SIAM J. IMAGING SCIENCES
Vol. 15, No. 2, pp. 738–761

Scheduled Restart Momentum for Accelerated Stochastic Gradient Descent*

Bao Wang†, Tan Nguyen‡, Tao Sun§, Andrea L. Bertozzi‡, Richard G. Baraniuk¶, and Stanley J. Osher‡

Abstract. Stochastic gradient descent (SGD) algorithms, with constant momentum and its variants such as Adam, are the optimization methods of choice for training deep neural networks (DNNs). There is great interest in speeding up the convergence of these methods due to their high computational expense. Nesterov accelerated gradient with a time-varying momentum (NAG) improves the convergence rate of gradient descent for convex optimization using a specially designed momentum; however, it accumulates error when the stochastic gradient is used, slowing convergence at best and diverging at worst. In this paper, we propose *scheduled restart SGD* (SRSGD), a new NAG-style scheme for training DNNs. SRSGD replaces the constant momentum in SGD by the increasing momentum in NAG but stabilizes the iterations by resetting the momentum to zero according to a schedule. Using a variety of models and benchmarks for image classification, we demonstrate that, in training DNNs, SRSGD significantly improves convergence and generalization; for instance, in training ResNet-200 for ImageNet classification, SRSGD achieves an error rate of 20.93% versus the benchmark of 22.13%. These improvements become more significant as the network grows deeper. Furthermore, on both CIFAR and ImageNet, SRSGD reaches similar or even better error rates with significantly fewer training epochs compared to the SGD baseline. Our implementation of SRSGD is available at https://github.com/minhtannguyen/SRSGD.

Key words. stochastic optimization, Nesterov accelerated gradient, restart, deep learning

AMS subject classifications. 65B99, 68T05, 68U01, 93E35

DOI. 10.1137/21M1453311

1. Introduction.

Training many machine learning models reduces to solving the finite-sum optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) := \min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} f_i(\boldsymbol{w}) := \mathcal{L}(g(\boldsymbol{x}_i,\boldsymbol{w}),y_i), \tag{1.1}$$

*Received by the editors October 15, 2021; accepted for publication (in revised form) January 5, 2022; published electronically May 31, 2022.
https://doi.org/10.1137/21M1453311
Funding: The work of the authors was supported by National Science Foundation grants DMS-1924935, DMS-1952339, CCF-1911094, IIS-1838177, and IIS-1730574, DOE grant DE-SC0021142, ONR grants N00014-18-12571, N00014-17-1-2551, and N00014-18-1-2047, AFOSR grant FA9550-18-1-0478, DARPA grant G001534-7500, a Vannevar Bush Faculty Fellowship, NSF grant 2030859 to the Computing Research Association for the CIFellows Project, the NSF Graduate Research Fellowship Program, and NSF IGERT Training Grant DGE-1250104.

†Department of Mathematics and Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT 84112 USA (wangbaonj@gmail.com).
‡Department of Mathematics, UCLA, Los Angeles, CA 90095 USA (mn15@rice.edu, bertozzi@math.ucla.edu, sjo@math.ucla.edu).
§College of Computer, NUDT, China, 999078 (tsun@math.ucla.edu).
¶Department of ECE, Rice University, Houston, TX 77005 USA (richb@rice.edu).

where $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ are the training samples and $\mathcal{L}$ is the loss function, e.g., cross-entropy loss for a classification task, that measure the discrepancy between the ground-truth label y_i and the prediction by the model $g(\cdot, \boldsymbol{w})$, parametrized by $\boldsymbol{w}$. The problem (1.1) is known as *empirical risk minimization* (ERM). In many applications, $f(\boldsymbol{w})$ is nonconvex, and $g(\cdot, \boldsymbol{w})$ is chosen among deep neural networks (DNNs) due to their preeminent performance across various tasks. These deep models are heavily overparametrized and require large amounts of training data. Thus, both N and the dimension of $\boldsymbol{w}$ can scale up to millions or even billions. These complications pose serious computational challenges.

One of the simplest algorithms to solve (1.1) is gradient descent (GD), which updates $\boldsymbol{w}$ according to

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - s_k \frac{1}{N} \sum_{i=1}^{N} \nabla f_i \left(\boldsymbol{w}^k \right), \tag{1.2}$$

where $s_k > 0$ is the step size at the kth iteration. Computing $\nabla f(\boldsymbol{w}^k)$ on the entire training set is memory intensive and often prohibitive for devices with limited random access memory such as GPUs used for deep learning (DL). In practice, we sample a subset of the training set, of size m with $m \ll N$, to approximate $\nabla f(\boldsymbol{w}^k)$ by the minibatch gradient $1/m \sum_{j=1}^{m} \nabla f_{i_j}(\boldsymbol{w}^k)$, resulting in the (minibatch)-stochastic gradient descent (SGD). SGD and its accelerated variants are among the most used optimization algorithms in machine learning. These gradient-based algorithms have low computational complexity, and they are easy to parallelize, making them suitable for large-scale and high dimensional problems [53, 52].

Nevertheless, GD and SGD have issues with slow convergence, especially when the problem is ill-conditioned. There are two common techniques to accelerate GD and SGD: adaptive step size [14, 23, 51] and momentum [41]. GD with constant momentum leverages the previous step information to accelerate GD according to

$$\boldsymbol{v}^{k+1} = \boldsymbol{w}^k - s_k \nabla f \left(\boldsymbol{w}^k \right); \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \mu \left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k \right), \tag{1.3}$$

where $\mu > 0$ is a constant. A similar acceleration can be achieved by the heavy-ball (HB) method [41]. HB have the same convergence rate of $O(1/k)$ as that of GD for convex smooth optimization. A breakthrough due to [38] replaces μ with $(k-1)/(k+2)$, which is known as the Nesterov accelerated gradient (NAG) with time-varying momentum. NAG accelerates the convergence rate to $O(1/k^2)$, which is optimal for convex and smooth loss functions [38]. In practice, NAG momentum can accelerate GD for nonconvex optimization, especially when the underlying problem is poorly conditioned [18]. However, NAG accumulates error and causes instability when the gradient is inexact [13, 2]. In many DL applications, constant momentum achieves state-of-the-art results, for instance, in training DNNs for image classification. Since NAG momentum achieves a much better convergence rate than constant momentum with exact gradient for general convex optimization, we consider the following question:

Can we leverage NAG with a time-varying momentum parameter to accelerate SGD in training DNNs and improve the test accuracy of the trained models?

Contributions. We answer the above question by proposing the first algorithm that integrates scheduled restart NAG momentum with plain SGD. Here, we restart the momen-

tum, which is orthogonal to the learning rate restart [33]. We name the resulting algorithm *scheduled restart SGD (SRSGD)*. The major practical benefits of SRSGD are fourfold:

- SRSGD remarkably speeds up DNN training. For image classification, SRSGD *significantly reduces the number of training epochs while preserving or even improving the deep network's accuracy*. In particular, on CIFAR10/100, the number of training epochs is reduced by half with SRSGD, while on ImageNet the reduction in training epochs is also remarkable.
- DNNs trained by SRSGD *generalize significantly better* than the benchmark optimizers. *The improvement becomes more significant as the network grows deeper* as shown in Figure 1.
- SRSGD *reduces overfitting in training very deep networks* such as ResNet-200 for ImageNet classification, enabling the accuracy to keep increasing with depth.
- SRSGD is *straightforward to implement* and only requires changing in a few lines of the SGD code. There is also no additional computational or memory overhead.

We focus on image classification with DNNs, in which SGD with momentum is the choice.

Related work. Momentum has long been used to accelerate SGD. SGD with momentum and a good initialization can handle the curvature issues in training DNNs and enable the trained models to generalize well [49]. In [27], the authors integrate momentum with adaptive step size to accelerate SGD. In this work, we study the time-varying momentum version of NAG with restart for stochastic optimization. Adaptive and scheduled restart have been used to accelerate NAG with the exact gradient [35, 37, 25, 31, 43, 39, 17, 48]; these studies of restart NAG momentum are for convex optimization with the exact gradient. In [15, 16], the authors provide analysis for the general stochastic gradient–based optimization algorithms. Restart techniques have also been used for stochastic optimization [28]. In particular, the authors of [4] have developed a multistage variant of NAG with momentum restart between stages. Our work focuses on developing NAG-based optimization for training DNNs. Efforts have also been devoted to studying the nonacceleration issues of SGD with HB and NAG momentum [26, 32], as well as accelerating first-order algorithms with noise-corrupted gradients [12, 3, 29].

Organization. In section 2, we review and discuss momentum for accelerating GD for convex smooth optimization. In section 3, we present the SRSGD algorithm and its theoretical guarantees. In section 4, we verify the efficacy of the proposed SRSGD in training DNNs for image classification on CIFAR and ImageNet. In section 4.3, we perform empirical analysis of

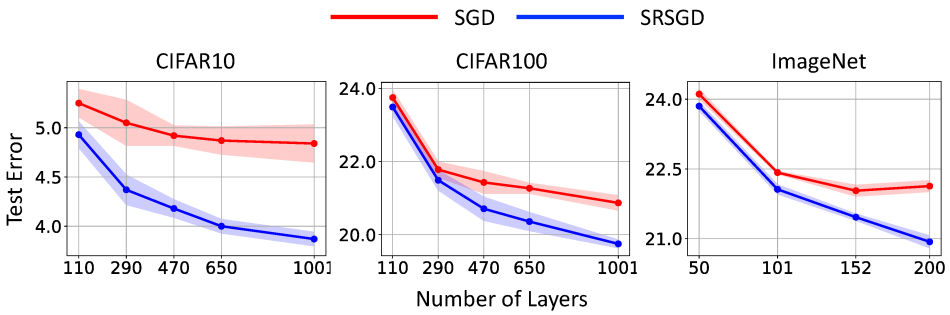

Figure 1. *Error rate versus depth of ResNet models trained with SRSGD and the baseline SGD with constant momemtum. Advantage of SRSGD continues to grow with depth.*

SRSGD. This paper ends with some concluding remarks. Technical proofs are provided in the appendix. Some experimental details and more results in training long short-term memories (LSTMs) [24] and wasserstein generative adversarial networks (WGANs) [1, 19] are provided in the supplementary materials.

Notation. We denote scalars/vectors by lowercase/lowercase boldface letters and matrices by uppercase boldface letters. For a vector $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$, we denote its ℓ_p norm by $\|\boldsymbol{x}\|_p = (\sum_{i=1}^d |x_i|^p)^{1/p}$. For a matrix $\mathbf{A}$, we use $\|\mathbf{A}\|_p$ to denote its induced norm by the vector ℓ_p norm. We denote the interval a to b (included) as $(a, b]$. For a function $f(\boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}$, we denote its gradient and Hessian as $\nabla f(\boldsymbol{w})$ and $\nabla^2 f(\boldsymbol{w})$, respectively.

2. Review: Momentum in gradient descent.

GD (1.2) is a popular approach to solve (1.1), which dates back to [9]. If $f(\boldsymbol{w})$ is convex and gradient L-Lipschitz, then GD converges with rate $O(1/k)$ by letting $s_k \equiv 1/L$ (we use this s_k in all the discussion below), which is independent of the dimension of $\boldsymbol{w}$.

HB [41] accelerates GD by using the history, which iterates as follows:

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - s_k \nabla f\left(\boldsymbol{w}^k\right) + \mu\left(\boldsymbol{w}^k - \boldsymbol{w}^{k-1}\right), \quad \mu > 0. \tag{2.1}$$

We can also accelerate GD by using the momentum scheme in (1.3). HB has a convergence rate of $O(1/k)$ for convex smooth optimization. Recently, several variants of (1.3) have been proposed for DL, e.g., [49] and [6].

