AN ADAPTIVE STOCHASTIC TRUST-REGION METHOD WITH VARIANCE REDUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a class of stochastic trust-region methods for unconstrained nonconvex optimization that integrates variance reduction techniques to accelerate convergence. Specifically, we develop two algorithmic variants, TR-SVRG and TR-SpiderBoost, by incorporating the stochastic variance-reduced gradient (SVRG) (Johnson & Zhang, 2013) and SpiderBoost (Wang et al., 2019) estimators, respectively, into a trust-region framework. Unlike classical trust-region approaches, our methods operate exclusively with stochastic gradient information and eliminate the need for function value evaluations. The trust-region radius is adaptively determined using a radius-control parameter and the variance-reduced gradient estimate. Under standard assumptions, we establish convergence in expectation to a first-order stationary point. Notably, our methods attain iteration and sample complexity bounds that match those of SVRG and SpiderBoost combined with stochastic gradient descent (SGD), while accommodating stochastic Hessian approximations that may be correlated with the gradient estimates, a regime beyond the scope of existing variance-reduction theory. Numerical experiments demonstrate that variance reduction techniques and trust-region methods mutually enhance performance, proper Hessian choice benefits algorithmic performance, and careful tuning of batch size and inner-loop parameters is vital for algorithm efficiency.

1 INTRODUCTION

We consider the unconstrained finite-sum optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) := \frac{1}{N}\sum_{i=1}^{N} f_i(\boldsymbol{x}), \tag{1}$$

where each $f_i(\boldsymbol{x})$ denotes an individual component function and $f(\boldsymbol{x})$ represents the aggregate objective. Problems of the form (1) arise ubiquitously in large-scale machine learning, where f_i typically corresponds to the loss associated with a single data sample, and solving (1) is commonly referred to as empirical risk minimization.

When both the sample size N and the dimension d are small to moderate, problem (1) can be efficiently solved using deterministic optimization methods that directly access the full objective function f and its gradients. However, such methods require evaluating the full gradient $\nabla f(\boldsymbol{x})$ or the objective value $f(\boldsymbol{x})$ at every iteration, incurring a computational cost that scales linearly with N. As a result, these approaches become prohibitively expensive when N or d is large, as is typical in modern machine learning applications.

To address this challenge, stochastic optimization methods have become the workhorse for large-scale learning. In these methods, only a single component function or a mini-batch of f_i's is sampled at each iteration, and the gradient or objective value of f is approximated using the sampled subset. This stochastic approximation significantly reduces per-iteration cost and enables scalable optimization. Among these, stochastic gradient descent (SGD) (Robbins & Monro, 1951; Bottou et al., 2018) is arguably the most widely adopted algorithm, owing to its conceptual simplicity and practical effectiveness. A vast literature has since extended SGD through momentum acceleration (Polyak, 1964; Liu et al., 2020), adaptive learning-rate schemes such as AdaGrad (Duchi et al., 2011) and Adam (Kingma, 2014), and refined theoretical analyses (Zhang et al., 2020; Fang et al., 2026a).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Despite their success, these methods rely primarily on first-order (gradient) information. In contrast, it is well known in deterministic optimization that incorporating second-order (Hessian) information can substantially accelerate convergence. In particular, under the Dennis–Moré condition Dennis & Moré (1974), local convergence rates can improve from linear to superlinear or even quadratic (Nocedal & Wright, 2006). Moreover, second-order methods have demonstrated superior empirical performance in a variety of large-scale learning tasks, including language model training (Liu et al., 2024) and computer vision applications (Höllein et al., 2025; Lan et al., 2025; Hsiao et al., 2026). Motivated by these observations, there has been growing interest in developing second-order or quasi-second-order methods for stochastic and large-scale optimization (Byrd et al., 2011; Curtis & Shi, 2022; Berahas et al., 2021; Fang et al., 2024a; Na et al., 2022).

Most existing second-order methods fall into two broad algorithmic frameworks: line-search methods (e.g., Berahas et al., 2021; Na et al., 2022) and trust-region methods (e.g., Curtis & Shi, 2022; Fang et al., 2024a). Line-search methods first compute a search direction—often based on gradient or approximate Newton directions—and then determine a suitable step size to ensure sufficient descent. In contrast, trust-region methods jointly determine the search direction and step size by approximately minimizing a local model of the objective within a prescribed neighborhood, known as the trust region (Conn et al., 2000). Trust-region methods are particularly appealing in nonconvex settings, as they do not require the Hessian approximation to be positive definite and often exhibit enhanced robustness and stability compared to line-search methods (see, e.g., Nocedal & Wright, 2006, Chapter 4).

Another fundamental challenge in stochastic optimization is slow convergence caused by the high variance of stochastic gradient estimates. Such variance can lead to oscillatory behavior and large sample complexity, especially when high-accuracy solutions are required. To mitigate this issue, a rich body of work has developed variance reduction techniques that significantly improve convergence rates, including SVRG (Johnson & Zhang, 2013), SAG/SAGA (Roux et al., 2012; Defazio et al., 2014), SARAH (Nguyen et al., 2017), STORM (Cutkosky & Orabona, 2019), and SPIDER (Fang et al., 2018; Wang et al., 2019), all of which achieve improved sample complexity compared to SGD.

However, most existing variance reduction methods are developed in conjunction with SGD, where second-order information is largely unexplored. Only recently have variance reduction techniques been incorporated into stochastic line-search methods (Berahas et al., 2023). Nevertheless, line-search frameworks typically require modifying the Hessian approximation to enforce positive definiteness, which can distort the underlying second-order information. In contrast, trust-region methods allow for the direct use of possibly indefinite curvature information. This observation naturally raises the following question:

Can one design an efficient stochastic optimization algorithm that combines trust-region methods with variance reduction techniques while retaining the favorable sample complexity of the latter?

In this paper, we provide an affirmative answer to this question. Our main contributions are summarized as follows:

1. We propose a novel trust-region framework that integrates SVRG (Johnson & Zhang, 2013) and SpiderBoost (Wang et al., 2019) estimators, giving rise to two methods: TR-SVRG and TR-SpiderBoost. The proposed methods rely solely on stochastic gradient information and do not require function value evaluations. At each iteration, the trust-region radius is adaptively set using a radius-control parameter and the variance-reduced gradient estimate.
2. Under standard assumptions, we establish that both TR-SVRG and TR-SpiderBoost converge in expectation to first-order stationary points. In terms of the stationarity measure $\mathbb{E}[\|g(\boldsymbol{x})\|^2] \leq \epsilon$, TR-SVRG attains an $\mathcal{O}(\epsilon^{-1})$ iteration complexity and an $\mathcal{O}(N+N^{2/3}\epsilon^{-1})$ sample complexity, while TR-SpiderBoost attains an $\mathcal{O}(\epsilon^{-1})$ iteration complexity and an $\mathcal{O}(N + N^{1/2}\epsilon^{-1})$ sample complexity. These rates match those of SVRG and SpiderBoost combined with SGD (Johnson & Zhang, 2013; Wang et al., 2019). Importantly, our analysis accommodates stochastic Hessian approximations that may be correlated with the gradient estimates—a regime not covered by existing variance reduction theory.
3. Numerical experiments demonstrate that variance reduction techniques accelerates convergence compared with vanilla trust-region methods, trust-region methods and proper choice of Hessian estimates further enhance the performance of variance reduction techniques, and proper tuning of batch size and inner-loop length is critical for algorithmic performance.

Notation. We use $\|\cdot\|$ to denote ℓ_2 norm for vectors and operator norm for matrices. For an integer $S > 0$, we denote $[S] = \{1, 2, \cdots, S\}$ and $[\tilde{S}] = \{0, 1, \cdots, S-1\}$. We use k to index outer loops and s to index inner loops, and denote the iterate at the k-th outer loop and s-th inner loop as $\boldsymbol{x}_{k,s}$. At $\boldsymbol{x}_{k,s}$, we define $g_{k,s} \coloneqq g(\boldsymbol{x}_{k,s}) = \nabla f(\boldsymbol{x}_{k,s})$ as the full gradient, and define $\mathbb{E}_{k,s}[\cdot]$ as the expectation conditional on $\boldsymbol{x}_{k,s}$.

2 Related Work

Stochastic second-order optimization methods. In deterministic optimization, it is well known that the proper incorporation of second-order information can substantially improve convergence rates (Dennis & Moré, 1974; Nocedal & Wright, 2006). Motivated by the rapid growth of large-scale machine learning applications, there has been increasing interest in stochastic second-order optimization algorithms.

The existing literature primarily considers two stochastic frameworks: the *fully stochastic* setup and the *random model* setup. The fully stochastic setup is commonly adopted in the analysis of SGD-type methods, where gradient estimators are assumed to be unbiased with bounded variance. In contrast, the random model setup requires gradient and function value estimates to satisfy adaptive accuracy conditions with a fixed probability; in this setting, gradient estimators may be biased and exhibit unbounded variance. For algorithms under the fully stochastic setup, we refer the reader to (Berahas et al., 2021; Fang et al., 2024a; Curtis & Shi, 2022; Na & Mahoney, 2025; Kuang et al., 2025; Shen et al., 2019; Bellavia et al., 2026). Representative results under the random model setup can be found in (Blanchet et al., 2019; Fang et al., 2024b; 2025; 2026b; Na et al., 2022; 2023; Wang & Yuan, 2022). In this work, we adopt the fully stochastic setup, as algorithms developed in this framework typically enjoy more favorable sample complexity guarantees.

Stochastic trust-region methods. Trust-region methods have been popular and effective in modern machine learning, with a wide applications from large language models (LLM) (Liu et al., 2024), policy optimization (Schulman et al., 2015), and computer vision (Hsiao et al., 2026). Among the existing stochastic trust-region methods, several works are particularly relevant to this paper. Wang & Yuan (2022) studies stochastic trust-region methods whose trust-region radii depend on a probabilistic model gradient. Their analysis also requires random-model accuracy conditions and uses a model-based framework with probabilistic objective value and gradient estimates. In contrast, our methods set the radius directly from the variance-reduced gradient estimator, and do not require function-value evaluations or fixed model-accuracy probabilities. In addition, Shen et al. (2019) develops a sample-efficient stochastic trust-region method for nonconvex optimization using inexact gradient and Hessian estimates, while Xie et al. (2024) analyzes stochastic trust-region methods beyond global Lipschitz smoothness and show how variance reduction improves second-order stationarity complexity. These works are complementary to ours: rather than improving the best known variance-reduction rate, we show that SVRG- and Spider-level complexities can be retained even when stochastic second-order geometry is introduced.

