$\boldsymbol{m}_n := (1 - \beta_n)\mathbf{G}_n + \beta_n \boldsymbol{m}_{n-1}, \tag{10b}$$

$$\boldsymbol{x}_{n+1} := P_X(\boldsymbol{x}_n - \alpha_n \boldsymbol{m}_n). \tag{10c}$$

Algorithm (10) uses the conjugate gradient-like direction $\mathbf{G}_n$ of $\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n)$. For simplicity, algorithm (10) with $\beta_n = 0$ is such that

$$\boldsymbol{x}_{n+1} := P_X\left(\boldsymbol{x}_n - \alpha_n \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) + \alpha_n \gamma_n \boldsymbol{m}_{n-1}\right), \tag{11}$$

which implies that algorithm (11) is the momentum method with a learning rate α_n and momentum coefficient $\alpha_n \gamma_n$.

The numerical comparisons in Section 4 show that Algorithm 1 with a constant learning rate performed better than Algorithm 1 with diminishing learning rates. For example, let us consider the text classification in Subsection 4.2 and compare AdamCG-C3 defined by

$$\begin{cases} \mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - 10^{-3}\mathbf{G}_{n-1}, \\ \boldsymbol{m}_n := 10^{-3}\boldsymbol{m}_{n-1} + (1 - 10^{-3})\mathbf{G}_n, \\ \hat{\boldsymbol{m}}_n := (1 - 0.9^{n+1})^{-1}\boldsymbol{m}_n, \\ \boldsymbol{v}_n := 0.999\boldsymbol{v}_{n-1} + (1 - 0.999)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \bar{\boldsymbol{v}}_n := (1 - 0.999^{n+1})^{-1}\boldsymbol{v}_n, \\ \hat{\boldsymbol{v}}_n = (\hat{v}_{n,i}) := \left(\max\{\hat{v}_{n-1,i}, \bar{v}_{n,i}\}\right), \\ \mathsf{H}_n := \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \\ \boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n - 10^{-3}\mathsf{H}_n^{-1}\boldsymbol{m}_n). \end{cases} \tag{12}$$

with AdamCG-D1 defined by

$$\begin{cases} \mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - 2^{-n}\mathbf{G}_{n-1}, \\ \boldsymbol{m}_n := 2^{-n}\boldsymbol{m}_{n-1} + (1-2^{-n})\mathbf{G}_n, \\ \hat{\boldsymbol{m}}_n := (1-0.9^{n+1})^{-1}\boldsymbol{m}_n, \\ \boldsymbol{v}_n := 0.999\boldsymbol{v}_{n-1} + (1-0.999)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \bar{\boldsymbol{v}}_n := (1-0.999^{n+1})^{-1}\boldsymbol{v}_n, \\ \hat{\boldsymbol{v}}_n = (\hat{v}_{n,i}) := (\max\{\hat{v}_{n-1,i}, \bar{v}_{n,i}\}), \\ \mathsf{H}_n := \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \\ \boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n - n^{-1/2}\mathsf{H}_n^{-1}\boldsymbol{m}_n). \end{cases} \tag{13}$$

AdamCG-C3 (algorithm (12)) works well for all $n \in \mathbb{N}$, since it uses a constant learning rate. Meanwhile, there is a possibility that AdamCG-D1 (algorithm (13)) does not work for a large number of iterations, because it uses diminishing learning rates. In fact, AdamCG-D1 (algorithm (13)) for a large n is as follows:

$$\begin{cases} \mathbf{G}_n := \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) - 2^{-n}\mathbf{G}_{n-1} \approx \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \boldsymbol{m}_n := 2^{-n}\boldsymbol{m}_{n-1} + (1-2^{-n})\mathbf{G}_n \approx \mathbf{G}_n \approx \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \hat{\boldsymbol{m}}_n := (1-0.9^{n+1})^{-1}\boldsymbol{m}_n, \\ \boldsymbol{v}_n := 0.999\boldsymbol{v}_{n-1} + (1-0.999)\mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n) \odot \mathsf{G}(\boldsymbol{x}_n, \boldsymbol{\xi}_n), \\ \bar{\boldsymbol{v}}_n := (1-0.999^{n+1})^{-1}\boldsymbol{v}_n, \\ \hat{\boldsymbol{v}}_n = (\hat{v}_{n,i}) := (\max\{\hat{v}_{n-1,i}, \bar{v}_{n,i}\}), \\ \mathsf{H}_n := \mathsf{diag}\left(\sqrt{\hat{v}_{n,i}}\right), \\ \boldsymbol{x}_{n+1} := P_{X,\mathsf{H}_n}(\boldsymbol{x}_n - n^{-1/2}\mathsf{H}_n^{-1}\boldsymbol{m}_n) \approx P_{X,\mathsf{H}_n}(\boldsymbol{x}_n) = \boldsymbol{x}_n, \end{cases} \tag{14}$$

which implies that algorithm (14) does not work. As can be seen in Figures 7–12, Algorithm 1 with diminishing learning rates would not be good for training neural networks.

Finally, let us compare the existing algorithm with Algorithm 1, in particular, AMSGrad in `torch.optim` using $\alpha_n = 10^{-3}$, $\beta_n = 0.9$, and $\zeta = 0.999$ with AMSGrad-C3 using $\alpha_n = 10^{-3}$, $\beta_n = 10^{-3}$, and $\zeta = 0.999$. The difference between AMSGrad and AMSGrad-C3 is the setting of β_n. According to Figures 7–9, AMSGrad-C3 performs comparably to AMSGrad, a useful algorithm. These results are guaranteed by Theorem 1, which indicates that Algorithm 1 with a small constant learning rate approximates a stationary point of the minimization problem in deep neural networks, and more specifically, the sequence $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ generated by AMSGrad-C3 (Algorithm 1 with $\delta = 0$) satisfying

$$\limsup_{n\to+\infty} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_n, \nabla f(\boldsymbol{x}_n)\rangle\right] \geq -\frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}}\frac{1}{10^3} - \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}}\frac{1}{10^3} \quad (\boldsymbol{x} \in X)$$

approximates $\boldsymbol{x}^\star \in X^\star := \{\boldsymbol{x}^\star \in X : \langle \boldsymbol{x} - \boldsymbol{x}^\star, \nabla f(\boldsymbol{x}^\star)\rangle \geq 0 \ (\boldsymbol{x} \in X)\}$.

