A Trainable Optimizer

Ruiqi Wang **Diego Klabjan**

IEMS, Northwestern University
2145 Sheridan Road, Evanston, IL 60208, USA

Abstract

The concept of learning to optimize involves utilizing a trainable optimization strategy rather than relying on manually defined full gradient estimations such as ADAM. We present a framework that jointly trains the full gradient estimator and the trainable weights of the model. Specifically, we prove that pseudo-linear TO (Trainable Optimizer), a linear approximation of the full gradient, matches SGD's convergence rate while effectively reducing variance. Pseudo-linear TO incurs negligible computational overhead, requiring only minimal additional tensor multiplications. To further improve computational efficiency, we introduce two simplified variants of Pseudo-linear TO. Experiments demonstrate that TO methods converge faster than benchmark algorithms (e.g., ADAM) in both strongly convex and non-convex settings, and fine tuning of an LLM.

1 Introduction

Gradient-based stochastic optimization methods are fundamental to solving many machine learning problems. Given a set of loss functions $\{f_n(w)\}_{n=1}^N$ where $w \in \mathbb{R}^d$, the full loss is defined as $F(w) = \frac{1}{N}\sum_{i=1}^N f_n(w)$. For a subset $\mathcal{B} \subset \{1,\dots,N\}$, the mini-batch loss is given by $F^{\mathcal{B}}(w) := \frac{1}{|\mathcal{B}|}\sum_{n\in\mathcal{B}} f_n(w)$. Let w^* be the global minimum of $F(w)$. Popular approaches, such as Stochastic Gradient Descent (SGD), RMSProp (Hinton, Srivastava, and Swersky 2012), Adagrad (Duchi, Hazan, and Singer 2011), and ADAM (Kingma and Ba 2015), can be unified under a general framework. In such a framework, the update direction depends on past iterations $w_1,\dots,w_t$, past stochastic gradients $g_1,\dots,g_t$ and the optimizer's variables θ_t, following the update rule

$$w_{t+1} = w_t - \gamma_t \widehat{G}_t,$$

where γ_t is the learning rate and $\widehat{G}_t$ denotes the update direction.

We list the examples of functions $\widehat{G}_t$ in Table 1.

Algorithm	$\widehat{G}_t$
SGD	g_t
Momentum	$(1-\beta_1)\sum_{k=1}^t \beta_1^{t-k} g_k$
Adagrad	$\frac{g_t}{\sqrt{\sum_{k=1}^t g_k \odot g_k / t + \epsilon}}$
RMSProp	$\frac{g_t}{\sqrt{(1-\beta)\sum_{k=1}^t \beta^{t-k} g_k \odot g_k + \epsilon}}$
ADAM	$\frac{(1-\beta_1)\sum_{j=1}^t \beta_1^{t-j} g_j}{\sqrt{(1-\beta_2)\sum_{k=1}^t \beta_2^{t-k} g_k \odot g_k + \epsilon}}$

Table 1: Examples of $\widehat{G}_t$

Learning to Optimize (L2O) automates the development of optimization strategies by replacing traditional, manually designed $\widehat{G}_t$ functions with learned counterparts (Chen et al. 2022). The core premise of L2O is to employ machine learning models, typically neural networks, to discover task-specific optimization policies that outperform conventional handcrafted rules. Unlike classical optimizers with fixed update rules and hyperparameters, L2O models are meta-trained across diverse problems to generalize to unseen tasks, making them particularly effective for complex, high-dimensional spaces such as neural architecture search and scientific computing. L2O models contain an offline procedure, where the learnable optimizer is trained on a set of similar tasks, and an online procedure, where the trained optimizer optimizes a new unseen task from the same task distribution.

Generally, an L2O approach parameterizes the update direction $\widehat{G}_t$ as a function of past values $w_1,\dots,w_t$ of trainable model weights and past gradients $g_1,\dots,g_t$ with optimizer variable θ being fixed. For a given task, θ is updated at the end of the optimization procedure, based on the performance of the optimizer on that task. With a learned θ, an unseen task is further handled with the following update rule for the model weights

$$w_{t+1} = w_t - \gamma_t \widehat{G}_t(w_1,\dots,w_t; g_1,\dots,g_t,\theta).$$

Practically, for a given new task, similar tasks are not necessarily available, which makes the offline procedure of L2O not possible. We propose a departure from standard L2O methods—which train $\widehat{G}_t$ across broad problem distributions—by instead co-training the optimizer and weights of the model for a single task. Our new approach replaces the fixed multidimensional optimizer variable $\theta \in \mathbb{R}^{\bar{d}}$ with θ_t, which is updated along with w_t simultaneously at each iteration. The trainable weights of the model are updated as

$$w_{t+1} = w_t - \gamma_t \widehat{G}_t(w_1, \ldots, w_t; g_1, \ldots, g_t; \theta_t). \tag{1}$$

It is easy to see that (1) captures all optimizers listed in Table 1. Optimizer variable θ_t is updated based on the gradient of an approximation loss function

$$l(\theta_t; w_1, \ldots, w_t; g_1, \ldots, g_t),$$

following the update rule:

$$\theta_{t+1} = \theta_t - \beta_t \nabla_{\theta_t} l(\theta_t; w_1, \ldots, w_t; g_1, \ldots, g_t). \tag{2}$$

Despite L2O's empirical success in accelerating convergence, its theoretical guarantees remain unstudied. We establish that for specific parameterizations of $\widehat{G}_t$ and setting of l to be an L_2 error with the ground truth being the gradient of F,

$$\begin{aligned} &l(\theta_t; w_1, \ldots, w_t; g_1 \ldots, g_t) \\ &= \frac{1}{2} \left\| g_t - \widehat{G}_t(w_1, \ldots, w_t; g_1, \ldots, g_t, \theta_t) \right\|_2^2, \end{aligned}$$

both the gradient approximation error $\left\| \widehat{G}_t - \nabla F(w_t) \right\|$ and the optimality gap $\|w_t - w^*\|$ converge to zero simultaneously.

The relationship between convergence and variance reduction in gradient-based methods has been well-studied. (Johnson and Zhang 2013a) proposed SVRG, achieving variance reduction through snapshot gradients and bias correction via full-batch gradients, guaranteeing exponential convergence for strongly convex losses. (Zaheer et al. 2018) proved ADAM's convergence under growing mini-batch sizes, while (Qian and Klabjan 2020) and (Wang and Klabjan 2025) further established a crucial link between variance reduction and convergence. A fundamental criterion for analyzing stochastic optimization methods involves verifying whether the estimation variance vanishes as $t \to \infty$. Our framework inherently satisfies this property, constituting a trainable variance reduction algorithm.

In this work, we focus on pseudo-linear approximations, where $\widehat{G}_t$ is formulated as a linear function with regards to w_t, i.e., $\widehat{G}_t = A_t w_t + b_t$ with optimizer variables $A_t \in \mathbb{R}^{d \times d}$, $b_t \in \mathbb{R}^d$, and $\theta_t = (A_t, b_t)$. We call it pseudo since A_t and b_t are functions of $w_1, \ldots, w_{t-1}$ and $g_1, \ldots, g_{t-1}$. This pseudo-linear parameterization is motivated by two key considerations. First, it draws inspiration from Taylor expansion principles where lower-order terms often provide effective approximations. Second, it enables efficient computation, as the gradient of the approximation loss can be calculated explicitly by simple tensor multiplications, which requires negligible additional computational time.

Our theoretical analysis establishes that for strongly convex loss functions, the parameters w_t under the pseudo-linearly approximated gradients converge to the optimal point w^* at the rate $\mathbb{E}\left[\|w_t - w^*\|_2^2\right] \leq \mathcal{O}(1/t)$. Furthermore, we show that the approximation variance $\mathbb{E}\left[\left\|\widehat{G}_t - \nabla F(w_t)\right\|_2^2\right]$ also converges to zero at the same $\mathcal{O}(1/t)$ rate. Comparing with the variance of SGD and ADAM, which does not converge to zero as $t \to \infty$, these results confirm that our trained optimizer functions as an effective variance reduction algorithm, which we name Trainable Optimizer (TO).

To address computational concerns, we propose two simplified variants of pseudo-linear TO:

- Diagonal TO restricts A_t to diagonal matrices,
- RankOne TO uses rank-one matrices for A_t.

The two variants significantly reduce the number of variables in the optimizer, which are more suitable with the practical cases with limited memory. Notably, we prove that momentum-based SGD emerges as a special case of pseudo-linear TO when appropriate learning rates and initializations A_0 and b_0 are selected.

Our comprehensive experimental evaluation compares three TO variants (Pseudo-linear (full), Diagonal, RankOne) across diverse machine learning tasks spanning three categories:

- Strongly Convex losses (e.g., L_2-regularized logistic regression),
- Convex but not strongly convex losses (e.g., logistic regression without regularization),
- Non-convex losses (e.g., ResNet classification, Llama fine-tuning).

The results demonstrate that TO significantly outperforms ADAM in both strongly convex and complex non-convex settings, while matching ADAM's performance on standard convex tasks.

Our main contributions are as follows.

- We propose a new framework that updates the optimizer variables simultaneously with trainable weights of the model, based on (1) and (2).
- We show the $\mathcal{O}(1/t)$ convergence of Pseudo-linear TO for strongly convex losses.
- We propose two variants of pseudo-linear TO: Diagonal TO and RankOne TO. They reduce the memory requirement of pseudo-linear TO and are better suitable for large models.
- We show by means of experiments that TO outperforms ADAM on strongly convex problems and complex non-convex problems.

