Improving Deep Neural Networks' Training for Image Classification with Nonlinear Conjugate Gradient-style Adaptive Momentum

Bao Wang and Qiang Ye

***Abstract*—Momentum is crucial in stochastic gradient-based optimization algorithms for accelerating or improving training deep neural networks (DNNs). In deep learning practice, the momentum is usually weighted by a well-calibrated constant. However, tuning the hyperparameter for momentum can be a significant computational burden. In this paper, we propose a novel *adaptive momentum* for improving DNNs training; this adaptive momentum, with no momentum-related hyperparameter required, is motivated by the nonlinear conjugate gradient (NCG) method. Stochastic gradient descent (SGD) with this new adaptive momentum eliminates the need for the momentum hyperparameter calibration, allows using a significantly larger learning rate, accelerates DNN training, and improves the final accuracy and robustness of the trained DNNs. For instance, SGD with this adaptive momentum reduces classification errors for training ResNet110 for CIFAR10 and CIFAR100 from 5.25% to 4.64% and 23.75% to 20.03%, respectively. Furthermore, SGD, with the new adaptive momentum, also benefits adversarial training and hence improves the adversarial robustness of the trained DNNs.**

***Index Terms*—Deep learning, image classification, adaptive momentum, nonlinear conjugate gradient.**

I. Introduction

Given a training dataset $\Omega_N := \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ with $\boldsymbol{x}_i$ and y_i being the data-label pair of the ith instance, natural training, i.e., training a machine learning (ML) classifier $y = g(\boldsymbol{x}, \boldsymbol{w})$, can be formulated as solving the following empirical risk minimization (ERM) problem [50]:

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) := \frac{1}{N}\sum_{i=1}^N f_i(\boldsymbol{w}) := \frac{1}{N}\sum_{i=1}^N \mathcal{L}(g(\boldsymbol{x}_i, \boldsymbol{w}), y_i), \quad (1)$$

where $\mathcal{L}$ is a loss function, e.g., cross-entropy, between the prediction $g(\boldsymbol{x}_i, \boldsymbol{w})$ and the ground truth y_i. Moreover, we can solve the following empirical adversarial risk minimization (EARM) problem to train an adversarially robust model [29]:

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^N \max_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_2\leq\epsilon} \mathcal{L}(g(\boldsymbol{x}_i', \boldsymbol{w}), y_i), \quad (2)$$

Manuscript received 9 September 2021; revised 19 October 2022; accepted 8 March 2023. The work of Bao Wang was supported in part by NSF under Grant DMS-1924935, Grant DMS-1952339, Grant DMS-2110145, Grant DMS-2152762, and Grant DMS-2208361; and in part by the Office of Science of the Department of Energy under Grant DE-SC0021142 and Grant DE-SC0023490. The work of Qiang Ye was supported by NSF under Grant DMS-1821144 and Grant DMS-2208314. *(Corresponding author: Bao Wang.)*

B. Wang is with the Department of Mathematics and Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT, 84112 USA e-mail: wangbaonj@gmail.com.

Q. Ye is with the Department of Mathematics, University of Kentucky, Lexington, KY 40506 USA e-mail: qye3@uky.edu.

where $\epsilon > 0$ is a constant. Training deep neural networks (DNNs) by solving (1) or (2) is challenging: 1) the objective function is highly nonconvex [24]; 2) N is very large, e.g., in ImageNet classification $N \sim 10^6$ [44], making computing the gradient of the loss function difficult and inefficient; 3) the dimension of $\boldsymbol{w}$ is very high; for instance, in training ResNet200 for ImageNet classification, $\boldsymbol{w}$ is of dimension $\sim 65M$ [17]. Due to the above challenges, stochastic gradient descent (SGD) becomes the method of choice for training DNNs for image classification [3]; momentum scaled by a well-calibrated scalar is usually integrated with SGD to accelerate or improve training DNNs [2, 38, 40, 48, 52].

Starting from $\boldsymbol{w}_0, \boldsymbol{p}_0(=\boldsymbol{0}) \in \mathbb{R}^d$, in the nth iteration (with $n \geq 1$) of SGD with momentum (scaled by a constant $\beta \geq 0$), we randomly sample a mini-batch $\{i_k\}_{k=1}^m \subset [N]$ ($m \ll N$); update $\boldsymbol{w}_n$ as follows [38]:

$$\boldsymbol{p}_n = \beta\boldsymbol{p}_{n-1} + \frac{1}{m}\sum_{k=1}^m \nabla f_{i_k}(\boldsymbol{w}_n); \quad \boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \alpha\boldsymbol{p}_n, \quad (3)$$

where $\alpha > 0$ is the step size, and $\nabla f_{i_k}(\boldsymbol{w}_n)$ can be replaced by $\nabla f_{i_k}(\boldsymbol{w}_n - \alpha\boldsymbol{p}_{n-1})$ to get the Nesterov momentum (NM) [48]. In training DNNs, *the hyperparameters α and β are manually tuned, which is time-consuming.* As a result, several adaptive step size algorithms have been developed and are widely used: Adagrad [9] adapts step size to the parameters based on the sum of the squares of the gradients; Adadelta [54] and RMSprop [20] modify Adagrad by restricting the window of accumulated past gradients to some fixed size; Adam integrates momentum with adaptive step size and achieves remarkable performances in many applications [21, 8, 28, 42]. However, there is limited work on developing adaptive strategies for β. It is shown in [40] that for any $0 \leq \beta < 1$ and $0 < \alpha < 2(1+\beta)/\lambda_{\max}$, the momentum method converges locally. Furthermore, the optimal convergence rate

$$\rho^* = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \quad (4)$$

is obtained when $\alpha = \alpha^* := 4/(\sqrt{\lambda_{\max}} + \sqrt{\lambda_{\min}})^2$ and $\beta = \beta^* := (\sqrt{\lambda_{\max}} - \sqrt{\lambda_{\min}})^2/(\sqrt{\lambda_{\max}} + \sqrt{\lambda_{\min}})^2$; where $\kappa = \lambda_{\max}/\lambda_{\min}$ and $\lambda_{\min}$ and $\lambda_{\max}$ are, respectively, the minimum and the maximum eigenvalues of the Hessian $\nabla^2 f(\boldsymbol{w}^*)$ at a local minimizer $\boldsymbol{w}^*$. Despite this elegant convergence theory, the optimal values α^* and β^* are not available in practice.

In this work, we tackle the problem of choosing the momentum weight β by observing that the classical (linear) conjugate gradient (CG) method [45] can be considered as a

generalization of the momentum method where β is chosen to enforce local orthogonality among the search directions $\boldsymbol{p}$ and α is chosen to minimize the objective function in the search direction. CG converges at the rate $\rho^* = (\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$, matching the rate of the classical momentum method for quadratic optimization problems. The nonlinear generalization of CG — nonlinear conjugate gradient (NCG) [10] — uses the same formulation of β as in CG (or an equivalent version) and determines α through line search, i.e., $\alpha_k := \operatorname{argmin}_\alpha f(\boldsymbol{w}_k - \alpha \boldsymbol{p}_k)$. NCG also significantly accelerates the convergence of gradient descent, and it has some nice theoretical convergence guarantees [1, 11, 6, 15, 41, 57]. A major obstacle to using NCG is the need for line search at each iteration; even an inexact search requires several function/derivative evaluations. As a result, it is not used very often in DNN applications [23].

A. Our Contributions

We recognize that the formulation of β in NCG is crucial to its success in accelerating GD/SGD, particularly in improving training DNNs for image classification. *The theory of NCG methods motivates us to propose an adaptive formulation of β using the Fletcher-Reeves (FR) formulation [10].* We call GD/SGD with an adaptive formulation of β an adaptive momentum method. *We establish some convergence results to show the global convergence of the proposed adaptive momentum method under certain conditions on the learning rate. For quadratic functions, we will show the accelerated convergence rate of the proposed adaptive momentum method under some mild conditions.* The major advantages of SGD with adaptive momentum are threefold:

- It converges faster and allows us to use significantly larger step sizes than the existing benchmark algorithms to train DNNs.
- Compared to the benchmark stochastic optimization algorithms, it significantly improves the accuracy and adversarial robustness of the trained DNNs for image classification. For instance, it reduces test errors of training ResNet110 for CIFAR10 and CIFAR100 classification from 5.25% to 4.64% and 23.75% to 20.03%, respectively. Furthermore, SGD with the new adaptive momentum benefits adversarial training and improves the robustness of the trained DNNs.
- It eliminates the work for momentum-related hyperparameter tuning with almost no computational overhead. It can be implemented simply by adapting the existing momentum optimization codes.

B. Related Works

Momentum scaled by an iteration-dependent scalar has been used to accelerate GD. One of the most prominent results is the Nesterov's accelerated gradient (NAG) [33, 32], which achieves a convergence rate of $O(1/n^2)$ with n being the number of iterations, for convex optimization (vs. GD with a convergence rate of $O(1/n)$). Both adaptive and scheduled restarts can further accelerate NAG with provable guarantees in certain circumstances [31, 12, 47, 37, 43]. However, directly applying NAG to SGD suffers from error accumulation [7, 52], which can be alleviated by using NAG with scheduled restart [52, 34, 53, 35] at the cost of hyperparameter calibration.

NCG [10] is a popular optimization method that has been studied extensively. For various formulations, it has been proved for a general function to have a descent property and global convergence under some assumptions on the line search known as Wolf conditions; see [1, 6, 11, 15, 41, 57]. NCG has been applied to deep learning; [23] empirically compares NCG, L-BFGS, and the momentum methods and found each to be superior in some problems. There are some related works in avoiding the line search in NCG, e.g., [30] uses some estimate of the Hessian to approximate the optimal step size α.

C. Organization

We organize this paper as follows: In Section II, we give a brief review of NCG, and then we leverage the adaptive momentum to accelerate GD and SGD. In Section III, we give the convergence guarantees of the proposed algorithms. We present the practical performance of SGD with adaptive momentum for training DNNs in Section IV. Technical proofs are provided in Sections VI and VII.

D. Notations

We denote scalars by lower or upper case letters; vectors/ matrices by lower/upper case boldface letters. For a vector $\boldsymbol{x} = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d$, we use $\|\boldsymbol{x}\| = (\sum_{i=1}^d |x_i|^2)^{1/2}$ to denote its ℓ_2 norm, and denote the ℓ_∞ norm of $\boldsymbol{x}$ by $\|\boldsymbol{x}\|_\infty = \max_{i=1}^d |x_i|$. For a matrix $\mathbf{A}$, we use $\|\mathbf{A}\|_{2/\infty}$ to denote its induced norm by the vector $\ell_{2/\infty}$ norm. We denote the set $\{1, 2, \ldots, N\}$ as $[N]$. For a function $f(\boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}$, we denote $\nabla f(\boldsymbol{w})$ and $\nabla^2 f(\boldsymbol{w})$ for the gradient and Hessian of $f(\boldsymbol{w})$, respectively.

II. Algorithm: GD/SGD with Adaptive Momentum

A. Nonlinear Conjugate Gradient Methods

The classical GD with an optimal learning rate (step size) has a local convergence rate that depends on the condition number κ of the Hessian matrix at a local minimum. The conjugate gradient (CG) method augments the gradient with a suitable momentum term as the search direction. In the quadratic case, i.e., $\min_{\boldsymbol{w}} f(\boldsymbol{w} \in \mathbb{R}^d) := 1/2 \boldsymbol{w}^\top \mathbf{A} \boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}$ with $\mathbf{A} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{b} \in \mathbb{R}^d$ being known. The modified directions of CG maintain orthogonality in the $\mathbf{A}$-inner product $\boldsymbol{w}^\top \mathbf{A} \boldsymbol{w}$, and it yields a significantly accelerated convergence rate $\rho^* = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$ that depends on $\sqrt{\kappa}$. CG has been generalized to general nonlinear functions as follows:

- In the first iteration, perform a line search along the direction $\boldsymbol{p}_0 := \nabla f(\boldsymbol{w}_0)$ to get the initial step size, i.e., $\alpha_0 := \arg\min_\alpha f(\boldsymbol{w}_0 - \alpha \boldsymbol{p}_0)$, and update $\boldsymbol{w}$ by $\boldsymbol{w}_1 = \boldsymbol{w}_0 - \alpha_0 \boldsymbol{p}_0$.
- For the nth iteration, we perform the following updates:
  - Compute
$$\beta_n = \beta_n^{FR} := \frac{(\nabla f(\boldsymbol{w}_n)^\top \nabla f(\boldsymbol{w}_n))}{(\nabla f(\boldsymbol{w}_{n-1})^\top \nabla f(\boldsymbol{w}_{n-1}))}.$$
  - Update the search direction:
$$\boldsymbol{p}_n = \nabla f(\boldsymbol{w}_n) + \beta_n \boldsymbol{p}_{n-1}.$$

– Perform a line search:

$$\alpha_n = \arg\min_{\alpha} f(\boldsymbol{w}_n - \alpha \boldsymbol{p}_n).$$

– Update the position, i.e., the weights of the model:

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \alpha_n \boldsymbol{p}_n.$$

There are several possible formulations of β_n in the literature, and the one we present β_n^{FR} is known as the Fletcher-Reeves formula [10]; see [19, 6, 39, 46, 15] for other formulations and related theoretical properties. The NCG has been empirically found to have some similar convergence properties as the classical linear CG method. There have been several analyses to show a descent property and convergence of NCG for a general function under some forms of the Wolf conditions for inexact line search of α_n; see [1, 11, 6, 15, 41, 57]. However, there appears to be no result characterizing its CG-like accelerated convergence rate.