6. Conclusion

We proposed an iterative algorithm with conjugate gradient-like directions for nonconvex optimization in deep neural networks to accelerate conventional adaptive learning rate optimization algorithms. We presented two convergence analyses of the algorithm. The first convergence analysis showed that the algorithm with a constant learning rate approximates a stationary point of a nonconvex optimization problem. The second analysis showed that the algorithm with a diminishing learning rate converges to a stationary point of the nonconvex optimization problem. We gave numerical results for concrete neural networks. The results showed that the proposed algorithm with a constant learning rate is superior for training neural networks from the viewpoints of theory and practice, while the proposed algorithm with a diminishing learning rate is not good for training neural networks. The reason behind these results is that using a constant learning rate guarantees that the algorithm works well, while a diminishing learning rate for a large number of iterations, which is approximately zero, implies that the algorithm is not updated.

Appendix A Proofs of Theorems 1 and 2 and Propositions 1 and 2

This section refers to [8]. Let us first prove the following lemma.

Lemma A1. *Suppose that (A1)–(A2) and (C1)–(C2) hold. Then, for all $x \in X$ and all $n \in \mathbb{N}$,*

$$\mathbb{E}\left[\|x_{n+1} - x\|_{\mathsf{H}_n}^2\right] \leq \mathbb{E}\left[\|x_n - x\|_{\mathsf{H}_n}^2\right] + \frac{2\alpha_n}{1-\delta^{n+1}}\Big\{(1-\beta_n)\mathbb{E}\left[\langle x - x_n, \nabla f(x_n)\rangle\right] \\ + \beta_n \mathbb{E}\left[\langle x - x_n, m_{n-1}\rangle\right] - (1-\beta_n)\gamma_n \mathbb{E}\left[\langle x - x_n, \mathbf{G}_{n-1}\rangle\right]\Big\} + \alpha_n^2 \mathbb{E}\left[\|\mathbf{d}_n\|_{\mathsf{H}_n}^2\right].$$

Proof. Choose $x \in X$ and $n \in \mathbb{N}$. The definition of x_{n+1} and the nonexpansivity of P_{X,H_n} imply that, almost surely,

$$\begin{aligned}\|x_{n+1} - x\|_{\mathsf{H}_n}^2 &\leq \|(x_n - x) + \alpha_n \mathbf{d}_n\|_{\mathsf{H}_n}^2 \\ &= \|x_n - x\|_{\mathsf{H}_n}^2 + 2\alpha_n \langle x_n - x, \mathbf{d}_n\rangle_{\mathsf{H}_n} + \alpha_n^2 \|\mathbf{d}_n\|_{\mathsf{H}_n}^2.\end{aligned}$$

The definitions of $\mathbf{d}_n$, m_n, and $\hat{m}_n$ ensure that

$$\langle x_n - x, \mathbf{d}_n\rangle_{\mathsf{H}_n} = \frac{1}{\tilde{\delta}_n}\langle x - x_n, m_n\rangle = \frac{\beta_n}{\tilde{\delta}_n}\langle x - x_n, m_{n-1}\rangle + \frac{1-\beta_n}{\tilde{\delta}_n}\langle x - x_n, \mathbf{G}_n\rangle,$$

where $\tilde{\delta}_n := 1 - \delta^{n+1}$. Moreover, the definition of $\mathbf{G}_n$ implies that

$$\langle x - x_n, \mathbf{G}_n\rangle = \langle x - x_n, \mathsf{G}(x_n, \xi_n)\rangle - \gamma_n \langle x - x_n, \mathbf{G}_{n-1}\rangle.$$

Hence, almost surely,

$$\|x_{n+1} - x\|_{\mathsf{H}_n}^2 \leq \|x_n - x\|_{\mathsf{H}_n}^2 + 2\alpha_n\Big\{\frac{\beta_n}{\tilde{\delta}_n}\langle x - x_n, m_{n-1}\rangle + \frac{1-\beta_n}{\tilde{\delta}_n}\langle x - x_n, \mathsf{G}(x_n, \xi_n)\rangle \\ - \frac{(1-\beta_n)\gamma_n}{\tilde{\delta}_n}\langle x - x_n, \mathbf{G}_{n-1}\rangle\Big\} + \alpha_n^2\|\mathbf{d}_n\|_{\mathsf{H}_n}^2. \tag{A1}$$

The conditions $x_n = x_n(\xi_{[n-1]})$ $(n \in \mathbb{N})$, (C1), and (C2) imply that

$$\begin{aligned}\mathbb{E}\left[\langle x - x_n, \mathsf{G}(x_n, \xi_n)\rangle\right] &= \mathbb{E}\left[\mathbb{E}\left[\langle x - x_n, \mathsf{G}(x_n, \xi_n)\rangle | \xi_{[n-1]}\right]\right] \\ &= \mathbb{E}\left[\left\langle x - x_n, \mathbb{E}\left[\mathsf{G}(x_n, \xi_n) | \xi_{[n-1]}\right]\right\rangle\right] \\ &= \mathbb{E}\left[\langle x - x_n, \nabla f(x_n)\rangle\right].\end{aligned}$$

Taking the expectation of (A1) leads to the assertion of Lemma A1. □

Lemma A2. *If (C3) holds, then, for all $n \in \mathbb{N}$, $\mathbb{E}[\|\mathbf{G}_n\|^2] \leq 4\hat{M}^2$ and $\mathbb{E}[\|m_n\|^2] \leq \tilde{M}^2$, where $\hat{M}^2 := \max\{M^2, \|\mathbf{G}_{-1}\|^2\}$ and $\tilde{M}^2 := \max\{\|m_{-1}\|^2, 4\hat{M}^2\}$. Moreover, if (A3) holds, then, for all $n \in \mathbb{N}$, $\mathbb{E}[\|\mathbf{d}_n\|_{\mathsf{H}_n}^2] \leq \tilde{B}^2\tilde{M}^2/(1-\delta)^2$, where $\tilde{B} := \sup\{\max_{i=1,2,\ldots,d} h_{n,i}^{-1/2} \colon n \in \mathbb{N}\} < +\infty$.*