Variance reduction techniques. A rich body of work has developed variance reduction techniques to improve the sample efficiency of stochastic optimization. For μ-strongly convex functions with L-Lipschitz continuous gradients, vanilla SGD requires $\mathcal{O}(1/\epsilon)$ samples to achieve $\mathbb{E}[\|\nabla f(\boldsymbol{x})\|^2] \le \epsilon$, whereas variance-reduced methods such as SVRG, SAG/SAGA, and SARAH achieve linear convergence with sample complexity $\mathcal{O}((N+\kappa)\log(1/\epsilon))$, where $\kappa = L/\mu$ is the condition number. For general convex objectives, SGD has sample complexity $\mathcal{O}(1/\epsilon^2)$, while SVRG attains $\mathcal{O}(N+\sqrt{N}/\epsilon)$, SAGA achieves $\mathcal{O}(N + N/\epsilon)$, and SARAH reaches $\mathcal{O}((N+1/\epsilon)\log(1/\epsilon))$. In the nonconvex finite-sum setting, SVRG achieves a sample complexity of $\mathcal{O}(N+N^{2/3}\epsilon^{-1})$. For online problems, STORM attains $\mathcal{O}(\epsilon^{-3/2})$, and SPIDER unifies several variance reduction schemes with the same $\mathcal{O}(\epsilon^{-3/2})$ complexity online and $\mathcal{O}(N+N^{1/2}\epsilon^{-1})$ in the finite-sum case. However, SPIDER requires a learning rate of order $\mathcal{O}(\epsilon)$, which can be impractically small; SpiderBoost (Wang et al., 2019) addresses this limitation through a modified algorithmic construction while maintaines the sample complexity. In this paper, we adapt SVRG and SpiderBoost into the trust-region framework—choosing SVRG for its broad applicability and SpiderBoost for its superior sample complexity. However, our algorithmic framework can also be incorporated with other variance reduction techniques.

3 ALGORITHM DESIGN

This section describes the design of our stochastic trust-region method, summarized in Algorithm 1. The method adopts the nested-loop structure typical of SVRG-type algorithms. We recast SpiderBoost into an equivalent nested-loop form to unify the presentation of the two algorithmic variants, which differ solely in the construction of the variance-reduced gradient estimate (see (3) and (4) below).

Specifically, at each outer iteration, the full gradient is computed over all samples; at each inner iteration, a mini-batch of samples is drawn to form a stochastic gradient estimate, to which the variance reduction technique is then applied.

Let $\boldsymbol{x}_{k,0}$ denote the iterate at the start of the k-th outer iteration; we refer to it as the *reference iterate*. The full gradient at this reference iterate is computed as

$$\bar{g}_{k,0} = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(\boldsymbol{x}_{k,0}), \tag{2}$$

which is evaluated only once per outer iteration and serves as a control variate for variance reduction. Note that, for a fixed dataset of N samples, this gradient is deterministic. We slightly abuse notation and use $\bar{(\cdot)}$ throughout to streamline the formulas for the variance-reduced gradient estimates.

The algorithm then enters the inner loop of length S. At each inner iteration s, a mini-batch $I_{k,s} \subset [N]$ of size b is sampled.

• **TR-SVRG:** Following the SVRG estimator, we form the variance-reduced gradient estimate as

$$\bar{g}_{k,s} = \frac{1}{b}\sum_{i\in I_{k,s}} \Big(\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0})\Big) + \bar{g}_{k,0}. \tag{3}$$

• **TR-SpiderBoost:** Following the SpiderBoost estimator, we form the variance-reduced gradient estimate for $s \geq 1$ as

$$\bar{g}_{k,s} = \frac{1}{b}\sum_{i\in I_{k,s}} \Big(\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,s-1})\Big) + \bar{g}_{k,s-1}. \tag{4}$$

and as in (2) for $s = 0$.

Based on the $\bar{g}_{k,s}$ obtained following either (3) or (4), we define the trust-region radius as

$$\Delta_{k,s} = \alpha\|\bar{g}_{k,s}\|, \tag{5}$$

where $\alpha > 0$ is a user-specified radius-control parameter.

Remark 3.1. The parameter α plays a role analogous to the learning rate in SGD. As we will show in our global convergence analysis, α is of the same order as the learning rate used in SGD-based algorithms. Our design of (5) is motivated by the goal of preserving the order of magnitude of the resulting step size relative to SGD. Specifically, in SGD, the step size is on the order of $\mathcal{O}(\alpha\|\bar{g}_k\|)$, where $\bar{g}_k$ denotes the gradient estimate at iterate $\boldsymbol{x}_k$. Meanwhile, (5) ensures that after solving the trust-region subproblem (6), the step size also scales as $\mathcal{O}(\alpha\|\bar{g}_{k,s}\|)$. Importantly, although the scalar α may be chosen to be small, the trust-region radius $\Delta_{k,s}$ retains full adaptivity to the magnitude of the variance-reduced gradient and can become large when the stochastic gradient estimate is large.

At each inner iteration, we construct a Hessian approximation $H_{k,s}$ and compute the step $\Delta\boldsymbol{x}_{k,s}$ by solving the trust-region subproblem approximately

$$\min_{\Delta\boldsymbol{x}\in\mathbb{R}^d} \quad m(\Delta\boldsymbol{x}) = \bar{g}_{k,s}^T\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T H_{k,s}\Delta\boldsymbol{x} \qquad \text{s.t. } \|\Delta\boldsymbol{x}\| \leq \Delta_{k,s}. \tag{6}$$

Remark 3.2. We allow $H_{k,s}$ to be stochastic and potentially dependent on the gradient estimate $\bar{g}_{k,s}$. This flexibility enhances the algorithm's applicability to real-world deep learning settings, enabling the use of diagonal preconditioners constructed from gradient estimates—such as those employed in Adam (Kingma, 2014), AdaGrad (Duchi et al., 2011), AdaBound (Luo et al., 2019), and others. Crucially, this setting goes beyond existing variance reduction analyses, which are largely confined to first-order methods and effectively assume $H_{k,s} = I$. Even recent work that incorporates variance reduction into line-search methods (Berahas et al., 2023) continues to require $H_{k,s}$ to be conditionally independent of $\bar{g}_{k,s}$.

Algorithm 1 An Adaptive Stochastic Trust-Region Method with Variance Reduction

Input: Initialization $\boldsymbol{x}_{0,0}$, batch size $b \in [1, N]$, inner-loop length $S > 0$, radius-control parameter $\alpha > 0$.
for $k = 0, 1, 2, \ldots$ **do**
  Set $\boldsymbol{x}_{k,0} = \boldsymbol{x}_{k-1,S}$ (if $k > 0$) or initialize $\boldsymbol{x}_{k,0}$.
  Compute $\bar{g}_{k,0}$ following (2).
  for $s = 0$ to $S - 1$ **do**
    Select a mini-batch $I_{k,s}$ of samples and compute the variance-reduced gradient estimate $\bar{g}_{k,s}$ following (3) for TR-SVRG or following (4) for TR-SpiderBoost.
    Generate the trust-region radius $\Delta_{k,s}$ as in (5).
    Construct $H_{k,s}$ and solve (6) to obtain $\Delta\boldsymbol{x}_{k,s}$.
    Update the iterate as $\boldsymbol{x}_{k,s+1} = \boldsymbol{x}_{k,s} + \Delta\boldsymbol{x}_{k,s}$.
  end for
  Update the iterate as $\boldsymbol{x}_{k+1,0} = \boldsymbol{x}_{k,S}$.
end for

We do not require (6) to be solved exactly. Instead, we impose the standard mild condition that the computed step $\Delta\boldsymbol{x}_{k,s}$ achieves at least as much reduction in the model $m(\cdot)$ as the Cauchy point (see, e.g., Nocedal & Wright, 2006). This requirement is satisfied by the Cauchy step itself and by a single conjugate gradient iteration of the Steihaug–CG solver, while circumventing the expense of high-precision subproblem solves. Under this condition, the following standard result holds (see, e.g., Curtis & Shi, 2022; Fang et al., 2024a).

Lemma 3.3 (Cauchy Reduction)**.** *Let $\Delta\boldsymbol{x}_{k,s}$ be an approximate solution to* (6) *that achieves at least the Cauchy reduction. Then, for all $k \geq 0$ and $s \in [\tilde{S}]$, we have*

$$m(\Delta\boldsymbol{x}_{k,s}) - m(\mathbf{0}) = \bar{g}_{k,s}^T \Delta\boldsymbol{x}_{k,s} + \frac{1}{2}\Delta\boldsymbol{x}_{k,s}^T H_{k,s}\Delta\boldsymbol{x}_{k,s} \leq -\|\bar{g}_{k,s}\|\Delta_{k,s} + \frac{1}{2}\|H_{k,s}\|\Delta_{k,s}^2.$$

4 Algorithm Analysis

In this section, we establish the theoretical properties of the proposed algorithm. We begin by stating the assumptions.

Assumption 4.1. All iterates $\boldsymbol{x}_{k,s}$ lie in an open convex set $\mathcal{X} \subseteq \mathbb{R}^d$. The objective $f : \mathbb{R}^d \to \mathbb{R}$ is bounded below by $f_{\inf}$. For each $i \in [N]$, the component function $f_i : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable with L-Lipschitz continuous gradient. For all $k \geq 0$ and $s \in [\tilde{S}]$, the Hessian approximation $H_{k,s}$ satisfies $\|H_{k,s}\| \leq \kappa_H$ for some constant κ_H.

Assumption 4.2. The mini-batch samples are sampled uniformly with replacement.

Both assumptions are mild and standard for stochastic optimization. Assumption 4.1 implies that the aggregate objective function f is continuously differentiable with L-Lipschitz continuous gradient. Assumption 4.2 implies that, at any iterate $\boldsymbol{x}_{k,s}$, the mini-batch gradient estimate $\frac{1}{b}\sum_{i\in I_{k,s}} \nabla f_i(\boldsymbol{x}_{k,s})$ is conditionally unbiased, i.e., $\mathbb{E}_{k,s}[\frac{1}{b}\sum_{i\in I_{k,s}} \nabla f_i(\boldsymbol{x}_{k,s})] = g(\boldsymbol{x}_{k,s})$.

Next, we define the first-order ϵ-stationary point based on the expected squared norm of the full gradient.

Definition 4.3. A point $\boldsymbol{x}$ is said to be a first-order ϵ-stationary point if the full gradient $g(\boldsymbol{x})$ satisfies

$$\mathbb{E}[\|g(\boldsymbol{x})\|^2] \leq \epsilon.$$

4.1 Main theorems

With the above assumptions, we establish the convergence properties for TR-SVRG and TR-SpiderBoost, respectively.