In the NCG method, a line search is performed to determine α_n, while β_n can be regarded as the momentum coefficient. Here, the adaptive momentum coefficient is determined by the gradient at the current and previous iterations. This has an advantage over the traditional momentum method in that the momentum coefficient is adaptively determined, and no tuning is needed. A disadvantage of NCG is that even an inexact line search for α_n requires several function/gradient evaluations and would make the method less appealing for training DNNs. Therefore, NCG is rarely used for DNNs.

B. (Stochastic) Gradient Descent with NCG Momentum

CG yields the optimal convergence rate of the momentum method without the spectral information needed to determine the optimal α and β. Although this is achieved with a line search, the formulation of the coefficient β_n should play a significant role as well. This motivates us to integrate β_n of CG into GD/SGD. We propose an adaptive momentum for GD/SGD, i.e., with a fixed α but with $\beta = \beta_n^{FR}$ as the momentum coefficient at each step. This has two potential benefits: (1) As a generalization of NCG, this may retain some acceleration effects of NCG. (2) As a momentum method, there is no need to determine or tune the hyperparameter for momentum. We have chosen the Fletcher-Reeves formula β_n^{FR} for its simplicity and robustness, as indicated by our preliminary numerical testing. We call the resulting algorithm FRGD/FRSGD, which we state as follows: Starting with $\boldsymbol{w}_0$, we set $\boldsymbol{p}_{-1} = \boldsymbol{0}$ and $\beta_0 = 0$ and iterate for $n \geq 0$ as follows:

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_n \boldsymbol{p}_{n-1}; \quad \boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \alpha \boldsymbol{p}_n, \tag{5}$$

where

$$\boldsymbol{r}_n = \begin{cases} \nabla f(\boldsymbol{w}_n), & \text{for FRGD} \\ \frac{1}{m}\sum_{j=1}^{m} \nabla f_{i_j}(\boldsymbol{w}_n), & \text{for FRSGD,} \end{cases}$$

and

$$\beta_n = (\boldsymbol{r}_n^\top \boldsymbol{r}_n)/(\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}).$$

We have found that this adaptive momentum method significantly accelerates the convergence of GD with momentum and outperforms NAG as well. The only extra computational cost over the momentum method is in computing an inner product $\boldsymbol{r}_n^\top \boldsymbol{r}_n$ at each step, which is negligible. Before testing FRGD/FRSGD in training DNNs, we first present an academic example to illustrate its potential advantage.

EXAMPLE 2.1: We consider the following quadratic optimization problem [16]:

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top \mathbf{L} \boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}, \tag{6}$$

where $\mathbf{L} \in \mathbb{R}^{500\times 500}$ is the Laplacian of a cycle graph, and $\boldsymbol{b}$ is a 500-dimensional vector whose first entry is 1 and all the other entries are 0. It is easy to see that $f(\boldsymbol{w})$ is convex (not strongly convex) with Lipschitz constant 4. We run GD, GD with momentum scaled by 0.9 (GD + Momentum), NAG, and FRGD with step size $1/4$ (the same hyperparameters as that used in [16]). As shown in Fig. 1, GD + Momentum converges faster than GD, while NAG speeds up GD + Momentum dramatically and converges to the minimum in an oscillatory fashion. Moreover, FRGD converges exponentially fast and significantly outperforms all other methods in this case.

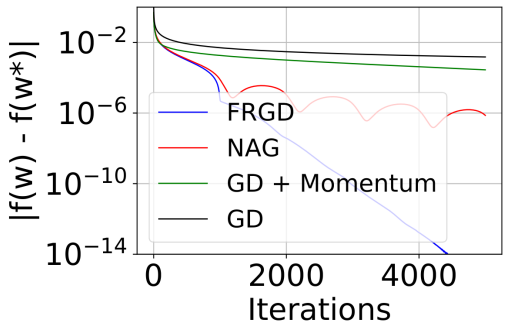


Fig. 1: Comparison between a few optimization algorithms for optimizing the quadratic function $f(\boldsymbol{w}) = 1/2\boldsymbol{w}^\top \mathbf{L}\boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}$, where $\mathbf{L} \in \mathbb{R}^{500\times 500}$ is the Laplacian of a cycle graph, and $\boldsymbol{w}, \boldsymbol{b} \in \mathbb{R}^{500}$. In particular, $\boldsymbol{b}$ is the vector whose first entry is 1 and all the others are 0s [16]. Momentum accelerates GD slightly; NAG oscillates to the minimum, $\boldsymbol{w}^*$, and converges much faster than GD (with momentum); FRGD converges exponentially fast to $\boldsymbol{w}^*$.

This example demonstrates that FRGD converges at a rate much faster than GD; we will present some theoretical results to demonstrate this property in the next section.

III. Main Theory

In this section, we present some theoretical results to demonstrate the descent property and convergence of FRGD and FRSGD. For FRGD, we shall focus on strongly convex functions and quadratic functions. Our results are applicable only locally for a general function. For the case of quadratic functions, we present convergence bounds to demonstrate an accelerated convergence rate. For FRSGD, we establish its convergence guarantee for general nonconvex functions with a small modification of the original scheme. The proofs of these results are provided in Sections VI and VII.

A. Convergence of FRGD

We first consider a convex function $f(\boldsymbol{w})$ with Lipschitz continuous Hessian matrix.

Theorem 1 (Convergence of FRGD for strongly convex functions)**.** *Consider applying the adaptive momentum method FRGD* (5) *for finding the minimum of* $f(\boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}$. *Assume that the Hessian matrix* $\mathbf{H}(\boldsymbol{w}) = [\frac{\partial^2 f(\boldsymbol{w})}{\partial w_i \partial w_j}]$ *is Lipschitz continuous with the Lipschiz constant* C *(i.e.,* $\|\mathbf{H}(\boldsymbol{w}) - \mathbf{H}(\tilde{\boldsymbol{w}})\| \leq C\|\boldsymbol{w} - \tilde{\boldsymbol{w}}\|$*) and its eigenvalues are in the interval* $[\lambda_{\min}, \lambda_{\max}]$ *with* $\lambda_{\min} > 0$*, i.e.,* $f(\boldsymbol{w})$ *is strongly convex. Assume* $\alpha \leq \frac{\lambda_{\min}}{(\lambda_{\max}^2 + C\|\boldsymbol{r}_0\|)K^2}$ *for some* $K > 0$*. Then*

$$\boldsymbol{p}_n^\top \boldsymbol{r}_n > 0$$

and

$$\|\boldsymbol{r}_n\| \leq \sqrt{1 - \alpha\lambda_{\min}}\|\boldsymbol{r}_{n-1}\|,$$

where $\boldsymbol{r}_n = \nabla f(\boldsymbol{w}_n)$ *and* $n \leq K$.

Theorem 1 shows that $\boldsymbol{p}_n$ is a descent direction and $\boldsymbol{r}_n$ converges monotonically with a rate of at least $\sqrt{1 - \alpha\lambda_{\min}}$. Although such properties are expected for GD, it is important that with the adaptive momentum, FRGD maintains these properties. However, NAG does not converge monotonically to the minimum; instead, it oscillates [47]. Moreover, it is worth noting that: 1) our assumption on the Hessian is different from the convergence theory of GD, we need Lipschitz Hessian while GD requires bounded Hessian; 2) both the step size constraint and the convergence rate are also different from that of GD.

1) Quadratic Functions: To further study the convergence rate FRGD, we consider a quadratic function $f(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top \mathbf{A}\boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}$, where $\mathbf{A}$ is a positive definite matrix. In this case, the optimization problem reduces to solving the linear system $\mathbf{A}\boldsymbol{w} = \boldsymbol{b}$. The classical CG method has been studied extensively for solving the linear system, and strong convergence results exist. With a fixed α, most properties that the analysis of CG relies on are no longer held. Fortunately, a technique used for the analysis of inexact CG due to round-off errors or inexact preconditioning [13, 49] can be adapted to our method. Using coupled two-term recurrences, our adaptive momentum becomes a Krylov subspace method, and we can derive the following convergence rate.[1]

Theorem 2 (Convergence of FRGD for quadratic functions at the beginning phase)**.** *Consider applying the adaptive momentum method FRGD* (5) *for minimizing* $f(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top \mathbf{A}\boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}$*. Let* $\boldsymbol{r}_n = \nabla f(\boldsymbol{w}_n) = \mathbf{A}\boldsymbol{w}_n - \boldsymbol{b}$*,* $\boldsymbol{z}_n = \boldsymbol{r}_n/\|\boldsymbol{r}_n\|$*, and* $\mathbf{Z}_n = [\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_{n-1}]$*. If* $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ *are linearly independent, then*

$$\|\boldsymbol{r}_n\| \leq 2(1 + K_n)\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^n \|\boldsymbol{r}_0\|, \tag{7}$$

where $K_n \leq n(1 + n\rho/2)\|\mathbf{A}\|\kappa(\mathbf{Z}_{n+1})$*,* $\rho = \max_{0 \leq j < i \leq n-1} \|\boldsymbol{r}_i\|^2/\|\boldsymbol{r}_j\|^2$*, and* $\kappa = \kappa(\mathbf{A})$ *is the spectral condition number of* $\mathbf{A}$.

The bound in (7) contains a linearly converging term with the rate $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$, but it also depends on the term K_n, which may grow with n. The key term in K_n is $\kappa(\mathbf{Z}_{n+1}) = \|\mathbf{Z}_{n+1}\|\|\mathbf{Z}_{n+1}^+\| \approx \sqrt{n+1}\|\mathbf{Z}_{n+1}^+\|$, which measures the linear independence among $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$. Thus, at the beginning phase of iterations, as long as $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ maintains some level of linear independence property as measured by the magnitude of $\kappa(\mathbf{Z}_{n+1})$, K_n may be a modestly increasing term so that $\|\boldsymbol{r}_n\|$ converges at a rate close to $(\sqrt{\kappa} - 1)/(\sqrt{\kappa} + 1)$. Note that ρ may be expected to be bounded. In particular, if $\|\boldsymbol{r}_n\|$ is monotonic, which holds under the condition of Theorem 1, then $\rho \leq 1$.

When the number of iterations n is large, the sequence $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ is expected to lose the linear independence property. Here, we present one way to circumvent this difficulty by considering the last $n-m$ iterates $\boldsymbol{z}_m, \boldsymbol{z}_{m+1}, \ldots, \boldsymbol{z}_n$, which can be expected to be linearly independent for any $m < n$ as long as $n - m$ is sufficiently small. In that case, the proof of Theorem 2 can be modified to obtain a similar but weaker bound, given in the following Corollary.