Proof. Let us define $\hat{M}^2 := \max\{M^2, \|\mathbf{G}_{-1}\|^2\} < +\infty$, where M is defined as in (C3). Let us consider the case where $n = 0$. The inequality $\|\boldsymbol{x}+\boldsymbol{y}\|^2 \leq 2\|\boldsymbol{x}\|^2 + 2\|\boldsymbol{y}\|^2$ ($\boldsymbol{x},\boldsymbol{y} \in \mathbb{R}^d$) ensures that

$$\|\mathbf{G}_0\|^2 \leq 2\,\|\mathsf{G}(\boldsymbol{x}_0,\boldsymbol{\xi}_0)\|^2 + 2\gamma_0^2\,\|\mathbf{G}_{-1}\|^2, \tag{A2}$$

which, together with $\gamma_n \leq 1/2$ ($n \in \mathbb{N}$) and the definition of $\hat{M}$, implies that

$$\mathbb{E}\left[\|\mathbf{G}_0\|^2\right] \leq 2M^2 + 2\cdot\frac{1}{4}\cdot 4\hat{M}^2 \leq 4\hat{M}^2.$$

Assume that $\mathbb{E}[\|\mathbf{G}_n\|^2] \leq 4\hat{M}^2$ for some $n \in \mathbb{N}$. The same discussion as for (A2) ensures that

$$\mathbb{E}\left[\|\mathbf{G}_{n+1}\|^2\right] \leq 2\mathbb{E}\left[\|\mathsf{G}(\boldsymbol{x}_{n+1},\boldsymbol{\xi}_{n+1})\|^2\right] + 2\gamma_{n+1}^2\mathbb{E}\left[\|\mathbf{G}_n\|^2\right] \leq 2M^2 + 2\cdot\frac{1}{4}\cdot 4\hat{M}^2 \leq 4\hat{M}^2.$$

Accordingly, we have, for all $n \in \mathbb{N}$,

$$\mathbb{E}\left[\|\mathbf{G}_n\|^2\right] \leq 4\hat{M}^2. \tag{A3}$$

From the definition of $\boldsymbol{m}_n$, the convexity of $\|\cdot\|^2$, and (A3), for all $n \in \mathbb{N}$,

$$\mathbb{E}\left[\|\boldsymbol{m}_n\|^2\right] \leq \beta_n\mathbb{E}\left[\|\boldsymbol{m}_{n-1}\|^2\right] + (1-\beta_n)\mathbb{E}\left[\|\mathbf{G}_n\|^2\right] \leq \beta_n\mathbb{E}\left[\|\boldsymbol{m}_{n-1}\|^2\right] + 4\hat{M}^2(1-\beta_n).$$

Hence, induction leads to, for all $n \in \mathbb{N}$,

$$\mathbb{E}\left[\|\boldsymbol{m}_n\|^2\right] \leq \tilde{M}^2 := \max\left\{\|\boldsymbol{m}_{-1}\|^2, 4\hat{M}^2\right\} < +\infty. \tag{A4}$$

Given $n \in \mathbb{N}$, $\mathsf{H}_n \succ O$ ensures that there exists a unique matrix $\overline{\mathsf{H}}_n \succ O$ such that $\mathsf{H}_n = \overline{\mathsf{H}}_n^2$ [22, Theorem 7.2.6]. From $\|\boldsymbol{x}\|_{\mathsf{H}_n}^2 = \|\overline{\mathsf{H}}_n\boldsymbol{x}\|^2$ ($\boldsymbol{x} \in \mathbb{R}^d$) and the definitions of $\mathbf{d}_n$ and $\hat{\boldsymbol{m}}_n$, we have, for all $n \in \mathbb{N}$,

$$\mathbb{E}\left[\|\mathbf{d}_n\|_{\mathsf{H}_n}^2\right] = \mathbb{E}\left[\left\|\overline{\mathsf{H}}_n^{-1}\mathsf{H}_n\mathbf{d}_n\right\|^2\right] \leq \frac{1}{\tilde{\delta}_n^2}\mathbb{E}\left[\left\|\overline{\mathsf{H}}_n^{-1}\right\|^2\|\boldsymbol{m}_n\|^2\right],$$

where $\tilde{\delta}_n := 1-\delta^{n+1} \geq 1-\delta$ and $\|\overline{\mathsf{H}}_n^{-1}\| = \|\mathsf{diag}(h_{n,i}^{-1/2})\| = \max_{i=1,2,\ldots,d} h_{n,i}^{-1/2}$ ($n \in \mathbb{N}$). From (A4) and $\tilde{B} := \sup\{\max_{i=1,2,\ldots,d} h_{n,i}^{-1/2} \colon n \in \mathbb{N}\} \leq \max_{i=1,2,\ldots,d} h_{0,i}^{-1/2} < +\infty$ (by (A3)), we have, for all $n \in \mathbb{N}$,

$$\mathbb{E}\left[\|\mathbf{d}_n\|_{\mathsf{H}_n}^2\right] \leq \frac{\tilde{B}^2\tilde{M}^2}{(1-\delta)^2},$$

which completes the proof. □

The convergence rate analysis of Algorithm 1 is as follows.