Theorem 4.4 (Convergence of TR-SVRG)**.** *Under Assumptions 4.1 and 4.2, and the variance-reduced gradient estimate is constructed following* (3)*. We select $\alpha = \frac{\mu_0 b}{2(L+2\kappa_H)N^\gamma} \in (0, 1]$, $b = \mu_1 N^\gamma$ with*

$\mu_0, \mu_1, \gamma \in (0, 1]$, *and* $S \leq \left\lfloor \frac{N^{3\gamma/2}}{\mu_0(b + \frac{\mu_0 L^2 b}{2(L+2\kappa_H)})} \right\rfloor$. *Then for all* $k \geq 0$ *and* $s \in [\tilde{S}]$, *there exist constants* μ_0 *and* $v_0 \in (0, 1)$ *such that*

$$\mathbb{E}\left[\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\|g_{k,s}\|^2\right] \leq \frac{2(L+\kappa_H)}{(K+1)S\mu_0\mu_1 v_0}(\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}).$$

Theorem 4.5 (Convergence of TR-SpiderBoost)**.** *Under Assumptions 4.1 and 4.2, and the variance-reduced gradient estimate is constructed following* (4)*. We select* $\alpha = \frac{1}{2(L+\kappa_H)}$, $b = \sqrt{N}$, *and* $S = \sqrt{N}$. *Then for all* $k \geq 0$ *and* $s \in [\tilde{S}]$, *we have*

$$\mathbb{E}\left[\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\|g_{k,s}\|^2\right] \leq \frac{32(L+2\kappa_H)}{(K+1)S}\left(1 + \frac{L^2}{4(L+2\kappa_H)^2}\right)(\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}).$$

The above results imply several important properties of TR-SVRG and TR-SpiderBoost. First, letting $K \to \infty$, the expected average squared gradient norm converges to zero, which shows that a subsequence of iterates generated by TR-SVRG and TR-SpiderBoost converges in expectation to a first-order stationary point. Second, TR-SVRG and TR-SpiderBoost both enjoy an $\mathcal{O}(1/\epsilon)$ iteration complexity for finding a first-order ϵ-stationary point as in Definition 4.3. Such an iteration complexity matches the SVRG and SpiderBoost methods implemented with SGD.

Based on the parameter selections in Theorems 4.4 and 4.5, we can derive the sample complexities of the two algorithmic variants, as stated below.

Corollary 4.6. *Under the conditions of Theorem 4.4 with* $\gamma = 2/3$*, TR-SVRG achieves a sample complexity of* $\mathcal{O}(N + N^{2/3}\epsilon^{-1})$.

Corollary 4.7. *Under the conditions of Theorem 4.5, TR-SpiderBoost achieves a sample complexity of* $\mathcal{O}(N + N^{1/2}\epsilon^{-1})$.

The above result can be derived using an analysis similar to (Reddi et al. (2016) Corollary 2) and (Wang et al. (2019) Theorem 1). Both sample complexities match the ones of SVRG and SpiderBoost combined with SGD. Importantly, our analysis allows the Hessian approximation to be stochastic and potentially dependent on the gradient estimator. This demonstrates that SVRG- and Spider-level sample complexity can be preserved even in the presence of stochastic second-order geometry, a regime not covered by existing variance reduction results.

5 Numerical Experiments

In this section, we empirically evaluate the proposed framework by addressing five key questions: (i) whether variance reduction techniques improve the performance of trust-region methods, (ii) whether a trust-region framework enhances variance reduction techniques, (iii) how the choice of Hessian estimator $H_{k,s}$ impacts algorithm performance, (iv) how mini-batch size b and inner-loop length S affect optimization, and (v) how the correlation between $H_{k,s}$ and $\bar{g}_{k,s}$ influences the algorithm. We address the first four questions in this section and defer the fifth to Appendix C.5.

We benchmark our methods on two finite-sum learning tasks: a single-hidden-layer MLP on MNIST and a ResNet-18 architecture on CIFAR-10. All algorithms operate under a fixed computational budget equivalent to 60 dataset passes in total gradient evaluations. When evaluating the impact of $H_{k,s}$, we additionally track the number of Hessian-vector products (HVPs). By default, $H_{k,s}$ is configured as a diagonal matrix derived from the second moments of the variance-reduced gradient estimators—a setup we refer to as the *Adam diagonal*. Unless stated otherwise (e.g., Section 5.4), we fix the mini-batch size to $b = 128$ and tune S. Experiment details are provided in Appendix C.

5.1 Variance reduction techniques benefit trust-region methods

To evaluate whether variance reduction techniques improve trust-region methods, we compare TR-SVRG and TR-SpiderBoost against the stochastic trust-region (TR) baseline without variance reduction. The baseline method uses the same Adam diagonal Hessian estimator and mini-batch size

Table 1: Training loss and test accuracy of stochastic trust-region (TR) baseline, TR-SVRG, and TR-SpiderBoost on MNIST MLP and CIFAR-10. Loss and accuracy are reported as mean $\pm$ standard deviation.

Dataset	Method	Training loss	Test accuracy (%)
MNIST MLP	TR	0.0538 ± 0.0037	94.87 ± 0.15
	TR-SVRG	$\mathbf{(7.38 \pm 1.81) \times 10^{-4}}$	$\mathbf{97.36 \pm 0.03}$
	TR-SpiderBoost	0.0438 ± 0.0022	95.72 ± 0.08
CIFAR-10	TR	0.839 ± 0.1012	33.04 ± 1.90
	TR-SVRG	$\mathbf{0.083 \pm 0.0438}$	$\mathbf{78.70 \pm 1.38}$
	TR-SpiderBoost	0.780 ± 0.0761	35.26 ± 1.27

$b = 128$. To ensure a fair comparison, we first tune the radius-control parameter α on the baseline TR method and apply the same parameter choice to both TR-SVRG and TR-SpiderBoost. Table 1 summarizes the resulting training loss and test accuracy.

As shown in the results, incorporating variance reduction consistently achieves lower training loss and higher test accuracy across both tasks. However, the magnitude of improvement depends significantly on the specific variance-reduction scheme. Combining SVRG with the trust-region framework delivers substantial gains, reducing training loss by up to 98.6% on MNIST and 90% on CIFAR-10. TR-SVRG also boosts CIFAR-10 test accuracy by 45.66 percentage points (from 33.04% to 78.70%). In contrast, integrating SpiderBoost yields more modest improvements over the baseline stochastic trust-region method.

5.2 Trust-region methods benefit variance reduction techniques

Now we explore whether trust-region methods can benefit variance reduction techniques. We compare TR-SVRG and TR-SpiderBoost with vanilla SVRG and SpiderBoost, which are based on SGD. To isolate the effects of trust-region methods, we set the learning rate of SGD equal to the radius-control parameter α, while keeping the mini-batch $I_{k,s}$ (and thus batch size b) and inner-loop length S identical within each pair. To ensure a fair comparison, we tune the parameters of SVRG and SpiderBoost first, and then keep the same parameter setting while incorporating the trust-region methods. We report the results in Table 2.

We observe that on MNIST MLP, TR-SVRG reduces the training loss from 0.00635 to 8.64×10^{-4}, an approximately 86% reduction, and improves the test accuracy from 97.99% to 98.28%; TR-SpiderBoost reduces the training loss by approximately 46% while keeping the test accuracy essentially identical. On CIFAR-10, TR-SVRG and TR-SpiderBoost reduce the training loss by 64% and 55%, respectively, while increasing test accuracy by over 10 and 3 percentage points. These observations demonstrate that trust-region mechanisms substantially improve the optimization performance of SVRG and SpiderBoost.

Table 2: Training loss and test accuracy of TR-SVRG, TR-SpiderBoost, SVRG, and SpiderBoost on MNIST MLP and CIFAR-10. Loss and accuracy are reported as mean $\pm$ standard deviation.

Dataset	Method	Training loss	Test accuracy (%)
MNIST MLP	SVRG	0.00635 ± 0.0021	97.99 ± 0.10
	TR-SVRG	$\mathbf{(8.64 \pm 1.62) \times 10^{-4}}$	$\mathbf{98.28 \pm 0.07}$
	SpiderBoost	0.0867 ± 0.0016	97.08 ± 0.05
	TR-SpiderBoost	0.0469 ± 0.0016	96.93 ± 0.08
CIFAR-10	SVRG	0.147 ± 0.058	70.02 ± 2.50
	TR-SVRG	$\mathbf{0.053 \pm 0.01}$	$\mathbf{80.70 \pm 2.56}$
	SpiderBoost	1.91 ± 0.13	30.07 ± 3.50
	TR-SpiderBoost	0.85 ± 0.061	33.12 ± 2.49

Table 3: Training loss, accuracy, and HVP counts of TR-SVRG with different choices of Hessian estimates. Values are reported as mean ± standard deviation. HVP counts are reported in millions.

Dataset	Hessian estimate	Training loss	Accuracy (%)	HVP (M)
MNIST MLP	Identity	0.00635 ± 0.0021	97.99 ± 0.10	0.00
	Adam diagonal	$\mathbf{(8.64 \pm 1.62) \times 10^{-4}}$	$\mathbf{98.28 \pm 0.07}$	0.00
	L-BFGS	0.00341 ± 0.0014	98.03 ± 0.12	0.00
	GGN	0.0416 ± 0.0071	96.98 ± 0.27	3.60
CIFAR-10	Identity	0.147 ± 0.058	70.02 ± 2.50	0.00
	Adam diagonal	0.053 ± 0.01	$\mathbf{80.70 \pm 2.56}$	0.00
	L-BFGS	0.063 ± 0.05	65.59 ± 9.35	0.00
	GGN	$\mathbf{0.039 \pm 0.01}$	72.31 ± 5.58	4.54

5.3 Proper choice of $H_{k,s}$ benefits the performance

A key advantage of trust-region methods is the seamless incorporation of various Hessian estimates. In this experiment, we compare four choices: the Identity matrix, the Adam diagonal matrix, an L-BFGS estimate (Liu & Nocedal, 1989), and a generalized Gauss-Newton (GGN) estimate (Wedderburn, 1974). Across all four Hessian constructions, we adopt the same radius-control parameter α, mini-batch sequence $I_{k,s}$, and inner-loop length S.

We report the results of TR-SVRG in Table 3, while deferring the results of TR-SpiderBoost to Appendix C.4. We observe that the most effective Hessian estimate depends on the task. On MNIST MLP, the Adam diagonal gives the lowest training loss, reducing it from 0.00635 with the Identity matrix to 8.64×10^{-4}, and achieves the highest test accuracy. L-BFGS improves over Identity but remains inferior to the Adam diagonal, while GGN performs worst in both loss and accuracy. On CIFAR-10, GGN offers the lowest training loss, whereas Adam diagonal maintains the highest accuracy. The Identity matrix performs worst in training loss, while L-BFGS yields the lowest accuracy. However, GGN incurs significant computational overhead due to required Hessian-vector products (HVPs).