Corollary 1 (Asymptotic convergence of FRGD for quadratic functions)**.** *Consider applying the adaptive momentum method FRGD* (5) *to minimize* $f(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top \mathbf{A}\boldsymbol{w} - \boldsymbol{b}^\top \boldsymbol{w}$*. Let* $\boldsymbol{r}_n = \nabla f(\boldsymbol{w}_n) = \mathbf{A}\boldsymbol{w}_n - \boldsymbol{b}$*,* $\boldsymbol{z}_n = \boldsymbol{r}_n/\|\boldsymbol{r}_n\|$*. If* m *(with* $m < n$*) is an integer such that* $\boldsymbol{z}_m, \boldsymbol{z}_{m+1}, \ldots, \boldsymbol{z}_n$ *are linearly independent, let* $\mathbf{Z}_n = [\boldsymbol{z}_m, \boldsymbol{z}_{m+1}, \ldots, \boldsymbol{z}_{n-1}]$*. Then we have*

$$\|\boldsymbol{r}_n\| \leq (1 + K_n) \min_{p \in \mathcal{P}_{n-m}, p(0)=1} \|p(\mathbf{AE})\| \, \|\boldsymbol{r}_m\| \tag{8}$$

where $K_n \leq n\alpha(1 + \frac{n-m}{2}\rho)(1 + m\sqrt{\rho})\|\mathbf{A}\|\kappa(\mathbf{Z}_{n+1})$*,* $\rho = \max_{0 \leq j < i \leq n-1} \|\boldsymbol{r}_i\|^2/\|\boldsymbol{r}_j\|^2$*,* $\mathbf{E} = \mathbf{I} + \sqrt{\beta_m}\frac{\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}\boldsymbol{e}_1^\top \mathbf{Z}_n^+$ *with* $\sqrt{\beta_m}\frac{\|\boldsymbol{p}_{m-1}\|}{\|\boldsymbol{r}_{m-1}\|} \leq m\sqrt{\rho}$*,* $\boldsymbol{e}_1 = [1, 0 \ldots, 0]^T$*, and* $\mathcal{P}_{n-m}$ *is the set of polynomials of degree* $n - m$.

The term $\min_{p \in \mathcal{P}_{n-m}, p(0)=1} \|p(\mathbf{AE})\| \, \|\boldsymbol{r}_m\|$, in (8), decays monotonically and is equivalent to the residual of the generalized minimal residual (GMRES) method [45] for the matrix $\mathbf{AE}$, where $\mathbf{E}$ is a rank-one perturbation of the identity matrix $\mathbf{I}$ with the norm of the perturbed matrix bounded by $m\sqrt{\rho}\|\mathbf{Z}_n^+\|$. Then the spectrum of $\mathbf{AE}$ is expected to have a similar distribution as $\mathbf{A}$ and hence $\min_{p \in \mathcal{P}_{n-m}, p(0)=1} \|p(\mathbf{AE})\| \, \|\boldsymbol{r}_m\|$ converges like CG iterates for $\mathbf{A}$. m can be chosen to be any integer less than n such that $\kappa(\mathbf{Z}_{n+1})$, which is a measure of linear independence of $\boldsymbol{z}_m, \boldsymbol{z}_{m+1}, \ldots, \boldsymbol{z}_n$, is bounded. Then K_n is a modest factor and we have a residual reduction $\|\boldsymbol{r}_n\|/\boldsymbol{r}_m\| \leq (1 + K_n) \min_{p \in \mathcal{P}_{n-m}, p(0)=1} \|p(\mathbf{AE})\|$ similar to GMRES over the last $n - m$ iteration. This characterizes the asymptotic convergence behavior of FRGD.

Note that the classical CG method converges at the rate of $(\sqrt{\kappa} - 1)/(\sqrt{\kappa} + 1)$ when $\mathbf{A}$ is positive definite — so does the momentum method (4) with the optimal α^* and β^*. However, they require strong conditions with the former requiring variable α_n (see its definition in the previous section), while the latter requires the optimal α^* and β^*. All these methods significantly accelerate GD with constant step size α, which has a convergence rate of $(\kappa - 1)/(\kappa + 1)$ at best.

We remark that a slightly more general bound $\|\boldsymbol{r}_n\| \leq (1 + K_n) \min_{p \in \mathcal{P}_n, p(0)=1} \|p(\mathbf{A})\boldsymbol{r}_0\|$ holds in place of (7) without the assumption that $\mathbf{A}$ is positive definite; see the proof in Section VI. Thus our method are applicable to more general

[1]Below, we use $\kappa(\mathbf{M}) := \|\mathbf{M}\|\|\mathbf{M}^+\|$ to denote the spectral condition number of a matrix $\mathbf{M}$ where $\mathbf{M}^+$ is the pseudo-inverse of $\mathbf{M}$.

situations of positive semi-definite $\mathbf{A}$ (Example 2.1) or the trust region problem with indefinite $\mathbf{A}$ [5, 55].

We also note that the theorem also *does not require any explicit condition on the learning rate* α. Although α may affect the quality of the basis $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ generated, as long as K_n increases gradually at a rate slower than $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$, we have the convergence of $\boldsymbol{r}_n$. This may explain the success of our method with quite large learning rates (see our numerical results in Section IV). Of course, this is only true to the extent that α is not so large that the condition number of the basis generated grows unbounded.

B. Convergence of FRSGD

This part consists of the convergence analysis for FRSGD. First, we collect several necessary assumptions that are widely used in (non)convex stochastic optimization.

Assumption 1: The stochastic gradient is an unbiased estimate of the gradient, i.e., $\mathbb{E}\boldsymbol{r}_n = \nabla f(\boldsymbol{w}_n)$.

Assumption 2: The gradient of f is L-Lipschitz, i.e., $\|\nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{u})\| \leq L\|\boldsymbol{w} - \boldsymbol{u}\|$ with $L > 0$.

Assumption 3: The stochastic gradient is uniformly bounded, i.e., $\sup_n\{\|\boldsymbol{r}_n\|\} \leq R$ with $R > 0$.

To prove the convergence of SGD with FR momentum, we need to clip the FR momentum such that the upper bound is $1 - \delta$ with $\delta > 0$ being any given small number, resulting in the following adaptive momentum scheme

$$\hat{\beta}_n := \min\left\{\frac{\|\boldsymbol{r}_n\|^2}{\|\boldsymbol{r}_{n-1}\|^2}, 1-\delta\right\}. \tag{9}$$

Note that under the condition of Theorem 1, $\hat{\beta}_n = \|r_n\|^2/\|r_{n-1}\|^2 \leq \sqrt{1-\alpha\lambda_{\min}} \leq 1-\delta$ if $\alpha\lambda_{\min} \geq 2\delta - \delta^2$. Indeed, in our numerical tests, the clipping above does not change the performance of FRSGD when $\delta \leq 10^{-3}$. In all the following deep learning experiments, we set $\delta = 10^{-3}$. In general, we also call SGD with the momentum in (9) FRSGD.

For general nonconvex optimization problems, we have the following convergence guarantee for FRSGD for nonconvex optimization.

Theorem 3 (Convergence of FRSGD for general nonconvex functions)**.** *Let* $\{\boldsymbol{w}_n\}_{n\geq 1}$ *be generated by FRSGD with momentum in* (9) *and suppose Assumptions 1, 2 and 3 hold. We have that*

$$\min_{1\leq n\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\} \leq C_1\alpha + \frac{(f(\boldsymbol{w}_1) - \min f)}{\alpha K},$$

where $C_1 > 0$ *is a constant independent of* α*, d, and* K*.*

According to Theorem 3, the upper bound of the error $\min_{1\leq n\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\}$ is the sum of $C_1\alpha$ and $\frac{(f(\boldsymbol{w}_1)-\min f)}{\alpha K}$. To get ϵ error for $\min_{1\leq n\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\}$ i.e., to ensure $\min_{1\leq n\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\} \leq \epsilon$, we need to set the step size to be $\alpha = \Theta(\epsilon)$ and the number of iteration to be $K = \mathcal{O}(\frac{1}{\epsilon^2})$. In summary, the obtained results above show that the speed of FRSGD can run as fast as SGD in the general nonconvex cases. It demonstrates the stability of the adaptive momentum in the stochastic setting, which also has a stronger convergence guarantee than the stochastic Nesterov's acceleration and its restart variants [7, 52].

IV. Experiments

In this section, we present numerical results to illustrate the advantage of FRSGD over the baseline SGD with constant momentum, Adam [21], AdamW [27], and RMSprop [20] in training DNNs for image classification. We run all experiments with five independent random seeds and report the means and standard deviations of the results.

a) Objective: Our experimental results will demonstrate the following advantages of FRSGD over the baseline methods of SGD with momentum or Nesterov momentum: 1) FRSGD converges significantly faster; 2) FRSGD is significantly more robust to large step sizes; 3) DNN trained by FRSGD is more accurate and more robust against adversarial attacks than that trained by the baseline methods, including SGD with momentum or Nesterov momentum, Adam, AdamW, and RMSprop.

b) Datasets: We consider the benchmark CIFAR10, CIFAR100 [22], and ImageNet datasets [44], and we follow the standard training/test splitting.

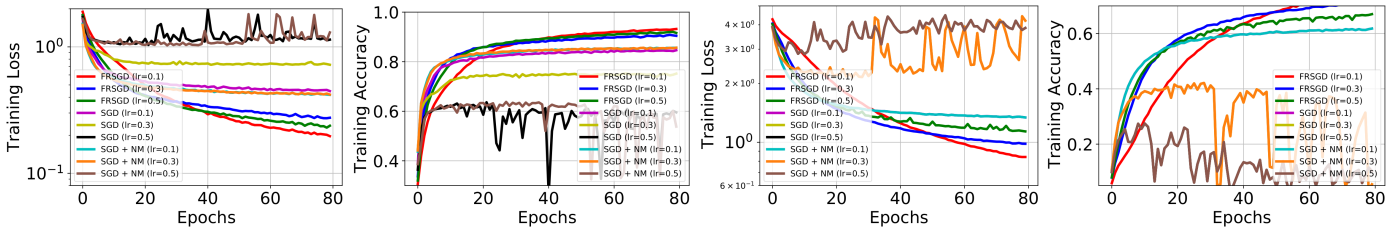
c) Tasks, Experimental Settings, and Baselines: For CIFAR10 and CIFAR100, we consider both natural and adversarial training. We use pre-activated ResNets (PreResNets) models of different depths [18]. As baseline optimization algorithms, we consider SGD with the standard momentum and with the Nesterov momentum scaled by 0.9, and we denote them as SGD and SGD+NM, respectively. We note that SGD is the optimizer used in the original ResNet implementations [17, 18]. For the SGD and SGD+NM baselines, we follow the standard setting of ResNets by running it for 200 epochs with an initial learning rate of 0.1 and decay it by a factor of 10 at the 80th, 120th, and 160th epoch, respectively. We run Adam, AdamW, and RMSprop for 240 epochs with an initial learning rate of 0.003 and reduce it by a factor of 10 at the 100th, 160th, and 200th epoch, respectively. For FRSGD, we run it for 240 epochs with an initial learning rate of 0.5 and reduce it by a factor of 10 at the 180th, 220th, and 230th epoch, respectively[2]. For adversarial training, we use the same SGD and FRSGD solvers described above to solve the outer minimization problem, and we run 10 iterations of the iterative fast gradient sign method (IFGSM^{10}) attack with $\alpha = 2/255$ and $\epsilon = 8/255$ to approximate the solution of the inner maximization problem. We provide the details of IFGSM [14] and a few other attacks in Section VIII.

For ImageNet, we only consider natural training and use the standard setting for SGD with momentum scaled by 0.9, i.e., run SGD with momentum for 90 epochs with an initial learning rate of 0.1 and decay it by a factor of 10 at the 30th and 60th epoch, respectively. As a comparison, we run FRSGD for 100 epochs with an initial learning rate of 0.5 and decay it by a factor of 10 at the 60th and 80th epoch, respectively.

A. FRSGD is Robust Under Large Learning Rates

In this subsection, we compare the performance of SGD, SGD+NM, and FRSGD in training PreResNet56 for CIFAR10

[2]Here, we are able to use a larger learning rate under which FRSGD is still stable, and we decay the learning rate when the decrease of the loss function becomes slower.