Theorem A1. *Suppose that (A1)–(A5) and (C1)–(C3) hold and $(\theta_n)_{n\in\mathbb{N}}$ defined by $\theta_n := \alpha_n(1-\beta_n)/(1-\delta^{n+1})$ and $(\beta_n)_{n\in\mathbb{N}}$ satisfy $\theta_{n+1} \leq \theta_n$ ($n \in \mathbb{N}$) and $\limsup_{n\to+\infty}\beta_n < 1$. Let $V_n(\boldsymbol{x}) := \mathbb{E}\left[\langle \boldsymbol{x}_n - \boldsymbol{x}, \nabla f(\boldsymbol{x}_n)\rangle\right]$ for all $\boldsymbol{x} \in X$ and all $n \in \mathbb{N}$. Then, for all $\boldsymbol{x} \in X$ and all $n \geq 1$,*

$$\frac{1}{n}\sum_{k=1}^{n} V_k(\boldsymbol{x}) \leq \frac{D\sum_{i=1}^{d} B_i}{2\tilde{b}n\alpha_n} + \frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2 n}\sum_{k=1}^{n}\alpha_k + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}n}\sum_{k=1}^{n}\beta_k + \frac{2\sqrt{Dd}\hat{M}}{n}\sum_{k=1}^{n}\gamma_k,$$

where $(\beta_n)_{n\in\mathbb{N}} \subset (0,b] \subset (0,1)$, $\tilde{b} := 1-b$, $\tilde{\delta} := 1-\delta$, $\hat{M}$, $\tilde{M}$ and $\tilde{B}$ are defined as in Lemma A2, and D and B_i are defined as in Assumption 1.

Proof. Let $\boldsymbol{x} \in X$ be fixed arbitrarily. Lemma A1 guarantees that, for all $k \in \mathbb{N}$,

$$\begin{aligned} V_k(\boldsymbol{x}) \leq \frac{1}{2\theta_k} & \left\{ \mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}\|^2_{\mathsf{H}_k} \right] - \mathbb{E}\left[\|\boldsymbol{x}_{k+1} - \boldsymbol{x}\|^2_{\mathsf{H}_k} \right] \right\} \\ & + \frac{\beta_k}{1-\beta_k} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_k, \boldsymbol{m}_{k-1} \rangle \right] + \gamma_k \mathbb{E}\left[\langle \boldsymbol{x}_k - \boldsymbol{x}, \mathbf{G}_{n-1} \rangle \right] + \frac{\alpha_k \tilde{\delta}_k}{2(1-\beta_k)} \mathbb{E}\left[\|\mathbf{d}_k\|^2_{\mathsf{H}_k} \right], \end{aligned}$$

where $\tilde{\delta}_n := 1 - \delta^{n+1} \leq 1$ ($n \in \mathbb{N}$). The condition $\limsup_{n \to +\infty} \beta_n < 1$ ensures the existence of $b > 0$ such that, for all $n \in \mathbb{N}$, $\beta_n \leq b < 1$. Let $\tilde{b} := 1 - b$. Then, for all $n \geq 1$, we have

$$\begin{aligned} \sum_{k=1}^{n} V_k(\boldsymbol{x}) \leq \frac{1}{2} & \underbrace{\sum_{k=1}^{n} \frac{1}{\theta_k} \left\{ \mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}\|^2_{\mathsf{H}_k} \right] - \mathbb{E}\left[\|\boldsymbol{x}_{k+1} - \boldsymbol{x}\|^2_{\mathsf{H}_k} \right] \right\}}_{\Theta_n} \\ & + \underbrace{\sum_{k=1}^{n} \frac{\beta_k}{1-\beta_k} \mathbb{E}\left[\langle \boldsymbol{x} - \boldsymbol{x}_k, \boldsymbol{m}_{k-1} \rangle \right]}_{B_n} + \underbrace{\sum_{k=1}^{n} \gamma_k \mathbb{E}\left[\langle \boldsymbol{x}_k - \boldsymbol{x}, \mathbf{G}_{n-1} \rangle \right]}_{\Gamma_n} + \frac{1}{2\tilde{b}} \underbrace{\sum_{k=1}^{n} \alpha_k \mathbb{E}\left[\|\mathbf{d}_k\|^2_{\mathsf{H}_k} \right]}_{A_n}. \end{aligned} \tag{A5}$$

The definition of Θ_n and $\mathbb{E}[\|\boldsymbol{x}_{n+1} - \boldsymbol{x}\|^2_{\mathsf{H}_n}]/\theta_n \geq 0$ imply that

$$\Theta_n \leq \frac{\mathbb{E}\left[\|\boldsymbol{x}_1 - \boldsymbol{x}\|^2_{\mathsf{H}_1} \right]}{\theta_1} + \underbrace{\sum_{k=2}^{n} \left\{ \frac{\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}\|^2_{\mathsf{H}_k} \right]}{\theta_k} - \frac{\mathbb{E}\left[\|\boldsymbol{x}_k - \boldsymbol{x}\|^2_{\mathsf{H}_{k-1}} \right]}{\theta_{k-1}} \right\}}_{\tilde{\Theta}_n}. \tag{A6}$$

Accordingly,

$$\tilde{\Theta}_n = \mathbb{E}\left[\sum_{k=2}^{n} \left\{ \frac{\left\| \overline{\mathsf{H}}_k (\boldsymbol{x}_k - \boldsymbol{x}) \right\|^2}{\theta_k} - \frac{\left\| \overline{\mathsf{H}}_{k-1} (\boldsymbol{x}_k - \boldsymbol{x}) \right\|^2}{\theta_{k-1}} \right\} \right],$$

where, for all $k \in \mathbb{N}$ and all $\boldsymbol{x} := (x_i) \in \mathbb{R}^d$,

$$\overline{\mathsf{H}}_k = \mathsf{diag}\left(\sqrt{h_{k,i}} \right) \text{ and } \left\| \overline{\mathsf{H}}_k \boldsymbol{x} \right\|^2 = \sum_{i=1}^{d} h_{k,i} x_i^2. \tag{A7}$$

Thus, for all $n \geq 2$,

$$\tilde{\Theta}_n = \mathbb{E}\left[\sum_{k=2}^{n} \sum_{i=1}^{d} \left(\frac{h_{k,i}}{\theta_k} - \frac{h_{k-1,i}}{\theta_{k-1}} \right) (x_{k,i} - x_i)^2 \right].$$