NAG [38, 5] iterates as

$$\boldsymbol{v}^{k+1} = \boldsymbol{w}^k - s_k \nabla f\left(\boldsymbol{w}^k\right); \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \frac{t_k - 1}{t_{k+1}}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right), \tag{2.2}$$

where $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$ with $t_0 = 1$. NAG achieves a convergence rate $O(1/k^2)$ with the step size $s_k = 1/L$, which is the optimal rate for convex smooth optimization.

Remark 1. In [48], the authors show that $(k-1)/(k+2)$ is the asymptotic limit of $(t_k - 1)/t_{k+1}$. In the following presentation of NAG with restart, for the ease of notation, we will replace the momentum coefficient $(t_k - 1)/t_{k+1}$ with $(k-1)/(k+2)$.

2.1. Adaptive Restart NAG.

The sequences, $\{f(\boldsymbol{w}^k) - f(\boldsymbol{w}^*)\}$ where $\boldsymbol{w}^*$ is the minimum of $f(\boldsymbol{w})$, generated by GD and GD with an appropriate constant momentum (GD + Momentum, which follows (1.3)) converge monotonically to zero. However, that sequence generated by NAG oscillates, as illustrated in Figure 2(a) when $f(\boldsymbol{w})$ is a quadratic function. The authors of [39] propose adaptive restart NAG (ARNAG) (2.3), which restarts the time-varying momentum of NAG according to the change of function values, to alleviate this oscillatory phenomenon. ARNAG iterates as follows:

$$\boldsymbol{v}^{k+1} = \boldsymbol{w}^k - s_k \nabla f\left(\boldsymbol{w}^k\right); \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \frac{m(k) - 1}{m(k) + 2}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right), \tag{2.3}$$

where $m(1) = 1$; $m(k+1) = m(k) + 1$ if $f(\boldsymbol{w}^{k+1}) \le f(\boldsymbol{w}^k)$, and $m(k+1) = 1$ otherwise.

2.2. Scheduled Restart NAG. Scheduled restart (SR) is another strategy to restart the time-varying momentum of NAG. We first divide the total iterations $(0, T]$ (integers only) into a few intervals $\{I_i\}_{i=1}^m = (T_{i-1}, T_i]$ such that $(0, T] = \bigcup_{i=1}^m I_i$. In each I_i we restart the momentum after every F_i iterations. The update rule is then given by

$$\boldsymbol{v}^{k+1} = \boldsymbol{w}^k - s_k \nabla f\left(\boldsymbol{w}^k\right); \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \frac{(k \bmod F_i)}{(k \bmod F_i) + 3}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right). \tag{2.4}$$

ARNAG/SRNAG converges linearly for convex optimization problems when the Polyak–Lojasiewicz (PL) condition holds [46].

2.3. Case study—Quadratic function. Consider the following quadratic optimization problem [20]:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \mathbf{L} \boldsymbol{x} - \boldsymbol{x}^T \boldsymbol{b}, \tag{2.5}$$

where $\mathbf{L} \in \mathbb{R}^{d \times d}$ is the Laplacian of a cycle graph, and $\boldsymbol{b}$ is a d-dimensional vector whose first entry is 1 and all the other entries are 0. Note that $f(\boldsymbol{x})$ is convex with Lipschitz constant 4. In particular, we set $d = 1\text{K}$ ($1\text{K} := 10^3$). We run $T = 50\text{K}$ iterations with step size $1/4$. In SRNAG, we restart, i.e., we set the momentum to 0, after every 1K iterations. Figure 2(a) shows that GD + Momentum as in (1.3) converges faster than GD, while NAG speeds up GD + Momentum dramatically and converges to the minimum in an oscillatory fashion. Both AR and SR further accelerate NAG significantly.

3. Algorithm proposed: Scheduled restart SGD. Computing the gradient for (1.1) can be computationally costly and memory intensive, especially when the training set is large. In many applications, such as training DNNs, SGD is used. In this section, we first prove the error accumulation of SGD with NAG momentum. *The proof informs us to restart the NAG momentum for a convergence guarantee, resulting in the proposed SRSGD scheme.*

3.1. Uncontrolled bound of Nesterov accelerated SGD. Replacing $\nabla f(\boldsymbol{w}^k) := 1/N \sum_{i=1}^N \nabla f_i(\boldsymbol{w}^k)$ in (2.2) with the minibatch gradient $1/m \sum_{j=1}^m \nabla f_{i_j}(\boldsymbol{w}^k) (m \ll N)$ will lead to uncontrolled error; Theorem 3.1 formulates this point for Nesterov accelerated SGD (NASGD).

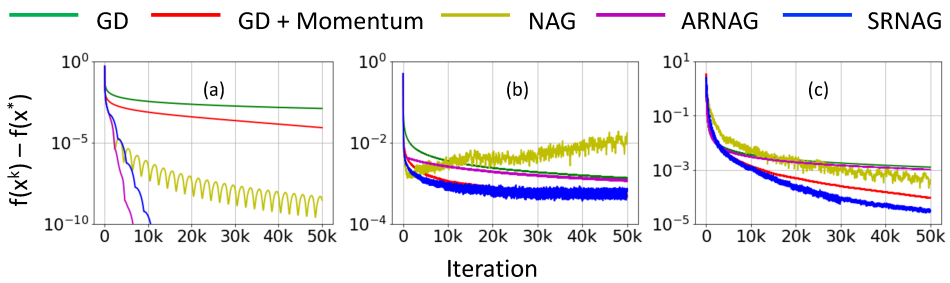

Figure 2. *Comparison between different schemes in optimizing the quadratic function in* (2.5) *with* (a) *exact gradient,* (b) *gradient with constant variance Gaussian noise, and* (c) *gradient with decaying variance Gaussian noise. NAG, ARNAG, and SRNAG can speed up convergence remarkably when an exact gradient is used. Also, SRNAG is more robust to a noisy gradient than NAG and ARNAG.*

Theorem 3.1 (uncontrolled bound of NASGD). *Let $f(\boldsymbol{w})$ be a convex and L-smooth function with $\|\nabla f(\boldsymbol{w})\| \leq R$. The sequence $\{\boldsymbol{w}^k\}_{k\geq 0}$ generated by (2.2), with stochastic gradient of bounded variance [8, 7] and using any constant step size $s_k \equiv s \leq 1/L$, satisfies*

$$\mathbb{E}\left(f\left(\boldsymbol{w}^k\right) - f(\boldsymbol{w}^*)\right) = O(k), \tag{3.1}$$

where $\boldsymbol{w}^*$ is the minimum of f, and the expectation is taken over the stochastic gradient.

The bound in Theorem 3.1 matches the bound given by [13] for a δ-inexact gradient. We will provide a proof of Theorem 3.1 in Appendix A. *The proof shows that the uncontrolled error bound is because the time-varying momentum gets closer and closer to* 1 *as iteration increases. To remedy this problem, we can restart the momentum in order to guarantee that the time-varying momentum with restart is upper bounded by a number that is strictly less than* 1.

We consider three different noisy gradients: Gaussian noise with constant and decaying variance corrupted gradients for the quadratic optimization (2.5), and a training logistic regression model for MNIST [30] classification. The detailed settings and discussion are provided in Appendix SM1 (in the supplementary material). We denote SGD with NAG momentum as NASGD and NASGD with AR and SR as ARSGD and SRSGD, respectively. The results shown in Figures 2(b) and (c) (iteration versus optimal gap for quadratic optimization (2.5)) and Figure 3(a) (iteration versus loss for training logistic regression) confirm Theorem 3.1. For these cases, SR improves the performance of NAG with inexact gradients. Moreover, when an inexact gradient is used, ARNAG/ARSGD performs almost the same as GD/SGD asymptotically because ARNAG/ARSGD restarts too often and almost degenerates to GD/SGD. The faster convergence of using SR in noisy gradient scenarios than the other algorithms motivates us to study the effectiveness of SRSGD in training deep networks.

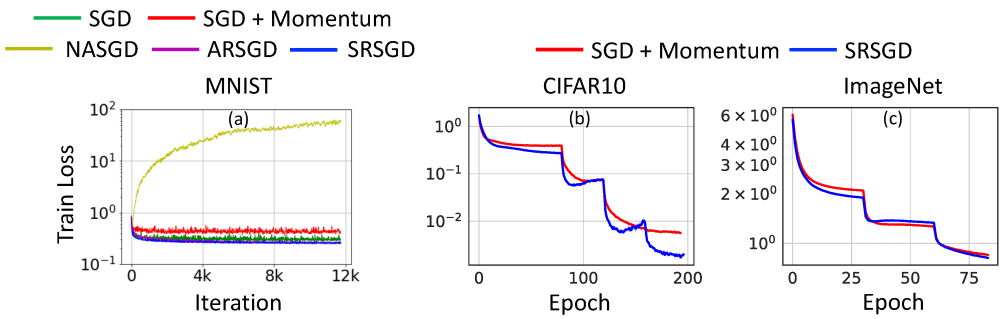

Figure 3. (a) *Training loss comparison between different schemes in training logistic regression for MNIST classification. Here, SGD is the plain SGD without momentum, and SGD + Momentum that follows (1.3) and replaces gradient with the minibatch stochastic gradient. NASGD is not robust to noisy gradient, ARSGD almost degenerates to SGD, and SRSGD performs the best in this case.* (b), (c) *Training loss versus training epoch of ResNet models trained with SRSGD (blue) and the SGD baseline with constant momentum as in PyTorch implementation, which is denoted by SGD in section 4 (red).*

3.2. SRSGD and its convergence. For ERM (1.1), SRSGD replaces $\nabla f(\boldsymbol{w})$ in (2.4) with stochastic gradient using batch size m, resulting in

$$\boldsymbol{v}^{k+1} = \boldsymbol{w}^k - s_k \frac{1}{m}\sum_{j=1}^{m} \nabla f_{i_j}\left(\boldsymbol{w}^k\right); \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \frac{(k \bmod F_i)}{(k \bmod F_i)+3}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right), \tag{3.2}$$

where F_i is the restart frequency used in the interval I_i. We implemented SRSGD, in both PyTorch [40] and Keras [11], by changing just a few lines of code on top of the existing implementation of the SGD optimizer. We formulate the convergence of SRSGD for general convex and nonconvex problems in Theorem 3.2 and provide its proof in Appendix B.

Theorem 3.2 (convergence of SRSGD). *Suppose $f(\boldsymbol{w})$ is L-smooth. Consider the sequence $\{\boldsymbol{w}^k\}_{k\geq 0}$ generated by (3.2) with stochastic gradient that is bounded and has bounded variance, and consider using any restart frequency F and using any constant step size $s_k := s \leq 1/L$. Assume that $\sum_{k\in\mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right) = \bar{R} < +\infty$ with $\bar{R}$ being a constant and the set $\mathcal{A} := \{k \in \mathbb{Z}^+ | \mathbb{E}f(\boldsymbol{w}^{k+1}) \geq \mathbb{E}f(\boldsymbol{w}^k)\}$; then we have*

$$\min_{1\leq k\leq K} \left\{\mathbb{E}\left\|\nabla f\left(\boldsymbol{w}^k\right)\right\|_2^2\right\} = O\left(s + \frac{1}{sK}\right). \tag{3.3}$$

If $f(\boldsymbol{w})$ is further convex and $\sum_{k\in\mathcal{B}}(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)) = \hat{R} < +\infty$ with $\hat{R}$ being a constant and the set $\mathcal{B} := \{k \in \mathbb{Z}^+ | \mathbb{E}\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|^2 \geq \mathbb{E}\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2\}$, then

$$\min_{1\leq k\leq K} \left\{\mathbb{E}\left(f\left(\boldsymbol{w}^k\right) - f(\boldsymbol{w}^*)\right)\right\} = O\left(s + \frac{1}{sK}\right), \tag{3.4}$$

where $\boldsymbol{w}^*$ is the minimum of f. To obtain any given ϵ error, we set $s = O(\epsilon)$ and $K = O(1/\epsilon^2)$.