CIFAR10 CIFAR10 CIFAR100 CIFAR100

Fig. 2: Plots of epochs vs. training loss and accuracy of PreResNet56 trained by FRSGD, SGD (with momentum), and SGD+NM with different learning rates. FRSGD is the most robust optimizer to large learning rates; where the performance of SGD and SGD+NM deteriorate when a large learning rate is used.

and CIFAR100 classification using different learning rates. We set the learning rate to be 0.1, 0.3, and 0.5, with all the other parameters the same as before. We plot epochs vs. training loss and training accuracy in Fig. 2. These results show that: 1) under the same small learning rate, e.g., 0.1, FRSGD converges remarkably faster than both SGD and SGD+NM; 2) the convergence of SGD and SGD+NM gets deteriorated severely when a larger learning rate is used; in particular, the training loss will not converge when 0.5 is used as the learning rate. However, FRSGD maintains convergence even when a very large learning rate is used; 3) the training loss and accuracy curves of SGD and SGD+NM get plateau very quickly at a large loss, while FRSGD continues to decay.

B. FRSGD Improves Accuracy of DNNs

a) CIFAR10: We consider training PreResNets with different depths using the settings mentioned before for the CIFAR10 classification. We list the test errors of different ResNets trained by different stochastic optimization algorithms in Table I. In general, SGD performs on par with SGD+NM; SGD+NM has small advantages over SGD for training shallow DNNs. SGD is significantly better than Adam, AdamW, and RMSprop. FRSGD outperforms SGD by $0.5 \sim 0.7\%$ for ResNets with the depth ranging from 56 to 470. These improvements over already small error rates are significant in the relative sense, e.g., for PreResNet470, the relative error reduction is $\sim 13\%$ (4.92% vs. 4.27%).

b) CIFAR100: Here, we consider CIFAR100 classification with the same DNNs and the same settings as those used for CIFAR10 classification. We report the test errors in Table II. In this case, FRSGD improves the test accuracy over both SGD and SGD+NM by ~ 1.0 to 1.6%. Again, Adam, AdamW, and RMSprop perform worse than the other optimization algorithms.

c) FRSGD vs. SGD with More Epochs: In the original ResNet experiments, we have run SGD and SGD+NM for 200 epochs, after which no training loss decay is observed. To compare over longer iterations, we train PreResNet110 by running SGD, SGD+NM, and FRSGD for 240 epochs [3]. Moreover, we also compare them with running Adam, AdamW, and RMSprop for 240 epochs using the aforementioned settings. Table III lists the training and testing losses as well as errors of PreResNet110 trained by different optimizers on CIFAR10. The best test error of SGD, SGD+NM, Adam, AdamW, and RMSprop for CIFAR10 classification are $5.23 \pm 0.15\%$, $5.19 \pm 0.16\%$, $6.47 \pm 0.31\%$, $6.54 \pm 0.15\%$, $10.88 \pm 0.35\%$, respectively, compared with 4.73 ± 0.12 for FRSGD. We see that adding 40 more epochs to SGD and SGD+NM does not improve classification accuracy much, and this is because the training loss has reached the plateau at each stage with a budget of 200 epochs. Adam and AdamW converge faster with a smaller final training loss, but the testing loss and accuracy are far behind those of SGD, SGD+NM, and FRSGD.

d) Training FRSGD with a Large Number of Epochs: In the previous experiments, considering the training efficiency, we limited the budget for training epochs of FRSGD by dropping the learning rate when the training loss convergence slows down but before reaching plateaus. This learning rate reduction may be premature and the result may not be the best accuracy our method can achieve. In this experiment, we relax this budget and use a much larger number of epochs for FRSGD and see if we can get more improvement in classification accuracy. In particular, we train PreResNet110/PreResNet290 by running 400 epochs of FRSGD with an initial learning rate of 0.5 and reducing the learning rate by a factor of 10 at the 200th, 300th, and 350th epochs, respectively.

In this setting, we get the best test error rates of $4.64 \pm 0.12/4.26 \pm 0.09\%$ for CIFAR10 and $20.03 \pm 0.29/19.89 \pm 0.19\%$ for CIFAR100, which remarkably improves what we get by using 240 epochs ($4.73 \pm 0.12/4.44 \pm 0.10$ for CIFAR10 and $22.52 \pm 0.35/20.66 \pm 0.31$ for CIFAR100). Furthermore, the training loss of FRSGD at the last epoch in training the PreResNet290 for CIFAR10 classification also becomes significantly smaller than that of SGD or SGD+NM (0.00138 ± 0.00012 (FRSGD), vs. 0.0048 ± 0.0003 (SGD), and 0.0052 ± 0.0003 (SGD+NM)).

e) ImageNet: We train ResNet18, whose implementation is available at [25], using both SGD with momentum and FRSGD with five different random seeds. We set the weight decay to be 0.0001 for both SGD with momentum and FRSGD. Table IV lists top-1 and top-5 accuracies, over five independent runs, of the models trained by two different optimization algorithms; we see that FRSGD can outperform SGD with momentum in classifying images.

[3] Based on trial and error, we found that adding 20 epochs each in the first and second learning rate stages gives the best performance. All the reported results are based on using this setting.

TABLE I: Test error (%) on CIFAR10 using the SGD (with momentum), SGD+NM, Adam, AdamW, RMSprop, and FRSGD. We also include the reported results from [18] (in parentheses) in addition to our reproduced results. ResNets trained by FRSGD are consistently more accurate than those trained by SGD, SGD+NM, Adam, AdamW, and RMSprop.

Network	SGD (baseline)	SGD+NM	Adam	AdamW	RMSprop	FRSGD
PreResNet56	6.12 ± 0.24	5.90 ± 0.17	7.54 ± 0.09	7.36 ± 0.13	11.31 ± 0.42	$\mathbf{5.39 \pm 0.13}$
PreResNet110	5.25 ± 0.14 (6.37)	5.24 ± 0.16	6.83 ± 0.10	6.54 ± 0.15	10.88 ± 0.35	$\mathbf{4.73 \pm 0.12}$
PreResNet164	5.10 ± 0.19 (5.46)	5.08 ± 0.21	6.65 ± 0.13	6.37 ± 0.19	10.60 ± 0.37	$\mathbf{4.50 \pm 0.16}$
PreResNet290	5.05 ± 0.23	5.04 ± 0.12	6.61 ± 0.11	6.31 ± 0.18	10.55 ± 0.29	$\mathbf{4.44 \pm 0.10}$
PreResNet470	4.92 ± 0.10	4.97 ± 0.15	6.50 ± 0.16	6.22 ± 0.21	10.52 ± 0.33	$\mathbf{4.27 \pm 0.09}$

TABLE II: Test error (%) on CIFAR100 using the SGD (with momentum), SGD+NM, Adam, AdamW, RMSprop, and FRSGD. We also include the reported results from [18] (in parentheses) in addition to our reproduced results. ResNets trained by FRSGD are uniformly more accurate than those trained by SGD, SGD+NM, Adam, AdamW, and RMSprop.

Network	SGD	SGD+NM	Adam	AdamW	RMSprop	FRSGD
PreResNet56	26.60 ± 0.33	26.14 ± 0.38	33.88 ± 0.57	32.95 ± 0.63	36.07 ± 0.98	$\mathbf{25.00 \pm 0.32}$
PreResNet110	23.75 ± 0.20	23.65 ± 0.36	30.05 ± 0.55	29.81 ± 0.48	33.05 ± 0.92	$\mathbf{22.52 \pm 0.35}$
PreResNet164	22.76 ± 0.37 (24.33)	22.79 ± 0.29	28.99 ± 0.50	28.93 ± 0.51	31.86 ± 0.75	$\mathbf{21.38 \pm 0.34}$
PreResNet290	21.78 ± 0.21	21.68 ± 0.21	28.07 ± 0.44	27.95 ± 0.40	30.05 ± 0.87	$\mathbf{20.66 \pm 0.31}$
PreResNet470	21.43 ± 0.30	21.21 ± 0.30	27.83 ± 0.48	27.70 ± 0.47	29.99 ± 0.86	$\mathbf{19.92 \pm 0.29}$

TABLE III: Lists of the optimal training/test loss and accuracy of PreResNet110 trained by FRSGD, SGD (with momentum), SGD+NM,Adam, AdamW, and RMSprop with 240 epochs. Adam has a smaller training loss than the others, but the PreResNet110 trained by FRSGD has the smallest test loss/error.

Optimizer	Training Loss	Training Error Rate (%)	Test Loss	Test Error rate (%)
SGD	0.00529 ± 0.00043	0.042 ± 0.006	0.1950 ± 0.00091	5.23 ± 0.15
SGD+NM	0.00462 ± 0.00047	0.032 ± 0.005	0.1846 ± 0.00101	5.19 ± 0.16
Adam	$\mathbf{0.00033 \pm 0.00003}$	0.002 ± 0.002	0.3237 ± 0.00125	6.47 ± 0.31
AdamW	0.00038 ± 0.00004	$\mathbf{0.001 \pm 0.002}$	0.3309 ± 0.00118	6.54 ± 0.15
RMSprop	0.19059 ± 0.00178	6.180 ± 0.162	0.4263 ± 0.0107	10.88 ± 0.35
FRSGD	0.00680 ± 0.00021	0.090 ± 0.002	$\mathbf{0.1611 \pm 0.00086}$	$\mathbf{4.73 \pm 0.12}$

TABLE IV: Test accuracy (%) of ResNet18 for ImageNet classification, where the models are trained by SGD with momentum (step size 0.1) and FRSGD.

Model	ResNet18
SGD (top-1)	69.86 ± 0.048 (69.86, [26])
SGD (top-5)	89.31 ± 0.090
FRSGD (top-1)	$\mathbf{69.99 \pm 0.057}$
FRSGD (top-5)	$\mathbf{89.62 \pm 0.088}$

C. FRSGD Improves Adversarial Training

Finally, we numerically demonstrate that FRSGD can also improve the adversarial robustness of the trained DNNs using adversarial training. We train the PreResNet110 by applying the adversarial training using the settings listed before. Then we apply the well-trained PreResNet110 to classify the test set under three kinds of benchmark adversarial attacks: fast gradient sign method (FGSM), m steps IFGSM (IFGSM^m with $m = 10, 20, 40, 100$) [14], and C&W attacks [4]. We apply the same set of hyperparameters for these attacks as that used in [51, 29] in the following experiments. A brief introduction of these attacks and the used hyperparameters are available in Section VIII.

Tables V and VI list the accuracy of the adversarially trained PreResNet110 for classifying CIFAR10 and CIFAR100 images with or without adversarial attacks [4]. First, we see that the robust PreResNet110 trained by FRSGD is slightly more accurate than that trained by SGD+NM for classifying the clean CIFAR10 and CIFAR100 images without any attack, e.g., the accuracy of FRSGD is $82.36 \pm 0.27\%$ and $54.95 \pm 0.49\%$ for CIFAR10 and CIFAR100 classification, while the corresponding accuracy of the model trained by SGD+NM is $82.19 \pm 0.29\%$ and $54.75 \pm 0.52\%$, respectively. Second, the model trained by FRSGD is more robust than that trained by SGD+NM under all three adversarial attacks mentioned before, e.g., under the IFGSM^{100} attack, the robust accuracy of these two models is $52.15 \pm 0.19\%$ vs. $51.08 \pm 0.35\%$ for CIFAR10 classification, and are $27.40 \pm 0.48\%$ vs. $29.01 \pm 0.39\%$ for CIFAR100 classification. Although the improvements are not very significant, the good performance of FRSGD in this difficult setting illustrates its robustness in different problem types.

[4]We only compare FRSGD with SGD+NM since SGD performance is weaker than SGD+NM in this case.

V. Concluding Remarks

In this paper, we leverage adaptive momentum from the NCG to improve SGD, and the resulting algorithm performs surprisingly well in the following sense: 1) It can accelerate GD significantly; in particular, we observed that it achieves exponential convergence for optimizing a specific convex function; 2) It allows us to use much larger step sizes and converges faster than SGD with (Nesterov) momentum in training DNNs; 3) DNNs trained by FRSGD have remarkably higher classification accuracy and are more robust to adversarial attacks for image classification. The method is as simple as SGD and is easy to implement. It is well suited for DNN training.

There are several interesting open questions that are worth further investigation. First, can we integrate the adaptive momentum with adaptive step size to further improve stochas-

TABLE V: Test accuracy (%) of PreResNet110 on CIFAR10 using PGD adversarial training with SGD+NM and FRSGD as the outer solver. FRSGD improves accuracies for classifying both clean and adversarial images.