The condition $\theta_k \leq \theta_{k-1}$ ($k \geq 1$) and (A3) imply that, for all $k \geq 1$ and all $i = 1, 2, \ldots, d$,

$$\frac{h_{k,i}}{\theta_k} - \frac{h_{k-1,i}}{\theta_{k-1}} \geq 0.$$

Hence, for all $n \geq 2$,

$$\tilde{\Theta}_n \leq D\mathbb{E}\left[\sum_{k=2}^{n} \sum_{i=1}^{d} \left(\frac{h_{k,i}}{\theta_k} - \frac{h_{k-1,i}}{\theta_{k-1}} \right) \right] = D\mathbb{E}\left[\sum_{i=1}^{d} \left(\frac{h_{n,i}}{\theta_n} - \frac{h_{1,i}}{\theta_1} \right) \right],$$

where $\max_{i=1,2,\ldots,d} \sup\{(x_{n,i} - x_i)^2 \colon n \in \mathbb{N}\} \leq D < +\infty$ (by (A5)). Therefore, (A6), $\mathbb{E}[\|\boldsymbol{x}_1 - \boldsymbol{x}\|^2_{\mathsf{H}_1}]/\theta_1 \leq D\mathbb{E}[\sum_{i=1}^d h_{1,i}/\theta_1]$, and (A4) imply, for all $n \in \mathbb{N}$,

$$\Theta_n \leq D\mathbb{E}\left[\sum_{i=1}^{d} \frac{h_{1,i}}{\theta_1}\right] + D\mathbb{E}\left[\sum_{i=1}^{d}\left(\frac{h_{n,i}}{\theta_n} - \frac{h_{1,i}}{\theta_1}\right)\right] = \frac{D}{\theta_n}\mathbb{E}\left[\sum_{i=1}^{d} h_{n,i}\right] \leq \frac{D}{\theta_n}\sum_{i=1}^{d} B_i,$$

which, together with $\theta_n := \alpha_n(1-\beta_n)/(1-\delta^{n+1}) \geq \tilde{b}\alpha_n$, implies

$$\Theta_n \leq \frac{D\sum_{i=1}^d B_i}{\tilde{b}\alpha_n}. \tag{A8}$$

The Cauchy-Schwarz inequality, together with $\max_{i=1,2,\ldots,d} \sup\{(x_{n,i} - x_i)^2 \colon n \in \mathbb{N}\} \leq D < +\infty$ (by (A5)) and $\mathbb{E}[\|\boldsymbol{m}_n\|] \leq \tilde{M}$ ($n \in \mathbb{N}$) (by Lemma A2), guarantees that, for all $n \in \mathbb{N}$,

$$B_n \leq \frac{\sqrt{Dd}}{\tilde{b}}\sum_{k=1}^{n} \beta_k \mathbb{E}\left[\|\boldsymbol{m}_{k-1}\|\right] \leq \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}}\sum_{k=1}^{n}\beta_k. \tag{A9}$$

A discussion similar to the one for obtaining (A9), together with $\mathbb{E}[\|\mathsf{G}_n\|] \leq 2\hat{M}$ ($n \in \mathbb{N}$) (by Lemma A2), implies that

$$\Gamma_n \leq \sqrt{Dd}\sum_{k=1}^{n}\gamma_k \mathbb{E}\left[\|\mathsf{G}_{k-1}\|\right] \leq 2\sqrt{Dd}\hat{M}\sum_{k=1}^{n}\gamma_k. \tag{A10}$$

Since $\mathbb{E}[\|\mathsf{d}_n\|^2_{\mathsf{H}_n}] \leq \tilde{B}^2\tilde{M}^2/(1-\delta)^2$ ($n \in \mathbb{N}$) holds (by Lemma A2), we have, for all $n \in \mathbb{N}$,

$$A_n := \sum_{k=1}^{n}\alpha_k \mathbb{E}\left[\|\mathsf{d}_k\|^2_{\mathsf{H}_k}\right] \leq \frac{\tilde{B}^2\tilde{M}^2}{(1-\delta)^2}\sum_{k=1}^{n}\alpha_k. \tag{A11}$$

Therefore, (A5), (A8), (A9), (A10), and (A11) leads to the assertion in Theorem A1. This completes the proof. □

Proof of Theorem 1. Let $\alpha_n := \alpha \in (0,1)$, $\beta_n := \beta = b \in (0,1)$, and $\gamma_n := \gamma \in [0, 1/2]$. We show that, for all $\epsilon > 0$ and all $\boldsymbol{x} \in X$,

$$\liminf_{n \to +\infty} V_n(\boldsymbol{x}) \leq \frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma + \frac{Dd\epsilon}{2\tilde{b}} + \epsilon. \tag{A12}$$

If (A12) does not hold for all $\epsilon > 0$ and all $\boldsymbol{x} \in X$, then there exist $\epsilon_0 > 0$ and $\hat{\boldsymbol{x}} \in X$ such that

$$\liminf_{n \to +\infty} V_n(\hat{\boldsymbol{x}}) > \frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma + \frac{Dd\epsilon_0}{2\tilde{b}} + \epsilon_0. \tag{A13}$$

Assumptions (A3) and (A4) ensure that there exists $n_0 \in \mathbb{N}$ such that, for all $n \in \mathbb{N}$, $n \geq n_0$ implies that

$$\mathbb{E}\left[\sum_{i=1}^{d}(h_{n+1,i} - h_{n,i})\right] \leq \frac{d\alpha\epsilon_0}{2}. \tag{A14}$$

Assumptions (A4) and (A5) and (A7) also imply that, for all $n \in \mathbb{N}$,

$$X_n := \mathbb{E}\left[\|\boldsymbol{x}_n - \hat{\boldsymbol{x}}\|^2_{\mathsf{H}_n}\right] = \mathbb{E}\left[\sum_{i=1}^{d} h_{n,i}(x_{n,i} - \hat{x}_i)^2\right] \leq D\sum_{i=1}^{d} B_i < +\infty. \tag{A15}$$

Moreover, Assumptions (A3) and (A5), (A7), and (A14) ensure that, for all $n \geq n_0$,

$$X_{n+1} - \mathbb{E}\left[\|\boldsymbol{x}_{n+1} - \hat{\boldsymbol{x}}\|^2_{\mathsf{H}_n}\right] = \mathbb{E}\left[\sum_{i=1}^{d}(h_{n+1,i} - h_{n,i})(x_{n+1,i} - \hat{x}_i)^2\right] \leq \frac{Dd\alpha\epsilon_0}{2}. \tag{A16}$$