In Section 2, we review related work on trainable optimizers and variance-reduction optimization methods. Section 3 presents the TO framework with linear approximation and demonstrates the convergence of model parameters to the optimal point while reducing the approximation variance to zero. This section also introduces two simplified versions—Diagonal TO and RankOne TO. In Section 4, we present numerical experiments illustrating the improved convergence rates of the proposed methods.

2 Related Work

2.1 Learning to Optimize

The L2O paradigm represents a significant shift from traditional optimization methods, replacing theoretically-derived update rules with data-driven approaches that leverage machine learning. Pioneering work by (Andrychowicz et al. 2016) demonstrated that recurrent neural networks (RNNs) can learn effective optimization strategies that outperform conventional methods like SGD. Subsequent advances incorporated reinforcement learning (RL) frameworks (Li and Malik 2017; Bello et al. 2017), enabling the discovery of specialized optimization algorithms for diverse applications ranging from neural architecture search to black-box optimization. Our work extends these foundations while introducing several key innovations. First, unlike standard L2O approaches that solely optimize the target model's parameters, our framework simultaneously optimizes both the model parameters and the optimization strategy itself. This co-optimization enables dynamic adaptation to problem-specific characteristics. Second, while most L2O methods lack theoretical guarantees, we provide rigorous convergence proofs showing simultaneous convergence of both trainable weights of the model and the gradient approximation error. By employing a carefully designed linear approximation, our approach achieves both computational efficiency and theoretical soundness.

2.2 Variance Reduction

Variance reduction has emerged as a fundamental technique for improving stability and the convergence rate of stochastic optimization methods. One particularly influential method is Stochastic Variance Reduced Gradient (SVRG), originally introduced by (Johnson and Zhang 2013b). SVRG operates by maintaining a snapshot model to effectively correct the noisy estimates obtained from stochastic gradients, thereby making the entire optimization process both faster and more numerically stable. Another important approach is SAGA, proposed by (Defazio, Bach, and Lacoste-Julien 2014), which builds upon SVRG's foundation while streamlining the process through an efficient history of gradients, making it more practical for large-scale implementations. Recent work includes the introduction of Variance Reduced Adam (VRADAM) by (Wang and Klabjan 2025), which combines variance reduction techniques with ADAM. This hybrid approach successfully resolves ADAM's well-known divergence issues while carefully preserving its advantages in training speed and the convergence rate for challenging deep learning tasks. While other recent works, such as (Huang, Li, and Huang 2022) and (Cutkosky and Orabona 2019), have demonstrated various approaches combining variance reduction techniques with modern optimization algorithms, our work makes distinct contributions by specifically focusing on utilizing the idea of trainable optimizers for variance reduction. We provide theoretical and empirical evidence showing that the gradient variance can be systematically reduced through the optimization process itself as the trainable optimizer variables are progressively refined during training.

3 Trainable Optimizer with Pseudo-linear Approximation

3.1 The Pseudo-linear TO Algorithm

In this section, we present our approach for approximating the full gradient with a trainable pseudo-linear function. Consider a stationary point w^* of F, where we assume that F is second-order differentiable at w^*. Drawing inspiration from the Taylor expansion of $\nabla F(w)$ centered at w^*, we have

$$\begin{aligned}\nabla F(w) &\approx \nabla F(w^*) + \nabla^2 F(w^*)(w - w^*)\\ &= \nabla^2 F(w^*)(w - w^*),\end{aligned}$$

which yields a linear function of w. We formulate the approximated gradient as $\widehat{G}_t = A_t w_t + b_t$, where A_t and b_t represent the variables of the optimizer. The corresponding approximation loss is defined by

$$l((A_t, b_t); w_t; g_t) := \frac{1}{2}\left\|g_t - A_t w_t - b_t\right\|_2^2,$$

with gradients computed as

$$\begin{aligned}\nabla_{A_t} l((A_t, b_t); w_t; g_t) &= -(g_t - A_t w_t - b_t) w_t^\top\\ \nabla_{b_t} l((A_t, b_t); w_t; g_t) &= -(g_t - A_t w_t - b_t).\end{aligned}$$

We introduced the Trainable Optimizer (TO) with pseudo-linear approximation in Algorithm 1. Notably, lines 5 and 6 of the algorithm implement updates to the parameters A and b through the gradient descent on the loss function $l((A_t, b_t); w_t; g_t)$.

The pseudo-linear TO framework admits an important interpretation as a generalized momentum-based SGD algorithm. This connection becomes exact under specific initialization conditions: when we set $A_0 = 0$, $b_0 = 0$, $\alpha_t = 0$ and maintain a constant $\beta_t = \beta$ for some $0 < \beta < 1$, the update rules in Algorithm 1 are simplified to

$$A_t = 0, b_t = \beta g_t + (1 - \beta) b_{t-1}, \widehat{G}_t = b_t,$$

which recovers precisely the standard momentum-based SGD formulation. The initialization of parameters A_0 and b_0 presents practical challenges. However, based on our theoretical analysis and empirical observations, we strongly recommend initializing both to zero, as this initialization strategy provides effective practical performance during the initial training phase.

3.2 Analyses

In this section we theoretically prove the convergence of Pseudo-linear TO under strongly convex loss. Theoretical analyses of gradient-based algorithms often rely on assumptions of bounded iterates and gradients. To this end, we modify Algorithm 1 by changing line 8 to

$$w_{t+1} \leftarrow \Pi_{\mathcal{F}}(w_t - \gamma_t \widehat{G}_t).$$

The projection operator $\Pi_{\mathcal{F}}(w)$, which maps vectors to the bounded feasible set $\mathcal{F}$, is mathematically defined as

$$\Pi_{\mathcal{F}}(w) := \arg\min_{w' \in \mathcal{F}} \|w - w'\|_2.$$

Algorithm 1: Pseudo-linear TO

```
Input: Learning rates $\alpha_t$, $\beta_t$, $\gamma_t$.
1: Initialize $w_1$, $A_0$ and $b_0$.
2: for $t = 1, \dots, T$ do
3:     Sample $\mathcal{B}_t \subset \{1, \dots, N\}$ such that $|\mathcal{B}_t| = b$
4:     $g_t \leftarrow \nabla F^{\mathcal{B}_t}(w_t)$
5:     $A_t \leftarrow A_{t-1} + \alpha_t(g_t - A_{t-1}w_t - b_{t-1})w_t^\top$
6:     $b_t \leftarrow b_{t-1} + \beta_t(g_t - A_{t-1}w_t - b_{t-1})$
7:     $\widehat{G}_t \leftarrow A_t w_t + b_t$
8:     $w_{t+1} \leftarrow w_t - \gamma_t \widehat{G}_t$
9: end for
```

Let $D_w = \sup_{w \in \mathcal{F}} \|w\|_2 < \infty$. In our context, we include additional optimizer variables A_t and b_t. In this section, we first show that under our arrangement of the algorithm, the trajectory of gradients are uniformly bounded. Then, we show that A_t and b_t are also bounded. All proofs are in the appendix.

For the theoretical analysis, we make the following assumptions. We assume that $\mathcal{F}$ is closed convex. This implies that w_{t+1} is a unique minimizer. Let $\bar{\mathcal{F}}$ be any open set that contains $\mathcal{F}$. Since $\bar{\mathcal{F}}$ is open, the derivatives on $\mathcal{F}$ can be appropriately defined.

Assumption. *The full loss F and minibatch losses $F^{\mathcal{B}}$ for all $\mathcal{B}$ satisfy the following conditions.*

1. *Feasibility set $\mathcal{F}$ must be such that $w^* \in \mathcal{F}$, where w^* is an optimal solution to the unconstrained problem.*
2. *$F(w)$ is a c strongly convex twice differentiable function.*
3. *For all $\mathcal{B} \subset \{1, \dots, N\}$ with $|\mathcal{B}| = b$, $\nabla F^{\mathcal{B}}(w)$ is Lipschitz continuous on $\bar{\mathcal{F}}$, i.e., there exists $L > 0$ such that for all $w, w' \in \bar{\mathcal{F}}$, we have $\left\|\nabla F^{\mathcal{B}}(w) - \nabla F^{\mathcal{B}}(w')\right\| \le L\left\|w - w'\right\|_2$.*
4. *There exists $0 < D_V < +\infty$ such that for all $w \in \bar{\mathcal{F}}$, $\mathbb{E}_{\mathcal{B}}\left[\left\|\nabla F^{\mathcal{B}}(w) - \nabla F(w)\right\|_2^2\right] \le D_V$, where $\mathcal{B}$ is a random subset of $\{1, \dots, N\}$ with $|\mathcal{B}| = b$.*

The conditions are commonly made when analyzing convergence of gradient based optimization methods. Remarkably, we avoid making the uniform bounded gradient assumption, which contradicts strongly convexity. However, given the Assumption and the bound for the trajectory w_t, i.e., $\|w_t\|_2 \le D_w$ for all t, the stochastic gradients g_t in each iterate are also uniformly bounded for each t, as shown in the following proposition.

Proposition 1. *There exists $0 < D_G < +\infty$ such that for all t, $\|g_t\|_2 \le D_G$.*

Furthermore, under specific requirements on learning rates, the optimizer variables A_t and b_t are also bounded. In what follows, all matrix norms are spectral.