Optimizer	Natural	FGSM	IFGSM^{10}	IFGSM^{20}	IFGSM^{40}	IFGSM^{100}	C&W
SGD+NM	82.19 ± 0.29	57.61 ± 0.33	55.35 ± 0.42	52.02 ± 0.34	51.45 ± 0.33	51.08 ± 0.35	62.92 ± 0.50
FRSGD	$\mathbf{82.36 \pm 0.27}$	$\mathbf{58.27 \pm 0.29}$	$\mathbf{55.83 \pm 0.31}$	$\mathbf{53.07 \pm 0.28}$	$\mathbf{52.39 \pm 0.25}$	$\mathbf{52.15 \pm 0.19}$	$\mathbf{63.05 \pm 0.33}$

TABLE VI: Test accuracy (%) of PreResNet110 on CIFAR100 using PGD adversarial training with SGD+NM and FRSGD as the outer solver. FRSGD improves accuracies for classifying both clean and adversarial images.

Optimizer	Clean	FGSM	IFGSM^{10}	IFGSM^{20}	IFGSM^{40}	IFGSM^{100}	C&W
SGD	54.75 ± 0.52	30.75 ± 0.41	29.61 ± 0.45	27.87 ± 0.44	27.51 ± 0.42	27.40 ± 0.48	38.97 ± 0.66
FRSGD	$\mathbf{54.95 \pm 0.49}$	$\mathbf{31.77 \pm 0.43}$	$\mathbf{30.79 \pm 0.33}$	$\mathbf{29.32 \pm 0.37}$	$\mathbf{29.09 \pm 0.40}$	$\mathbf{29.01 \pm 0.39}$	$\mathbf{39.01 \pm 0.50}$

tic optimization algorithms? Second, can we prove stronger convergence results for FRGD/FRSGD under more general conditions? Third, can we leverage adaptive momentum to improve training DNNs for other deep learning tasks beyond image classification? Application of the FRSGD to other computer vision tasks, e.g., objection detection, is another interesting future work. One particular question is whether FRSGD can improve the performance of existing object detection frameworks [56, 36].

VI. Proof of the convergence of FRGD

First, using the integral form of the mean value theorem, we write

$$\begin{aligned}&\nabla f(\boldsymbol{w}_{n+1}) - \nabla f(\boldsymbol{w}_n)\\ =&\int_0^1 \mathbf{H}(\boldsymbol{w}_n + t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))dt(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n) = \mathbf{H}_n(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n)\end{aligned}$$

where $\mathbf{H}(\boldsymbol{w}) = [\frac{\partial^2 f}{\partial w_i \partial w_j}(\boldsymbol{w})]$ is the Hessian matrix and

$$\mathbf{H}_n := \int_0^1 \mathbf{H}(\boldsymbol{w}_n + t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))dt.$$

We now consider the iterates of FRGD and give some lemmas and then prove Theorem 1.

Lemma 1. *Assume that the Hessian $\mathbf{H}(\boldsymbol{w})$ is Lipschitz continuous with the Lipschiz constant C, i.e., $\|\mathbf{H}(\boldsymbol{w}) - \mathbf{H}(\tilde{\boldsymbol{w}})\| \leq C\|\boldsymbol{w} - \tilde{\boldsymbol{w}}\|$. Then,*

$$\|\mathbf{H}_{n+1} - \mathbf{H}_n\| \leq \frac{1}{2}C(\|\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1}\| + \|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\|).$$

Proof. Using the definition of $\mathbf{H}_n$, we have

$$\begin{aligned}&\|\mathbf{H}_{n+1} - \mathbf{H}_n\|\\ =&\|\int_0^1 \mathbf{H}(\boldsymbol{w}_{n+1} + t(\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1}))dt\\ &- \int_0^1 \mathbf{H}(\boldsymbol{w}_n + t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))dt\|\\ \leq&\int_0^1 \|\mathbf{H}(\boldsymbol{w}_{n+1} + t(\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1}))dt - \mathbf{H}(\boldsymbol{w}_n + t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))\|dt\\ \leq&\int_0^1 C\|\boldsymbol{w}_{n+1} + t(\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1})) - \boldsymbol{w}_n - t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))\|dt\\ \leq&C\int_0^1 t\|\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1})\| + (1-t)\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\|dt\\ =&\frac{1}{2}C\|\boldsymbol{w}_{n+2} - \boldsymbol{w}_{n+1})\| + \frac{1}{2}C\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\|.\end{aligned}$$

□

Lemma 2. *If the eigenvalues of the Hessian $\mathbf{H}(\boldsymbol{w})$ are in $[\lambda_{\min}, \lambda_{\max}]$ for all $\boldsymbol{w}$, then the eigenvalues of $\mathbf{H}_n$ are in $[\lambda_{\min}, \lambda_{\max}]$ for all n.*

Proof. For any nonzero vector $\boldsymbol{x} \in \mathbb{R}^d$, using the minimax theorem, we have

$$\lambda_{\min} \leq \frac{\boldsymbol{x}^T\mathbf{H}(\boldsymbol{w})\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}} \leq \lambda_{\max}.$$

Then it follows from

$$\boldsymbol{x}^T\mathbf{H}_n\boldsymbol{x} = \int_0^1 \boldsymbol{x}^T\mathbf{H}(\boldsymbol{w}_n + t(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n))\boldsymbol{x}dt$$

i.e.,

$$\lambda_{\min} \leq \frac{\boldsymbol{x}^T\mathbf{H}_n\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}} \leq \lambda_{\max}.$$

Using the minimax theorem again, we have that the eigenvalues of $\mathbf{H}_n$ are in $[\lambda_{\min}, \lambda_{\max}]$. □

Proof of Theorem 1. Consider $\boldsymbol{w}_{n-1}, \boldsymbol{w}_n, \boldsymbol{w}_{n+1}$. By Lemma 1, we have

$$\begin{aligned}\boldsymbol{r}_{n+1} - \boldsymbol{r}_n &= \nabla f(\boldsymbol{w}_{n+1}) - \nabla f(\boldsymbol{w}_n)\\ &= \mathbf{H}_n(\boldsymbol{w}_{n+1} - \boldsymbol{w}_n) = -\alpha\mathbf{H}_n\boldsymbol{p}_n.\end{aligned} \tag{10}$$

Furthermore,

$$\begin{aligned}\|\mathbf{H}_n - \mathbf{H}_{n-1}\| &\leq \frac{1}{2}C(\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\| + \|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|)\\ &= \frac{1}{2}C\alpha(\|\boldsymbol{p}_n\| + \|\boldsymbol{p}_{n-1}\|).\end{aligned} \tag{11}$$

On the other hand, by Lemma 2, the eigenvalues of $\mathbf{H}_n$ are on the interval $[\lambda_{\min}, \lambda_{\max}]$.

Now we prove by induction in n that, for $1 \leq n \leq K$,

$$\frac{\boldsymbol{r}_{n-1}^\top\mathbf{H}_{n-1}\boldsymbol{p}_{n-1}}{\boldsymbol{r}_{n-1}^\top\boldsymbol{r}_{n-1}} \geq \lambda_{\min},\ \|\boldsymbol{p}_{n-1}\| \leq n\|\boldsymbol{r}_{n-1}\|, \tag{12}$$

and $\|\boldsymbol{r}_n\| \leq \sqrt{1 - \alpha\lambda_{\min}}\|\boldsymbol{r}_{n-1}\|$.

First, consider the case $n = 1$. Since $\boldsymbol{p}_0 = \boldsymbol{r}_0$ and $\boldsymbol{r}_1 = \boldsymbol{r}_0 - \alpha\mathbf{H}_0\boldsymbol{p}_0 = \boldsymbol{r}_0 - \alpha\mathbf{H}_0\boldsymbol{r}_0$, we have $\boldsymbol{r}_0^\top\mathbf{H}_0\boldsymbol{p}_0 = \boldsymbol{r}_0^\top\mathbf{H}_0\boldsymbol{r}_0 \geq \lambda_{\min}\boldsymbol{r}_0^\top\boldsymbol{r}_0$, $\|\boldsymbol{p}_0\| = \|\boldsymbol{r}_0\|$, and

$$\begin{aligned}\boldsymbol{r}_1^\top\boldsymbol{r}_1 =&\boldsymbol{r}_0^\top\boldsymbol{r}_0 - 2\alpha\boldsymbol{r}_0^\top\mathbf{H}_0\boldsymbol{r}_0 + \alpha^2\boldsymbol{r}_0^\top\mathbf{H}_0^2\boldsymbol{r}_0\\ \leq&\boldsymbol{r}_0^\top\boldsymbol{r}_0 - 2\alpha\lambda_{\min}\boldsymbol{r}_0^\top\boldsymbol{r}_0 + \alpha^2\lambda_{\max}^2\boldsymbol{r}_0^\top\boldsymbol{r}_0\\ \leq&(1 - \alpha\lambda_{\min})\boldsymbol{r}_0^\top\boldsymbol{r}_0,\end{aligned}$$

where we have used $-\lambda_{\min} + \alpha\lambda_{\max}^2 \leq 0$ as implied from the condition on α. So, (12) holds for $n = 1$.

Assume that (12) holds for some $n \le K-1$. We prove it for $n+1$. Using (5) and the induction assumption, we have $\beta_n = \frac{\|\boldsymbol{r}_n\|^2}{\|\boldsymbol{r}_{n-1}\|^2} \le 1$ and then

$$\begin{aligned}\boldsymbol{p}_n^\top \boldsymbol{p}_n =& \boldsymbol{r}_n^\top \boldsymbol{r}_n + 2\beta_n \boldsymbol{r}_n^\top \boldsymbol{p}_{n-1} + \beta_n^2 \boldsymbol{p}_{n-1}^\top \boldsymbol{p}_{n-1}\\ \le& \|\boldsymbol{r}_n\|^2 + 2\frac{\|\boldsymbol{r}_n\|^2}{\|\boldsymbol{r}_{n-1}\|^2}\|\boldsymbol{r}_n\|\|\boldsymbol{p}_{n-1}\| + \frac{\|\boldsymbol{r}_n\|^2}{\|\boldsymbol{r}_{n-1}\|^2}\|\boldsymbol{p}_{n-1}\|^2\\ \le& \|\boldsymbol{r}_n\|^2 + 2\|\boldsymbol{r}_n\|^2 n + \|\boldsymbol{r}_n\|^2 n^2 = (1+n)^2\|\boldsymbol{r}_n\|^2.\end{aligned}$$

where we have used $\|\boldsymbol{p}_{n-1}\| \le n\|\boldsymbol{r}_{n-1}\|$.

Next, using (5) and (10), we have

$$\begin{aligned}&\boldsymbol{r}_n^\top \mathbf{H}_n \boldsymbol{p}_n\\ =&\boldsymbol{r}_n^\top \mathbf{H}_n \boldsymbol{r}_n + \beta_n \boldsymbol{r}_n^\top \mathbf{H}_n \boldsymbol{p}_{n-1}\\ =&\boldsymbol{r}_n^\top \mathbf{H}_n \boldsymbol{r}_n + \beta_n \boldsymbol{r}_{n-1}^\top \mathbf{H}_n \boldsymbol{p}_{n-1} - \alpha\beta_n \boldsymbol{p}_{n-1}^\top \mathbf{H}_{n-1}\mathbf{H}_n \boldsymbol{p}_{n-1}\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\boldsymbol{r}_{n-1}^\top \mathbf{H}_{n-1}\boldsymbol{p}_{n-1}}{\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}}\\ &+\boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\boldsymbol{r}_{n-1}^\top (\mathbf{H}_n - \mathbf{H}_{n-1})\boldsymbol{p}_{n-1}}{\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}} - \alpha\frac{\boldsymbol{r}_n^\top \boldsymbol{r}_n}{\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}}\lambda_{\max}^2 \boldsymbol{p}_{n-1}^\top \boldsymbol{p}_{n-1}\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \lambda_{\min} - \boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\|\boldsymbol{p}_{n-1}\|}{\|\boldsymbol{r}_{n-1}\|}\|\mathbf{H}_n - \mathbf{H}_{n-1}\|\\ &- \alpha\lambda_{\max}^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\|\boldsymbol{p}_{n-1}\|^2}{\|\boldsymbol{r}_{n-1}\|^2}\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \lambda_{\min} - \boldsymbol{r}_n^\top \boldsymbol{r}_n n\frac{C}{2}\alpha(\|\boldsymbol{p}_n\| + \|\boldsymbol{p}_{n-1}\|)\\ &- \alpha\lambda_{\max}^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n n^2\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \lambda_{\min} - \alpha\boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{C}{2} n((n+1)\|\boldsymbol{r}_n\|\\ &+ n\|\boldsymbol{r}_{n-1}\|) - \alpha\lambda_{\max}^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n n^2\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \lambda_{\min} - \alpha\boldsymbol{r}_n^\top \boldsymbol{r}_n CK^2\|\boldsymbol{r}_0\| - \alpha\lambda_{\max}^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n K^2\\ \ge&\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n,\end{aligned}$$

where the last inequality follows from the condition on α.