Theorem 3.2 relies on the assumption that $\sum_{k\in\mathcal{A}}$ or $_{\mathcal{B}}\left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right)$ is bounded, and we provide empirical verification of this assumption on training DNNs for MNIST and CIFAR10 classification in Appendix B.1. We provide the detailed bounds of (3.3) and (3.4) in Theorem B.2 in Appendix B.

Theorem 3.2 shows that SRSGD converges as long as the momentum coefficient does not get infinitely close to 1. However, the convergence rate in Theorem 3.2 is independent of the restart frequency, which in theory is suboptimal. Establishing acceleration results with optimal restart under certain extra assumptions is an interesting problem. As far as we are aware, in the exact gradient scenario, NAG with appropriate restart can achieve linear convergence for convex optimization with the PL condition [46].

4. Experimental results. We evaluate SRSGD on a variety of benchmarks for image classification, including CIFAR10, CIFAR100, and ImageNet. In all experiments, we show the advantage of SRSGD over the widely used and well-calibrated SGD baselines with a constant momentum of 0.9[1] and decreasing learning rate at certain epochs, which we denote as SGD. We also compare SRSGD with the well-calibrated SGD in which we switch momentum to the Nesterov momentum of 0.9, and we denote this optimizer as SGD + NM. We fine tune the

[1]For SGD, we have tested momentum from 0 to 1.0 with an interval of 0.5 and confirmed 0.9 is optimal.

SGD and SGD + NM baselines to obtain the best validation performance, and we then adopt the same set of parameters for training with SRSGD. In the SRSGD experiments, *we tune the restart frequencies on small DNNs for each task based on the validation performance and apply the calibrated restart frequencies to large DNNs for the same task.* Note that ARSGD is impractical for training on large-scale datasets since it requires computing the loss over the whole training set at each iteration, which is very computationally inefficient. Alternatively, ARSGD can estimate loss and restart using minibatches, but then ARSGD restarts too often and degenerates to SGD without momentum as we mentioned in section 3. Thus, we do not compare with ARSGD in our CIFAR and ImageNet experiments. The details about hyper-parameters calibration can be found in Appendix SM2.4 (in the supplementary material). We provide a detailed description of datasets and experimental settings in Appendix SM2 (in the supplementary material). Additional experimental results in training LSTMs [24] and WGANs [1, 19] with SRSGD, as well as a comparison between SRSGD and SGD + NM on ImageNet classification task, are provided in Appendix SM3 (in the supplementary material). We also note that in all the following experiments, the training loss will blow up if we apply NASGD without restart. These further confirm the stabilizing effect of scheduled restart in training DNNs.

4.1. CIFAR10 and CIFAR100. We summarize our results for CIFAR in Tables 1 and 2. We also explore two different restarting frequency schedules for SRSGD: *linear* and *exponential* schedules. These schedules are governed by two parameters: the initial restarting frequency F_1 and the growth rate r. In both scheduling schemes, the restarting frequency at the first learning rate stage is set to F_1 during training. Then the restarting frequency at the $(k+1)$th learning rate stage is determined by

$$F_{k+1} = \begin{cases} F_1 \times r^k & \text{exponential schedule,} \\ F_1 \times (1 + (r-1) \times k) & \text{linear schedule.} \end{cases}$$

We search F_1 and r using the method outlined in Appendix SM2.4 (in the supplementary material) for both schedules on the smallest DNN used for each task. For CIFAR10, $(F_1 = 40, r = 1.25)$ and $(F_1 = 30, r = 2)$ are good initial restarting frequencies and growth rates for the exponential and linear schedules, respectively. For CIFAR100, those values are $(F_1 = 45, r = 1.5)$ for the exponential schedule and $(F_1 = 50, r = 2)$ for the linear schedule.

Improvement in accuracy increases with depth. We observe that the linear schedule of restarting frequency yields better test error on CIFAR than the exponential schedule for most of the models except for Pre-ResNet-470 and Pre-ResNet-1001 on CIFAR100 (see Tables 1 and 2). SRSGD with either a linear or an exponential schedule for restarting frequency outperforms SGD. Furthermore, the advantage of SRSGD over SGD is more significant for deeper networks. This observation holds strictly when using the linear schedule (see Figure 1) and is generally true when using the exponential schedule with only a few exceptions.

Faster convergence reduces the training time by half. SRSGD also converges faster than SGD. This result is consistent with our MNIST case study in section 3 and indeed expected since SRSGD can avoid the error accumulation when there is an inexact oracle. For CIFAR, Figure 3(b) shows that SRSGD yields smaller training loss than SGD during the training. Interestingly, SRSGD converges quickly to good loss values in the second and third stages. This suggests that *the model can be trained with SRSGD in many fewer epochs*

Table 1

Classification test error (%) on CIFAR10 using SGD, SGD + NM, and SRSGD. We report the results of SRSGD with two restarting schedules: linear (lin) and exponential (exp). The numbers of iterations after which we restart the momentum in the lin schedule are 30, 60, 90, 120 *for the* 1*st,* 2*nd,* 3*rd, and* 4*th learning rate stages. Those numbers for the exp schedule are* 40, 50, 63, 78. *We include the reported baseline results from* [22] *(in parentheses) in addition to our reproduced results.*

Network	#Params	SGD (baseline)	SGD+NM	SRSGD (lin)	SRSGD (exp)	Improve over SGD (lin/exp)	Improve over SGD+NM (lin/exp)
Pre-Res-110	1.1M	5.25 ± 0.14 (6.37)	5.24 ± 0.16	**4.93 ± 0.13**	5.00 ± 0.47	**0.32**/0.25	**0.31**/0.24
Pre-Res-290	3.0M	5.05 ± 0.23	5.04 ± 0.12	**4.37 ± 0.15**	4.50 ± 0.18	**0.68**/0.55	**0.67**/0.54
Pre-Res-470	4.9M	4.92 ± 0.10	4.97 ± 0.15	**4.18 ± 0.09**	4.49 ± 0.19	**0.74**/0.43	**0.79**/0.48
Pre-Res-650	6.7M	4.87 ± 0.14	4.80 ± 0.14	**4.00 ± 0.07**	4.40 ± 0.13	**0.87**/0.47	**0.80**/0.40
Pre-Res-1001	10.3M	4.84 ± 0.19 (4.92)	4.62 ± 0.14	**3.87 ± 0.07**	4.13 ± 0.10	**0.97**/0.71	**0.75**/0.49

Table 2

Classification test error (%) on CIFAR100 using SGD, SGD + NM, and SRSGD. We report the results of SRSGD with two restarting schedules: linear (lin) and exponential (exp). The numbers of iterations after which we restart the momentum in the lin schedule are 50, 100, 150, 200 *for the* 1*st,* 2*nd,* 3*rd, and* 4*th stages. Those numbers for the exp schedule are* 45, 68, 101, 152. *We include the reported results from* [22] *(in parentheses) in addition to our reproduced results.*

Network	#Params	SGD (baseline)	SGD+NM	SRSGD (lin)	SRSGD (exp)	Improve over SGD (lin/exp)	Improve over SGD+NM (lin/exp)
Pre-Res-110	1.2M	23.75 ± 0.20	23.65 ± 0.36	**23.49 ± 0.23**	23.50 ± 0.39	**0.26**/0.25	**0.16**/0.15
Pre-Res-290	3.0M	21.78 ± 0.21	21.68 ± 0.21	**21.49 ± 0.27**	21.58 ± 0.20	**0.29**/0.20	**0.19**/0.10
Pre-Res-470	4.9M	21.43 ± 0.30	21.21 ± 0.30	20.71 ± 0.32	**20.64 ± 0.18**	0.72/**0.79**	0.50/**0.57**
Pre-Res-650	6.7M	21.27 ± 0.14	21.04 ± 0.38	**20.36 ± 0.25**	20.41 ± 0.21	**0.91**/0.86	**0.68**/0.63
Pre-Res-1001	10.4M	20.87 ± 0.20 (22.71)	20.13 ± 0.16	19.75 ± 0.11	**19.53 ± 0.19**	1.12/**1.34**	0.38/**0.60**

compared to SGD while achieving a similar error rate. Another interesting result demonstrated in Figure 3(b) is that the training loss of SRSGD may increase in each training stage with a given learning rate and an appropriate restart, which can escape some local minima that do not generalize well. It is interesting to study this intuition for future work.

Results in Table 3 confirm the hypothesis above. We train Pre-ResNet models with SRSGD *in only* 100 *epochs*, decreasing the learning rate by a factor of 10 at the 80th, 90th, and 95th epochs while using the same linear schedule for restarting frequency as before with $(F_1 = 30, r = 2)$ for CIFAR10 and $(F_1 = 50, r = 2)$ for CIFAR100. We compare the test error of the trained models with those trained by the SGD baseline in 200 epochs. We observe that SRSGD training consistently yields lower test errors than SGD except for the case of Pre-ResNet-110 even though *the number of training epochs of our method is only half of the number of training epochs required by SGD*. For Pre-ResNet-110, SRSGD needs 110 epochs with learning rate decreased at the 80th, 90th, and 100th epochs to achieve the same error rate as the 200-epoch SGD training on CIFAR10. On CIFAR100, SRSGD training for Pre-ResNet-110 needs 140 epochs with learning rate decreased at the 80th, 100th, and 120th epochs to outperform the 200-epoch SGD. Comparison with SGD short training is provided in Appendix SM4.2 (in the supplementary material).

4.2. ImageNet. Next, we discuss our experimental results on the 1000-way ImageNet classification task [47]. We conduct our experiments on ResNet-50, 101, 152, and 200 with

Table 3

*On CIFAR*10/100 *(%), SRSGD training* with only 100 epochs *achieves classification errors (%) comparable to the SGD baseline training with* 200 *epochs.*

	CIFAR10		CIFAR100	
Network	SRSGD	Improvement	SRSGD	Improvement
Pre-Res-110	5.43 ± 0.18	-0.18	23.85 ± 0.19	-0.10
Pre-Res-290	4.83 ± 0.11	0.22	21.77 ± 0.43	0.01
Pre-Res-470	4.64 ± 0.17	0.28	21.42 ± 0.19	0.01
Pre-Res-650	4.43 ± 0.14	0.44	21.04 ± 0.20	0.23
Pre-Res-1001	4.17 ± 0.20	0.67	20.27 ± 0.11	0.60
Pre-Res-110	5.25 ± 0.10 (110 epochs)	0.00	23.73 ± 0.23 (140 epochs)	0.02

Table 4

Single crop validation errors (%) on ImageNet of ResNets trained with SGD baseline and SRSGD. We report the results of SRSGD with the increasing restarting frequency in the first two learning rates. In the last learning rate, the restarting frequency is linearly decreased to 1*. For baseline results, we also include the reported single-crop validation errors* [21] *(in parentheses).*

Network	# Params	SGD		SRSGD		Improvement	
		top-1	top-5	top-1	top-5	top-1	top-5
ResNet-50	25.56M	24.11 ± 0.10 (24.70)	7.22 ± 0.14 (7.80)	$\mathbf{23.85 \pm 0.09}$	$\mathbf{7.10 \pm 0.09}$	0.26	0.12
ResNet-101	44.55M	22.42 ± 0.03 (23.60)	6.22 ± 0.01 (7.10)	$\mathbf{22.06 \pm 0.10}$	$\mathbf{6.09 \pm 0.07}$	0.36	0.13
ResNet-152	60.19M	22.03 ± 0.12 (23.00)	6.04 ± 0.07 (6.70)	$\mathbf{21.46 \pm 0.07}$	$\mathbf{5.69 \pm 0.03}$	0.57	0.35
ResNet-200	64.67M	22.13 ± 0.12	6.00 ± 0.07	$\mathbf{20.93 \pm 0.13}$	$\mathbf{5.57 \pm 0.05}$	1.20	0.43

five different seeds. We use the official PyTorch implementation [42] for all of our ResNet models [40]. Following common practice, we train each model for 90 epochs and decrease the learning rate by a factor of 10 at the 30th and 60th epochs. We use an initial learning rate of 0.1, a momentum scaled by 0.9, and a weight decay value of 0.0001. Additional details and comparisons between SRSGD and SGD + NM are given in Appendix SM3 (in the supplementary material).