Proposition 2. *Let $S_\alpha := \sum_{s=1}^{\infty} \alpha_s < 1/D_w^2 < \infty$ and let $\beta_t \le 1$ for all t. There exist D_A and D_b independent on t, such that $\|A_t\|_2 \le D_A$ and $\|b_t\|_2 \le D_b$ for all t.*

The requirement $\sum_{t=1}^{\infty} \alpha_s < \infty$ can be simply satisfied by $\alpha_s = 6/\pi^2 D_w^2 s^2$. We prove this proposition by properly applying triangle inequalities and the principle of induction.

Since $F(w) = \frac{1}{\binom{N}{b}} \sum_{\mathcal{B}:|\mathcal{B}|=b} F^{\mathcal{B}}(w)$, Item 3 in Assumption naturally implies that

$$\begin{aligned}\|\nabla F(w) - \nabla F(w')\|_2 &\le \frac{1}{\binom{N}{b}} \sum_{\mathcal{B}:|\mathcal{B}|=b} \|\nabla F^{\mathcal{B}}(w) - \nabla F^{\mathcal{B}}(w')\|_2 \\ &\le L\|w - w'\|_2,\end{aligned}$$

i.e., ∇F is also Lipchitz continuous. Therefore we have $\|\nabla F(w_t)\|_2 \le L\left\|w_t - w^*\right\| \le 2LD_w < D_G$ and we have $\max\left\{\|\nabla F(w_t)\|_2, \left\|\widehat{G}_t\right\|_2\right\} \le D_G$. Using the previously defined constants L, D_A and D_G, we present the convergence result of Pseudo-linear TO.

Theorem 1. *Given Assumption, setting $\gamma_t = \gamma/(t+\mu)$ and $\beta_t = \beta/(t-1+\mu)$, where γ and β satisfy*

$$\begin{aligned}\gamma &> \frac{1}{c} \\ \beta &> 1 + \frac{4\gamma^2 L^2 \sqrt{D_A^2 + L^2}}{c(\gamma c - 1)} + 2\gamma\sqrt{D_A^2 + L^2},\end{aligned}$$

and $\alpha_t = \alpha/(t-1+\mu)^2$ with $\sum_{s=1}^{\infty} \alpha_s < 1/D_w^2$, and μ to be large enough such that $\alpha_1 < 1, \beta_1 < 1$, $\gamma_1 < 1$, and $\alpha D_w^2/\mu + \beta < \mu$, there exist $M_1, M_2 > 0$ such that for all t, we have

$$\begin{aligned}\mathbb{E}\left[\|w_t - w^*\|_2^2\right] &\le \frac{M_1}{t+\mu} \\ \mathbb{E}\left[\left\|\widehat{G}_t - \nabla F(w_t)\right\|_2^2\right] &\le \frac{M_2}{t+\mu}.\end{aligned}$$

To prove the theorem, we jointly bound $\mathbb{E}\left[\|w_t - w^*\|_2^2\right]$ and $\mathbb{E}\left[\left\|\widehat{G}_t - \nabla F(w_t)\right\|_2^2\right]$ recurrently and apply induction. The expectations are over history up to t. Theorem 1 shows that for strongly convex losses, Pseudo-linear TO achieves an $\mathcal{O}(1/t)$ convergence rate, matching the rates of SGD and Momentum. Comparing to ADAM, which has strongly convex examples with non-convergent variance (Wang and Klabjan 2025), namely

$$\mathbb{E}\left[\|\widehat{G}_t^{(\text{ADAM})} - \nabla F(w_t)\|^2\right] \not\to 0,$$

the convergence result for Pseudo-linear TO automatically achieves the reduction of variance.

3.3 Simplified Variants of Pseudo-linear TO

A notable limitation of Pseudo-linear TO lies in its requirement of memory. For model weights $w \in \mathbb{R}^d$, the optimizer variables A and b introduce $d^2 + d$ new parameters, which is impractical for large models, when d is large. To address this challenge, we propose two memory-efficient variants of Pseudo-linear TO.

- **Diagonal TO** (Algorithm 2) enforces $A_t = \operatorname{diag}\{a_t\}$, where $a_t \in \mathbb{R}^d$, reducing the parameter count to $\mathcal{O}(d)$. Here, $\odot$ denotes element-wise vector multiplication.

Algorithm 2: Diagonal TO

```
Input: Learning rates $\alpha_t, \beta_t, \gamma_t$.
1: Initialize $w_1, a_0$ and $b_0$.
2: for $t = 1, \ldots, T$ do
3:     Sample $\mathcal{B}_t \subset \{1, \ldots, N\}$ such that $|\mathcal{B}_t| = b$
4:     $g_t \leftarrow \nabla F^{\mathcal{B}_t}(w_t)$
5:     $a_t \leftarrow a_{t-1} + \alpha_t(g_t - a_{t-1} \odot w_t - b_{t-1}) \odot w_t$
6:     $b_t \leftarrow b_{t-1} + \beta_t(g_t - a_{t-1} \odot w_t - b_{t-1})$
7:     $\widehat{G}_t \leftarrow a_t \odot w_t + b_t$
8:     $w_{t+1} \leftarrow w_t - \gamma_t \widehat{G}_t$
9: end for
```

Algorithm 3: RankOne TO

```
Input: Learning rates $\alpha_t, \beta_t, \gamma_t$.
1: Initialize $w_1, a_0, c_0$ and $b_0$.
2: for $t = 1, \ldots, T$ do
3:     Sample $\mathcal{B}_t \subset \{1, \ldots, N\}$ such that $|\mathcal{B}_t| = b$
4:     $g_t \leftarrow \nabla F^{\mathcal{B}_t}(w_t)$
5:     $a_t \leftarrow a_{t-1} + \alpha_t(g_t - a_{t-1}c_{t-1}^\top w_t - b_{t-1})c_{t-1}^\top w_t$
6:     $c_t \leftarrow c_{t-1} + \alpha_t(g_t - a_{t-1}c_{t-1}^\top w_t - b_{t-1})^\top a_{t-1} w_t$
7:     $b_t \leftarrow b_{t-1} + \beta_t(g_t - a_{t-1}c_{t-1}^\top w_t - b_{t-1})$
8:     $\widehat{G}_t \leftarrow a_t c_t^\top w_t + b_t$
9:     $w_{t+1} \leftarrow w_t - \gamma_t \widehat{G}_t$
10: end for
```

- **RankOne TO** (Algorithm 3) employs factorization $A_t = a_t c_t^\top$ with $a_t, c_t \in \mathbb{R}^d$, similarly achieving memory complexity $\mathcal{O}(d)$ while maintaining a richer parameter interaction.

The gradient update rules in both algorithms (Lines 5-6 of Algorithm 2 and Lines 5-7 of Algorithm 3) are derived from the corresponding structured gradient computations for each parameterization. These modifications enlarge the scale of problems that are suitable under the Pseudo-linear TO framework. It is easy to extend the proof of Theorem 1 and show the convergence of Diagonal TO and RankOne TO.

4 Experiments

4.1 Datasets and Implementations

We conduct comprehensive empirical evaluations comparing various TO implementations against ADAM and Momentum across multiple real-world datasets.

- **MNIST** (Deng 2012): A handwritten digit dataset comprising of 60,000 grayscale images (28×28 pixels)
- **News20** (Lang 1995): Approximately 18,000 newsgroup posts across 20 topics
- **CovType** (Blackard, Dean, and Anderson 1998): Forest cover type prediction dataset with 581,012 samples and 54 features across 7 classes
- **SmallNorb** (LeCun, Huang, and Bottou 2004): 3D object recognition dataset containing images of 50 toys from 5 categories
- **KDD** (Tavallaee et al. 2009): Network intrusion detection subset from KDD CUP 99
- **CIFAR-10** (Krizhevsky and Hinton 2009): 60,000 color images across 10 object classes
- **Alpaca** (Taori et al. 2023): 52,000 instruction-following demonstrations generated by text-davinci-003

The models employed are logistic regressions or deep neural networks. The architectures are described later in the appendix. The last dataset corresponds to fine-tuning of an LLM. The experiments span strongly convex, convex, non-convex and LLM tasks.

Given the negligible computational overhead beyond gradient computation, we focus on convergence speed measured in epochs. For each experiment, we perform extensive hyperparameter tuning:

- ADAM learning rates: $\gamma \in [10^{-5},\ 2 \times 10^{-5},\ 5 \times 10^{-5},\ 10^{-4},\ 2 \times 10^{-4},\ 5 \times 10^{-4},\ 10^{-3},\ 2 \times 10^{-3},\ 5 \times 10^{-3}]$,
- TO methods: $\gamma \in [10^{-3},\ 2 \times 10^{-3},\ 5 \times 10^{-3},\ 0.01,\ 0.02,\ 0.05,\ 0.1,\ 0.2,\ 0.5]$,
- TO-specific parameters: $\alpha, \beta \in [0.0, 0.01, 0.1, 0.5, 1.0]$.

All the hyperparameters are tested under constant and exponentially decaying scheme. The exponential decay rates are selected among $[0.6,\ 0.8,\ 0.95]$. The trainable weights of logistic regression and FFN are initialized from standard normal sampling. The mini-batch size is set to be 64.