Finally, using the two inequalities above, we have

$$\begin{aligned}\boldsymbol{r}_{n+1}^\top \boldsymbol{r}_{n+1} =& \boldsymbol{r}_n^\top \boldsymbol{r}_n - 2\alpha \boldsymbol{r}_n^\top \mathbf{H}_n \boldsymbol{p}_n + \alpha^2 \boldsymbol{p}_n^\top \mathbf{H}_n^2 \boldsymbol{p}_n\\ \le& \boldsymbol{r}_n^\top \boldsymbol{r}_n - 2\alpha\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \alpha^2\lambda_{\max}^2 \boldsymbol{p}_n^\top \boldsymbol{p}_n\\ \le& \boldsymbol{r}_n^\top \boldsymbol{r}_n - 2\alpha\lambda_{\min}\boldsymbol{r}_n^\top \boldsymbol{r}_n + \alpha^2\lambda_{\max}^2 (n+1)^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n\\ \le& (1-\alpha\lambda_{\min})\boldsymbol{r}_n^\top \boldsymbol{r}_n - \alpha(\lambda_{\min} - \alpha\lambda_{\max}^2(n+1)^2)\boldsymbol{r}_n^\top \boldsymbol{r}_n\\ \le& (1-\alpha\lambda_{\min})\boldsymbol{r}_n^\top \boldsymbol{r}_n,\end{aligned}$$

where we note that $n+1 \le K$ and hence $\lambda_{\min} - \alpha\lambda_{\max}^2(n+1)^2 \ge 0$. This completes the proof of (12).

We now prove $\boldsymbol{r}_n^\top \boldsymbol{p}_n \ge \boldsymbol{r}_n^\top \boldsymbol{r}_n > 0$ by induction. The case $n=0$ is trivial and assume it holds for $n-1$. Then

$$\begin{aligned}\boldsymbol{r}_n^\top \boldsymbol{p}_n =& \boldsymbol{r}_n^\top \boldsymbol{r}_n + \beta_n \boldsymbol{r}_n^\top \boldsymbol{p}_{n-1}\\ =& \boldsymbol{r}_n^\top \boldsymbol{r}_n + \beta_n \boldsymbol{r}_{n-1}^\top \boldsymbol{p}_{n-1} - \alpha\beta_n \boldsymbol{p}_{n-1}^\top H_{n-1}\boldsymbol{p}_{n-1}\\ =& \boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\boldsymbol{r}_{n-1}^\top \boldsymbol{p}_{n-1}}{\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}} - \alpha\boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\boldsymbol{p}_{n-1}^\top H_{n-1}\boldsymbol{p}_{n-1}}{\boldsymbol{r}_{n-1}^\top \boldsymbol{r}_{n-1}}\\ \ge& \boldsymbol{r}_n^\top \boldsymbol{r}_n + \boldsymbol{r}_n^\top \boldsymbol{r}_n - \alpha\lambda_{\max}\boldsymbol{r}_n^\top \boldsymbol{r}_n \frac{\|\boldsymbol{p}_{n-1}\|^2}{\|\boldsymbol{r}_{n-1}\|^2}\\ \ge& 2\boldsymbol{r}_n^\top \boldsymbol{r}_n - \alpha\lambda_{\max} n^2 \boldsymbol{r}_n^\top \boldsymbol{r}_n\\ \ge& \boldsymbol{r}_n^\top \boldsymbol{r}_n\end{aligned}$$

where the last inequality follows from $\alpha \le \frac{\lambda_{\min}}{(\lambda_{\max}^2 + C\|\boldsymbol{r}_0\|)K^2} < \frac{\lambda_{\min}}{\lambda_{\max}^2 K^2} \le \frac{1}{\lambda_{\max}K^2}$. This completes the proof of the theorem. □

Proof of Theorem 2. Let $\mathbf{R}_n = [\boldsymbol{r}_0, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n-1}]$, $\mathbf{D}_n = \operatorname{diag}\{\|\boldsymbol{r}_0\|, \|\boldsymbol{r}_1\|, \ldots, \|\boldsymbol{r}_{n-1}\|\}$ and $\mathbf{P}_n = [\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{n-1}]$. Then $\mathbf{Z}_n = \mathbf{R}_n\mathbf{D}_n^{-1}$. Using $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha\mathbf{A}\boldsymbol{p}_k$, we have

$$\begin{aligned}\alpha\mathbf{A}\mathbf{P}_n &= [\boldsymbol{r}_0 - \boldsymbol{r}_1, \boldsymbol{r}_1 - \boldsymbol{r}_2, \ldots, \boldsymbol{r}_{n-1} - \boldsymbol{r}_n]\\ &= \mathbf{R}_n\mathbf{L}_n - \boldsymbol{r}_n e_n^\top = \mathbf{Z}_n\mathbf{D}_n\mathbf{L}_n - \boldsymbol{r}_n e_n^\top,\end{aligned}$$

and using $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k \boldsymbol{p}_{k-1}$

$$\mathbf{Z}_n = \mathbf{R}_n\mathbf{D}_n^{-1} = \mathbf{P}_n\mathbf{U}_n\mathbf{D}_n^{-1},$$

where $\boldsymbol{e}_n = [0, \ldots, 0, 1]^\top$ and $\mathbf{L}_n$ is the $n\times n$ lower bidiagonal matrix with 1 on the diagonal and -1 on the subdiagonal, and $\mathbf{U}_n$ is the upper bidiagonal matrix with 1 on the diagonal and $-\beta_1, \ldots, -\beta_{n-1}$ on the superdiagonal. Combining the two equations, we obtain

$$\mathbf{A}\mathbf{Z}_n = \mathbf{Z}_n\mathbf{T}_n - \frac{1}{\alpha'}\frac{\boldsymbol{r}_n}{\|\boldsymbol{r}_0\|}\boldsymbol{e}_n^\top,$$

where $\mathbf{T}_n = \frac{1}{\alpha}\mathbf{D}_n\mathbf{L}_n\mathbf{U}_n\mathbf{D}_n^{-1}$ and $\alpha' = \alpha\|\boldsymbol{r}_{n-1}\|/\|\boldsymbol{r}_0\|$. Note that $\alpha' = \boldsymbol{e}_n^\top \mathbf{T}_n^{-1}\boldsymbol{e}_1$. Apply Theorem 3.5 of [49] (with $\hat{\Delta}_n$ there equal to 0 and the indexes shifted by 1) to the above equation, we have

$$\|\boldsymbol{r}_n\| \le (1+K_n)\min_{p\in\mathcal{P}_n, p(0)=1}\|p(\mathbf{A})\boldsymbol{r}_0\|, \tag{13}$$

where $K_n = \|\mathbf{A}\mathbf{Z}_n\mathbf{T}_n^{-1}[\mathbf{I}_n \quad 0]\mathbf{Z}_{n+1}^+\| \le \|\mathbf{A}\|\|\mathbf{T}_n^{-1}\|\|\mathbf{Z}_n\|\|\mathbf{Z}_{n+1}^+\|$ and $\mathcal{P}_n$ is the set of polynomials of degree n.

Note that $\beta_k = \|\boldsymbol{r}_k\|^2/\|\boldsymbol{r}_{k-1}\|^2$. Write $\mathbf{T}_n = \frac{1}{\alpha}\mathbf{D}_n\mathbf{L}_n\mathbf{D}_n^{-1}\mathbf{D}_n\mathbf{U}_n\mathbf{D}_n^{-1} = \frac{1}{\alpha}\hat{\mathbf{L}}_n\hat{\mathbf{U}}_n$, where

$$\begin{aligned}\hat{\mathbf{L}}_n :=& \mathbf{D}_n\mathbf{L}_n\mathbf{D}_n^{-1}\\ =& \begin{pmatrix} 1 & & & & \\ -\frac{\|\boldsymbol{r}_1\|}{\|\boldsymbol{r}_0\|} & 1 & & & \\ & \ddots & \ddots & & \\ & & & 1 & \\ & & & -\frac{\|\boldsymbol{r}_{n-1}\|}{\|\boldsymbol{r}_{n-2}\|} & 1 \end{pmatrix}\\ =& \begin{pmatrix} 1 & & & & \\ -\sqrt{\beta_1} & 1 & & & \\ & \ddots & \ddots & & \\ & & & 1 & \\ & & & -\sqrt{\beta_{n-1}} & 1 \end{pmatrix},\end{aligned}$$

and

$$\begin{aligned}\hat{\mathbf{U}}_n :=& \mathbf{D}_n\mathbf{U}_n\mathbf{D}_n^{-1}\\ =& \begin{pmatrix} 1 & -\beta_1\frac{\|\boldsymbol{r}_0\|}{\|\boldsymbol{r}_1\|} & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & 1 & -\beta_{n-1}\frac{\|\boldsymbol{r}_{n-2}\|}{\|\boldsymbol{r}_{n-1}\|} \\ & & & & 1 \end{pmatrix}\\ =& \begin{pmatrix} 1 & -\sqrt{\beta_1} & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & 1 & -\sqrt{\beta_{n-1}} \\ & & & & 1 \end{pmatrix} = \hat{\mathbf{L}}_n^\top.\end{aligned}$$

Then $\hat{\mathbf{L}}_n^{-1}$ is a lower triangular matrix with the diagonals being 1 and with the (i,j) entry being $\sqrt{\beta_j\beta_{j+1}\ldots\beta_{i-1}} = \|\boldsymbol{r}_{i-1}\|/\|\boldsymbol{r}_{j-1}\|$ for $i > j$. Then bounding $\|\boldsymbol{r}_{i-1}\|^2/\|\boldsymbol{r}_{j-1}\|^2$ by ρ, we have $\|\hat{\mathbf{L}}_n^{-1}\|_F^2 \le n + n(n-1)\rho/2$. So, $\|\mathbf{T}_n^{-1}\| =$

$\alpha\|\hat{\mathbf{L}}_n^{-1}\|^2 \leq \alpha\|\hat{\mathbf{L}}_n^{-1}\|_F^2 \leq \alpha n(1+n\rho/2)$. Combining this with $\|\mathbf{Z}_n\|\|\mathbf{Z}_{n+1}^+\| \leq \|\mathbf{Z}_{n+1}\|\|\mathbf{Z}_{n+1}\| = \kappa(\mathbf{Z}_{n+1})$ results in $K_n \leq n\alpha(1+\frac{n}{2}\rho)\|\mathbf{A}\|\kappa(\mathbf{Z}_{n+1})$. Finally, the bound follows from the standard CG convergence bound [45, p.215] that shows

$$\begin{aligned}\min_{p\in\mathcal{P}_n, p(0)=1}\|p(\mathbf{A})\boldsymbol{r}_0\| &\leq \min_{p\in\mathcal{P}_n, p(0)=1}\max_i |p(\lambda_i)|\,\|\boldsymbol{r}_0\| \\ &\leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^n\|\boldsymbol{r}_0\|,\end{aligned}$$

where λ_i (for $1\leq i\leq d$) are eigenvalues of $\mathbf{A}$. □

We give a lemma for the proof of Corollary 1.