We report single crop validation errors of ResNet models trained with SGD and SRSGD on ImageNet in Table 4. In contrast to our CIFAR experiments, we observe that for ResNets trained on ImageNet with SRSGD, linearly decreasing the restarting frequency to 1 at the last learning rate stage (i.e., after the 60th epoch) helps improve the generalization of the models. Thus, in our experiments, we use linear scheduling with $(F_1 = 40, r = 2)$. From epochs 60 to 90, the restarting frequency decays to 1 linearly.

Advantage of SRSGD continues to grow with depth. Similar to the CIFAR experiments, we observe that SRSGD outperforms the SGD baseline for all ResNet models that we study. As shown in Figure 1, *the advantage of SRSGD over SGD grows with network depth*, just as in our CIFAR experiments with Pre-ResNet architectures.

Avoiding overfitting in ResNet-200. ResNet-200 is an interesting model that demonstrates that *SRSGD is better than the SGD baseline at avoiding overfitting.*[2] The ResNet-200 trained with SGD has a top-1 error of 22.13%, higher than the ResNet-152 trained with SGD, which achieves a top-1 error of 22.03% (see Table 4). [22] pointed out that ResNet-200 suffers

[2]By overfitting, we mean that the model achieves low training error but high test error.

Table 5

Comparison of single crop validation errors on ImageNet (%) between SRSGD training with fewer epochs *and SGD training with full* 90 *epochs.*

Network	SRSGD	Reduction	Improvement	Network	SRSGD	Reduction	Improvement
ResNet-50	24.30 ± 0.21	10	-0.19	ResNet-152	21.79 ± 0.07	15	0.24
ResNet-101	22.32 ± 0.06	10	0.1	ResNet-200	21.92 ± 0.17	30	0.21

from overfitting. The ResNet-200 trained with our SRSGD has a top-1 error of 20.93%, which is 1.2% lower than the ResNet-200 trained with the SGD and also lower than the ResNet-152 trained with both SRSGD and SGD, an improvement by 0.53% and 1.1%, respectively. We hypothesize that SRSGD with appropriate restart frequency is locally not monotonic (see Figures 3(b), (c)), and this property allows SRSGD to escape from bad minima in order to reach a better one, which helps avoid overfitting in very deep networks. Theoretical analysis of the observation that SRSGD is less overfitting in training DNNs is under investigation.

Training ImageNet in fewer epochs. As in the CIFAR experiments, we note that when training on ImageNet, SRSGD converges faster than SGD at the first and last learning rates while quickly reaching a good loss value at the second learning rate (see Figure 3(c)). This observation suggests that ResNets can be trained with SRSGD in fewer epochs while achieving error rates comparable to the same models trained by the SGD baseline using all 90 epochs. We summarize the results in Table 5. On ImageNet, we note that SRSGD helps reduce the number of training epochs for very deep networks (ResNet-101, 152, 200). For smaller networks like ResNet-50, training with fewer epochs slightly decreases the accuracy.

4.3. Empirical analysis. SRSGD helps reduce the training time. We find that SRSGD training using fewer epochs yields error rates comparable to both the SGD baseline and the SRSGD full training with 200 epochs on CIFAR. We conduct an ablation study to understand the impact of reducing the number of epochs on the final error rate when training with SRSGD on CIFAR10 and ImageNet. In the CIFAR10 experiments, we vary the number of epoch reductions from 15 to 90, while in the ImageNet experiments, we vary the number of epoch reductions from 10 to 30. We summarize our results in Figure 4 and provide detailed results in Appendix SM4 (in the supplementary material). For CIFAR10, we can train with 30 fewer epochs while still maintaining an error rate comparable to the full SRSGD training, and with a better error rate than the SGD baseline trained in full 200 epochs. For ImageNet, SRSGD training with fewer epochs decreases the accuracy but still obtains results comparable to the 90-epoch SGD baseline.

Impact of restarting frequency. We examine the impact of restarting frequency on the network training. We choose a case study of training a Pre-ResNet-290 on CIFAR10 using SRSGD with a linear schedule scheme for the restarting frequency. We fix the growth rate $r = 2$ and vary the initial restarting frequency F_1 from 1 to 80. As shown in Figure 5, SRSGD with a large F_1, e.g., $F_1 = 80$, approximates NASGD (yellow). We also show the training loss and test accuracy of NASGD in red. As discussed in section 3, it suffers from error accumulation due to stochastic gradients and converges slowly or even diverges. SRSGD with small F_1, e.g., $F_1 = 1$, approximates SGD without momentum (green). It converges faster initially but reaches a worse local minimum (i.e., larger loss). Typical SRSGD (blue)

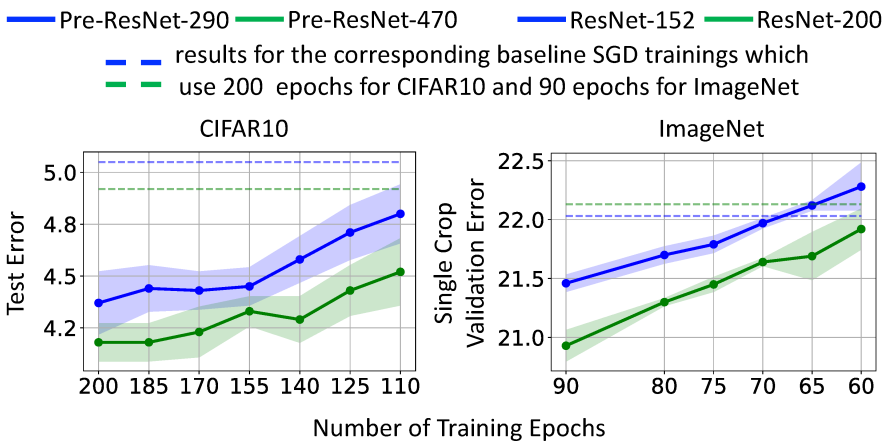

Figure 4. *Test error versus number of training epochs. Dashed lines are test errors of SGD trained with* 200 *epochs for CIFAR*10 *(left) and* 90 *epochs for ImageNet (right). For CIFAR, SRSGD with fewer epochs achieves results comparable to SRSGD with* 200 *epochs. For ImageNet, training with less epochs slightly decreases the performance of SRSGD but still achieves results comparable to* 200*-epoch SGD.*

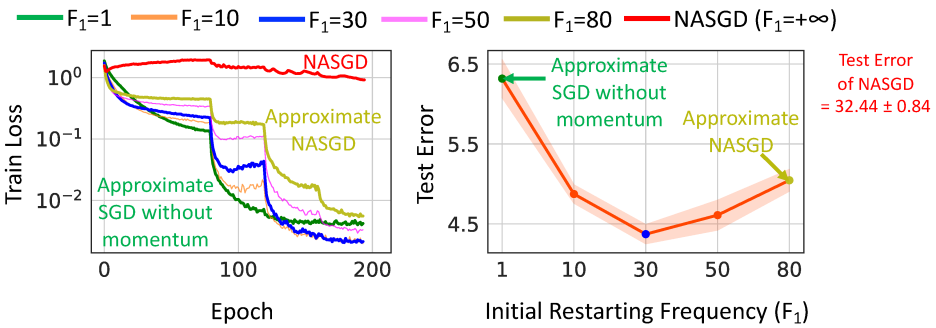

Figure 5. *Training loss (left) and test error (right) of Pre-ResNet-*290 *trained on CIFAR*10 *with different initial restarting frequencies* F_1 *(linear schedule). SRSGD with small* F_1 *approximates SGD without momentum, while SRSGD with large* F_1 *approximates NASGD. The training loss and test accuracy of NASGD are shown in red and confirm the result of Theorem* 3.1 *that NASGD accumulates error due to the stochastic gradients.*

converges faster than NASGD and to a better local minimum than both NASGD and SGD without momentum. It also achieves the best test error. We provide more empirical analysis results in Appendix SM4, SM5, and SM6 (in the supplementary material). The impact of the growth rate r is studied in Appendix SM5.2 (in the supplementary material).

5. Concluding remarks and adaptive step size algorithms.

5.1. Comparison with Adam and RMSProp. SRSGD outperforms not only SGD with momentum but also other popular optimizers including Adam and RMSProp [50] for image classification tasks. In fact, for image classification tasks, Adam and RMSProp yield worse performance than the baseline SGD with momentum [10]. Table 6 compares SRSGD with Adam and RMSprop on CIFAR10.

Table 6
Test errors on CIFAR10 of (left) Pre-ResNet-110 and (right) Pre-ResNet-290 using different optimizers.

SRSGD	Adam	RMSProp	SRSGD	Adam	RMSProp
4.93 ± 0.13%	6.83 ± 0.10%	7.31 ± 0.31%	**4.37 ± 0.15**%	6.12 ± 0.18%	7.18 ± 0.05%

5.2. Conclusion and future work. We propose the scheduled restart stochastic gradient descent (SRSGD), with two major changes from the widely used SGD with constant momentum. First, we replace the momentum in SGD with the iteration-dependent momentum that used in the Nesterov accelerated gradient (NAG). Second, we restart the NAG momentum according to a schedule to prevent error accumulation when the stochastic gradient is used. For image classification, SRSGD can significantly improve the accuracy of the trained DNNs. Also, compared to the SGD baseline, SRSGD requires fewer training epochs to reach the same trained model's accuracy. There are numerous avenues for future work: (1) deriving the optimal restart scheduling and the corresponding convergence rate of SRSGD and (2) integrating the scheduled restart NAG momentum with adaptive learning rate algorithms, e.g., Adam [27].

Appendix A. Uncontrolled bound of NASGD. Consider the optimization problem

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}), \tag{A.1}$$

where $f(\boldsymbol{w})$ is L-smooth and convex.

Starting from $\boldsymbol{w}^k$, GD with step size $\frac{1}{r}$ can be formulated as the minimization of the function

$$Q_r\left(\boldsymbol{v}, \boldsymbol{w}^k\right) := \left\langle \boldsymbol{v} - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle + \frac{r}{2}\|\boldsymbol{v} - \boldsymbol{w}^k\|_2^2. \tag{A.2}$$

With direct computation, we can get that

$$Q_r\left(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k\right) - \min Q_r\left(\boldsymbol{v}, \boldsymbol{w}^k\right) = \frac{\|\mathbf{g}^k - \nabla f(\mathbf{w}^k)\|_2}{2r},$$

where $\mathbf{g}^k := \frac{1}{m}\sum_{j=1}^m \nabla f_{i_j}(\boldsymbol{w}^k)$. We assume the variance is bounded, which gives that the stochastic gradient rule, $\mathcal{R}_s$, satisfies $\mathbb{E}[Q_r(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k) - \min Q_r(\boldsymbol{v}, \boldsymbol{w}^k)|\chi^k] \le \delta$, with δ being a constant and χ^k being the sigma algebra generated by $\boldsymbol{w}^1, \boldsymbol{w}^2, \ldots, \boldsymbol{w}^k$, i.e.,

$$\chi^k := \sigma\left(\boldsymbol{w}^1, \boldsymbol{w}^2, \ldots, \boldsymbol{w}^k\right).$$

NASGD can be reformulated as

$$\boldsymbol{v}^{k+1} \approx \arg\min_{\boldsymbol{v}} Q_r\left(\boldsymbol{v}, \boldsymbol{w}^k\right) \text{ with rule } \mathcal{R}_s; \quad \boldsymbol{w}^{k+1} = \boldsymbol{v}^{k+1} + \frac{t_k - 1}{t_{k+1}}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right), \tag{A.3}$$

where $t_0 = 1$ and $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$.