We use the full training loss as our primary evaluation metric. To ensure statistical reliability, we repeat each experiment five times with different random seeds. Using ADAM as our baseline, we calculate the relative performance difference ρ for each method. Let loss_k be the training loss of a considered algorithm in epoch k. The relative difference is defined as

$$\rho = \text{avg}\left(\frac{\min_{1\le k\le T}\{\text{ADAM loss}_k\} - \min_{1\le k\le T}\{\text{loss}_k\}}{\min_{1\le k\le T}\{\text{ADAM loss}_k\}}\right),$$

where T is the number of epochs and the average is taken across multiple runs. Metric $\rho > 0$ indicates superior performance to ADAM in terms of achieved minimum loss. We complement this with two-sided Wald tests on the relative differences, reporting significance levels s. Our statistical analysis follows these conventions:

- $s < 0.05$: Significantly better than ADAM
- $s > 0.95$: Significantly worse than ADAM
- $0.05 \le s \le 0.95$: No significant difference from ADAM.

Our theoretical analyses show that pseudo-linear and other variants of TO achieve an $\mathcal{O}(1/t)$ convergence rate for strongly convex losses, and a convergence of the approximate variance with the same convergence rate. This suggests the expectation of TO's advantage on strongly convex task and noisy and complex tasks.

4.2 Convex Experiments

For convex losses, we train logistic regression models on five of the previously mentioned datasets. Since our theoretical analyses primarily address strongly convex losses,

Dataset	λ	Full TO	Diagonal TO	RankOne TO
SmallNorb	λ^*	N/A	$\mathbf{18.0}$ $(+)$	13.0 $(+)$
CovType	λ^*	$\mathbf{3.3}$ $(\sim)$	-2.0 $(\sim)$	-2.1 $(\sim)$
KDD	λ_1	15.0 $(+)$	$\mathbf{16.0}$ $(+)$	14.0 $(+)$
	λ_2	6.0 $(+)$	$\mathbf{5.8}$ $(+)$	-3.0 $(\sim)$
	λ_3	10.0 $(+)$	10.0 $(+)$	$\mathbf{10.2}$ $(+)$
	λ_4	$\mathbf{9.0}$ $(+)$	7.6 $(+)$	-6.0 $(\sim)$
	λ_5	11.0 $(+)$	$\mathbf{11.2}$ $(+)$	8.7 $(+)$
News20	λ_1	N/A	$\mathbf{0.4}$ $(+)$	0.4 $(+)$
	λ_2	N/A	$\mathbf{0.2}$ $(+)$	0.2 $(+)$
	λ_3	N/A	$\mathbf{4.4}$ $(+)$	4.3 $(+)$
	λ_4	N/A	$\mathbf{4.8}$ $(+)$	4.8 $(+)$
	λ_5	N/A	$\mathbf{16.8}$ $(+)$	16.7 $(+)$
MNIST	λ_1	$\mathbf{4.2}$ $(+)$	3.9 $(+)$	-1.1 $(-)$
	λ_2	$\mathbf{4.4}$ $(+)$	3.9 $(+)$	-1.0 $(-)$
	λ_3	-7.1 $(-)$	$\mathbf{0.7}$ $(+)$	$\mathbf{0.7}$ $(+)$
	λ_4	$\mathbf{0.3}$ $(+)$	0.2 $(+)$	0.2 $(+)$
	λ_5	$\mathbf{0.7}$ $(+)$	0.6 $(+)$	-3.4 $(-)$

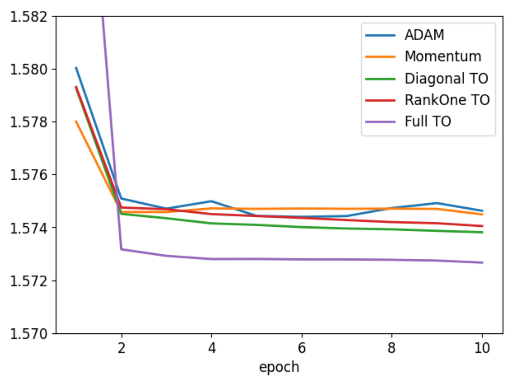
Table 2: Values of ρ for strongly convex tasks (all values $\times 10^{-3}$). Full TO denotes for full version of pseudo-linear TO in Algorithm 1. '+', '$\sim$' and '-' denote for $s < 0.05$, $0.05 \leq s \leq 0.95$, and $s > 0.95$, respectively.

we emphasize results using logistic regression with cross-entropy loss and L2 regularization. Our experimental protocol involves three key validation steps to determine the optimal regularization parameter λ. First, we randomly partition each dataset into training and validation sets. Second, we train models using ADAM with λ values from the discrete set $[0,\ 5\times10^{-4},\ 10^{-3},\ 5\times10^{-3},\ 0.01,\ 0.05]$, selecting the λ yielding best validation performance with ADAM. When the optimal λ equals zero, we conduct additional evaluations by randomly sampling five normally distributed λ values with mean being 0 and standard deviation of 1, and filter the negative values, in order to thoroughly compare TO variants against ADAM on strongly convex losses. The final selected λ values for each dataset appear in the appendix.

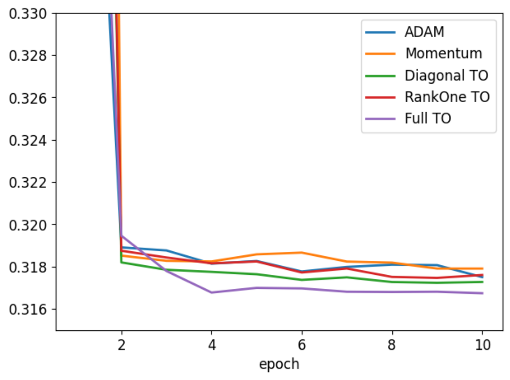
Table 2 presents the values of ρ and significance for all strongly convex experiments. Due to memory constraints, we exclude pseudo-linear TO results for SmallNorb and News20 datasets. Our analysis of 17 strongly convex experiments reveals that TO variants significantly outperform ADAM in 16 cases. TO variants also outperform Momentum in 15 cases. The complete values of ρ including Momentum are in the appendix.

To investigate performance on convex but non-strongly-convex losses, we conduct parallel experiments without regularization $\lambda = 0$ on the same datasets. In Figure 1, we plot the curve of training loss of the $\lambda = 0$ experiments for the three datasets where Full TO could be evaluated (CovType, MNIST and KKD) and find that pseudo-linear TO exceeds ADAM's performance. Additional results, such as training losses for different additional datasets, and the standard deviation of the train loss among different random seeds can be found in the appendix.

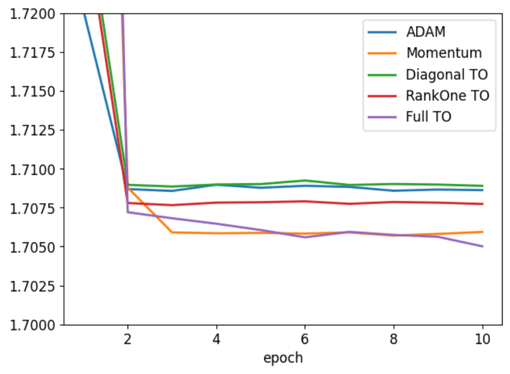
Our empirical findings suggest initializing with $\alpha \approx 0.01$ while $\beta \approx 1$. Exponential decaying of α and β also plays



(a) CovType



(b) KDD

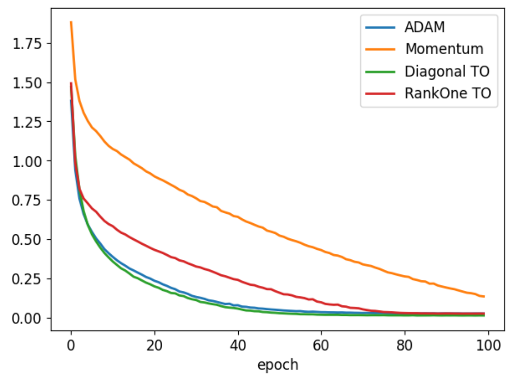


(c) MNIST

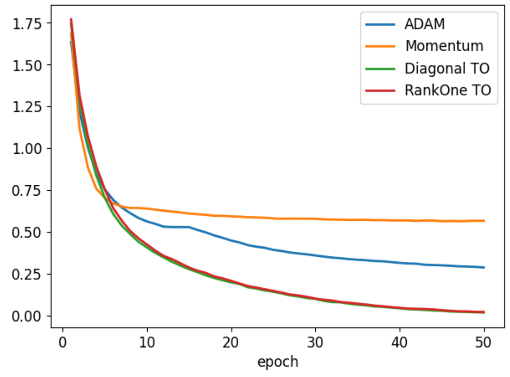
Figure 1: Training loss curves for logistic regression ($\lambda = 0$) on different datasets

crucial rules practically. These settings provide robust performance across different datasets.