Lemma 3. *We have*

$$\frac{\|\boldsymbol{p}_k\|}{\|\boldsymbol{r}_k\|} \leq 1 + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|} + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-2}\|} + \ldots + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_0\|}.$$

Proof. It follows from $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{p}_{k-1}$ and $\beta_k = \frac{\|\boldsymbol{r}_k\|^2}{\|\boldsymbol{r}_{k-1}\|^2}$ that $\|\boldsymbol{p}_k\| \leq \|\boldsymbol{r}_k\| + \beta_k\|\boldsymbol{p}_{k-1}\| \leq \|\boldsymbol{r}_k\|(1+\frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|}\frac{\|\boldsymbol{p}_{k-1}\|}{\|\boldsymbol{r}_{k-1}\|})$. Then

$$\frac{\|\boldsymbol{p}_k\|}{\|\boldsymbol{r}_k\|} \leq 1 + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|}\frac{\|\boldsymbol{p}_{k-1}\|}{\|\boldsymbol{r}_{k-1}\|}.$$

Applying this repeatedly leads to

$$\begin{aligned}\frac{\|\boldsymbol{p}_k\|}{\|\boldsymbol{r}_k\|} &\leq 1 + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|} + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|}\frac{\|\boldsymbol{r}_{k-1}\|}{\|\boldsymbol{r}_{k-2}\|}\frac{\|\boldsymbol{p}_{k-2}\|}{\|\boldsymbol{r}_{k-2}\|} \\ &\leq 1 + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|} + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|}\frac{\|\boldsymbol{r}_{k-1}\|}{\|\boldsymbol{r}_{k-2}\|} + \ldots \\ &\quad + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|}\frac{\|\boldsymbol{r}_{k-1}\|}{\|\boldsymbol{r}_{k-2}\|}\cdots\frac{\|\boldsymbol{r}_1\|}{\|\boldsymbol{r}_0\|}\frac{\|\boldsymbol{p}_0\|}{\|\boldsymbol{r}_0\|} \\ &= 1 + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-1}\|} + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_{k-2}\|} + \ldots + \frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_0\|}\end{aligned}$$

where we note that $\boldsymbol{p}_0 = \boldsymbol{r}_0$. □

Proof of Corollary 1. The proof is similar to the one for Theorem 2 but uses the iterates from m to n. For ease of notation, we use the same $\mathbf{R}_n, \mathbf{Z}_n, \mathbf{D}_n$ to denote similar matrices defined from the iterates from m to n. Namely, let $\mathbf{R}_n = [\boldsymbol{r}_m, \boldsymbol{r}_{m+1}, \ldots, \boldsymbol{r}_{n-1}]$, $\mathbf{Z}_n = [\boldsymbol{z}_m, \boldsymbol{z}_{m+1}, \ldots, \boldsymbol{z}_{n-1}]$, $\mathbf{D}_n = \mathrm{diag}\{\|\boldsymbol{r}_m\|, \|\boldsymbol{r}_{m+1}\|, \ldots, \|\boldsymbol{r}_{n-1}\|\}$ and $\mathbf{P}_n = [\boldsymbol{p}_m, \ldots, \boldsymbol{p}_{n-1}]$. Then $\mathbf{Z}_n = \mathbf{R}_n\mathbf{D}_n^{-1}$. Using $\boldsymbol{r}_{m+1} = \boldsymbol{r}_m - \alpha\mathbf{A}\boldsymbol{p}_m$, we have

$$\begin{aligned}\alpha\mathbf{A}\mathbf{P}_n &= [\boldsymbol{r}_m - \boldsymbol{r}_{m+1}, \boldsymbol{r}_{m+1} - \boldsymbol{r}_{m+2}, \ldots, \boldsymbol{r}_{n-1} - \boldsymbol{r}_n] \\ &= \mathbf{Z}_n\mathbf{D}_n\mathbf{L}_n - \boldsymbol{r}_n\boldsymbol{e}_{n-m}^\top,\end{aligned}$$

and using $\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_k\boldsymbol{p}_{k-1}$

$$\mathbf{R}_n = \mathbf{P}_n\mathbf{U}_n - \beta_m\boldsymbol{p}_{m-1}\boldsymbol{e}_1^T,$$

where $\boldsymbol{e}_n = [0,\ldots,0,1]^\top$, $\boldsymbol{e}_1 = [1,0,\ldots,0]^\top$, and $\mathbf{L}_n$ is the $(n-m)\times(n-m)$ lower bidiagonal matrix with 1 on the diagonal and -1 on the subdiagonal, and $\mathbf{U}_n$ is the upper bidiagonal matrix with 1 on the diagonal and $-\beta_{m+1}, \ldots, -\beta_{n-1}$ on the superdiagonal. Then, noting that $\frac{\beta_m\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_m\|} = \sqrt{\beta_m}\frac{\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}$, we have

$$\mathbf{Z}_n = \mathbf{R}_n\mathbf{D}_n^{-1} = \mathbf{P}_n\mathbf{U}_n\mathbf{D}_n^{-1} - \sqrt{\beta_m}\frac{\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}\boldsymbol{e}_1^T.$$

Combining this with the equation on $\alpha\mathbf{A}\mathbf{P}_n$, we obtain

$$\mathbf{A}\mathbf{Z}_n + \sqrt{\beta_m}\frac{\mathbf{A}\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}\boldsymbol{e}_1^T = \mathbf{Z}_n\mathbf{T}_n - \frac{1}{\alpha'}\frac{\boldsymbol{r}_n}{\|\boldsymbol{r}_0\|}\boldsymbol{e}_{n-m}^\top,$$

where $\mathbf{T}_n = \frac{1}{\alpha}\mathbf{D}_n\mathbf{L}_n\mathbf{U}_n\mathbf{D}_n^{-1}$ and $\alpha' = \alpha\|\boldsymbol{r}_{n-1}\|/\|\boldsymbol{r}_m\|$. Since $\mathbf{Z}_n$ has full column rank, we have $\mathbf{Z}_n^+\mathbf{Z}_n = I$. So we can write

$$\mathbf{A}\mathbf{E}\mathbf{Z}_n = \mathbf{Z}_n\mathbf{T}_n - \frac{1}{\alpha'}\frac{\boldsymbol{r}_n}{\|\boldsymbol{r}_0\|}\boldsymbol{e}_{n-m}^\top,$$

where $\mathbf{E} = I + \sqrt{\beta_m}\frac{\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}e_1^T\mathbf{Z}_n^+$. Now, as in (13) and the bound for K_n in the proof of of Theorem 2, we have

$$\|\boldsymbol{r}_n\| \leq (1+K_n)\min_{p\in\mathcal{P}_{n-m}, p(0)=1}\|p(\mathbf{A}E)\boldsymbol{r}_m\|, \tag{14}$$

where

$$\begin{aligned}K_n =& \|\mathbf{A}\mathbf{E}\mathbf{Z}_n\mathbf{T}_n^{-1}[\mathbf{I}_n\ 0]\mathbf{Z}_{n+1}^+\| \\ \leq& \|\mathbf{A}\mathbf{Z}_n + \sqrt{\beta_m}\frac{\mathbf{A}\boldsymbol{p}_{m-1}}{\|\boldsymbol{r}_{m-1}\|}e_1^T\|\,\|\mathbf{T}_n^{-1}\|\,\|\mathbf{Z}_{n+1}^+\| \\ \leq& (\|\mathbf{A}\|\,\|\mathbf{Z}_n\| + \|\mathbf{A}\|\sqrt{\beta_m}\frac{\|\boldsymbol{p}_{m-1}\|}{\|\boldsymbol{r}_{m-1}\|})\,\|\mathbf{T}_n^{-1}\|\,\|\mathbf{Z}_{n+1}^+\| \\ \leq& \|\mathbf{A}\|\,\|\mathbf{Z}_n\|(1+\sqrt{\beta_m}\frac{\|\boldsymbol{p}_{m-1}\|}{\|\boldsymbol{r}_{m-1}\|})\,\|\mathbf{T}_n^{-1}\|\,\|\mathbf{Z}_{n+1}^+\| \\ \leq& n\alpha(1+\frac{n-m}{2}\rho)(1+\sqrt{\beta_m}\frac{\|\boldsymbol{p}_{m-1}\|}{\|\boldsymbol{r}_{m-1}\|})\,\|\mathbf{A}\|\kappa(\mathbf{Z}_{n+1}).\end{aligned}$$

Now, by Lemma 3, we have

$$\sqrt{\beta_m}\frac{\|\boldsymbol{p}_{m-1}\|}{\|\boldsymbol{r}_{m-1}\|} \leq \frac{\|\boldsymbol{r}_m\|}{\|\boldsymbol{r}_{m-1}\|} + \frac{\|\boldsymbol{r}_m\|}{\|\boldsymbol{r}_{m-2}\|} + \ldots + \frac{\|\boldsymbol{r}_m\|}{\|\boldsymbol{r}_0\|} \leq m\sqrt{\rho}.$$

Thus $K_n \leq n\alpha(1+\frac{n-m}{2}\rho)(1+m\sqrt{\rho})\|\mathbf{A}\|\kappa(\mathbf{Z}_{n+1})$. □

VII. Proof of the Convergence of FRSGD

Lemma 4. *Let f have Lipschitz gradient with constant $L>0$ and let $\{\boldsymbol{w}_n\}_{n\geq 0}$ be generated by FRSGD with momentum in* (9)*, we have*

$$\sup_n\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\| \leq \frac{\alpha R}{\delta} \tag{15}$$

and

$$\sum_{n=1}^K\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|^2 \leq \frac{\alpha^2KR^2}{\delta^2}. \tag{16}$$

Proof. Note that FRSGD can be rewritten as

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \alpha\boldsymbol{r}_n + \hat{\beta}_n(\boldsymbol{w}_n - \boldsymbol{w}_{n-1}).$$

It holds that

$$\begin{aligned}\|\alpha\boldsymbol{r}_n\| =& \|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n - \hat{\beta}_n(\boldsymbol{w}_n - \boldsymbol{w}_{n-1})\| \\ \geq& \|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\| - \|\hat{\beta}_n(\boldsymbol{w}_n - \boldsymbol{w}_{n-1})\| \\ \geq& \|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\| - (1-\delta)\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|.\end{aligned}$$

Thus,

$$\begin{aligned}\|\alpha\boldsymbol{r}_n\|^2 \geq& (\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\| - (1-\delta)\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|)^2 \\ =& \|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\|^2 - 2(1-\delta)\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\| \\ &\times\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\| + (1-\delta)^2\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|^2 \\ \geq& \delta\|\boldsymbol{w}_{n+1} - \boldsymbol{w}_n\|^2 - \delta(1-\delta)\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|^2.\end{aligned}$$

By using mathematical induction, we then get (15).

Summing the above equation from $n=1$ to $K-1$, we get

$$\begin{aligned}&\delta^2\sum_{n=1}^K\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|^2 \\ \leq& \delta\|\boldsymbol{w}_{K+1} - \boldsymbol{w}_{K-1}\|^2 - \delta\|\boldsymbol{w}_1 - \boldsymbol{w}_0\|^2 + \delta^2\sum_{n=1}^{K-1}\|\boldsymbol{w}_n - \boldsymbol{w}_{n-1}\|^2 \\ =& \sum_{n=1}^{K-1}\alpha^2\|\boldsymbol{r}_n\| \leq \alpha^2KR^2.\end{aligned}$$

□

Lemma 5. *Let f have Lipschitz gradient with constant $L > 0$ and let $\{\boldsymbol{w}_n\}_{n\geq 0}$ be generated by FRSGD with momentum in* (9)*, we have*

$$\sum_{n=1}^{K}\hat{\beta}_n\mathbb{E}\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\leq\frac{(1-\delta)L}{\delta}\sum_{n=1}^{K}\mathbb{E}\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2.$$

Proof. Direct calculation yields

$$\begin{aligned}
&\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
=&\langle\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
&+\langle\nabla f(\boldsymbol{w}_n)-\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
\leq&\langle\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
&+\|\nabla f(\boldsymbol{w}_n)-\nabla f(\boldsymbol{w}_{n-1})\|\cdot\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|\\
\leq&\langle\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle+L\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2\\
=&\langle\nabla f(\boldsymbol{w}_{n-1}),-\alpha\boldsymbol{r}_{n-1}+\hat{\beta}_{n-1}(\boldsymbol{w}_{n-1}-\boldsymbol{w}_{n-2})\rangle\\
&+L\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2\\
=&-\alpha\langle\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{r}_{n-1}\rangle+\hat{\beta}_{n-1}\langle\nabla f(\boldsymbol{w}_{n-1}),\boldsymbol{w}_{n-1}-\boldsymbol{w}_{n-2}\rangle\\
&+L\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2
\end{aligned}$$