5.4 Proper choice of b and S benefits the algorithm performance

To explore how b and S affect algorithm performance, we vary the values of b and S while keeping $b \times S$ fixed, and plot the test accuracy. We use the b and S settings in Section 5.2 as the reference. Specifically, for TR-SVRG, we fix $b \times S = 153600$ for MNIST MLP and $b \times S = 128000$ for CIFAR-10, evaluating $b \in \{32, 64, 128, 256, 512\}$ with $S = 153600/b$ and $S = 128000/b$, respectively. For TR-SpiderBoost, we fix $b \times S = 31360$ for MNIST MLP and $b \times S = 7168$ for CIFAR-10, evaluating $b \in \{32, 64, 128, 256, 512\}$ with $S = 153600/b$ and $S = 128000/b$, respectively. The test accuracy curves for TR-SVRG and TR-SpiderBoost are shown in Figure 1.

Optimization dynamics vary notably across tasks and variance-reduction schemes. On the MNIST MLP task, TR-SVRG yields consistently high, robust test accuracies across various b, S configurations, whereas TR-SpiderBoost exhibits marked performance degradation as the mini-batch size b decreases. Conversely, on CIFAR-10, TR-SVRG achieves optimal performance with smaller batch sizes b, with accuracy diminishing as b increases. Meanwhile, TR-SpiderBoost records uniformly low test accuracies across all b and S combinations, suggesting that this particular variance-reduction mechanism struggles to adapt effectively to this benchmark.

6 Limitations

While our proposed trust-region framework achieves optimal sample complexities and strong practical performance, several limitations remain:

Theoretical vs. Empirical Smoothness Guarantees: Our convergence proofs rely on L-Lipschitz gradient continuity for all component functions (Assumption 4.1). However, modern deep architectures (such as MLPs and ResNets utilizing ReLU activations) are non-smooth, meaning that the empirical success on these models extends beyond our formal theoretical guarantees. Extending

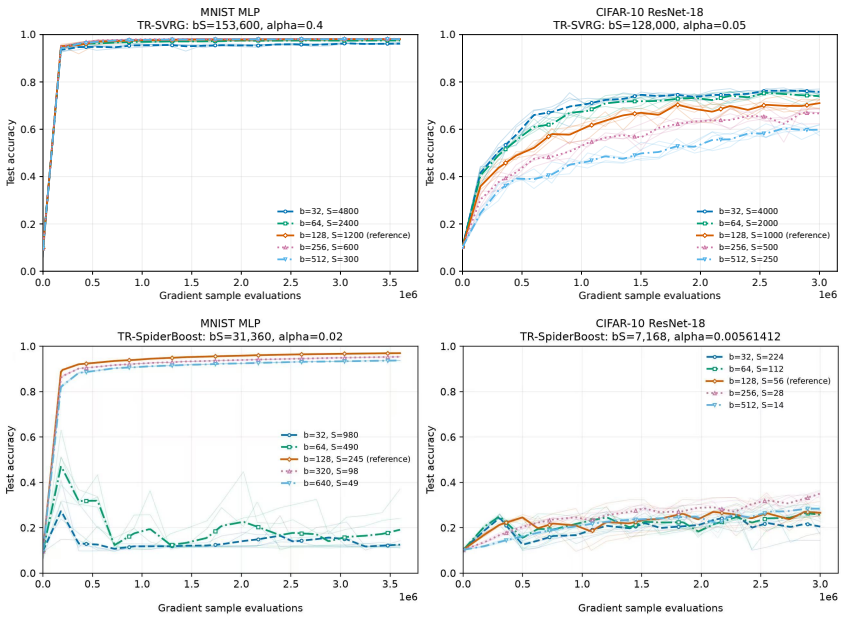

Figure 1: Test accuracy of TR-SVRG and TR-SpiderBoost on MNIST MLP and CIFAR-10 with different combinations of b and S.

the analysis to accommodate weaker smoothness notions—such as generalized or coordinate-wise smoothness—would help close this gap between theory and practice.

Curvature Computation Overhead: Although diagonal Hessian approximations (e.g., Adam diagonal) introduce minimal computational overhead, employing structured or full curvature models like GGN requires additional Hessian-vector products. This overhead can offset the gains in solution quality that richer curvature information provides, particularly on larger architectures where each HVP is expensive. Moreover, our current experiments only explore a few curvature choices; a systematic study of a broader family of Hessian approximations, though outside the scope of this paper, would be valuable.

Sensitivity to b and S: The performance of TR-SVRG and TR-SpiderBoost remains sensitive to the choice of inner-loop length S and batch size b. Such variability brings extra difficulty in parameter tuning and suggests that a purely theory-driven choice of these hyperparameters (as in Theorems 4.4 and 4.5) may not directly translate into empirical performance. Therefore, developing adaptive schemes for b and S could substantially improve the practical usability of the proposed methods.

7 Conclusion

In this paper, we presented TR-SVRG and TR-SpiderBoost, a new class of stochastic trust-region methods for non-convex finite-sum optimization that seamlessly integrate variance reduction. By adaptively controlling the trust-region radius using variance-reduced gradient estimates, our framework eliminates objective function evaluations while accommodating stochastic Hessian approximations that may be correlated with gradient estimates. Theoretically, we established first-order stationary point convergence matching the optimal sample complexities of SVRG ($\mathcal{O}(N + N^{2/3}\epsilon^{-1})$) and SpiderBoost ($\mathcal{O}(N + N^{1/2}\epsilon^{-1})$), closing a gap between the trust-region and variance-reduction literatures that previously relied on restrictive independence assumptions. Empirically, we demonstrated that variance reduction techniques and trust-region methods mutually enhance performance, proper Hessian choice benefits algorithmic performance, and careful tuning of batch size and inner-loop parameters is vital for peak efficiency. We view this as a first step toward variance-reduced trust-region methods more broadly, motivating future work on adaptive curvature selection and weaker smoothness assumptions.

AI USE STATEMENT

Generative AI tools were used to assist with paper polishing and the implementation. We neither use language models in the process of theory formulation nor in theoretical proof. We also do not use AI to collect any cited sources. The authors reviewed all AI-assisted experimental artifacts, tests, and compiled outputs, and take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The appendices document the theoretical derivations, and additional experimental results. The experiment code will be released after the double-blind review period.

REFERENCES

Stefania Bellavia, Benedetta Morini, and Mahsa Yousefi. Fully stochastic trust-region methods with barzilai–borwein steplengths. *Journal of Computational and Applied Mathematics*, 476:117059, 2026.

Albert S. Berahas, Frank E. Curtis, Daniel Robinson, and Baoyu Zhou. Sequential quadratic optimization for nonlinear equality constrained stochastic optimization. *SIAM Journal on Optimization*, 31 (2):1352–1379, jan 2021. doi: 10.1137/20m1354556.

Albert S. Berahas, Jiahao Shi, Zihong Yi, and Baoyu Zhou. Accelerating stochastic sequential quadratic programming for equality constrained optimization using predictive variance reduction. *Computational Optimization and Applications*, 86(1):79–116, April 2023. ISSN 1573-2894. doi: 10.1007/s10589-023-00483-2.

Jose Blanchet, Coralia Cartis, Matt Menickelly, and Katya Scheinberg. Convergence rate analysis of a stochastic trust-region method via supermartingales. *INFORMS journal on optimization*, 1(2): 92–119, 2019.

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

Richard H Byrd, Gillian M Chin, Will Neveitt, and Jorge Nocedal. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, 21(3): 977–995, 2011.

Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust Region Methods*. Society for Industrial and Applied Mathematics, jan 2000. doi: 10.1137/1.9780898719857.

Frank E Curtis and Rui Shi. A fully stochastic second-order trust region method. *Optimization Methods and Software*, 37(3):844–877, 2022.

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32, 2019.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

J. E. Dennis and Jorge J. Moré. A characterization of superlinear convergence and its application to quasi-newton methods. *Mathematics of Computation*, 28(126):549–560, 1974. ISSN 1088-6842. doi: 10.1090/s0025-5718-1974-0343581-1.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. *Advances in neural information processing systems*, 31, 2018.

Yuchen Fang, Sen Na, Michael W. Mahoney, and Mladen Kolar. Fully stochastic trust-region sequential quadratic programming for equality-constrained optimization problems. *SIAM Journal on Optimization*, 34(2):2007–2037, June 2024a. ISSN 1095-7189. doi: 10.1137/22m1537862.

Yuchen Fang, Sen Na, Michael W. Mahoney, and Mladen Kolar. Trust-region sequential quadratic programming for stochastic optimization with random models. *arXiv preprint arXiv:2409.15734*, 2024b.

Yuchen Fang, Javad Lavaei, and Sen Na. High probability complexity bounds of trust-region stochastic sequential quadratic programming with heavy-tailed noise. *arXiv preprint arXiv:2503.19091*, 2025.

Yuchen Fang, James Demmel, and Javad Lavaei. Why is normalization preferred? a worst-case complexity theory for stochastically preconditioned sgd under heavy-tailed noise. *arXiv preprint arXiv:2602.13413*, 2026a.

Yuchen Fang, Jihun Kim, Sen Na, James Demmel, and Javad Lavaei. A trust-region interior-point stochastic sequential quadratic programming method. *arXiv preprint arXiv:2603.10230*, 2026b.

Lukas Höllein, Aljaž Božič, Michael Zollhöfer, and Matthias Nießner. 3dgs-lm: Faster gaussian-splatting optimization with levenberg-marquardt. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26740–26750, 2025.

Roger Hsiao, Yuchen Fang, Xiangru Huang, Ruilong Li, Hesam Rabeti, Zan Gojcic, Javad Lavaei, James Demmel, and Sophia Shao. 3dgs ^{2}-tr: Scalable second-order trust-region method for 3d gaussian splatting. In *Forty-third International Conference on Machine Learning*, 2026.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Wei Kuang, Mihai Anitescu, and Sen Na. Online covariance matrix estimation in sketched newton methods. *arXiv preprint arXiv:2502.07114*, 2025.

Lei Lan, Tianjia Shao, Zixuan Lu, Yu Zhang, Chenfanfu Jiang, and Yin Yang. 3dgs2: Near second-order converging 3d gaussian splatting. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–10, 2025.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Hong Liu, Zhiyuan Li, David Leo Wright Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3xHDeA8Noi.

Yanli Liu, Yuan Gao, and Wotao Yin. An improved analysis of stochastic gradient descent with momentum. *Advances in Neural Information Processing Systems*, 33:18261–18271, 2020.

Liangchen Luo, Yuanhao Xiong, and Yan Liu. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*, 2019.

Sen Na and Michael W. Mahoney. Statistical inference of constrained stochastic optimization via sketched sequential quadratic programming. *Journal of Machine Learning Research*, 26(33):1–75, 2025.

Sen Na, Mihai Anitescu, and Mladen Kolar. An adaptive stochastic sequential quadratic programming with differentiable exact augmented lagrangians. *Mathematical Programming*, 199(1–2):721–791, June 2022. ISSN 1436-4646. doi: 10.1007/s10107-022-01846-z.

Sen Na, Mihai Anitescu, and Mladen Kolar. Inequality constrained stochastic nonlinear optimization via active-set sequential quadratic programming. *Mathematical Programming*, 202(1–2):279–353, March 2023. ISSN 1436-4646. doi: 10.1007/s10107-023-01935-7.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *International conference on machine learning*, pp. 2613–2621. PMLR, 2017.

Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 2006.

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323. PMLR, 2016.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Nicolas Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. *Advances in neural information processing systems*, 25, 2012.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

Zebang Shen, Pan Zhou, Cong Fang, and Alejandro Ribeiro. A stochastic trust region method for non-convex minimization, 2019. URL https://arxiv.org/abs/1903.01540.

Xiaoyu Wang and Ya-xiang Yuan. Stochastic trust-region methods with trust-region radius depending on probabilistic models. *Journal of Computational Mathematics*, 40(2):294–334, 2022. doi: 10.4208/jcm.2012-m2020-0144.

Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost and momentum: Faster variance reduction algorithms. *Advances in Neural Information Processing Systems*, 32, 2019.

Robert WM Wedderburn. Quasi-likelihood functions, generalized linear models, and the gauss—newton method. *Biometrika*, 61(3):439–447, 1974.

Chenghan Xie, Chenxi Li, Chuwen Zhang, Qi Deng, Dongdong Ge, and Yinyu Ye. Trust region methods for nonconvex stochastic optimization beyond lipschitz smoothness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16049–16057, 2024. doi: 10.1609/aaai.v38i14.29537.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33:15383–15393, 2020.

A USEFUL LEMMAS

Lemma A.1. *For any random variable z, we have*

$$\mathbb{E}[\|z - \mathbb{E}[z]\|^2] \leq \mathbb{E}[\|z\|^2].$$

Proof. We have

$$\begin{aligned}\mathbb{E}[\|z - \mathbb{E}[z]\|^2] &= \mathbb{E}[\|z\|^2 - 2z^T\mathbb{E}[z] + \|\mathbb{E}[z]\|^2] \\ &= \mathbb{E}[\|z\|^2] - 2\|\mathbb{E}[z]\|^2 + \|\mathbb{E}[z]\|^2 = \mathbb{E}[\|z\|^2] - \|\mathbb{E}[z]\|^2 \leq \mathbb{E}[\|z\|^2].\end{aligned}$$

We thus finish the proof. □

Lemma A.2. *For independent mean-zero random variables $z_1, z_2, \cdots, z_n$, we have*

$$\mathbb{E}[\|\sum_{i=1}^n z_i\|^2] = \sum_{i=1}^n \mathbb{E}[\|z_i\|^2].$$

Proof. We have

$$\mathbb{E}[\|\sum_{i=1}^n z_i\|^2] = \sum_{i=1}^n \mathbb{E}[\|z_i\|^2] + \sum_{i\neq j} \mathbb{E}[z_i^T z_j].$$

Since z_i, z_j are independent with mean zero, we have $\mathbb{E}[z_i^T z_j] = 0$ for all $i \neq j$. We thus complete the proof. □

B ANALYSIS OF TR-SVRG AND TR-SPIDERBOOST

Lemma B.1 (One-Step Reduction)**.** *Under Assumption 4.1, for any iteration $(k, s) \in \mathbb{N} \times [\tilde{S}]$, we have*

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq -\|\bar{g}_{k,s}\|\Delta_{k,s} + \frac{1}{2}\|H_{k,s}\|\Delta_{k,s}^2 + \|\bar{g}_{k,s} - g_{k,s}\|\Delta_{k,s} + \frac{1}{2}(L + \kappa_H)\Delta_{k,s}^2.$$

With $\Delta_{k,s} = \alpha\|\bar{g}_{k,s}\|$, we further have

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{2}\alpha\|\bar{g}_{k,s}\|^2 + \frac{1}{2}\alpha\|\bar{g}_{k,s} - g_{k,s}\|^2 + \frac{1}{2}(L + 2\kappa_H)\alpha^2\|\bar{g}_{k,s}\|^2.$$

Proof. By the Lipschitz continuity of $g(\boldsymbol{x})$ (cf. Assumption 4.1), we have

$$f(\boldsymbol{x}_{k,s+1}) \leq f(\boldsymbol{x}_{k,s}) + g_{k,s}^T\Delta\boldsymbol{x}_{k,s} + \frac{1}{2}L\|\Delta\boldsymbol{x}_{k,s}\|^2.$$

Therefore,

$$\begin{aligned}f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) - \bar{g}_{k,s}^T\Delta\boldsymbol{x}_{k,s} - \frac{1}{2}\Delta\boldsymbol{x}_{k,s}^T H_{k,s}\Delta\boldsymbol{x}_{k,s} \\ \leq (g_{k,s} - \bar{g}_{k,s})^T\Delta\boldsymbol{x}_{k,s} + \frac{1}{2}L\|\Delta\boldsymbol{x}_{k,s}\|^2 - \frac{1}{2}\Delta\boldsymbol{x}_{k,s}^T H_{k,s}\Delta\boldsymbol{x}_{k,s}.\end{aligned}$$

Using the fact that $\|H_{k,s}\| \leq \kappa_H$, we have

$$\begin{aligned}f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) - \bar{g}_{k,s}^T\Delta\boldsymbol{x}_{k,s} - \frac{1}{2}\Delta\boldsymbol{x}_{k,s}^T H_{k,s}\Delta\boldsymbol{x}_{k,s} &\leq (g_{k,s} - \bar{g}_{k,s})^T\Delta\boldsymbol{x}_{k,s} + \frac{1}{2}(L + \kappa_H)\|\Delta\boldsymbol{x}_{k,s}\|^2 \\ &\leq \|g_{k,s} - \bar{g}_{k,s}\|\|\Delta\boldsymbol{x}_{k,s}\| + \frac{1}{2}(L + \kappa_H)\|\Delta\boldsymbol{x}_{k,s}\|^2 \\ &\leq \|g_{k,s} - \bar{g}_{k,s}\|\|\Delta_{k,s}\| + \frac{1}{2}(L + \kappa_H)\|\Delta_{k,s}\|^2.\end{aligned}$$

The second inequality is by Cauchy-Schwarz inequality and the last inequality is by the fact that $\|\Delta x_{k,s}\| \leq \Delta_{k,s}$. Rearranging the terms, we have

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq \bar{g}_{k,s}^T\Delta\boldsymbol{x}_{k,s} + \frac{1}{2}\Delta\boldsymbol{x}_{k,s}^T H_{k,s}\Delta\boldsymbol{x}_{k,s} + \|g_{k,s} - \bar{g}_{k,s}\|\|\Delta_{k,s}\| + \frac{1}{2}(L + \kappa_H)\|\Delta_{k,s}\|^2.$$

Combining the conclusion in Lemma 3.3 with the above display, we have

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq -\|\bar{g}_{k,s}\|\Delta_{k,s} + \frac{1}{2}\|H_{k,s}\|\Delta_{k,s}^2 + \|\bar{g}_{k,s} - g_{k,s}\|\Delta_{k,s} + \frac{1}{2}(L+\kappa_H)\Delta_{k,s}^2.$$

With $\Delta_{k,s} = \alpha\|\bar{g}_{k,s}\|$ and $\|H_{k,s}\| \leq \kappa_H$, we further have

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq -\alpha\|\bar{g}_{k,s}\|^2 + \alpha\|\bar{g}_{k,s} - g_{k,s}\|\|\bar{g}_{k,s}\| + \frac{1}{2}(L+2\kappa_H)\alpha^2\|\bar{g}_{k,s}\|^2.$$

By Young's inequality (i.e., $ab \leq \frac{1}{2}a^2 + \frac{1}{2}b^2$), we obtain

$$\begin{aligned} f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) &\leq -\alpha\|\bar{g}_{k,s}\|^2 + \frac{1}{2}\alpha\|\bar{g}_{k,s} - g_{k,s}\|^2 + \frac{1}{2}\alpha\|\bar{g}_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\|\bar{g}_{k,s}\|^2 \\ &= -\frac{1}{2}\alpha\|\bar{g}_{k,s}\|^2 + \frac{1}{2}\alpha\|\bar{g}_{k,s} - g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\|\bar{g}_{k,s}\|^2. \end{aligned}$$

We thus finish the proof. □

B.1 Analysis of TR-SVRG

Lemma B.2 (One-Step Expected Reduction)**.** *Under Assumptions 4.1 and 4.2, suppose* $\alpha \leq \frac{1}{2(L+2\kappa_H)}$ *holds, then for any iteration* $(k,s) \in \mathbb{N} \times [\tilde{S}]$*, we have*

$$\mathbb{E}_{k,s}[f(\boldsymbol{x}_{k,s+1})] - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{4}\alpha\|g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\mathbb{E}_{k,s}[\|\bar{g}_{k,s} - g_{k,s}\|^2].$$

Proof. Since $\bar{g}_{k,s}$ is an unbiased estimator of $g_{k,s}$, we have $\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] = \mathbb{E}_{k,s}[\|g_{k,s} - \bar{g}_{k,s}\|^2] + \|g_{k,s}\|^2$. Therefore, after taking expectation conditional on $\boldsymbol{x}_{k,s}$, we have

$$\mathbb{E}_{k,s}[f(\boldsymbol{x}_{k,s+1})] - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{2}\alpha\|g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\|g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\mathbb{E}_{k,s}[\|g_{k,s} - \bar{g}_{k,s}\|^2].$$

Since $\alpha \leq \frac{1}{2(L+2\kappa_H)}$, we have $\frac{1}{2}(L+2\kappa_H)\alpha^2\|g_{k,s}\|^2 \leq \frac{1}{4}\alpha\|g_{k,s}\|^2$. Therefore, we have

$$\mathbb{E}_{k,s}[f(\boldsymbol{x}_{k,s+1})] - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{4}\alpha\|g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\mathbb{E}_{k,s}[\|\bar{g}_{k,s} - g_{k,s}\|^2].$$

We complete the proof. □

Lemma B.3 (Upper Bound of Variance)**.** *Let the variance-reduced gradient estimate* $\bar{g}_{k,s}$ *be computed as in* (3)*. Then for all* $(k,s) \in \mathbb{N} \times [\tilde{S}]$*, we have*

$$\mathbb{E}_{k,s}[\|\bar{g}_{k,s} - g_{k,s}\|^2] \leq \frac{L^2}{b}\|\boldsymbol{x}_{k,s} - \boldsymbol{x}_{k,0}\|^2.$$

Proof. Denote $J_{k,s} = \tilde{g}_{k,s} - \tilde{g}_{k,0}$, then the unbiasedness condition in Assumption 4.2 implies that $\mathbb{E}_{k,s}[J_{k,s}] = g_{k,s} - g_{k,0}$. Recall that $\bar{g}_{k,s} = \tilde{g}_{k,s} - (\tilde{g}_{k,0} - g_{k,0})$, we then have

$$\begin{aligned} \mathbb{E}_{k,s}[\|\bar{g}_{k,s} - g_{k,s}\|^2] &= \mathbb{E}_{k,s}[\|\tilde{g}_{k,s} - (\tilde{g}_{k,0} - g_{k,0}) - g_{k,s}\|^2] \\ &= \mathbb{E}_{k,s}[\|J_{k,s} + g_{k,0} - g_{k,s}\|^2] \\ &= \mathbb{E}_{k,s}[\|J_{k,s} - \mathbb{E}_{k,s}[J_{k,s}]\|^2]. \end{aligned}$$