A.1. Preliminaries. We first introduce several definitions and some useful properties in variational and convex analysis. More detailed background can be found at [34, 36, 45, 44].

Letting f be convex, we say f is *L-smooth (gradient Lipschitz)* if f is differentiable and

$$\|\nabla f(\boldsymbol{v}) - \nabla f(\boldsymbol{w})\|_2 \leq L\|\boldsymbol{v} - \boldsymbol{w}\|_2,$$

and we say f is *ν-strongly convex* if for any $\boldsymbol{w}, \boldsymbol{v} \in \mathrm{dom}(f)$

$$f(\boldsymbol{w}) \geq f(\boldsymbol{v}) + \langle \nabla f(\boldsymbol{v}), \boldsymbol{w} - \boldsymbol{v}\rangle + \frac{\nu}{2}\|\boldsymbol{w} - \boldsymbol{v}\|_2^2.$$

In the following, we list several basic but useful lemmas; the proof can be found in [36].

Lemma A.1. *If f is ν-strongly convex, then for any $\boldsymbol{v} \in \mathrm{dom}(f)$ we have*

$$f(\boldsymbol{v}) - f(\boldsymbol{v}^*) \geq \frac{\nu}{2}\|\boldsymbol{v} - \boldsymbol{v}^*\|_2^2, \tag{A.4}$$

where $\boldsymbol{v}^*$ is the minimizer of f.

Lemma A.2. *If f is L-smooth, for any $\boldsymbol{w}, \boldsymbol{v} \in \mathrm{dom}(f)$,*

$$f(\boldsymbol{w}) \leq f(\boldsymbol{v}) + \langle \nabla f(\boldsymbol{v}), \boldsymbol{w} - \boldsymbol{v}\rangle + \frac{L}{2}\|\boldsymbol{w} - \boldsymbol{v}\|_2^2.$$

A.2. Uncontrolled bound of NASGD: Analysis. In this part, we denote

$$\tilde{\boldsymbol{v}}^{k+1} := \arg\min_{\boldsymbol{v}} Q_r\left(\boldsymbol{v}, \boldsymbol{w}^k\right). \tag{A.5}$$

Lemma A.3. *If the constant $r > 0$, then*

$$\mathbb{E}\left(\left\|\boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2 \middle| \chi^k\right) \leq \frac{2\delta}{r}. \tag{A.6}$$

Proof. Note that $Q_r(\boldsymbol{v}, \boldsymbol{w}^k)$ is strongly convex with constant r, and $\tilde{\boldsymbol{v}}^{k+1}$ in (A.5) is the minimizer of $Q_r(\boldsymbol{v}, \boldsymbol{w}^k)$. With Lemma A.1 we have

$$Q_r\left(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k\right) - Q_r\left(\tilde{\boldsymbol{v}}^{k+1}, \boldsymbol{w}^k\right) \geq \frac{r}{2}\left\|\boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2. \tag{A.7}$$

Notice that

$$\mathbb{E}\left[Q_r\left(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k\right) - Q_r\left(\tilde{\boldsymbol{v}}^{k+1}, \boldsymbol{w}^k\right)\right] = \mathbb{E}\left[Q_r\left(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k\right) - \min_{\boldsymbol{v}} Q_r\left(\boldsymbol{v}, \boldsymbol{w}^k\right)\right] \leq \delta.$$

The inequality (A.6) can be established by combining the above two inequalities. ∎

Lemma A.4. *If the constant satisfies $r > L$, then we have*

$$\begin{aligned}\mathbb{E}\left(f\left(\tilde{\boldsymbol{v}}^{k+1}\right) + \frac{r}{2}\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2 - \left(f\left(\boldsymbol{v}^{k+1}\right) + \frac{r}{2}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right)\right) \\ \geq -\tau\delta - \frac{r-L}{2}\mathbb{E}\left[\left\|\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2\right],\end{aligned} \tag{A.8}$$

where $\tau = \frac{L^2}{r(r-L)} + 1$.

Proof. The convexity of f gives us

$$0 \leq \left\langle \nabla f\left(\boldsymbol{v}^{k+1}\right), \boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1} \right\rangle + f\left(\tilde{\boldsymbol{v}}^{k+1}\right) - f\left(\boldsymbol{v}^{k+1}\right). \tag{A.9}$$

From the definition of the stochastic gradient rule $\mathcal{R}_s$, we have

$$\begin{aligned} -\delta &\leq \mathbb{E}\left(Q_r\left(\tilde{\boldsymbol{v}}^{k+1}, \boldsymbol{w}^k\right) - Q_r\left(\boldsymbol{v}^{k+1}, \boldsymbol{w}^k\right)\right) \\ &= \mathbb{E}\left[\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle + \frac{r}{2}\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right] \end{aligned} \tag{A.10}$$

$$- \mathbb{E}\left[\left\langle \boldsymbol{v}^{k+1} - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle + \frac{r}{2}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right]. \tag{A.11}$$

With (A.9) and (A.10), we have

$$\begin{aligned} -\delta \leq &\left(f\left(\tilde{\boldsymbol{v}}^{k+1}\right) + \frac{r}{2}\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right) - \left(f\left(\boldsymbol{v}^{k+1}\right) + \frac{r}{2}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right) \\ &+ \mathbb{E}\left\langle \nabla f\left(\boldsymbol{w}^k\right) - \nabla f\left(\tilde{\boldsymbol{v}}^{k+1}\right), \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{v}^{k+1}\right\rangle. \end{aligned} \tag{A.12}$$

With the Schwarz inequality $\langle \boldsymbol{a}, \boldsymbol{b}\rangle \leq \frac{\|\boldsymbol{a}\|_2^2}{2\mu} + \frac{\mu}{2}\|\boldsymbol{b}\|_2^2$ with $\mu = \frac{L^2}{r-L}$, $a = \nabla f(\boldsymbol{v}^{k+1}) - \nabla f(\tilde{\boldsymbol{v}}^{k+1})$ and $b = \boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}$,

$$\begin{aligned} &\left\langle \nabla f\left(\boldsymbol{w}^k\right) - \nabla f\left(\tilde{\boldsymbol{v}}^{k+1}\right), \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{v}^{k+1}\right\rangle \\ &\leq \frac{(r-L)}{2L^2}\left\|\nabla f\left(\boldsymbol{w}^k\right) - \nabla f\left(\tilde{\boldsymbol{v}}^{k+1}\right)\right\|_2^2 + \frac{L^2}{2(r-L)}\left\|\boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2 \\ &\leq \frac{(r-L)}{2}\left\|\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2 + \frac{L^2}{2(r-L)}\left\|\boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2. \end{aligned} \tag{A.13}$$

Combining (A.12) and (A.13), we have

$$\begin{aligned} -\delta \leq &\mathbb{E}\left(f\left(\tilde{\boldsymbol{v}}^{k+1}\right) + \frac{r}{2}\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right) - \mathbb{E}\left(f\left(\boldsymbol{v}^{k+1}\right) + \frac{r}{2}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^k\right\|_2^2\right) \\ &+ \frac{L^2}{2(r-L)}\mathbb{E}\left\|\boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2 + \frac{r-L}{2}\mathbb{E}\left\|\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\right\|_2^2. \end{aligned} \tag{A.14}$$

By rearrangement of the above inequality (A.14) and using Lemma A.3, we obtain the result. ∎

Lemma A.5. *If the constants satisfy $r > L$, then we have the following bounds:*

$$\mathbb{E}\left(f(\boldsymbol{v}^k) - f\left(\boldsymbol{v}^{k+1}\right)\right) \geq \frac{r}{2}\mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{v}^{k+1}\right\|_2^2 + r\mathbb{E}\left\langle \boldsymbol{w}^k - \boldsymbol{v}^k, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle - \tau\delta, \tag{A.15}$$

$$\mathbb{E}\left(f(\boldsymbol{v}^*) - f\left(\boldsymbol{v}^{k+1}\right)\right) \geq \frac{r}{2}\mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{v}^{k+1}\right\|_2^2 + r\mathbb{E}\left\langle \boldsymbol{w}^k - \boldsymbol{v}^*, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle - \tau\delta, \tag{A.16}$$

where $\tau := \frac{L^2}{r(r-L)} + 1$ and $\boldsymbol{v}^*$ is the minimum.

Proof. With Lemma A.2, we have

$$-f\left(\tilde{\boldsymbol{v}}^{k+1}\right) \geq -f\left(\boldsymbol{w}^k\right) - \left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle - \frac{L}{2}\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2. \tag{A.17}$$

Using the convexity of f, we have

$$f(\boldsymbol{v}^k) - f\left(\boldsymbol{w}^k\right) \geq \left\langle \boldsymbol{v}^k - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle,$$

i.e.,

$$f(\boldsymbol{v}^k) \geq f\left(\boldsymbol{w}^k\right) + \left\langle \boldsymbol{v}^k - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle. \tag{A.18}$$

According to the definition of $\tilde{\boldsymbol{v}}^{k+1}$ in (A.2), i.e.,

$$\tilde{\boldsymbol{v}}^{k+1} = \arg\min_{\boldsymbol{v}} Q_r(\boldsymbol{v}, \boldsymbol{w}^k) = \arg\min_{\boldsymbol{v}} \left\langle \boldsymbol{v} - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle + \frac{r}{2}\left\|\boldsymbol{v} - \boldsymbol{w}^k\right\|_2^2,$$

and the optimization condition gives

$$\tilde{\boldsymbol{v}}^{k+1} = \boldsymbol{w}^k - \frac{1}{r}\nabla f\left(\boldsymbol{w}^k\right). \tag{A.19}$$

Substituting (A.19) into (A.18), we obtain

$$f(\boldsymbol{v}^k) \geq f\left(\boldsymbol{w}^k\right) + \left\langle \boldsymbol{v}^k - \boldsymbol{w}^k, r\left(\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\right)\right\rangle. \tag{A.20}$$

Direct summation of (A.17) and (A.20) gives

$$f(\boldsymbol{v}^k) - f\left(\tilde{\boldsymbol{v}}^{k+1}\right) \geq \left(r - \frac{L}{2}\right)\left\|\tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\|_2^2 + r\left\langle \boldsymbol{w}^k - \boldsymbol{v}^k, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle. \tag{A.21}$$

Summing (A.21) and (A.8), we obtain the inequality (A.15)

$$\mathbb{E}\left[f(\boldsymbol{v}^k) - f\left(\boldsymbol{v}^{k+1}\right)\right] \geq \frac{r}{2}\mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{v}^{k+1}\right\|_2^2 + r\mathbb{E}\left\langle \boldsymbol{w}^k - \boldsymbol{v}^k, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle - \tau\delta. \tag{A.22}$$

On the other hand, with the convexity of f, we have

$$f(\boldsymbol{v}^*) - f\left(\boldsymbol{w}^k\right) \geq \left\langle \boldsymbol{v}^* - \boldsymbol{w}^k, \nabla f\left(\boldsymbol{w}^k\right)\right\rangle = \left\langle \boldsymbol{v}^* - \boldsymbol{w}^k, r\left(\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\right)\right\rangle. \tag{A.23}$$