Taking expectation, we then get

$$\begin{aligned}
&\mathbb{E}\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
\leq&-\alpha\mathbb{E}\|\nabla f(\boldsymbol{w}_{n-1})\|^2+\hat{\beta}_{n-1}\mathbb{E}\langle\nabla f(\boldsymbol{w}_{n-1}),(\boldsymbol{w}_{n-1}-\boldsymbol{w}_{n-2})\rangle\\
&+L\mathbb{E}\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2\\
\leq&\hat{\beta}_{n-1}\mathbb{E}\langle\nabla f(\boldsymbol{w}_{n-1}),(\boldsymbol{w}_{n-1}-\boldsymbol{w}_{n-2})\rangle+L\mathbb{E}\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2.
\end{aligned}$$

With induction and the fact that $\boldsymbol{w}_1=\boldsymbol{w}_0$, we then get

$$\begin{aligned}
&\hat{\beta}_n\mathbb{E}\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\leq L\sum_{i=2}^{n}(\prod_{j=i}^{n}\hat{\beta}_j)\mathbb{E}\|\boldsymbol{w}_i-\boldsymbol{w}_{i-1}\|^2\\
&\leq L\sum_{i=2}^{n}(1-\delta)^{n+1-i}\mathbb{E}\|\boldsymbol{w}_i-\boldsymbol{w}_{i-1}\|^2.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
&\sum_{n=1}^{K}\hat{\beta}_k\mathbb{E}\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
\leq&L\sum_{n=1}^{K}\sum_{i=1}^{n}(1-\delta)^{n+1-i}\mathbb{E}\|\boldsymbol{w}_i-\boldsymbol{w}_{i-1}\|^2\\
\leq&\frac{(1-\delta)L}{\delta}\sum_{n=1}^{K}\mathbb{E}\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2.
\end{aligned}$$

□

Proof of Theorem 3. The Lipschitz property yields

$$\begin{aligned}
&f(\boldsymbol{w}_{n+1})\leq f(\boldsymbol{w}_n)+\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_{n+1}-\boldsymbol{w}_n\rangle+\frac{L}{2}\|\boldsymbol{w}_{n+1}-\boldsymbol{w}_n\|^2\\
=&f(\boldsymbol{w}_n)-\alpha\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{r}_n\rangle+\hat{\beta}_n\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\\
&+\frac{L}{2}\|\boldsymbol{w}_{n+1}-\boldsymbol{w}_n\|^2.
\end{aligned}$$

Taking expectation, we then get

$$\begin{aligned}
&\mathbb{E}f(\boldsymbol{w}_{n+1})\leq\mathbb{E}f(\boldsymbol{w}_n)-\alpha\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\\
&+\mathbb{E}(\hat{\beta}_n\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle)+\frac{L}{2}\mathbb{E}\|\boldsymbol{w}_{n+1}-\boldsymbol{w}_n\|^2.
\end{aligned}$$

With Lemmas 4 and 5, we are then led to

$$\begin{aligned}
&\sum_{n=1}^{K}\alpha\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\leq\sum_{n=1}^{K}\mathbb{E}\Big(\hat{\beta}_n\langle\nabla f(\boldsymbol{w}_n),\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\rangle\Big)\\
&+\frac{L}{2}\sum_{n=1}^{K}\mathbb{E}\|\boldsymbol{w}_n-\boldsymbol{w}_{n-1}\|^2+f(\boldsymbol{w}_1)-\min f.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\min_{1\leq n\leq K}\{\mathbb{E}\|\nabla f(\boldsymbol{w}_n)\|^2\}\leq&\frac{\frac{L}{2}\alpha R^2\delta+(1-\delta)L\alpha R^2}{\delta^3}\\
&+\frac{f(\boldsymbol{w}_1)-\min f}{K\alpha}
\end{aligned}$$

By denoting $C_1:=\frac{LR^2\delta+2LR^2(1-\delta)}{2\delta^3}$, we get the result. □

VIII. Adversarial Attacks

We focus on the ℓ_∞ norm-based FGSM, IFGSM, and C&W white-box attacks. For a given image-label pair $\{\boldsymbol{x},y\}$, a given ML model $g(\boldsymbol{x},\boldsymbol{w})$, and the associated loss $f(\boldsymbol{x},y):=f(g(\boldsymbol{x},\boldsymbol{w}),y)$:

- Fast gradient sign method (FGSM) searches an adversarial, $\boldsymbol{x}'$, within an ℓ_∞-ball as
$$\boldsymbol{x}'=\boldsymbol{x}+\epsilon\cdot\operatorname{sign}\left(\nabla_{\boldsymbol{x}}f(\boldsymbol{x},y)\right),$$
and we set $\epsilon=8/255$ in all of our experiments.
- Iterative FGSM (IFGSM^M) [14] iterates FGSM and clip the range as
$$\boldsymbol{x}^{(m)}=\operatorname{Clip}_{\boldsymbol{x},\epsilon}\left\{\boldsymbol{x}^{(m-1)}+\alpha\cdot\operatorname{sign}\left(\nabla_{\boldsymbol{x}^{(m-1)}}f\right)\right\},$$
with $\boldsymbol{x}^{(0)}=\boldsymbol{x},\ m=1,\ldots,M$ and we set $\epsilon=8/255$ and $\alpha=1/255$ in IFGSM attacks with different number of iterations.
- C&W attack [4] searches the minimal perturbation (δ) attack as
$$\min_{\boldsymbol{\delta}}||\boldsymbol{\delta}||_\infty,\ \text{ subject to }\ g(\boldsymbol{w},\boldsymbol{x}+\boldsymbol{\delta})=t,\ \boldsymbol{x}+\boldsymbol{\delta}\in[0,1]^d,$$
for $\forall t\neq y$.

we use the same setting as that used in [51] for C&W attack.

References

[1] M. AL-BAALI. Descent Property and Global Convergence of the Fletcher—Reeves Method with Inexact Line Search. *IMA Journal of Numerical Analysis*, 5(1):121–124, 01 1985.

[2] Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8624–8628. IEEE, 2013.

[3] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[4] N. Carlini and D.A. Wagner. Towards evaluating the robustness of neural networks. *IEEE European Symposium on Security and Privacy*, pages 39–57, 2016.

[5] Yair Carmon and John C Duchi. Analysis of krylov subspace solutions of regularized non-convex quadratic

problems. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10705–10715. Curran Associates, Inc., 2018.

[6] Yu-Hong Dai and Yaxiang Yuan. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM Journal on optimization*, 10(1):177–182, 1999.

[7] Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146(1-2):37–75, 2014.

[8] Timothy Dozat. Incorporating nesterov momentum into adam. 2016.

[9] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(Jul):2121–2159, 2011.

[10] Reeves Fletcher and Colin M Reeves. Function minimization by conjugate gradients. *The computer journal*, 7(2):149–154, 1964.

[11] Jean Charles Gilbert and Jorge Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, 2(1):21–42, 1992.

[12] Pontus Giselsson and Stephen Boyd. Monotonicity and restart in fast gradient methods. In *53rd IEEE Conference on Decision and Control*, pages 5058–5063. IEEE, 2014.

[13] Gene H. Golub and Qiang Ye. Inexact preconditioned conjugate gradient method with inner-outer iteration. *SIAM Journal on Scientific Computing*, 21(4):1305–1320, 1999.

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[15] William W Hager and Hongchao Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on optimization*, 16(1):170–192, 2005.

[16] Moritz Hardt. Robustness versus acceleration. http://blog.mrtz.org/2014/08/18/robustness-versus-acceleration.html, 2014.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[19] Magnus R Hestenes et al. Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49(6):409–436, 1952.

[20] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. https://www.cs.toronto.edu/~kriz/cifar.html, 2009.

[23] Quoc V. Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y. Ng. On optimization methods for deep learning, 2011.

[24] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[25] Liyuan Liu. Radam. https://github.com/LiyuanLucasLiu/RAdam/tree/master/cifar_imagenet, 2019.

[26] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2020.

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[28] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.

[29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[30] Martin Moller. Moller, m.f.: A scaled conjugate gradient algorithm for fast supervised learning. neural networks 6, 525-533. *Neural Networks*, 6:525–533, 12 1993.

[31] Arkaddii S Nemirovskii and Yu E Nesterov. Optimal methods of smooth convex minimization. *USSR Computational Mathematics and Mathematical Physics*, 25(2):21–30, 1985.

[32] Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. 1998.

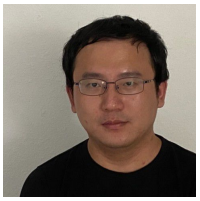
[33] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[34] Tan Nguyen, Richard Baraniuk, Andrea Bertozzi, Stanley Osher, and Bao Wang. Momentumrnn: Integrating momentum into recurrent neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1924–1936. Curran Associates, Inc., 2020.

[35] Tan Minh Nguyen, Richard Baraniuk, Robert Kirby, Stanley Osher, and Bao Wang. Momentum transformer: Closing the performance gap between self-attention and its linearization. In Bin Dong, Qianxiao Li, Lei Wang, and Zhi-Qin John Xu, editors, *Proceedings of Mathematical and Scientific Machine Learning*, volume 190 of *Proceedings of Machine Learning Research*, pages 189–204. PMLR, 15–17 Aug 2022.

[36] Chengcheng Ning, Huajun Zhou, Yan Song, and Jinhui Tang. Inception single shot multibox detector for object detection. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 549–

554. IEEE, 2017.
[37] Brendan O'donoghue and Emmanuel Candes. Adaptive restart for accelerated gradient schemes. *Foundations of computational mathematics*, 15(3):715–732, 2015.
[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.
[39] Elijah Polak and Gerard Ribiere. Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 3(R1):35–43, 1969.
[40] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.
[41] Michael James David Powell. Some convergence properties of the conjugate gradient method. *Mathematical Programming*, 11(1):42–49, 1976.
[42] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.
[43] Vincent Roulet and Alexandre d'Aspremont. Sharpness, restart and acceleration. In *Advances in Neural Information Processing Systems*, pages 1119–1129, 2017.
[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.
[45] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.
[46] JR SHEWCHUCK. An introduction to the conjugate gradient method without agonizing pain. *Technical Report CMU-CS-94-125*, 1994.
[47] Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2014.
[48] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.
[49] Charles H. Tong and Qiang Ye. Analysis of the finite precision bi-conjugate gradient algorithm for nonsymmetric linear systems. *Math. Comput.*, 69:1559–1575, 2000.
[50] Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.
[51] B. Wang, B. Yuan, Z. Shi, and S. Osher. ResNet ensemble via the Feynman-Kac formalism to improve natural and robust acurcies. In *Advances in Neural Information Processing Systems*, 2019.
[52] Bao Wang, Tan M Nguyen, Andrea L Bertozzi, Richard G Baraniuk, and Stanley J Osher. Scheduled restart momentum for accelerated stochastic gradient descent. *arXiv preprint arXiv:2002.10583*, 2020.
[53] Hedi Xia, Vai Suliafu, Hangjie Ji, Tan Nguyen, Andrea Bertozzi, Stanley Osher, and Bao Wang. Heavy ball neural ordinary differential equations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 18646–18659. Curran Associates, Inc., 2021.
[54] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
[55] Lei-Hong Zhang, Chungen Shen, and Ren-Cang Li. On the generalized lanczos trust-region method. *SIAM Journal on Optimization*, 27(3):2110–2142, 2017.
[56] Huajun Zhou, Zechao Li, Chengcheng Ning, and Jinhui Tang. Cad: Scale invariant framework for real-time object detection. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 760–768, 2017.
[57] G Zoutendijk. Nonlinear programming, computational methods. *Integer and nonlinear programming*, pages 37–86, 1970.



Bao Wang received Ph.D. in 2016 from Michigan State University. He is an assistant professor of mathematics at the University of Utah. He is a recipient of the Chancellor's award for postdoc research at UCLA. His research interests include scientific computing and deep learning.

Qiang Ye received Ph.D. in 1989 from the University of Calgary. He held a faculty position at the University of Manitoba before moving to the University of Kentucky, where he is currently a Professor of Mathematics. He is a recipient of the Edwards Research Professorship and the University Research Professorship at the University of Kentucky. He has received the Marcel F. Neuts Prize for best paper in the journal Stochastic Models. His research interests include scientific computing and deep learning.