Note that $J_{k,s} = \frac{1}{b}\sum_{i \in I_{k,s}}(\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0}))$, and $\mathbb{E}_{k,s}[\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0})] = \mathbb{E}_{k,s}[J_{k,s}]$ for all $i \in [N]$, we have

$$\begin{aligned} \mathbb{E}_{k,s}[\|\bar{g}_{k,s} - g_{k,s}\|^2] &= \frac{1}{b^2}\mathbb{E}_{k,s}[\|\sum_{i \in I_{k,s}}(\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0}) - \mathbb{E}_{k,s}[J_{k,s}])\|^2] \\ &= \frac{1}{b^2}\sum_{i \in I_{k,s}}\mathbb{E}_{k,s}[\|\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0}) - \mathbb{E}_{k,s}[J_{k,s}]\|^2] \\ &\leq \frac{1}{b^2}\sum_{i \in I_{k,s}}\mathbb{E}_{k,s}[\|\nabla f_i(\boldsymbol{x}_{k,s}) - \nabla f_i(\boldsymbol{x}_{k,0})\|^2] \\ &\leq \frac{L^2}{b}\|\boldsymbol{x}_{k,s} - \boldsymbol{x}_{k,0}\|^2, \end{aligned}$$

where the second equality is by Lemma A.2, the first inequality is by Lemma A.1, and the last inequality is by the Lipschitz continuity of the component gradients (cf. Assumption 4.1). □

Theorem B.4 (Global Convergence). *Under Assumptions 4.1 and 4.2, we select* $\alpha = \frac{\mu_0 b}{2(L+2\kappa_H)N^\gamma} \in (0,1]$, $b = \mu_1 N^\gamma$ *with* $\mu_0, \mu_1, \gamma \in (0,1]$, *and* $S \leq \left\lfloor \frac{N^{3\gamma/2}}{\mu_0(b+\frac{\mu_0 L^2 b}{2(L+2\kappa_H)})} \right\rfloor$, *then for all* $k \geq 0$ *and* $s \in [\tilde{S}]$, *there exist universal constants* μ_0 *and* $v_0 \in (0,1)$ *such that*

$$\mathbb{E}\left[\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\|g_{k,s}\|^2\right] \leq \frac{2(L+\kappa_H)(\mathbb{E}[f(\boldsymbol{x}_{0,0})]-f_{\inf})}{(K+1)S\mu_0\mu_1 v_0}.$$

Proof. Combining the results of Lemma B.2 and Lemma B.3, we have

$$\mathbb{E}_{k,s}[f(\boldsymbol{x}_{k,s+1})] - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{4}\alpha\|g_{k,s}\|^2 + \frac{L^2(L+2\kappa_H)}{2b}\alpha^2\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2.$$

Next we analyze the term $\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2$. Note that

$$\begin{aligned}
\mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,0}\|^2] &= \mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,s}+\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2] \\
&= \mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,s}\|^2] + 2\mathbb{E}_{k,s}[(\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,s})^T(\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0})] + \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&\leq \mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,s}\|^2] + 2\mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,s}\|]\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\| + \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&\leq \mathbb{E}_{k,s}[\Delta_{k,s}^2] + 2\mathbb{E}_{k,s}[\Delta_{k,s}]\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\| + \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&= \alpha^2\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] + 2\alpha\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|]\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\| + \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&\leq \alpha^2\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] + \frac{\alpha}{z}\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] + \alpha z\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 + \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&= \left(1+\frac{1}{\alpha z}\right)\alpha^2\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] + (1+\alpha z)\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2.
\end{aligned}$$

Recall that $\mathbb{E}_{k,s}[\|\bar{g}_{k,s}\|^2] = \|g_{k,s}\|^2 + \mathbb{E}_{k,s}[\|g_{k,s}-\bar{g}_{k,s}\|^2] \leq \|g_{k,s}\|^2 + L^2/b \cdot \|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2$, we have

$$\mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,0}\|^2] \leq \left(1+\frac{1}{\alpha z}\right)\alpha^2\|g_{k,s}\|^2 + \left(1+\alpha z+\left(\alpha^2+\frac{\alpha}{z}\right)\frac{L^2}{b}\right)\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2.$$

To further the analysis, we define the following Lyapunov function $\Phi_{k,s} = f(\boldsymbol{x}_{k,s}) + \lambda_s\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2$, where

$$\lambda_s = \frac{\alpha^2L^2(L+2\kappa_H)}{2b} + \lambda_{s+1}\left(1+\alpha z+\left(\alpha^2+\frac{\alpha}{z}\right)\frac{L^2}{b}\right)$$

and $\lambda_S = 0$. Note that $\Phi_{k,0} = f(\boldsymbol{x}_{k,0})$. Based on the above definition, we find

$$\begin{aligned}
\mathbb{E}_{k,s}[\Phi_{k,s+1}-\Phi_{k,s}] &= \mathbb{E}_{k,s}[f(\boldsymbol{x}_{k,s+1})-f(\boldsymbol{x}_{k,s})] + \lambda_{s+1}\mathbb{E}_{k,s}[\|\boldsymbol{x}_{k,s+1}-\boldsymbol{x}_{k,0}\|^2] - \lambda_s\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&\leq -\frac{1}{4}\alpha\|g_{k,s}\|^2 + \frac{L^2(L+2\kappa_H)}{2b}\alpha^2\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 - \lambda_s\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&\quad + \lambda_{s+1}\left(1+\frac{1}{\alpha z}\right)\alpha^2\|g_{k,s}\|^2 + \left(1+\alpha z+\left(\alpha^2+\frac{\alpha}{z}\right)\frac{L^2}{b}\right)\|\boldsymbol{x}_{k,s}-\boldsymbol{x}_{k,0}\|^2 \\
&= -\left(\frac{1}{4}\alpha - \lambda_{s+1}\left(1+\frac{1}{\alpha z}\right)\alpha^2\right)\|g_{k,s}\|^2.
\end{aligned}$$

Denoting

$$\Lambda_s = \frac{1}{4}\alpha - \lambda_{s+1}\left(1+\frac{1}{\alpha z}\right)\alpha^2,$$

and setting $\Lambda_{\min} = \min_{s\in[\tilde{S}]}\Lambda_s$, we have $\mathbb{E}_{k,s}[\Phi_{k,s+1}-\Phi_{k,s}] \leq -\Lambda_{\min}\|g_{k,s}\|^2$. Rearranging the terms, we have

$$\|g_{k,s}\|^2 \leq \frac{\mathbb{E}_{k,s}[\Phi_{k,s}-\Phi_{k,s+1}]}{\Lambda_{\min}}.$$

Taking full expectation and summing over $s = 0, \ldots, S-1$, we have

$$\sum_{s=0}^{S-1} \mathbb{E}[\|g_{k,s}\|^2] \leq \frac{\mathbb{E}[\Phi_{k,0}] - \mathbb{E}[\Phi_{k,S}]}{\Lambda_{\min}} = \frac{\mathbb{E}[f(\boldsymbol{x}_{k,0}) - f(\boldsymbol{x}_{k+1,0})]}{\Lambda_{\min}},$$

where we use the facts that $\Phi_{k,0} = f(\boldsymbol{x}_{k,0})$ and $\lambda_S = 0$. Summing over $k = 0, 1, 2, \cdots, K$ for any finite K, we have

$$\sum_{k=0}^{K} \sum_{s=0}^{S-1} \mathbb{E}[\|g_{k,s}\|^2] \leq \frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}}{\Lambda_{\min}}.$$

We then divide both sides by $(K+1)S$ and yield

$$\mathbb{E}\left[\frac{1}{(K+1)S} \sum_{k=0}^{K} \sum_{s=0}^{S-1} \|g_{k,s}\|^2\right] \leq \frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}}{(K+1)S \cdot \Lambda_{\min}}. \tag{7}$$

Next we lower bound $\Lambda_{\min}$ by setting parameters properly. Specifically, we set $\mu_0 \in (0, 1]$ as a parameter to be specified later, $\gamma \in (0, 1]$, and

$$\alpha = \frac{\mu_0 b}{2(L + 2\kappa_H)N^\gamma}, \quad z = \frac{2(L + 2\kappa_H)}{N^{\gamma/2}}, \quad b = N^\gamma, \quad S \leq \left\lfloor \frac{N^{3\gamma/2}}{\mu_0(b + \frac{\mu_0 L^2 b}{2(L+2\kappa_H)})} \right\rfloor.$$

By the recursive structure of λ_s, we have

$$\lambda_0 = \frac{\alpha^2 L^2 (L + 2\kappa_H)}{2b} \cdot \frac{(1+\rho)^S - 1}{\rho},$$

where

$$\begin{aligned}
\rho &= \alpha z + \left(\alpha^2 + \frac{\alpha}{z}\right) \frac{L^2}{b} \\
&= \frac{\mu_0 b}{N^{3\gamma/2}} + \frac{\mu_0^2 L^2 b}{4(L + 2\kappa_H)^2 N^{2\gamma}} + \frac{\mu_0 L^2}{4(L + \kappa_H)^2 N^{\gamma/2}} \\
&= \frac{\mu_0 b}{N^{3\gamma/2}} + \frac{\mu_0^2 L^2 b}{4(L + 2\kappa_H)^2 N^{2\gamma}} + \frac{\mu_0 L^2 b}{4(L + \kappa_H)^2 N^{3\gamma/2}} \\
&\leq \mu_0 N^{-3\gamma/2} \left(b + \frac{L^2 b}{2(L + 2\kappa_H)^2}\right).
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
\lambda_0 &= \frac{\mu_0^2 L^2 b}{8(L + 2\kappa_H)N^{2\gamma}} \cdot \frac{(1+\rho)^S - 1}{\frac{\mu_0 b}{N^{3\gamma/2}} + \frac{\mu_0^2 L^2 b}{4(L+2\kappa_H)^2 N^{2\gamma}} + \frac{\mu_0 L^2 b}{4(L+\kappa_H)^2 N^{3\gamma/2}}} \\
&\leq \frac{\mu_0^2 L^2 b}{8(L + 2\kappa_H)N^{2\gamma}} \cdot \frac{(1+\rho)^S - 1}{\frac{\mu_0 b}{N^{3\gamma/2}}} \\
&= \frac{\mu_0 L^2 (e-1)}{8(L + 2\kappa_H)} N^{-\gamma/2},
\end{aligned}$$

where we also use the upper bound of S and the monotonicity of the function $(1 + 1/l)^l$ and the fact that $(1 + 1/l)^l \to e$ as $l \to \infty$. Plugging in the upper bound of λ_0 to the definition of Λ_s, noticing that λ_s is monotone decreasing with respect to s, we have

$$\begin{aligned}
\Lambda_{\min} &= \frac{1}{4}\alpha - \lambda_0 \left(1 + \frac{1}{\alpha z}\right) \alpha^2 \\
&\geq \frac{1}{4}\alpha - \frac{\alpha \mu_0 L^2 (e-1)}{8(L + 2\kappa_H)} N^{-\gamma/2} \left(\alpha + \frac{1}{z}\right) \\
&= \alpha \left[\frac{1}{4} - \frac{\mu_0 L^2 (e-1)}{8(L + 2\kappa_H)} N^{-\gamma/2} \left(\frac{\mu_0 b}{2(L + 2\kappa_H)N^\gamma} + \frac{N^{\gamma/2}}{2(L + 2\kappa_H)}\right)\right] \\
&\geq \alpha \left[\frac{1}{4} - \frac{\mu_0 L^2 (e-1)(\mu_0 b + 1)}{16(L + 2\kappa_H)^2}\right].
\end{aligned}$$