The summation of (A.17) and (A.23) results in

$$f(\boldsymbol{v}^*) - f\left(\tilde{\boldsymbol{v}}^{k+1}\right) \geq \left(r - \frac{L}{2}\right)\|\boldsymbol{w}^k - \tilde{\boldsymbol{v}}^{k+1}\|_2^2 + r\left\langle \boldsymbol{w}^k - \boldsymbol{v}^*, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle. \tag{A.24}$$

Summing (A.24) and (A.8), we obtain

$$\mathbb{E}\left(f(\boldsymbol{v}^*) - f\left(\boldsymbol{v}^{k+1}\right)\right) \geq \frac{r}{2}\mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{v}^{k+1}\right\|_2^2 + r\mathbb{E}\left\langle \boldsymbol{w}^k - \boldsymbol{v}^*, \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k\right\rangle - \tau\delta, \tag{A.25}$$

which is the same as (A.16). ∎

Theorem A.6 (uncontrolled bound of NASGD). *(Theorem 1 with detailed bound) Let the constant r satisfy $r < L$ and the sequence $\{\boldsymbol{v}^k\}_{k\geq 0}$ be generated by NASGD with stochastic gradient that has bounded variance. By using any constant step size $s_k \equiv s \leq 1/L$, we have*

$$\mathbb{E}\left[f(\boldsymbol{v}^k) - \min_{\boldsymbol{v}} f(\boldsymbol{v})\right] \leq \left(\frac{2\tau\delta}{r} + R^2\right)\frac{4k}{3}. \tag{A.26}$$

Proof. We denote

$$F^k := \mathbb{E}(f(\boldsymbol{v}^k) - f(\boldsymbol{v}^*)).$$

By (A.15) $\times\,(t_k - 1) +$ (A.16), we have

$$\begin{aligned}\frac{2[(t_k-1)F^k - t_kF^{k+1}]}{r} &\geq t_k\mathbb{E}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^k\right\|_2^2 \\ &\quad + 2\mathbb{E}\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\rangle - \frac{2\tau t_k\delta}{r}.\end{aligned} \tag{A.27}$$

With $t_{k-1}^2 = t_k^2 - t_k$, (A.27) $\times\, t_k$ yields

$$\begin{aligned}\frac{2\left[t_{k-1}^2F^k - t_k^2F^{k+1}\right]}{r} &\geq \mathbb{E}\left\|t_k\boldsymbol{v}^{k+1} - t_k\boldsymbol{w}^k\right\|_2^2 \\ &\quad + 2t_k\mathbb{E}\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\rangle - \frac{2\tau t_k^2\delta}{r}.\end{aligned} \tag{A.28}$$

Substituting $\boldsymbol{a} = t_k\boldsymbol{v}^{k+1} - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*$ and $\boldsymbol{b} = t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*$ into identity

$$\|\boldsymbol{a}-\boldsymbol{b}\|_2^2 + 2\langle \boldsymbol{a}-\boldsymbol{b}, \boldsymbol{b}\rangle = \|\boldsymbol{a}\|_2^2 - \|\boldsymbol{b}\|_2^2, \tag{A.29}$$

it follows that

$$\begin{aligned}&\mathbb{E}\left\|t_k\boldsymbol{v}^{k+1} - t_k\boldsymbol{w}^k\right\|_2^2 + 2t_k\mathbb{E}\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{w}^k, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\rangle \\ =\;& \mathbb{E}\left\|t_k\boldsymbol{v}^{k+1} - t_k\boldsymbol{w}^k\right\|_2^2 + 2t_k\mathbb{E}\left\langle \boldsymbol{v}^{k+1} - \boldsymbol{w}^k, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\rangle \\ &+2t_k\mathbb{E}\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{v}^{k+1}, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\rangle \\ \underset{\text{(A.29)}}{=}\;& \mathbb{E}\left\|t_k\boldsymbol{v}^{k+1} - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\|_2^2 - \left\|t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\|_2^2 \\ &+2t_k\mathbb{E}\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{v}^{k+1}, t_k\boldsymbol{w}^k - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\rangle \\ =\;& \mathbb{E}\left\|t_k\boldsymbol{v}^{k+1} - (t_k-1)\boldsymbol{v}^k - \boldsymbol{v}^*\right\|_2^2 - \mathbb{E}\left\|t_{k-1}\boldsymbol{v}^k - (t_{k-1}-1)\boldsymbol{v}^{k-1} - \boldsymbol{v}^*\right\|_2^2 \\ &+2t_k\mathbb{E}\left\langle \tilde{\boldsymbol{v}}^{k+1} - \boldsymbol{v}^{k+1}, t_{k-1}\boldsymbol{v}^k - (t_{k-1}-1)\boldsymbol{v}^{k-1} - \boldsymbol{v}^*\right\rangle.\end{aligned} \tag{A.30}$$

In the third identity, we used the fact $t_k\boldsymbol{w}^k = t_k\boldsymbol{v}^k + (t_{k-1}-1)(\boldsymbol{v}^k - \boldsymbol{v}^{k-1})$. If we denote $u^k = \mathbb{E}\|t_{k-1}\boldsymbol{v}^k - (t_{k-1}-1)\boldsymbol{v}^{k-1} - \boldsymbol{v}^*\|_2^2$, (A.28) can be rewritten as

$$\begin{aligned}\frac{2t_k^2F^{k+1}}{r} + u^{k+1} &\leq \frac{2t_{k-1}^2F^k}{r} + u^k + \frac{2\tau t_k^2\delta}{r} \\ &\quad + 2t_k\mathbb{E}\left\langle \boldsymbol{v}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}, t_{k-1}\boldsymbol{v}^k - (t_{k-1}-1)\boldsymbol{v}^{k-1} - \boldsymbol{v}^*\right\rangle \\ &\leq \frac{2t_k^2F^k}{r} + u^k + \frac{2\tau t_k^2\delta}{r} + t_{k-1}^2R^2,\end{aligned} \tag{A.31}$$

where we used

$$2t_k\mathbb{E}\left\langle \boldsymbol{v}^{k+1}-\tilde{\boldsymbol{v}}^{k+1}, t_{k-1}\boldsymbol{v}^k-(t_{k-1}-1)\boldsymbol{v}^{k-1}-\boldsymbol{v}^*\right\rangle$$
$$\le t_k^2\mathbb{E}\left\|\boldsymbol{v}^{k+1}-\tilde{\boldsymbol{v}}^{k+1}\right\|_2^2+\mathbb{E}\left\|t_{k-1}\boldsymbol{v}^k-\left(t_{k-1}\boldsymbol{v}^k-(t_{k-1}-1)\boldsymbol{v}^{k-1}-\boldsymbol{v}^*\right)\right\|_2^2=2t_k^2\delta/r+t_{k-1}^2R^2.$$

Denoting

$$\xi_k:=\frac{2t_{k-1}^2F^k}{r}+u^k,$$

then, we have

$$\xi_{k+1}\le\xi_0+\left(\frac{2\tau\delta}{r}+R^2\right)\sum_{i=1}^k t_i^2=\left(\frac{2\tau\delta}{r}+R^2\right)\frac{k^3}{3}. \tag{A.32}$$

With the fact that $\xi_k\ge\frac{2t_{k-1}^2F^k}{r}\ge k^2F^k/4$, we then proved the result. ∎

Appendix B. Convergence of SRSGD. We prove the convergence of NASGD with SR, i.e., the convergence of SRSGD. We denote that $\theta^k:=\frac{t_k-1}{t_{k+1}}$ in the Nesterov iteration and $\hat{\theta}^k$ is its use in SRSGD. For any restart frequency F (positive integer), we have $\hat{\theta}^k=\theta^{k-\lfloor k/F\rfloor*F}$. In the restart version, we can see that

$$\hat{\theta}^k\le\theta^F=:\bar{\theta}<1.$$

Lemma B.1. *Let the constant satisfy $r>L$ and the sequence $\{\boldsymbol{v}^k\}_{k\ge0}$ be generated by the SRSGD with restart frequency F (any positive integer); we have*

$$\sum_{i=1}^k\|\boldsymbol{v}^i-\boldsymbol{v}^{i-1}\|_2^2\le\frac{r^2kR^2}{(1-\bar{\theta})^2}, \tag{B.1}$$

where $\bar{\theta}:=\theta^F<1$ and $R:=\sup_{\mathbf{x}}\{\|\nabla f(\mathbf{x})\|_2\}$.

Proof. It holds that

$$\begin{aligned}\left\|\boldsymbol{v}^{k+1}-\boldsymbol{w}^k\right\|_2&=\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k+\boldsymbol{v}^k-\boldsymbol{w}^k\right\|_2\\&\ge\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2-\left\|\boldsymbol{v}^k-\boldsymbol{w}^k\right\|_2\ge\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2-\bar{\theta}\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2.\end{aligned} \tag{B.2}$$

Thus,

$$\begin{aligned}\left\|\boldsymbol{v}^{k+1}-\boldsymbol{w}^k\right\|_2^2&\ge\left(\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2-\bar{\theta}\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2\right)^2\\&=\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2^2-2\bar{\theta}\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2+\bar{\theta}^2\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2^2\\&\ge(1-\bar{\theta})\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2^2-\bar{\theta}(1-\bar{\theta})\left\|\boldsymbol{v}^{k+1}-\boldsymbol{v}^k\right\|_2^2.\end{aligned} \tag{B.3}$$

Summing (B.3) from $k=1$ to K, we get

$$(1-\bar{\theta})^2\sum_{k=1}^K\left\|\boldsymbol{v}^k-\boldsymbol{v}^{k-1}\right\|_2^2\le\sum_{k=1}^K\left\|\boldsymbol{v}^{k+1}-\boldsymbol{w}^k\right\|_2^2\le r^2KR^2. \tag{B.4}$$

∎

In the following, we denote $\mathcal{A}:=\{k\in Z^+|\mathbb{E}f(\boldsymbol{v}^k)\ge\mathbb{E}f(\boldsymbol{v}^{k-1})\}$.

Theorem B.2 (convergence of SRSGD). *(Theorem 2 with detailed bound) Suppose $f(\boldsymbol{w})$ is L-smooth. Consider the sequence $\{\boldsymbol{w}^k\}_{k\geq 0}$ generated by (3.2) with a stochastic gradient that is bounded and has bound variance. Using any restart frequency F and any constant step size $s_k := s \leq 1/L$, assume that $\sum_{k\in\mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right) = \bar{R} < +\infty$; then we have*

$$\min_{1\leq k\leq K} \left\{ \mathbb{E} \left\| \nabla f\left(\boldsymbol{w}^k\right) \right\|_2^2 \right\} \leq \frac{rR^2}{(1-\bar{\theta})^2} \frac{L(1+\tilde{\theta})}{2} + \frac{rLR^2}{2} + \frac{\tilde{\theta}\tilde{R}}{rK}. \tag{B.5}$$

If $f(\boldsymbol{w})$ is further convex and the set $\mathcal{B} := \{k \in \mathbb{Z}^+ | \mathbb{E}\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|^2 \geq \mathbb{E}\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2\}$ obeys $\sum_{k\in\mathcal{B}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right) = \hat{R} < +\infty$, then

$$\min_{1\leq k\leq K} \left\{ \mathbb{E} \left(f\left(\boldsymbol{w}^k\right) - f(\boldsymbol{w}^*) \right) \right\} \leq \frac{\|\boldsymbol{w}^0 - \boldsymbol{w}^*\|^2 + \hat{R}}{2\gamma k} + \frac{\gamma R^2}{2}, \tag{B.6}$$

where $\boldsymbol{w}^*$ is the minimum of f. To obtain ϵ error, we set $s = O(\epsilon)$ and $K = O(1/\epsilon^2)$.