The condition $\delta \in [0, 1)$ and $X_{n+1} < +\infty$ (by (A15)) ensure that there exists $n_1 \in \mathbb{N}$ such that, for all $n \in \mathbb{N}$, $n \geq n_1$ implies that

$$X_{n+1}\delta^{n+1} \leq \frac{Dd\alpha\epsilon_0}{2}. \tag{A17}$$

The definition of the limit inferior of $(V_n(\hat{\boldsymbol{x}}))_{n\in\mathbb{N}}$ guarantees that there exists $n_2 \in \mathbb{N}$ such that, for all $n \geq n_2$,

$$\liminf_{n\to+\infty} V_n(\hat{\boldsymbol{x}}) - \frac{1}{2}\epsilon_0 \leq V_n(\hat{\boldsymbol{x}}),$$

which, together with (A13), implies that, for all $n \geq n_1$,

$$V_n(\hat{\boldsymbol{x}}) > \frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma + \frac{Dd\epsilon_0}{2\tilde{b}} + \frac{1}{2}\epsilon_0. \tag{A18}$$

Thus, Lemmas A1 and A2 and (A16) lead to the finding that, for all $n \geq n_3 := \max\{n_0, n_1, n_2\}$,

$$X_{n+1} \leq X_n + \frac{Dd\alpha\epsilon_0}{2} - \frac{2\alpha\tilde{b}}{1-\delta^{n+1}}V_n(\hat{\boldsymbol{x}}) + \frac{2\sqrt{Dd}\tilde{M}}{\tilde{\delta}}\alpha\beta + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha\gamma + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha^2,$$

where $\tilde{b} := 1 - b$ and $\tilde{\delta} := 1 - \delta$. Hence, from (A17), $1 - \delta^{n+1} \leq 1$, and $(X_{n+1} - X_n)\delta^{n+1} \leq X_{n+1}\delta^{n+1}$ ($n \in \mathbb{N}$), we have, for all $n \geq n_3$,

$$\begin{aligned} X_{n+1} &\leq X_n + \frac{Dd\alpha\epsilon_0}{2} - 2\alpha\tilde{b}V_n(\hat{\boldsymbol{x}}) + \frac{2\sqrt{Dd}\tilde{M}}{\tilde{\delta}}\alpha\beta + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha\gamma + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha^2 + X_{n+1}\delta^{n+1} \\ &\leq X_n + Dd\alpha\epsilon_0 - 2\alpha\tilde{b}V_n(\hat{\boldsymbol{x}}) + \frac{2\sqrt{Dd}\tilde{M}}{\tilde{\delta}}\alpha\beta + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha\gamma + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha^2. \end{aligned} \tag{A19}$$

Therefore, (A18) ensures that, for all $n \geq n_3$,

$$\begin{aligned} X_{n+1} &< X_n + Dd\alpha\epsilon_0 - 2\alpha\tilde{b}\left\{\frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma + \frac{Dd\epsilon_0}{2\tilde{b}} + \frac{1}{2}\epsilon_0\right\} \\ &\quad + \frac{2\sqrt{Dd}\tilde{M}}{\tilde{\delta}}\alpha\beta + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha\gamma + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha^2 \\ &= X_n - \alpha\tilde{b}\epsilon_0 \\ &< X_{n_3} - \alpha\tilde{b}\epsilon_0(n + 1 - n_3). \end{aligned}$$

Since the right-hand side of the above inequality approaches minus infinity when n diverges, we have a contradiction. Hence, (A12) holds for all $\epsilon > 0$ and all $\boldsymbol{x} \in X$. From the arbitrary condition of ϵ, we have, for all $\boldsymbol{x} \in X$,

$$\liminf_{n\to+\infty} V_n(\boldsymbol{x}) \leq \frac{\tilde{B}^2\tilde{M}^2}{2\tilde{b}\tilde{\delta}^2}\alpha + \frac{\sqrt{Dd}\tilde{M}}{\tilde{b}\tilde{\delta}}\beta + \frac{2\sqrt{Dd}\hat{M}}{\tilde{\delta}}\gamma,$$

which completes the proof. □

Proof of Theorem 2. Let $\boldsymbol{x} \in X$. Lemmas A1 and A2 and (A15), together with a discussion similar to the one for obtaining (A19), ensure that, for all $k \in \mathbb{N}$,

$$X_{k+1} \leq X_k + D\mathbb{E}\left[\sum_{i=1}^{d}(h_{k+1,i} - h_{k,i})\right] - 2\alpha_k(1-\beta_k)V_k(\boldsymbol{x}) \\ + \frac{2\sqrt{Dd}\tilde{M}}{\tilde{\delta}}\alpha_k\beta_k + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha_k\gamma_k + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha_k^2 + D\sum_{i=1}^{d}B_i\delta^{k+1},$$

which implies that

$$2\alpha_k V_k(\boldsymbol{x}) \leq X_k - X_{k+1} + D\mathbb{E}\left[\sum_{i=1}^{d}(h_{k+1,i} - h_{k,i})\right] + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\alpha_k\gamma_k + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\alpha_k^2 \\ + 2\left(\frac{\sqrt{Dd}\tilde{M}}{\tilde{\delta}} + F\right)\alpha_k\beta_k + D\sum_{i=1}^{d}B_i\delta^{k+1},$$

where $F := \sup\{|V_n(\boldsymbol{x})| \colon n \in \mathbb{N}\} < +\infty$ holds from Assumptions (A2) and (A5). Summing up the above inequality from $k = 0$ to $k = n$ ensures that

$$2\sum_{k=0}^{n}\alpha_k V_k(\boldsymbol{x}) \leq X_0 + D\mathbb{E}\left[\sum_{i=1}^{d}(h_{n+1,i} - h_{0,i})\right] + \frac{4\sqrt{Dd}\hat{M}\tilde{b}}{\tilde{\delta}}\sum_{k=0}^{n}\alpha_k\gamma_k + \frac{\tilde{B}^2\tilde{M}^2}{\tilde{\delta}^2}\sum_{k=0}^{n}\alpha_k^2 \\ + 2\left(\frac{\sqrt{Dd}\tilde{M}}{\tilde{\delta}} + F\right)\sum_{k=0}^{n}\alpha_k\beta_k + D\hat{B}\sum_{k=0}^{n}\delta^{k+1},$$