We denote $v_0 := \frac{1}{4} - \frac{\mu_0 L^2(e-1)(\mu_0 b+1)}{16(L+2\kappa_H)^2}$. When μ_0 is set properly, we can ensure $v_0 > 0$. Therefore, we have

$$\Lambda_{\min} \geq \frac{\mu_0 b v_0}{2(L+\kappa_H)N^\gamma}.$$

Combining the above lower bound of $\Lambda_{\min}$ with (7) and the selection of b leads to the conclusion. □

B.2 Analysis of TR-SpiderBoost

Lemma B.5 (Fang et al. (2018), Lemma 1)**.** *Under Assumptions 4.1 and 4.2, the gradient estimator $\bar{g}_{k,s}$ constructed following* (4) *satisfies for $1 \leq s \leq S-1$,*

$$\mathbb{E}[\|\bar{g}_{k,s} - g_{k,s}\|^2] \leq \frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,s} - \boldsymbol{x}_{k,s-1}\|^2] + \mathbb{E}[\|\bar{g}_{k,s-1} - g_{k,s-1}\|^2].$$

For $s \geq 1$, telescoping Lemma B.5 over $1 \leq i \leq s-1$ leads to

$$\begin{aligned}
\mathbb{E}[\|\bar{g}_{k,s} - g_{k,s}\|^2] &\leq \sum_{i=1}^{s} \frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] + \mathbb{E}[\|\bar{g}_{k,0} - g_{k,0}\|^2] \\
&\leq \sum_{i=1}^{s+1} \frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] + \mathbb{E}[\|\bar{g}_{k,0} - g_{k,0}\|^2] \\
&= \sum_{i=1}^{s+1} \frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2],
\end{aligned}$$

since $\bar{g}_{k,0} - g_{k,0} = 0$.

For $s = 0$, we directly have $\mathbb{E}[\|\bar{g}_{k,0} - g_{k,0}\|^2] = 0$.

Lemma B.6. *Under Assumptions 4.1 and 4.2, if $S = \sqrt{N}, b = \sqrt{N}$, and $\alpha = \frac{1}{2(L+2\kappa_H)}$, then then for all $k \geq 0$ and $s \in [\tilde{S}]$, the TR-SpiderBoost ensures*

$$\mathbb{E}\left[\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\|g_{k,s}\|^2\right] \leq \frac{32(L+2\kappa_H)}{(K+1)S}\left(1 + \frac{L^2}{4(L+2\kappa_H)^2}\right)(\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}).$$

Proof. By Lemma B.1, we have

$$f(\boldsymbol{x}_{k,s+1}) - f(\boldsymbol{x}_{k,s}) \leq -\frac{1}{2}\alpha\|\bar{g}_{k,s}\|^2 + \frac{1}{2}\alpha\|\bar{g}_{k,s} - g_{k,s}\|^2 + \frac{1}{2}(L+2\kappa_H)\alpha^2\|\bar{g}_{k,s}\|^2.$$

Taking expectation on both sides of the above inequality yields that

$$\begin{aligned}
\mathbb{E}[f(\boldsymbol{x}_{k,s+1})] - \mathbb{E}[f(\boldsymbol{x}_{k,s})] &\leq -\frac{\alpha}{2}\mathbb{E}[\|\bar{g}_{k,s}\|^2] + \frac{\alpha}{2}\mathbb{E}[\|\bar{g}_{k,s} - g_{k,s}\|^2] + \frac{(L+2\kappa_H)\alpha^2}{2}\mathbb{E}[\|\bar{g}_{k,s}\|^2] \\
&\leq -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\mathbb{E}[\|\bar{g}_{k,s}\|^2] + \frac{\alpha}{2}\sum_{i=1}^{s+1}\frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] \\
&\leq -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\mathbb{E}[\|\bar{g}_{k,s}\|^2] + \frac{\alpha}{2}\sum_{i=1}^{s+1}\frac{L^2}{b}\mathbb{E}[\Delta_{k,i-1}^2] \\
&= -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\mathbb{E}[\|\bar{g}_{k,s}\|^2] + \frac{\alpha^3 L^2}{2b}\sum_{i=1}^{s+1}\mathbb{E}[\|\bar{g}_{k,i-1}\|^2]. \qquad (8)
\end{aligned}$$

Next, telescoping (8) over s from 0 to s where $s \leq S-1$, we obtain

$$\begin{aligned}
&\mathbb{E}[f(\boldsymbol{x}_{k,s+1})] - \mathbb{E}[f(\boldsymbol{x}_{k,0})] \\
&\qquad \leq -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\sum_{j=0}^{s}\mathbb{E}[\|\bar{g}_{k,j}\|^2] + \frac{\alpha^3 L^2}{2b}\sum_{j=0}^{s}\sum_{i=0}^{j}\mathbb{E}[\|\bar{g}_{k,i}\|^2] \\
&\qquad \leq -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\sum_{j=0}^{s}\mathbb{E}[\|\bar{g}_{k,j}\|^2] + \frac{\alpha^3 L^2}{2b}\sum_{j=0}^{s}\sum_{i=0}^{s}\mathbb{E}[\|\bar{g}_{k,i}\|^2] \\
&\qquad \leq -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2}\right)\sum_{j=0}^{s}\mathbb{E}[\|\bar{g}_{k,j}\|^2] + \frac{\alpha^3 L^2 S}{2b}\sum_{i=0}^{s}\mathbb{E}[\|\bar{g}_{k,i}\|^2] \\
&\qquad = -\left(\frac{\alpha}{2} - \frac{(L+2\kappa_H)\alpha^2}{2} - \frac{\alpha^3 L^2 S}{2b}\right)\sum_{j=0}^{s}\mathbb{E}[\|\bar{g}_{k,j}\|^2] \\
&\qquad = -\beta_1\sum_{j=0}^{s}\mathbb{E}[\|\bar{g}_{k,j}\|^2].
\end{aligned}$$

Therefore, when $s = S-1$, after rearranging the terms, we have

$$\sum_{j=0}^{S-1}\mathbb{E}[\|\bar{g}_{k,j}\|^2] \leq \frac{\mathbb{E}[f(\boldsymbol{x}_{k,0})] - \mathbb{E}[f(\boldsymbol{x}_{k,S})]}{\beta_1} = \frac{\mathbb{E}[f(\boldsymbol{x}_{k,0})] - \mathbb{E}[f(\boldsymbol{x}_{k+1,0})]}{\beta_1}.$$

where the last equality follows from the fact that $\boldsymbol{x}_{k,S} = \boldsymbol{x}_{k+1,0}$.

We then sum over $k = 0, 1, \cdots, K$, and replace the index j by s, and obtain

$$\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|\bar{g}_{k,s}\|^2] \leq \sum_{k=0}^{K}\frac{\mathbb{E}[f(\boldsymbol{x}_{k,0})] - \mathbb{E}[f(\boldsymbol{x}_{k+1,0})]}{\beta_1} \leq \frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}}{\beta_1}.$$

Dividing $(k+1)S$ on both sides yields

$$\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|\bar{g}_{k,s}\|^2] \leq \frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}}{(K+1)S\beta_1}. \tag{9}$$

We next bound $\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|g_{k,s}\|^2]$. Observe that

$$\mathbb{E}[\|g_{k,s}\|^2] = \mathbb{E}[\|g_{k,s} - \bar{g}_{k,s} + \bar{g}_{k,s}\|^2] \leq 2\mathbb{E}[\|g_{k,s} - \bar{g}_{k,s}\|^2] + 2\mathbb{E}[\|\bar{g}_{k,s}\|^2], \tag{10}$$

it suffices to bound $\frac{2}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|g_{k,s} - \bar{g}_{k,s}\|^2]$ and $\frac{2}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|\bar{g}_{k,s}\|^2]$.

For the first term, we have

$$\begin{aligned}
\frac{2}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|g_{k,s} - \bar{g}_{k,s}\|^2] &\leq \frac{2}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\sum_{i=1}^{s+1}\frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] \\
&\leq \frac{2}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\sum_{i=1}^{S}\frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] \\
&= \frac{2}{K+1}\sum_{k=0}^{K}\sum_{i=1}^{S}\frac{L^2}{b}\mathbb{E}[\|\boldsymbol{x}_{k,i} - \boldsymbol{x}_{k,i-1}\|^2] \\
&\leq \frac{2}{K+1}\sum_{k=0}^{K}\sum_{i=1}^{S}\frac{\alpha^2 L^2}{b}\mathbb{E}[\|\bar{g}_{k,i-1}\|^2] \\
&= \frac{2\alpha^2 L^2}{(K+1)b}\sum_{k=0}^{K}\sum_{i=0}^{S-1}\mathbb{E}[\|\bar{g}_{k,i}\|^2] \\
&\leq \frac{2\alpha^2 L^2}{(K+1)b}\frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})] - f_{\inf}}{\beta_1}.
\end{aligned} \tag{11}$$

We combine (9) with (10) and (11), and obtain

$$\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|g_{k,s}\|^2] \le \frac{2\alpha^2L^2S}{(K+1)Sb}\frac{\mathbb{E}[f(\boldsymbol{x}_{0,0})]-f_{\inf}}{\beta_1}+\frac{2(\mathbb{E}[f(\boldsymbol{x}_{0,0})]-f_{\inf})}{(K+1)S\beta_1}$$
$$=\frac{2}{(K+1)S\beta_1}\left(1+\frac{\alpha^2L^2S}{b}\right)(\mathbb{E}[f(\boldsymbol{x}_{0,0})]-f_{\inf}).$$

Based on the parameter setting in Theorem 4.5 that

$$S=\sqrt{N},\quad b=\sqrt{N},\quad \text{and } \alpha=\frac{1}{2(L+2\kappa_H)},$$

we obtain

$$\beta_1=\frac{\alpha}{2}-\frac{\alpha^2(L+2\kappa_H)}{2}-\frac{\alpha^3L^2S}{2b}\ge\frac{1}{16(L+2\kappa_H)}>0.$$

Therefore, TR-SpiderBoost satisfies

$$\frac{1}{(K+1)S}\sum_{k=0}^{K}\sum_{s=0}^{S-1}\mathbb{E}[\|g_{k,s}\|^2]\le\frac{32(L+2\kappa_H)}{(K+1)S}\left(1+\frac{L^2}{4(L+2\kappa_H)^2}\right)(\mathbb{E}[f(\boldsymbol{x}_{0,0})]-f_{\inf}).$$

This completes the proof. □

C Experiment Details

C.1 Data and evaluation

We evaluate our approach on two standard image classification tasks: MNIST and CIFAR-10. For MNIST, we train a Multilayer Perceptron (MLP) with a hidden width of 256 and ReLU activations on the full 60,000-image training set. For CIFAR-10, we train a CIFAR-stem ResNet-18 with GroupNorm on the 50,000-image training set. Both tasks aim to minimize the mean multiclass cross-entropy loss without explicit regularization.