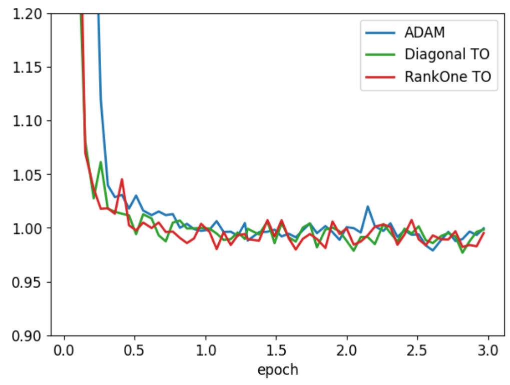
4.3 Nonconvex Experiments



(a) ResNet18-CIFAR10



(b) ResNet50-CIFAR10



(c) Llama7b-Alpaca

Figure 2: Training loss curves for non-convex tasks

For non-convex optimization tasks, we conduct three experimental evaluations. First, we train FFNs on both the CovType and KDD datasets. Second, we train ResNets of varying depths (18 and 50 layers) on the CIFAR-10 classification task. Third, we perform fine-tuning of a selected subset of pretrained weights from the Llama-7b model using the Alpaca dataset.

Model	Dataset	Full TO	Diagonal TO	RankOne TO
FFN	CovType	-1.3×10^{-4}	-1.8×10^{-4}	-2.0×10^{-4}
	KDD	-1.1×10^{-3}	-3.2×10^{-3}	-2.1×10^{-3}
ResNet18	CIFAR10	N/A	**0.5**	8.2×10^{-3}
ResNet50		N/A	**0.9**	**0.9**
Llama-7b	Alpaca	N/A	$\mathbf{2.1 \times 10^{-3}}$	-9.2×10^{-4}

Table 3: Values of ρ for non-convex tasks

The relative performance differences are reported in Table 3. Figure 2 shows the curves of training loss for the tasks of training CIFAR10 with ResNet and finetuning Llama. These experiments demonstrate TO's superior performance on challenging non-convex problems. For the Llama finetuning task, TO achieves faster initial convergence compared to ADAM. In the ResNet classification tasks, TO consistently discovers solutions with better final optimality. The training curves for FFN are in the appendix, which show comparable performance between TO and ADAM, with no statistically significant difference observed.

5 Conclusion

This paper introduces Trainable Optimizer (TO), a novel optimization framework that jointly learns model parameters and optimization dynamics through a trainable linear gradient approximation, with theoretical guarantees showing simultaneous convergence of both parameters and approximate error. We develop memory-efficient Diagonal and RankOne TO variants that maintain $\mathcal{O}(d)$ complexity while preserving convergence properties. Experiments across convex and non-convex tasks on seven datasets demonstrate TO's advantages: superior performance to ADAM in 16/17 strongly convex cases, faster convergence and better solutions for ResNet training and Llama fine-tuning, and comparable performance on simpler non-convex tasks.

References

Andrychowicz, M.; Denil, M.; Gómez, S.; Hoffman, M. W.; Pfau, D.; Schaul, T.; Shillingford, B.; and de Freitas, N. 2016. Learning to learn by gradient descent by gradient descent. In Lee, D.; Sugiyama, M.; Luxburg, U.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Bello, I.; Zoph, B.; Vasudevan, V.; and Le, Q. V. 2017. Neural optimizer search with reinforcement learning. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 459–468. International Convention Centre, Sydney, Australia: PMLR.

Blackard, J. A.; Dean, D. J.; and Anderson, C. W. 1998. UCI Machine Learning Repository: Covertype Data Set. https://archive.ics.uci.edu/ml/datasets/covertype.

Chen, T.; Chen, X.; Chen, W.; Heaton, H.; Liu, J.; Wang, Z.; and Yin, W. 2022. Learning to Optimize: A Primer and A Benchmark. *Journal of Machine Learning Research*, 23(189): 1–59.

Cutkosky, A.; and Orabona, F. 2019. Momentum-based Variance Reduction in Non-Convex SGD. In *Advances in*

Neural Information Processing Systems (NeurIPS), 15210–15219.

Defazio, A.; Bach, F.; and Lacoste-Julien, S. 2014. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Deng, L. 2012. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6): 141–142.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7).

Hinton, G.; Srivastava, N.; and Swersky, K. 2012. 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.

Huang, F.; Li, J.; and Huang, H. 2022. SUPER-ADAM: Faster and Universal Framework of Adaptive Gradients. arXiv:2106.08208.

Johnson, R.; and Zhang, T. 2013a. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems*.

Johnson, R.; and Zhang, T. 2013b. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. In Burges, C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Kingma, D. P.; and Ba, J. 2015. ADAM: A method for stochastic optimization. *International Conference on Learning Representations*.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf.

Lang, K. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, 331–339.

LeCun, Y.; Huang, F. J.; and Bottou, L. 2004. Learning methods for generic object recognition with invariance to pose and lighting. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2: II–104 Vol.2.

Li, K.; and Malik, J. 2017. Learning to optimize. In *International Conference on Learning Representations (ICLR)*.

Qian, X.; and Klabjan, D. 2020. The impact of the mini-batch size on the variance of gradients in stochastic gradient descent. *arXiv preprint arXiv:2004.13146*.

Reddi, S. J.; Kale, S.; and Kumar, S. 2018. On the convergence of ADAM and beyond. *International Conference on Learning Representations*.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Tavallaee, M.; Bagheri, E.; Lu, W.; and Ghorbani, A. A. 2009. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on Computational Intelligence for Security and Defense Applications*.

Wang, R.; and Klabjan, D. 2025. Divergence Results and Convergence of a Variance Reduced Version of ADAM. *arXiv:2210.05607*.

Zaheer, M.; Reddi, S.; Sachan, D.; Kale, S.; and Kumar, S. 2018. Adaptive methods for nonconvex optimization. *Advances in Neural Information Processing Systems*.

A Trainable Optimizer: Supplementary Materials

A Proofs

A.1 Technical Lemmas

Lemma 1. *For all t, $\|w_{t+1} - w_t\|_2 \le \gamma_t \left\|\widehat{G}_t\right\|_2$.*

Proof. (Reddi, Kale, and Kumar 2018) shows that for all u and u', $\|\Pi_{\mathcal{F}}(u) - \Pi_{\mathcal{F}}(u')\|_2 \le \|u - u'\|_2$. In our case, we have $w_t \in \mathcal{F}$, therefore $\|w_{t+1} - w_t\|_2 = \|\Pi_{\mathcal{F}}(w_t - \gamma_t \widehat{G}_t) - \Pi_{\mathcal{F}}(w_t)\|_2 \le \|\gamma_t \widehat{G}_t\|_2$. □

Lemma 2. *Let $0 < k_2 < k_1$ and $b_1 < 0 < b_2$, for any x_0 and y_0. There exists $x \ge x_0$ and $y \ge y_0$ such that $k_2 x + b_2 \le y \le k_1 x + b_1$.*

Proof. Let $x^* = \frac{b_2 - b_1}{k_1 - k_2}$, $y^* = \frac{k_1 b_2 - k_2 b_1}{k_1 - k_2}$, and let $x_n = x^* + n$, $y_n = y^* + \frac{k_1 + k_2}{2} n$. It is easy to check that $k_2 x_n + b_2 \le y_n \le k_1 x_n + b_1$ for all $n > 0$. We can pick n large enough such that $x_n \ge x_0$ and $y_n \ge y_0$. □

A.2 Proof to Proposition 1

Proof. By definition g_t reads

$$\begin{aligned}
\|g_t\|_2 &= \|\nabla F^{\mathcal{B}_t}(w_t)\|_2 \\
&\le \|\nabla F^{\mathcal{B}_t}(w_t) - \nabla F^{\mathcal{B}_t}(w^*)\|_2 + \|\nabla F^{\mathcal{B}_t}(w^*)\|_2 \\
&\le L\|w_t - w^*\| + \max_{\mathcal{B}} \|\nabla F^{\mathcal{B}}(w^*)\|_2 \\
&\le 2LD_w + \max_{\mathcal{B}} \|\nabla F^{\mathcal{B}}(w^*)\|_2,
\end{aligned}$$

where the inequalities hold according to Assumption. The proposition is proved by letting $D_G = 2LD_w + \max_{\mathcal{B}} \|\nabla F^{\mathcal{B}}(w^*)\|_2$. □

A.3 Proof to Proposition 2

Proof. Consider

$$\begin{aligned}
D_b &= \max\left\{\|b_0\|_2, \frac{1}{1 - S_\alpha D_w^2}\left(D_G(1 + S_\alpha D_w^2) + \|A_0\|_2 D_w\right)\right\} \\
D_A &= \|A_0\|_2 + S_\alpha D_w (D_b + D_G).
\end{aligned}$$

Apparently $\|b_0\|_2 \le D_b$. We assume that for all $0 \le t \le T-1$, $\|b_t\| \le D_b$. Reformulate the updating rule for A_t yields

$$\begin{aligned}
A_t &= A_{t-1} + \alpha_t (g_t - A_{t-1} w_t - b_{t-1}) w_t^\top \\
&= A_{t-1}(I_d - \alpha_t w_t w_t^\top) + \alpha_t (g_t - b_t) w_t^\top.
\end{aligned}$$

Then, for all $1 \le t \le T-1$,

$$\begin{aligned}
\|A_t\|_2 &= \left\|A_{t-1}(I_d - \alpha_t w_t w_t^\top) + \alpha_t (g_t - b_t) w_t^\top\right\|_2 \\
&\le \left\|I_d - \alpha_t w_t w_t^\top\right\|_2 \|A_{t-1}\|_2 + \alpha_t \|g_t - b_t\|_2 \|w_t\|_2 \\
&\le \|A_{t-1}\|_2 + \alpha_t (D_b + D_G) D_w,
\end{aligned}$$

where the last inequality holds because $\|g_t\| \le D_G$. We recursively obtain

$$\begin{aligned}
\|A_t\|_2 &\le \|A_0\|_2 + \sum_{s=1}^{t} \alpha_s (D_b + D_G) D_w \\
&< \|A_0\|_2 + S_\alpha (D_b + D_G) D_w = D_A.
\end{aligned}$$