where $\hat{B} := \sum_{i=1}^{d} B_i$. Let $(\alpha_n)_{n\in\mathbb{N}}$, $(\beta_n)_{n\in\mathbb{N}}$, and $(\gamma_n)_{n\in\mathbb{N}}$ satisfy $\sum_{n=0}^{+\infty}\alpha_n = +\infty$, $\sum_{n=0}^{+\infty}\alpha_n^2 < +\infty$, $\sum_{n=0}^{+\infty}\alpha_n\beta_n < +\infty$, and $\sum_{n=0}^{+\infty}\alpha_n\gamma_n < +\infty$. Assumption (A4) and $\delta \in [0,1)$ imply that

$$\sum_{k=0}^{+\infty}\alpha_k V_k(\boldsymbol{x}) < +\infty. \tag{A20}$$

We prove that, for all $\boldsymbol{x} \in X$, $\liminf_{n\to+\infty} V_n(\boldsymbol{x}) \leq 0$. Assume that $\liminf_{n\to+\infty} V_n(\boldsymbol{x}) \leq 0$ does not hold for all $\boldsymbol{x} \in X$. Then there exist $\hat{\boldsymbol{x}} \in X$, $\zeta > 0$, and $m_0 \in \mathbb{N}$ such that, for all $n \geq m_0$, $V_n(\hat{\boldsymbol{x}}) \geq \zeta$. Accordingly, (A20) and $\sum_{n=0}^{+\infty}\alpha_n = +\infty$ guarantee that

$$+\infty = \zeta\sum_{k=m_0}^{+\infty}\alpha_k \leq \sum_{k=m_0}^{+\infty}\alpha_k V_k(\hat{\boldsymbol{x}}) < +\infty,$$

which is a contradiction. Hence, $\liminf_{n\to+\infty} V_n(\boldsymbol{x}) \leq 0$ holds for all $\boldsymbol{x} \in X$.

Let $\alpha_n := 1/n^\eta$ ($\eta \in [1/2, 1)$) and $\beta_n := \beta^n$ ($\beta \in (0,1)$). First, we consider the case where $\gamma_n := \gamma^n$ ($\gamma \in (0,1)$).[7] Then, $\theta_{n+1} \leq \theta_n$ ($n \in \mathbb{N}$) and $\limsup_{n\to+\infty}\beta_n < 1$. When $\eta = 1/2$, we have

$$\frac{1}{n\alpha_n} = \frac{1}{\sqrt{n}}$$

[7] Footnote 2 implies that $\gamma_n \leq 1/2$ ($n \geq k_0$) and $\max\{M^2, \|\mathsf{G}_{k_0}\|\} < +\infty$. Accordingly, Theorem A1 holds for all $n \geq k_0$. Since Theorem 2 discusses the convergence of Algorithm 1, we may assume, without loss of generality, that Theorem A1 holds for all $n \geq 1$. Or, we may replace $\gamma \in (0,1)$ with $\gamma \in (0, 1/2]$.

and

$$\frac{1}{n}\sum_{k=1}^{n}\alpha_k \leq \frac{1}{n}\sqrt{\sum_{k=1}^{n}1^2}\sqrt{\sum_{k=1}^{n}\left(\frac{1}{\sqrt{k}}\right)^2} \leq \sqrt{\frac{1+\ln n}{n}},$$

where the first inequality comes from the Cauchy-Schwarz inequality and the second inequality comes from $\sum_{k=1}^{n}(1/k) \leq 1+\ln n$. We also have

$$\frac{1}{n}\sum_{k=1}^{n}\beta_k \leq \frac{1}{n}\sum_{k=1}^{+\infty}\beta^k = \frac{\beta}{(1-\beta)n} \text{ and } \frac{1}{n}\sum_{k=1}^{n}\gamma_k \leq \frac{1}{n}\sum_{k=1}^{+\infty}\gamma^k = \frac{\gamma}{(1-\gamma)n}. \tag{A21}$$

Therefore, Theorem A1 implies that

$$\frac{1}{n}\sum_{k=1}^{n}V_k(\boldsymbol{x}) \leq \mathcal{O}\left(\sqrt{\frac{1+\ln n}{n}}\right).$$

In the case where $\eta \in (1/2, 1)$, we have

$$\frac{1}{n\alpha_n} = \frac{1}{n^{1-\eta}} \text{ and } \frac{1}{n}\sum_{k=1}^{n}\alpha_k \leq \frac{1}{n}\sqrt{\sum_{k=1}^{n}1^2}\sqrt{\sum_{k=1}^{n}\left(\frac{1}{k^\eta}\right)^2} \leq \frac{B}{\sqrt{n}}, \tag{A22}$$

where $B := \sum_{n=1}^{+\infty}(1/k^{2\eta}) < +\infty$. Therefore, Theorem A1, together with (A21), ensures that

$$\frac{1}{n}\sum_{k=1}^{n}V_k(\boldsymbol{x}) \leq \mathcal{O}\left(\frac{1}{n^{1-\eta}}\right).$$

Next, we consider the case where $\gamma_n := 1/n^\kappa$ ($\kappa > 1-\eta$). Since $\kappa > 1/2$ holds, an argument similar to the one for obtaining (A22) implies that

$$\frac{1}{n}\sum_{k=1}^{n}\gamma_k = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right).$$

The discussion in the above paragraph and Theorem A1 lead to the same convergence rate of $(1/n)\sum_{k=1}^{n}V_k(\boldsymbol{x})$ as the one for $\gamma_n := \gamma^n$ ($\gamma \in (0,1)$). This completes the proof. □