C.2 Parameter tuning

For every method-task combination, we evaluate 30 hyperparameter configurations by testing ten different values for either the learning rate or the radius-control parameter. We also tune the generalized Gauss–Newton (GGN) damping factor across $\{0.001, 0.1, 1\}$ and the conjugate gradient (CG) iteration caps across $\{5, 10\}$. For L-BFGS, we search over initial scales of $\{0.1, 1, 10\}$ and update frequencies of $\{10, 20\}$.

Table 4: Selected parameters for Section 5.1: batch size b, inner-loop length S, and learning rate (for SVRG/SpiderBoost) or radius-related parameter α (for TR-SVRG/TR-SpiderBoost).

Task	Method	b	S	Rate or α
MNIST MLP	TR	128	–	0.2
MNIST MLP	TR-SVRG	128	1200	0.2
MNIST MLP	TR-SpiderBoost	128	245	0.2
CIFAR-10	TR	128	–	0.06
CIFAR-10	TR-SVRG	128	800	0.06
CIFAR-10	TR-SpiderBoost	128	300	0.06

C.3 Details on $H_{k,s}$ construction and solving trust-region subproblems

When using a scaled identity matrix $H = I$ as the Hessian approximation, the exact trust-region step is $-\min\{\alpha, 1\}\bar{g}$. For a positive diagonal Hessian, the solver computes the step as $p_i = -\bar{g}_i/(h_i+\lambda)$,

Table 5: Selected parameters for Section 5.2: batch size b, inner-loop length S, and learning rate (for SVRG/SpiderBoost) or radius-related parameter α (for TR-SVRG/TR-SpiderBoost).

Task	Method	b	S	Rate or α
MNIST MLP	SVRG	128	1200	0.4
MNIST MLP	SpiderBoost	128	245	0.02
MNIST MLP	TR-SVRG	128	1200	0.4
MNIST MLP	TR-SpiderBoost	128	245	0.02
CIFAR-10	SVRG	128	1000	0.05
CIFAR-10	SpiderBoost	128	245	0.02
CIFAR-10	TR-SVRG	128	1000	0.05
CIFAR-10	TR-SpiderBoost	128	245	0.02

where $\lambda \geq 0$ is chosen to satisfy both the radius constraint and complementary slackness up to numerical tolerance.

For both L-BFGS and GGN models, we solve the trust-region subproblem using Steihaug conjugate gradients (CG). Both the diagonal and CG solvers compare their proposed step against the Cauchy step, adopting the one that yields a lower quadratic model value.

Next, we introduce the construction of $H_{k,s}$. To facilitate the introduction, we use a single index t rather than the nested indices $\{k, s\}$.

To construct the Adam diagonal matrix, let z_t represent the gradient signal at step $t \geq 1$. Starting from $v_0 = 0$, we update the diagonal elements as follows:

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) z_t^{\odot 2}, \qquad u_t = \left(\frac{v_t}{1-\beta_2^t} + \epsilon \mathbf{1} \right)^{\odot r},$$

$$h_{t,j} = \min\left\{ 100,\ c \left(\frac{u_{t,j}}{\mathrm{rms}(u_t)} + \rho \right) \right\}, \qquad H_t = \mathrm{diag}(h_t).$$

Here, all exponents are applied element-wise, and the root-mean-square over a d-dimensional parameter space is defined as $\mathrm{rms}(u) = \|u\|/\sqrt{d}$. The trust-region step relies directly on $\bar{g}_t$ without Adam's typical first-moment momentum, as the compound of momentum and variance reduction techniques causes instability in training.

For the L-BFGS Hessian operator, we compute gradient differences using the exact same mini-batch for successive matrix-update iterations. If a pair fails the positive-curvature condition, we discard it; subsequent valid pairs are stabilized with Powell damping if necessary. Both gradient evaluations count toward the overall sample budget.

The GGN matrix is constructed as:

$$H_t = \frac{1}{|\mathcal{C}_t|} \sum_{i \in \mathcal{C}_t} J_i^\top \left(\mathrm{diag}(q_i) - q_i q_i^\top \right) J_i + \delta I,$$

where J_i is the logit Jacobian and q_i is the softmax probability vector. The curvature batch $\mathcal{C}_t$ matches the estimator batch between full gradient computation; upon a full gradient computation, a completely new batch is drawn. Hessian-vector product (HVP) counts reflect the actual number of samples processed by GGN products, including any Cauchy safeguards. We do not count algebraic products involving the Identity, diagonal, or L-BFGS matrices against this sample-based HVP budget.

C.4 Proper choice of $H_{k,s}$ benefits the performance

We continue the investigation in Section 5.3 on the choice of $H_{k,s}$. We report the training loss, test accuracy, and HVP counts for TR-SpiderBoost in Table 6. We observe that in MNIST MLP dataset, L-BFGS yields the lowest training loss and the highest accuracy at the same time, while on CIFAR-10 dataset, GGN generates the lowest training loss, but the highest accuracy is obtained by using L-BFGS.

Table 6: Training loss, test accuracy, and HVP counts of TR-SpiderBoost with different choices of Hessian estimates. Values are reported as mean ± standard deviation. HVP counts are reported in millions.

Model matrix / control	Training loss	Accuracy (%)	HVP (M)
MNIST MLP			
Identity	0.0528 ± 0.018	95.12 ± 0.05	0.00
Adam diagonal	0.0469 ± 0.0016	96.93 ± 0.08	0.00
L-BFGS	$\mathbf{0.0189 \pm 0.009}$	$\mathbf{97.38 \pm 0.07}$	0.00
GGN	0.0381 ± 0.0120	95.80 ± 0.18	4.80
CIFAR-10			
Identity	0.699 ± 0.141	38.38 ± 1.40	0.00
Adam diagonal	0.85 ± 0.061	33.12 ± 2.49	0.00
L-BFGS	0.675 ± 0.15	$\mathbf{46.59 \pm 8.78}$	0.00
GGN	$\mathbf{0.47 \pm 0.18}$	42.31 ± 4.99	5.38

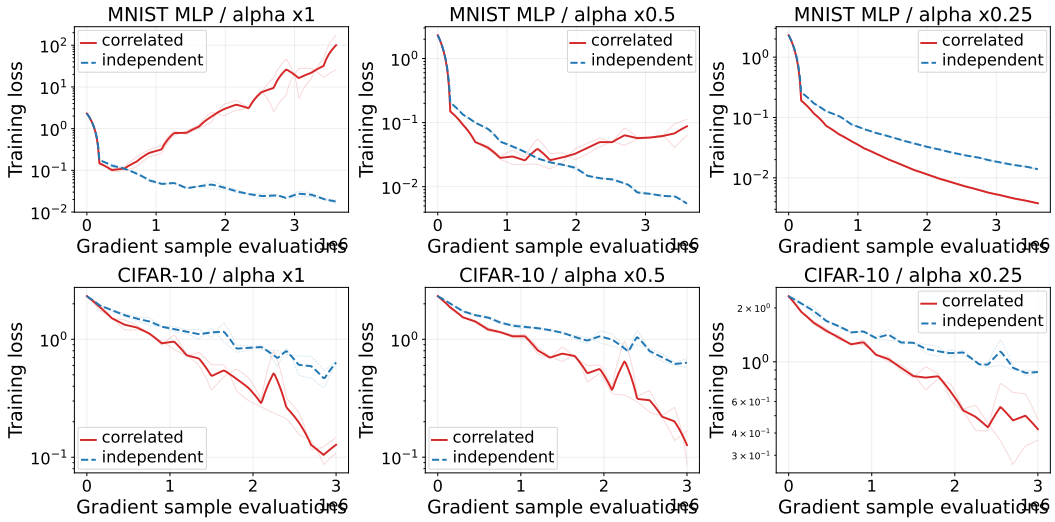

Figure 2: The impact of using correlated versus independent batches to construct the Adam diagonal matrix. Each row corresponds to a task and each column to a trust-region radius equal to α, $\alpha/2$, or $\alpha/4$. Training loss is plotted against gradient sample evaluations.

C.5 Dependence between Hessian estimates and gradient estimates

Our theoretical framework permits the Hessian estimate to be stochastic and potentially dependent on the variance-reduced gradient estimates. To see if this dependence impacts practical performance, we constructed the Adam diagonal Hessian using two different sampling strategies: a *correlated-batch* approach (which uses the exact same mini-batch for both the variance-reduced gradient and the Hessian estimate) and an *independent-batch* approach (which draws a fresh mini-batch for the Hessian estimate).

As shown in Figure 2, the impact of this choice depends heavily on both the dataset and the radius-control parameter α. On the MNIST MLP task, the correlated-batch approach becomes highly unstable at the largest radius, causing the training loss to spike after an initial decrease. Shrinking the radius α mitigates this issue: at $\alpha/2$, the two methods perform similarly for a longer period, and at $\alpha/4$, the correlated-batch method actually outperforms the independent-batch approach. This suggests that coupling the gradient and Hessian estimators is not inherently detrimental, but it can severely amplify instability when the trust-region radius is too large.

The CIFAR-10 results are even more favorable for the correlated-batch method. Across all three radius settings (Figure 2), the correlated-batch variant consistently achieves a lower training loss than the independent-batch variant, with the performance gap widening at the largest radius.

Our findings suggest that independent sampling is a valuable stabilizing tool when stochastic model matrices are tightly coupled to the gradient estimator—particularly for MLPs trained with large trust-region radii. However, at smaller radii, correlated batches is highly competitive and saves significant computational overhead. The optimal strategy ultimately depends on the interplay between the Hessian matrix, the trust-region radius, and the specific optimization landscape.