Furthermore, we have

$$\begin{aligned}
\|b_T\|_2 &= \|(1 - \beta_T) b_{T-1} + \beta_T (g_T - A_{T-1} w_T)\|_2 \\
&\le (1 - \beta_T)\|b_{T-1}\|_2 + \beta_T (\|g_T\|_2 + \|A_{T-1}\|_2 \|w_T\|_2) \\
&\le (1 - \beta_T) D_b + \beta_T (D_G + D_A D_w) \\
&\le D_b.
\end{aligned}$$

The last inequality holds because

$$\begin{aligned} D_b &= \max\left\{ \|b_0\|_2, \frac{1}{1-S_\alpha D_w^2}\left(D_G(1+S_\alpha D_w^2) + \|A_0\|_2 D_w\right)\right\} \\ &\geq \frac{1}{1-S_\alpha D_w^2}\left(D_G(1+S_\alpha D_w^2) + \|A_0\|_2 D_w\right), \end{aligned}$$

which implies that

$$(1-S_\alpha D_w^2)D_b \geq D_G(1+S_\alpha D_w^2) + \|A_0\|_2 D_w,$$

and therefore

$$\begin{aligned} D_b &\geq D_b S_\alpha D_w^2 + D_G(1+S_\alpha D_w^2) + \|A_0\|_2 D_w \\ &= D_G + D_w(\|A_0\|_2 + S_\alpha D_w(D_G + D_b)) = D_G + D_w D_A. \end{aligned}$$

According to the principle of induction, this finishes the proof. □

A.4 Proof to Theorem 1

Proof. We start with bounding the approximation loss of the full gradient. Letting $G_t = \nabla F(w_t)$, the approximation error reads

$$\begin{aligned} \widehat{G}_t - G_t &= A_t w_t + b_t - G_t \\ &= \left(A_{t-1} + \alpha_t\left(g_t - A_{t-1}w_t - b_{t-1}\right)w_t^\top\right)w_t + b_{t-1} + \beta_t\left(g_t - A_{t-1}w_t - b_{t-1}\right) - G_t \\ &= A_{t-1}w_t + b_{t-1} + (\alpha_t\|w_t\|_2^2 + \beta_t)\left(g_t - A_{t-1}w_t - b_{t-1}\right) - G_t \\ &= (1-\delta_t)\left(A_{t-1}w_t + b_{t-1} - G_t\right) + \delta_t\left(g_t - G_t\right), \end{aligned}$$

where $\delta_t = \alpha_t\|w_t\|_2^2 + \beta_t$. Letting $\delta = \alpha D_w^2/\mu + \beta$, apparently we have $\delta_t = (\alpha\|w_t\|_2^2/(t-1+\mu) + \beta)/(t-1+\mu) < \delta/(t-1+\mu)$. Taking the square norm of the approximation error yields

$$\begin{aligned} \left\|\widehat{G}_t - G_t\right\|_2^2 &= (1-\delta_t)^2\|A_{t-1}w_t + b_{t-1} - G_t\|_2^2 + \delta_t^2\|g_t - G_t\|_2^2 \\ &\quad +2(1-\delta_t)\delta_t\left(A_{t-1}w_t + b_{t-1} - G_t\right)^\top(g_t - G_t). \end{aligned}$$

Letting $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot\,|\mathcal{B}_1, \cdots, \mathcal{B}_{t-1}]$, we notice that A_{t-1}, b_{t-1}, G_t and w_t are known given $\mathcal{B}_1, \cdots, \mathcal{B}_{t-1}$. Then, we have

$$\begin{aligned} \mathbb{E}_t\left[\left\|\widehat{G}_t - G_t\right\|_2^2\right] &= (1-\delta_t)^2\|A_{t-1}w_t + b_{t-1} - G_t\|_2^2 + \delta_t^2\mathbb{E}_t\left[\|g_t - G_t\|_2^2\right] \\ &\quad +2(1-\delta_t)\delta_t\left(A_{t-1}w_t + b_{t-1} - G_t\right)^\top\mathbb{E}_t\left[g_t - G_t\right] \\ &= (1-\delta_t)^2\|A_{t-1}w_t + b_{t-1} - G_t\|_2^2 + \delta_t^2\mathbb{E}_t\left[\|g_t - G_t\|_2^2\right] \\ &\leq (1-\delta_t)^2\|A_{t-1}w_t + b_{t-1} - G_t\|_2^2 + D_V\frac{\delta^2}{(t-1+\mu)^2} \\ &= (1-\delta_t)^2\underbrace{\left\|\widehat{G}_{t-1} - G_{t-1} + A_{t-1}(w_t - w_{t-1}) - (G_t - G_{t-1})\right\|_2^2}_{T_1} + D_V\frac{\delta^2}{(t-1+\mu)^2}, \end{aligned}$$

where the second equality holds because $\mathbb{E}_t g_t = G_t$.

For three arbitrary vectors a_1, a_2, a_3, the Cauchy inequality implies that for any $\lambda > 0$,

$$\begin{aligned} \|a_1 + a_2 + a_3\|_2^2 &= \|a_1\|_2^2 + \|a_2\|_2^2 + \|a_3\|_2^2 + 2a_1^\top a_2 + 2a_2^\top a_3 + 2a_3^\top a_1 \\ &\leq \|a_1\|_2^2 + \|a_2\|_2^2 + \|a_3\|_2^2 + \frac{1}{2\lambda}\|a_1\|_2^2 + 2\lambda\|a_2\|_2^2 + \frac{1}{2\lambda}\|a_1\|_2^2 + 2\lambda\|a_3\|_2^2 + \|a_2\|_2^2 + \|a_3\|_2^2 \\ &= (1+\frac{1}{\lambda})\|a_1\|_2^2 + (2+2\lambda)\|a_2\|_2^2 + (2+2\lambda)\|a_3\|_2^2. \end{aligned}$$

Applying this inequality to T_1 and replacing λ with $\lambda/(t-1+\mu)$, we have

$$\begin{aligned}
T_1 &\leq \left(1+\frac{\lambda}{t-1+\mu}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+2\left(1+\frac{t-1+\mu}{\lambda}\right)\left(\|A_{t-1}(w_t-w_{t-1})\|_2^2+\|G_t-G_{t-1}\|_2^2\right)\\
&\leq \left(1+\frac{\lambda}{t-1+\mu}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+2\left(1+\frac{t-1+\mu}{\lambda}\right)\left(D_A^2+L^2\right)\|w_t-w_{t-1}\|_2^2\\
&\leq \left(1+\frac{\lambda}{t-1+\mu}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+2\left(1+\frac{t-1+\mu}{\lambda}\right)\left(D_A^2+L^2\right)\gamma_{t-1}^2\|\widehat{G}_{t-1}\|_2^2\\
&\leq \left(1+\frac{\lambda}{t-1+\mu}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+4\left(1+\frac{t-1+\mu}{\lambda}\right)\left(D_A^2+L^2\right)\gamma_{t-1}^2(\|\widehat{G}_{t-1}-G_{t-1}\|_2^2+\|G_{t-1}\|_2^2)\\
&= \left(1+\frac{\lambda}{t-1+\mu}+4\left(1+\frac{t-1+\mu}{\lambda}\right)\left(D_A^2+L^2\right)\gamma_{t-1}^2\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2\\
&\quad +4\left(1+\frac{t-1+\mu}{\lambda}\right)\left(D_A^2+L^2\right)\gamma_{t-1}^2\|G_{t-1}\|_2^2,
\end{aligned}$$

where the second inequality applies Proposition 2 and Lipschitz continuity of ∇F, the third inequality applies Lemma 1 and the forth inequality further applies the Cauchy inequality. We let $\lambda = 2\gamma\sqrt{D_A^2+L^2}$. The upper bound is simplified to

$$\begin{aligned}
T_1 &\leq \left(1+\frac{4\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}+\frac{4\gamma^2(D_A^2+L^2)}{(t-1+\mu)^2}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+4\left(1+\frac{t-1+\mu}{2\gamma\sqrt{D_A^2+L^2}}\right)\frac{\gamma^2(D_A^2+L^2)}{(t-1+\mu)^2}\|G_{t-1}\|_2^2\\
&= \left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+4\left(1+\frac{t-1+\mu}{2\gamma\sqrt{D_A^2+L^2}}\right)\frac{\gamma^2(D_A^2+L^2)}{(t-1+\mu)^2}\|G_{t-1}\|_2^2\\
&\leq \left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+4\left(1+\frac{t-1+\mu}{2\gamma\sqrt{D_A^2+L^2}}\right)\frac{\gamma^2L^2(D_A^2+L^2)}{(t-1+\mu)^2}\|w_{t-1}-w^*\|_2^2\\
&\leq \left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+\frac{4\gamma L^2\sqrt{D_A^2+L^2}}{t-1+\mu}\|w_{t-1}-w^*\|_2^2,
\end{aligned}$$

where the second inequality further applies the Lipschitz continuity of ∇F. The last inequality holds because μ is large enough such that

$$\frac{1}{t-1+\mu}\leq\frac{1}{\beta}<\frac{1}{2\gamma\sqrt{D_A^2+L^2}},$$

and thus

$$\frac{1}{t-1+\mu}\left(1+\frac{t-1+\mu}{2\gamma\sqrt{D_A^2+L^2}}\right)<\frac{1}{\gamma\sqrt{D_A^2+L^2}}.$$