Proof of Proposition 1. Since $F(\cdot, \boldsymbol{\xi})$ is convex for almost every $\boldsymbol{\xi} \in \Xi$, we have, for all $n \in \mathbb{N}$,

$$\begin{aligned}
&\mathbb{E}[f(\boldsymbol{x}_n) - f^\star] \leq V_n(\boldsymbol{x}^\star),\\
&\mathbb{E}[f(\tilde{\boldsymbol{x}}_n) - f^\star] \leq \frac{1}{n}\sum_{k=1}^{n}\mathbb{E}[f(\boldsymbol{x}_k) - f^\star] \leq \frac{1}{n}\sum_{k=1}^{n}V_k(\boldsymbol{x}^\star),
\end{aligned}$$

which, together with Theorem 1, leads to Proposition 1. □

Proof of Proposition 2. Theorem 2 and the proof of Proposition 1 lead to the finding that $\liminf_{n\to+\infty}\mathbb{E}[f(\boldsymbol{x}_n) - f^\star] = 0$ and $\lim_{n\to+\infty}\mathbb{E}[f(\tilde{\boldsymbol{x}}_n) - f^\star] = 0$. Let $\hat{\boldsymbol{x}} \in X$ be an arbitrary accumulation point of $(\tilde{\boldsymbol{x}}_n)_{n\in\mathbb{N}} \subset X$. Since there exists $(\tilde{\boldsymbol{x}}_{n_i})_{i\in\mathbb{N}} \subset (\tilde{\boldsymbol{x}}_n)_{n\in\mathbb{N}}$ such that $(\tilde{\boldsymbol{x}}_{n_i})_{i\in\mathbb{N}}$ converges almost surely to $\hat{\boldsymbol{x}}$, the continuity of f and $\lim_{n\to+\infty}\mathbb{E}[f(\tilde{\boldsymbol{x}}_n) - f^\star] = 0$ imply that $\mathbb{E}[f(\hat{\boldsymbol{x}}) - f^\star] = 0$, and hence, $\hat{\boldsymbol{x}} \in X^\star$. The convergence rate of $\mathbb{E}[f(\tilde{\boldsymbol{x}}_n) - f^\star]$ follows from Theorem A1. □

Author Contributions: Conceptualization, H. Iiduka; methodology, H. Iiduka; software, Y. Kobayashi; validation, H. Iiduka and Y. Kobayashi; formal analysis, H. Iiduka; investigation, H. Iiduka and Y. Kobayashi; resources, H. Iiduka; data curation, Y. Kobayashi; writing–original draft preparation, H. Iiduka; writing–review and editing,

H. Iiduka; visualization, H. Iiduka and Y. Kobayashi; supervision, H. Iiduka; project administration, H. Iiduka; funding acquisition, H. Iiduka. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by JSPS KAKENHI, Grant Number JP18K11184.

Acknowledgments: The authors would like to thank Michelle Zhou for giving us a chance to submit our paper to this journal. We are sincerely grateful to Assistant Editor Elliot Guo and the two referees for helping us improve the original manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Caciotta, M.; Giarnetti, S.; Leccese, F. Hybrid neural network system for electric load forecasting of telecomunication station. 19th IMEKO World Congress 2009, 2009, Vol. 1, pp. 657–661.
2. Caciotta, M.; Giarnetti, S.; Leccese, F.; Orioni, B.; Oreggia, M.; Pucci, C.; Rametta, S. Flavors mapping by Kohonen network classification of Panel Tests of Extra Virgin Olive Oil. *Measurement* **2016**, *78*, 366–372.
3. Proietti, A.; Liparulo, L.; Leccese, F.; Panella, M. Shapes classification of dust deposition using fuzzy kernel-based approaches. *Measurement* **2016**, *77*, 344–350.
4. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press, Cambridge, 2016.
5. Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* **2011**, *12*, 2121–2159.
6. Kingma, D.P.; Ba, J.L. Adam: A method for stochastic optimization. Proceedings of The International Conference on Learning Representations, 2015, pp. 1–15.
7. Reddi, S.J.; Kale, S.; Kumar, S. On the convergence of Adam and beyond. Proceedings of The International Conference on Learning Representations, 2018, pp. 1–23.
8. Iiduka, H. Appropriate learning rates of adaptive learning rate optimization algorithms for training deep neural networks. arXiv:2002.09647.
9. Hager, W.H.; Zhang, H. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization* **2006**, *2*, 35–58.
10. Iiduka, H. Acceleration method for convex optimization over the fixed point set of a nonexpansive mapping. *Mathematical Programming* **2015**, *149*, 131–165.
11. Iiduka, H. Hybrid conjugate gradient method for a convex optimization problem over the fixed-point set of a nonexpansive mapping. *Journal of Optimization Theory and Applications* **2009**, *140*, 463–475.
12. Iiduka, H.; Yamada, I. A use of conjugate gradient direction for the convex optimization problem over the fixed point set of a nonexpansive mapping. *SIAM Journal on Optimization* **2009**, *19*, 1881–1893.
13. Iiduka, H. Three-term conjugate gradient method for the convex optimization problem over the fixed point set of a nonexpansive mapping. *Applied Mathematics and Computation* **2011**, *217*, 6315–6327.
14. Kobayashi, Y.; Iiduka, H. Conjugate-gradient-based Adam for stochastic optimization and its application to deep learning. arXiv:2003.00231.
15. Bauschke, H.H.; Combettes, P.L. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*; Springer: New York, 2011.
16. Facchinei, F.; Pang, J.S. *Finite-Dimensional Variational Inequalities and Complementarity Problems I*; Springer, New York, 2003.
17. Nemirovski, A.; Juditsky, A.; Lan, G.; Shapiro, A. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization* **2009**, *19*, 1574–1609.
18. Polyak, B.T. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics* **1964**, *4*, 1–17.
19. Sutskever, I.; Martens, J.; Dahl, G.; Hinton, G. On the importance of initialization and momentum in deep learning. Proceedings of the 30 th International Conference on Machine Learning, 2013, pp. 1–14.
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
21. Iiduka, H. Stochastic fixed point optimization algorithm for classifier ensemble. *IEEE Transactions on Cybernetics* **2020**, *50*, 4370–4380.
22. Horn, R.A.; Johnson, C.R. *Matrix Analysis*; Cambridge University Press, Cambridge, 1985.