Proof. First, we show the convergence of SRSGD for nonconvex optimization. L-smoothness of f, i.e., Lipschitz gradient continuity, gives us

$$f\left(\boldsymbol{v}^{k+1}\right) \leq f\left(\boldsymbol{w}^k\right) + \left\langle \nabla f\left(\boldsymbol{w}^k\right), \boldsymbol{v}^{k+1} - \boldsymbol{w}^k \right\rangle + \frac{L}{2} \left\| \boldsymbol{v}^{k+1} - \boldsymbol{w}^k \right\|_2^2. \tag{B.7}$$

Taking expectation, we get

$$\mathbb{E}f\left(\boldsymbol{v}^{k+1}\right) \leq \mathbb{E}f\left(\boldsymbol{w}^k\right) - r\mathbb{E} \left\| \nabla f\left(\boldsymbol{w}^k\right) \right\|_2^2 + \frac{r^2LR^2}{2}. \tag{B.8}$$

On the other hand, we have

$$f\left(\boldsymbol{w}^k\right) \leq f(\boldsymbol{v}^k) + \hat{\theta}^k \left\langle \nabla f(\boldsymbol{v}^k), \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\rangle + \frac{L(\hat{\theta}^k)^2}{2} \left\| \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\|_2^2. \tag{B.9}$$

Then, we have

$$\begin{aligned} \mathbb{E}f\left(\boldsymbol{v}^{k+1}\right) \leq{}& \mathbb{E}f(\boldsymbol{v}^k) + \hat{\theta}^k \mathbb{E} \left\langle \nabla f(\boldsymbol{v}^k), \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\rangle \\ &+ \frac{L(\hat{\theta}^k)^2}{2} \mathbb{E} \left\| \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\|_2^2 - r\mathbb{E} \left\| \nabla f\left(\boldsymbol{w}^k\right) \right\|_2^2 + \frac{r^2LR^2}{2}. \end{aligned} \tag{B.10}$$

We also have

$$\hat{\theta}^k \langle \nabla f(\boldsymbol{v}^k), \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \rangle \leq \hat{\theta}^k \left(f(\boldsymbol{v}^k) - f\left(\boldsymbol{v}^{k-1}\right) + \frac{L}{2} \left\| \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\|_2^2 \right). \tag{B.11}$$

We then get that

$$\mathbb{E}f\left(\boldsymbol{v}^{k+1}\right) \leq \mathbb{E}f(\boldsymbol{v}^k) + \hat{\theta}^k \left(\mathbb{E}f(\boldsymbol{v}^k) - \mathbb{E}f\left(\boldsymbol{v}^{k-1}\right) \right) - r\mathbb{E} \left\| \nabla f\left(\boldsymbol{w}^k\right) \right\|_2^2 + A_k, \tag{B.12}$$

where

$$A_k := \mathbb{E}\frac{L}{2} \left\| \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\|_2^2 + \frac{L(\hat{\theta}^k)^2}{2} \mathbb{E} \left\| \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \right\|_2^2 + \frac{r^2LR^2}{2}.$$

Summing the inequality gives us

(B.13)
$$\mathbb{E}f(\boldsymbol{v}^{K+1}) \leq \mathbb{E}f(\boldsymbol{v}^0) + \tilde{\theta}\sum_{k\in\mathcal{A}}\left(\mathbb{E}f(\boldsymbol{v}^k) - \mathbb{E}f\left(\boldsymbol{v}^{k-1}\right)\right) - r\sum_{k=1}^{K}\mathbb{E}\left\|\nabla f\left(\boldsymbol{w}^k\right)\right\|_2^2 + \sum_{k=1}^{K}A_k.$$

It is easy to see that

$$\tilde{\theta}\sum_{k\in\mathcal{A}}\left(\mathbb{E}f(\boldsymbol{v}^k) - \mathbb{E}f\left(\boldsymbol{v}^{k-1}\right)\right) = \tilde{\theta}\tilde{R}.$$

We get the result by using Lemma B.1

Second, we prove the convergence of SRSGD for convex optimization. Let $\boldsymbol{w}^*$ be the minimizer of f. We have

(B.14)
$$\begin{aligned}\mathbb{E}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^*\right\|_2^2 &= \mathbb{E}\left\|\boldsymbol{w}^k - \gamma\nabla f\left(\boldsymbol{w}^k\right) - \boldsymbol{w}^*\right\|_2^2 \\ &= \mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{w}^*\right\|_2^2 - 2\gamma\mathbb{E}\left\langle\nabla f\left(\boldsymbol{w}^k\right), \boldsymbol{w}^k - \boldsymbol{w}^*\right\rangle + \gamma^2\mathbb{E}\left\|\nabla f\left(\boldsymbol{w}^k\right)\right\|_2^2 \\ &\leq \mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{x}^*\right\|_2^2 - 2\gamma\mathbb{E}\left\langle\nabla f\left(\boldsymbol{w}^k\right), \boldsymbol{w}^k - \boldsymbol{w}^*\right\rangle + \gamma^2R^2.\end{aligned}$$

We can also derive

$$\begin{aligned}\mathbb{E}\left\|\boldsymbol{w}^k - \boldsymbol{w}^*\right\|_2 &= \mathbb{E}\left\|\boldsymbol{v}^k + \hat{\theta}^k\left(\boldsymbol{v}^k - \boldsymbol{v}^{k-1}\right) - \boldsymbol{w}^*\right\|_2^2 \\ &= \mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 + 2\hat{\theta}^k\mathbb{E}\left\langle\boldsymbol{v}^k - \boldsymbol{v}^{k-1}, \boldsymbol{v}^k - \boldsymbol{w}^*\right\rangle + (\hat{\theta}^k)^2\mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{v}^{k-1}\right\|_2^2 \\ &= \mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 + \hat{\theta}^k\mathbb{E}\left(\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 + \left\|\boldsymbol{v}^{k-1} - \boldsymbol{v}^k\right\|_2^2 - \left\|\boldsymbol{v}^{k-1} - \boldsymbol{w}^*\right\|_2^2\right) \\ &\quad + (\hat{\theta})^2\mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{v}^{k-1}\right\|_2^2 \\ &= \mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 + \hat{\theta}^k\mathbb{E}\left(\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 - \left\|\boldsymbol{v}^{k-1} - \boldsymbol{w}^*\right\|_2^2\right) + 2(\hat{\theta}^k)^2\mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{v}^{k-1}\right\|_2^2,\end{aligned}$$

where we used the following identity:

$$(\boldsymbol{a}-\boldsymbol{b})^T(\boldsymbol{a}-\boldsymbol{b}) = \frac{1}{2}[\|\boldsymbol{a}-\boldsymbol{d}\|_2^2 - \|\boldsymbol{a}-\boldsymbol{c}\|_2^2 + \|\boldsymbol{b}-\boldsymbol{c}\|_2^2 - \|\boldsymbol{b}-\boldsymbol{d}\|_2^2].$$

Then, we have

(B.15)
$$\begin{aligned}\mathbb{E}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^*\right\|_2^2 \leq \mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 - 2\gamma\mathbb{E}\langle\nabla f(\boldsymbol{w}^k), \boldsymbol{w}^k - \boldsymbol{w}^*\rangle + 2(\hat{\theta}^k)^2\mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{v}^{k-1}\right\|_2^2 \\ + r^2R^2 + \hat{\theta}^k\mathbb{E}\left(\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 - \left\|\boldsymbol{v}^{k-1} - \boldsymbol{w}^*\right\|_2^2\right).\end{aligned}$$

We then get that

(B.16)
$$\begin{aligned}2\gamma\mathbb{E}\left(f\left(\boldsymbol{w}^k\right) - f(\boldsymbol{w}^*)\right) \leq \mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 - \mathbb{E}\left\|\boldsymbol{v}^{k+1} - \boldsymbol{w}^*\right\|_2^2 \\ + \hat{\theta}^k\left(\mathbb{E}\left\|\boldsymbol{v}^k - \boldsymbol{w}^*\right\|_2^2 - \mathbb{E}\left\|\boldsymbol{v}^{k-1} - \boldsymbol{w}^*\right\|_2^2\right) + r^2R^2.\end{aligned}$$

Summing the inequality gives us the desired convergence result for convex optimization. ∎

B.1. Numerical verification of the assumptions in Theorem 3.2. In this part, we numerically verify the assumptions in Theorem 2. In particular, we apply SRSGD with learning rate 0.1 to train LeNet[3] for MNIST classification. We conduct numerical verification as follows: starting from a given point $\boldsymbol{w}^0$, we randomly sample 469 minibatches (note in total we have 469 batches in the training data) with batch size 128 and compute the stochastic gradient using each minibatch. Next, we advance to the next step with each of these 469 stochastic gradients and get the approximated $\mathbb{E}f(\boldsymbol{w}^1)$. We randomly choose one of these 469 positions as the updated weights of our model. By iterating the above procedure, we can get $\boldsymbol{w}^1, \boldsymbol{w}^2, \cdots$ and $\mathbb{E}f(\boldsymbol{w}^1), \mathbb{E}f(\boldsymbol{w}^2), \cdots$ and we use these values to verify our assumptions in Theorem 2. We set restart frequencies to be 20, 40, and 80, respectively. Figure 6, top panels, plots k versus the cardinality of the set $\mathcal{A} := \{k \in \mathbb{Z}^+ | \mathbb{E}f(\boldsymbol{w}^{k+1}) \geq \mathbb{E}f(\boldsymbol{w}^k)\}$, and Figure 6, bottom panels, plots k versus $\sum_{k\in\mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right)$. Figure 6 shows that $\sum_{k\in\mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right)$ converges to a constant $\bar{R} < +\infty$. We also noticed that when the training gets plateaued, $\mathbb{E}(f(\boldsymbol{w}^k))$ still oscillates, but the magnitude of the oscillation diminishes as iterations goes, which is consistent with our plots that the cardinality of $\mathcal{A}$ increases linearly, but $\bar{R}$ converges to a finite number. These numerical results show that our assumption in Theorem 2 is reasonable.

We repeat a similar process as above and further verify the assumptions in Theorem 2 for Pre-ResNet-20 trained on CIFAR10.[4] Figure 7 confirms that the assumptions in Theorem 2 still hold in this case of CIFAR10 training with a larger and more advanced network architecture. This implies that our assumptions in Theorem 2 are reasonable across different datasets, network architectures, and training procedures.