The expectation $\mathbb{E}_t[\|\widehat{G}_t-G_t\|_2^2]$ is upper bounded by

$$\mathbb{E}_t\left[\left\|\widehat{G}_t-G_t\right\|_2^2\right] \leq (1-\delta_t)^2\left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+\frac{C_1\gamma}{t-1+\mu}\|w_{t-1}-w^*\|_2^2+\frac{D_V\delta^2}{(t-1+\mu)^2},$$

where $C_1 = 4L^2\sqrt{D_A^2+L^2}$. The setting of parameters satisfies that $\beta_t<\delta_t<1$ and therefore

$$\begin{aligned}
(1-\delta_t)^2\left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2 &< \left(1-\frac{\beta}{t-1+\mu}\right)^2\left(1+\frac{2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2\\
&< \left(1-\frac{\beta-2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}\right)^2<1-\frac{\beta-2\gamma\sqrt{D_A^2+L^2}}{t-1+\mu}:=1-\frac{\beta^*}{t-1+\mu},
\end{aligned}$$

where $\beta^* := \beta-2\gamma\sqrt{D_A^2+L^2}>0$. We obtain

$$\mathbb{E}_t\left[\left\|\widehat{G}_t-G_t\right\|_2^2\right] \leq \left(1-\frac{\beta^*}{t-1+\mu}\right)\left\|\widehat{G}_{t-1}-G_{t-1}\right\|_2^2+\frac{C_1\gamma}{t-1+\mu}\|w_{t-1}-w^*\|_2^2+\frac{D_V\delta}{(t-1+\mu)^2}.$$

Taking full expectation on both side yields

$$\mathbb{E}\left[\left\|\widehat{G}_t - G_t\right\|_2^2\right] \leq \left(1 - \frac{\beta^*}{t-1+\mu}\right)\mathbb{E}\left[\left\|\widehat{G}_{t-1} - G_{t-1}\right\|_2^2\right] + \frac{C_1\gamma}{t-1+\mu}\mathbb{E}\left[\|w_{t-1} - w^*\|_2^2\right] + \frac{D_V\delta}{(t-1+\mu)^2}. \quad (3)$$

Next, we consider the distance between the iterate and the optimal point. We obtain

$$\begin{aligned}
\|w_t - w^*\|_2^2 &\leq \left\|w_{t-1} - \gamma_{t-1}\widehat{G}_{t-1} - w^*\right\|_2^2 \\
&= \|w_{t-1} - w^*\|_2^2 + \gamma_{t-1}^2\left\|\widehat{G}_{t-1}\right\|_2^2 - 2\gamma_{t-1}\widehat{G}_{t-1}^\top(w_{t-1} - w^*) \\
&\leq \|w_{t-1} - w^*\|_2^2 + \gamma_{t-1}^2 D_G^2 - 2\gamma_{t-1}(\widehat{G}_{t-1} - G_{t-1})^\top(w_{t-1} - w^*) - 2\gamma_{t-1}G_{t-1}^\top(w_{t-1} - w^*) \\
&\leq (1 - 2c\gamma_{t-1})\|w_{t-1} - w^*\|_2^2 - 2\gamma_{t-1}(\widehat{G}_{t-1} - G_{t-1})^\top(w_{t-1} - w^*) + \gamma_{t-1}^2 D_G^2 \\
&\leq (1 - 2c\gamma_{t-1})\|w_{t-1} - w^*\|_2^2 + \gamma_{t-1}c\|w_{t-1} - w^*\|_2^2 + \frac{\gamma_{t-1}}{c}\left\|\widehat{G}_{t-1} - G_{t-1}\right\|_2^2 + \gamma_{t-1}^2 D_G^2 \\
&= \left(1 - \frac{c\gamma}{t-1+\mu}\right)\|w_{t-1} - w^*\|_2^2 + \frac{\gamma}{c(t-1+\mu)}\left\|\widehat{G}_{t-1} - G_{t-1}\right\|_2^2 + \frac{D_G^2\gamma^2}{(t-1+\mu)^2},
\end{aligned}$$

where the third inequality holds because of strong convexity of $F(w)$. Taking expectation we have

$$\begin{aligned}
\mathbb{E}\left[\|w_t - w^*\|_2^2\right] &\leq \left(1 - \frac{\gamma c}{t-1+\mu}\right)\mathbb{E}\left[\|w_{t-1} - w^*\|_2^2\right] + \frac{\gamma}{c(t-1+\mu)}\mathbb{E}\left[\left\|\widehat{G}_{t-1} - G_{t-1}\right\|_2^2\right] \\
&\quad + \frac{D_G^2\gamma^2}{(t-1+\mu)^2}. \qquad (4)
\end{aligned}$$

The setting of hyperparameters guarantees that $\frac{\beta^*-1}{C_1\gamma} > \frac{\gamma}{c(\gamma c-1)} > 0$. Applying Lemma 2, there exist $M_1 \geq (\mu + 1)\|w_1 - w^*\|_2^2$ and $M_2 \geq (\mu+1)\mathbb{E}\left[\left\|\widehat{G}_1 - G_1\right\|_2^2\right]$, such that

$$\frac{\gamma}{c(\gamma c-1)}M_2 + \frac{D_G^2\gamma^2}{\gamma c - 1} \leq M_1 \leq \frac{\beta^*-1}{C_1\gamma}M_2 - \frac{D_V\delta^2}{C_1\gamma}. \quad (5)$$

Apparently,

$$\begin{aligned}
\mathbb{E}\left[\|w_1 - w^*\|_2^2\right] &\leq \frac{M_1}{1+\mu}, \\
\mathbb{E}\left[\left\|\widehat{G}_1 - G_1\right\|_2^2\right] &\leq \frac{M_2}{1+\mu}.
\end{aligned}$$

We use induction. Assuming

$$\begin{aligned}
\mathbb{E}\left[\|w_{t-1} - w^*\|_2^2\right] &\leq \frac{M_1}{t-1+\mu}, \\
\mathbb{E}\left[\left\|\widehat{G}_{t-1} - G_{t-1}\right\|_2^2\right] &\leq \frac{M_2}{t-1+\mu},
\end{aligned}$$

and according to (3) and (4), we obtain

$$\begin{aligned}
\mathbb{E}\left[\left\|\widehat{G}_t - G_t\right\|_2^2\right] &\leq \left(1 - \frac{\beta^*}{t-1+\mu}\right)\frac{M_2}{t-1+\mu} + \frac{C_1\gamma}{(t-1+\mu)^2}M_1 + \frac{D_V\delta^2}{(t-1+\mu)^2} \\
&= \frac{t-2+\mu}{(t-1+\mu)^2}M_2 - \frac{1}{(t-1+\mu)^2}\left((\beta^*-1)M_2 - C_1\gamma M_1 - D_V\delta^2\right) \\
&\leq \frac{M_2}{t+\mu},
\end{aligned}$$

where the last inequality holds because $(t+\mu)(t-2+\mu) \leq (t-1+\mu)^2$ and the second term is guaranteed by (5) to be negative. We also have

$$\begin{aligned}
\mathbb{E}\left[\|w_t - w^*\|_2^2\right] &\leq \left(1 - \frac{\gamma c}{t-1+\mu}\right)\frac{M_1}{t-1+\mu} + \frac{\gamma}{c(t-1+\mu)^2}M_2 + \frac{D_G^2\gamma^2}{(t-1+\mu)^2} \\
&= \frac{t-2+\mu}{(t-1+\mu)^2}M_1 - \frac{1}{(t-1+\mu)^2}\left((\gamma c-1)M_1 - \frac{\gamma}{c}M_2 - D_G^2\gamma^2\right) \\
&\leq \frac{M_1}{t+\mu}.
\end{aligned}$$

This completes the proof. □

B Experimental Details

B.1 Values of λ

In the experiments for strongly convex losses, the l_2 regularization parameters λ are decided as described in Section 4.2. In Table 4, λ^* denotes the optimal value of λ and λ_1 to λ_5 denote the sampled values of λ when $\lambda^* = 0$.

Dataset	SmallNorb	CovType	News20	KDD	MNIST
λ^*	0.001	$5 \cdot 10^{-4}$	-	-	-
λ_1	-	-	0.460	0.970	1.600
λ_2	-	-	1.100	0.023	1.400
λ_3	-	-	0.034	0.210	0.034
λ_4	-	-	0.078	0.088	0.680
λ_5	-	-	0.212	0.390	0.460

Table 4: l_2 regularization parameters

B.2 Network Architectures

Next, we describe the FFNs used in the non-convex experiments. Each FFN has two fully connected layers with the dimensions shown in Table 5. The underlying activation function is ReLU.