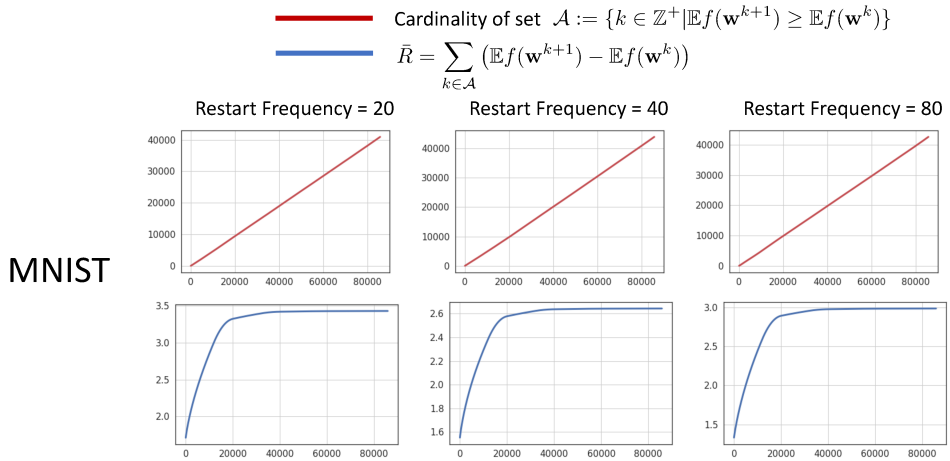

Figure 6. *Cardinality of the set $\mathcal{A} := \{k \in \mathbb{Z}^+ | \mathbb{E}f(\boldsymbol{w}^{k+1}) \geq \mathbb{E}f(\boldsymbol{w}^k)\}$ (top panels) and the value of $\bar{R} = \sum_{k\in\mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right)$ (bottom panels) of LeNet trained on MNIST. We notice that when the training gets plateaued, $\mathbb{E}(f(\boldsymbol{w}^k))$ still oscillates, but the magnitude of the oscillation diminishes as iterations go, which is consistent with our plots that the cardinality of $\mathcal{A}$ increases linearly, but $\bar{R}$ converges to a finite number under different restart frequencies. These results confirm that our assumption in Theorem 2 is reasonable.*

[3] We used the PyTorch implementation of LeNet at https://github.com/activatedgeek/LeNet-5.
[4] Implementation available at https://github.com/bearpaw/pytorch-classification.

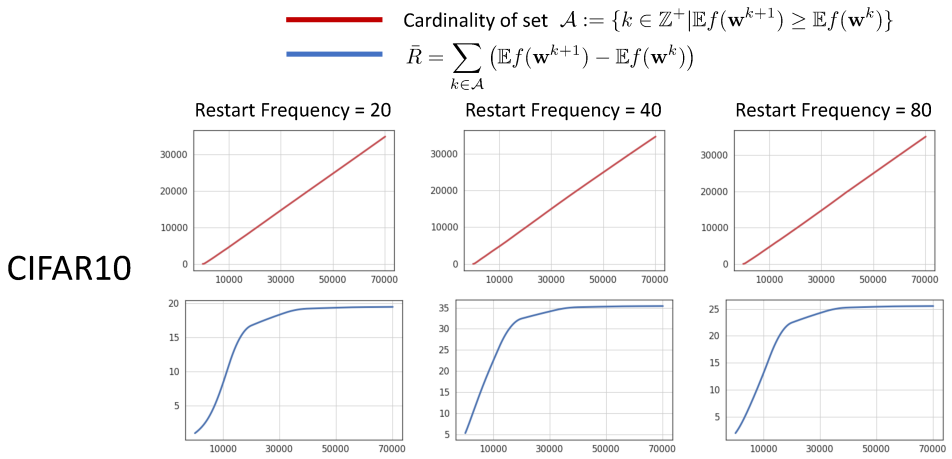

Figure 7. *Cardinality of the set* $\mathcal{A} := \{k \in \mathbb{Z}^+ | \mathbb{E}f(\boldsymbol{w}^{k+1}) \geq \mathbb{E}f(\boldsymbol{w}^k)\}$ *(top panels) and the value of* $\bar{R} = \sum_{k \in \mathcal{A}} \left(\mathbb{E}f(\boldsymbol{w}^{k+1}) - \mathbb{E}f(\boldsymbol{w}^k)\right)$ *(bottom panels) of Pre-ResNet-20 trained on CIFAR10. We notice that when the training gets plateaued,* $\mathbb{E}(f(\boldsymbol{w}^k))$ *still oscillates, but the magnitude of the oscillation diminishes as iterations go, which is consistent with our plots that the cardinality of* $\mathcal{A}$ *increases linearly, but* $\bar{R}$ *converges to a finite number under different restart frequencies. These results confirm that our assumption in Theorem 2 is reasonable.*

REFERENCES

[1] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein generative adversarial networks*, in Proceedings of the 34th International Conference on Machine Learning, D. Precup and Y. W. Teh, eds., Proceedings of Machine Learning Research 70, PMLR, pp. 214–223, http://proceedings.mlr.press/v70/arjovsky17a.html.

[2] M. Assran and M. Rabbat, *On the Convergence of Nesterov's Accelerated Gradient Method in Stochastic Settings*, preprint, arXiv:2002.12414, 2020.

[3] N. S. Aybat, A. Fallah, M. Gurbuzbalaban, and A. Ozdaglar, *Robust Accelerated Gradient Methods for Smooth Strongly Convex Functions*, preprint, arXiv:1805.10579, 2018.

[4] N. S. Aybat, A. Fallah, M. Gurbuzbalaban, and A. Ozdaglar, *A universally optimal multistage accelerated stochastic gradient method*, in Advances in Neural Information Processing Systems, 2019, pp. 8525–8536.

[5] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.

[6] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, *Advances in optimizing recurrent networks*, in Proceedings of the International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8624–8628.

[7] L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for large-scale machine learning*, SIAM Rev., 60 (2018), pp. 223–311.

[8] S. Bubeck, *Convex Optimization: Algorithms and Complexity*, preprint, arXiv:1405.4980, 2014.

[9] A. Cauchy, *Méthode générale pour la résolution des systemes d'équations simultanées*, Comp. Rend. Sci. Paris (1847).

[10] J. Chen and A. Kyrillidis, *Decaying Momentum Helps Neural Network Training*, preprint, arXiv:1910.04952, 2019.

[11] F. Chollet et al., *Keras*, https://keras.io, 2015.

[12] M. B. Cohen, J. Diakonikolas, and L. Orecchia, *On Acceleration with Noise-Corrupted Gradients*, preprint, arXiv:1805.12591, 2018.

[13] O. Devolder, F. Glineur, and Y. Nesterov, *First-order methods of smooth convex optimization with inexact oracle*, Math. Program., 146 (2014), pp. 37–75.

[14] J. Duchi, E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization*, J. Mach. Learn. Res., 12 (2011), pp. 2121–2159.
[15] S. Ghadimi and G. Lan, *Stochastic first-and zeroth-order methods for nonconvex stochastic programming*, SIAM J. Optim., 23 (2013), pp. 2341–2368.
[16] S. Ghadimi and G. Lan, *Accelerated gradient methods for nonconvex nonlinear and stochastic programming*, Math. Program., 156 (2016), pp. 59–99.
[17] P. Giselsson and S. Boyd, *Monotonicity and restart in fast gradient methods*, in Proceedings of the 53rd IEEE Conference on Decision and Control, IEEE, 2014, pp. 5058–5063.
[18] G. Goh, *Why momentum really works*, Distill, 2 (2017), e6.
[19] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, *Improved training of Wasserstein GANs*, in Advances in Neural Information Processing Systems, 2017, pp. 5767–5777.
[20] M. Hardt, *Robustness Versus Acceleration*, http://blog.mrtz.org/2014/08/18/robustness-versus-acceleration.html, 2014.
[21] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Networks*, https://github.com/KaimingHe/deep-residual-networks, 2016.
[22] K. He, X. Zhang, S. Ren, and J. Sun, *Identity mappings in deep residual networks*, in European Conference on Computer Vision, Springer, New York, 2016, pp. 630–645.
[23] G. Hinton, N. Srivastava, and K. Swersky, *Neural Networks for Machine Learning Lecture 6a: Overview of Mini-Batch Gradient Descent*, https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.
[24] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Comput., 9 (1997), pp. 1735–1780.
[25] A. Iouditski and Y. Nesterov, *Primal-Dual Subgradient Methods for Minimizing Uniformly Convex Functions*, preprint, arXiv:1401.1792, 2014.
[26] R. Kidambi, P. Netrapalli, P. Jain, and S. Kakade, *On the insufficiency of existing momentum schemes for stochastic optimization*, in Proceedings of the Information Theory and Applications Workshop, IEEE, 2018, pp. 1–9.
[27] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, preprint, arXiv:1412.6980, 2014.
[28] A. Kulunchakov and J. Mairal, *A generic acceleration framework for stochastic composite optimization*, in Advances in Neural Information Processing Systems, 2019, pp. 12556–12567.
[29] G. Lan, *An optimal method for stochastic composite optimization*, Math. Program., 133 (2012), pp. 365–397.
[30] Y. LeCun and C. Cortes, *MNIST Handwritten Digit Database*, http://yann.lecun.com/exdb/mnist/, 2010.
[31] Q. Lin and L. Xiao, *An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization*, in Proceedings of the International Conference on Machine Learning, 2014, pp. 73–81.
[32] C. Liu and M. Belkin, *Accelerating SGD with momentum for over-parameterized learning*, in Proceedings of the International Conference on Learning Representations, 2020, https://openreview.net/forum?id=r1gixp4FPH.
[33] I. Loshchilov and F. Hutter, *SGDR: Stochastic Gradient Descent with Warm Restarts*, preprint, arXiv:1608.03983, 2016.
[34] B. S. Mordukhovich, *Variational analysis and generalized differentiation* I*: Basic theory*, Grundlehren Math. Wiss. 330, Springer, New York, 2006.
[35] A. S. Nemirovskii and Y. E. Nesterov, *Optimal methods of smooth convex minimization*, USSR Comput. Math. Math. Phys., 25 (1985), pp. 21–30.
[36] Y. Nesterov, *Introductory Lectures on Convex Programming Volume* I*: A Basic Course*, Appl. Optim. 87, Springer, New York, 2004.
[37] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.
[38] Y. E. Nesterov, *A method for solving the convex programming problem with convergence rate o (1/k^ 2)*, Dokl. Akad. Nauk SSSR, 269 (1983), pp. 543–547.
[39] B. O'donoghue and E. Candes, *Adaptive restart for accelerated gradient schemes*, Found. Comput. Math., 15 (2015), pp. 715–732.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., *PyTorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems, 2019, pp. 8024–8035.
[41] B. T. Polyak, *Some methods of speeding up the convergence of iteration methods*, USSR Comput. Math. Math. Phys., 4 (1964), pp. 1–17.
[42] PyTorch, *ImageNet Training in PyTorch*, https://github.com/pytorch/examples/tree/master/imagenet.
[43] J. Renegar, *Efficient First-Order Methods for Linear Programming and Semidefinite Programming*, preprint, arXiv:1409.5832, 2014.
[44] R. T. Rockafellar, *Convex Analysis*, Princeton Landmarks Math. Phys. 28, Princeton University Press, Princeton, NJ, 1970.
[45] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, Grundlehren Math. Wiss. 317, Springer, New York, 2009.
[46] V. Roulet and A. d'Aspremont, *Sharpness, restart and acceleration*, in Advances in Neural Information Processing Systems, 2017, pp. 1119–1129.
[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., *Imagenet large scale visual recognition challenge*, Int. J. Comput. Vis., 115 (2015), pp. 211–252.
[48] W. Su, S. Boyd, and E. Candes, *A differential equation for modeling nesterov's accelerated gradient method: Theory and insights*, in Advances in Neural Information Processing Systems, 2014, pp. 2510–2518.
[49] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, *On the importance of initialization and momentum in deep learning*, in Proceedings of the International Conference on Machine Learning, 2013, pp. 1139–1147.
[50] T. Tieleman and G. Hinton, *Lecture 6.5—RmsProp: Divide the Gradient by a Running Average of Its Recent Magnitude*, COURSERA: Neural Networks for Machine Learning, 2012.
[51] M. D. Zeiler, *ADADELTA: An Adaptive Learning Rate Method*, preprint, arXiv:1212.5701, 2012.
[52] S. Zhang, A. E. Choromanska, and Y. LeCun, *Deep learning with elastic averaging SGD*, in Advances in Neural Information Processing Systems, 2015, pp. 685–693.
[53] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, *Parallelized stochastic gradient descent*, in Advances in Neural Information Processing Systems, 2010, pp. 2595–2603.