Dataset	Input dimension	Hidden dimension	Output dimension
CovType	98	10	7
KDD	116	10	2

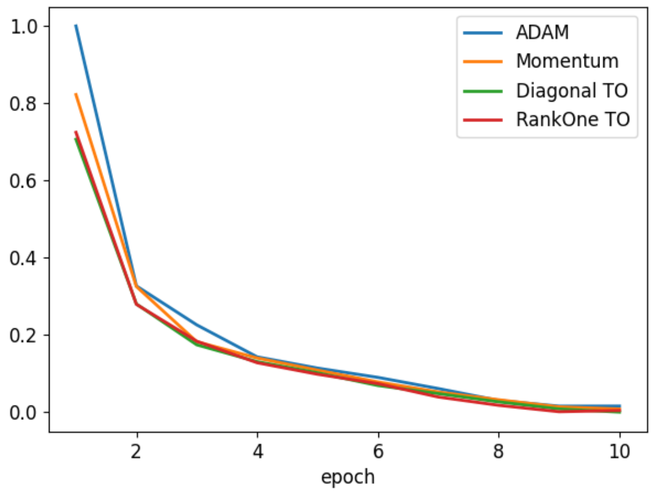
Table 5: Feedforward network structure

B.3 Additional Results

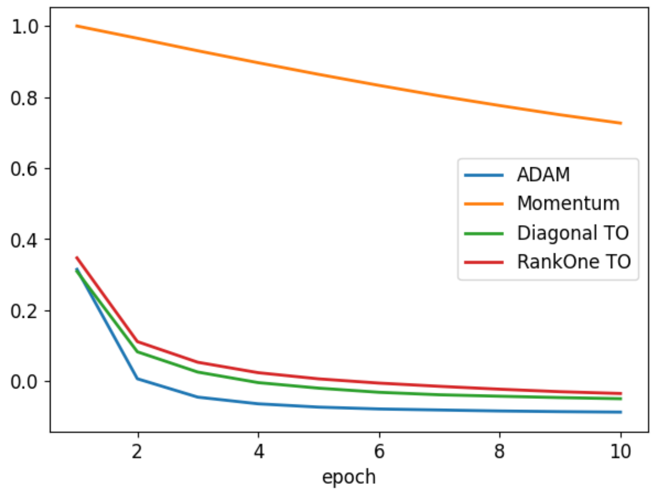
Additional experimental results are displayed next. Table 6 extends the results in Table 2 by adding the column of Momentum. Figures 3 and 5 display the training loss curves of additional convex and non-convex experiments. Specifically, the tasks are training logistic models on SmallNorb, News20 and FFNs on CovType and KDD. The y-axes show the min-max standardized training loss values. Figure 4 shows the standard deviation of training loss over multiple runs for convex experiments. The standard deviation for KDD is not included as the difference among runs is minor. The y-axes show the min-max standardized standard deviation values.

Dataset	λ	Momentum	Full TO	Diagonal TO	RankOne TO
SmallNorb	λ^*	-2.3 $(-)$	N/A	**18.0** $(+)$	13.0 $(+)$
CovType	λ^*	-1.3 $(\sim)$	**3.3** $(\sim)$	-2.0 $(\sim)$	-2.1 $(\sim)$
KDD	λ_1	-85.9 $(-)$	15.0 $(+)$	**16.0** $(+)$	14.0 $(+)$
	λ_2	-2.6 $(\sim)$	6.0 $(+)$	**5.8** $(+)$	-3.0 $(\sim)$
	λ_3	-21.2 $(-)$	10.0 $(+)$	10.0 $(+)$	**10.2** $(+)$
	λ_4	-11.8 $(-)$	**9.0** $(+)$	7.6 $(+)$	-6.0 $(\sim)$
	λ_5	-40.2 $(-)$	11.0 $(+)$	**11.2** $(+)$	8.7 $(+)$
News20	λ_1	-1.4 $(-)$	N/A	**0.4** $(+)$	0.4 $(+)$
	λ_2	0.2 $(+)$	N/A	**0.2** $(+)$	0.2 $(+)$
	λ_3	-1.0 $(\sim)$	N/A	**4.4** $(+)$	4.3 $(+)$
	λ_4	2.2 $(+)$	N/A	**4.8** $(+)$	4.8 $(+)$
	λ_5	2.0 $(\sim)$	N/A	**16.8** $(+)$	16.7 $(+)$
MNIST	λ_1	-9.1 $(-)$	**4.2** $(+)$	3.9 $(+)$	-1.1 $(-)$
	λ_2	-12.1 $(-)$	**4.4** $(+)$	3.9 $(+)$	-1.0 $(-)$
	λ_3	-13.9 $(-)$	-7.1 $(-)$	**0.7** $(+)$	**0.7** $(+)$
	λ_4	**12.7** $(+)$	0.3 $(+)$	0.2 $(+)$	0.2 $(+)$
	λ_5	**3.1** $(+)$	0.7 $(+)$	0.6 $(+)$	-3.4 $(-)$

Table 6: Values of ρ for strongly convex tasks (all values $\times 10^{-3}$). Full TO denotes for full version of pseudo-linear TO in Algorithm 1. '+', '$\sim$' and '-' denote for $s < 0.05$, $0.05 \leq s \leq 0.95$, and $s > 0.95$, respectively.

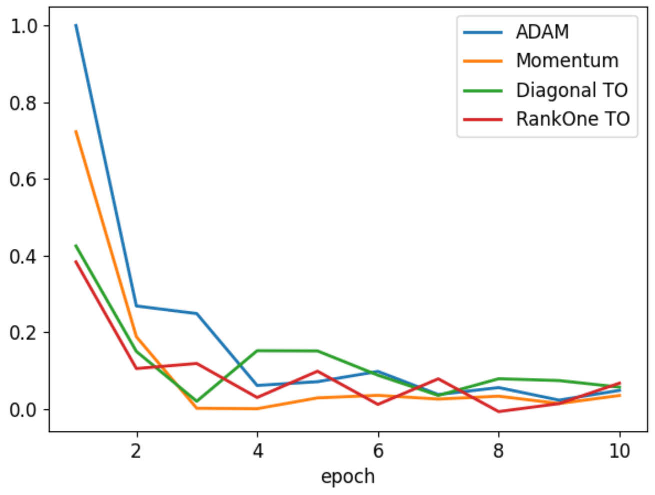


(a) SmallNorb

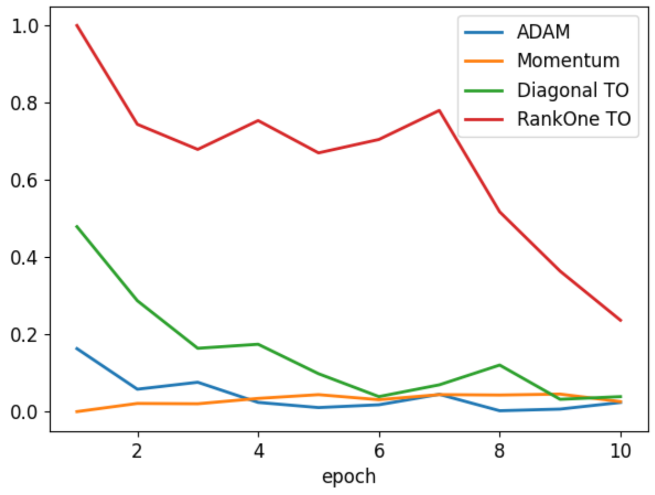


(b) News20

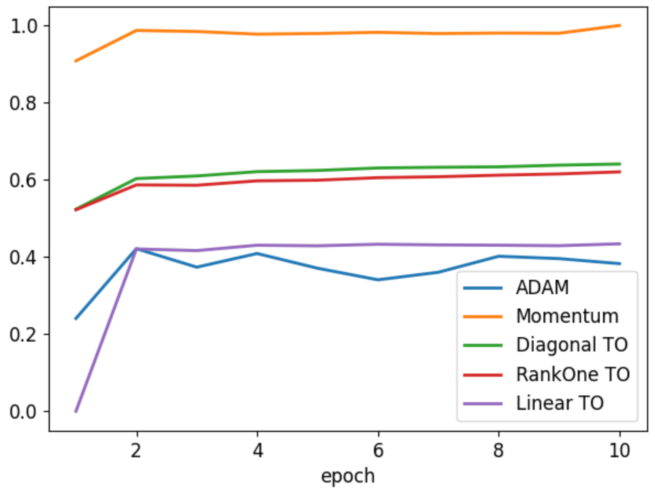
Figure 3: Training loss curves for logistic regression tasks with $\lambda = 0$. Standardization is based on minimum=0.96 and maximum=1.31 in the left figure, and minimum=2.18 and maximum=2.99 in the right figure.



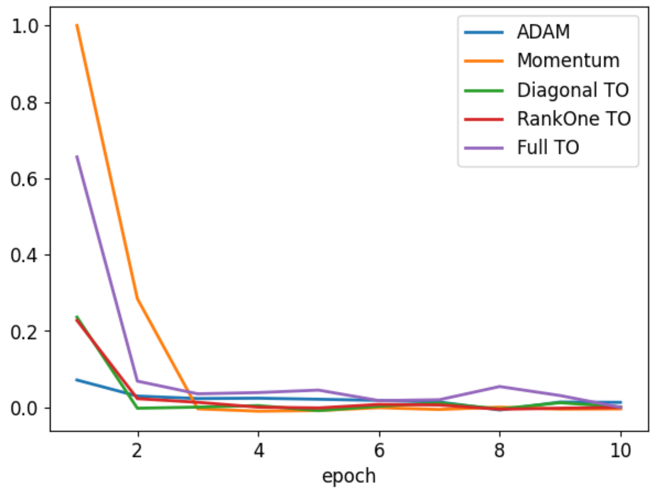
(a) SmallNorb



(b) News20

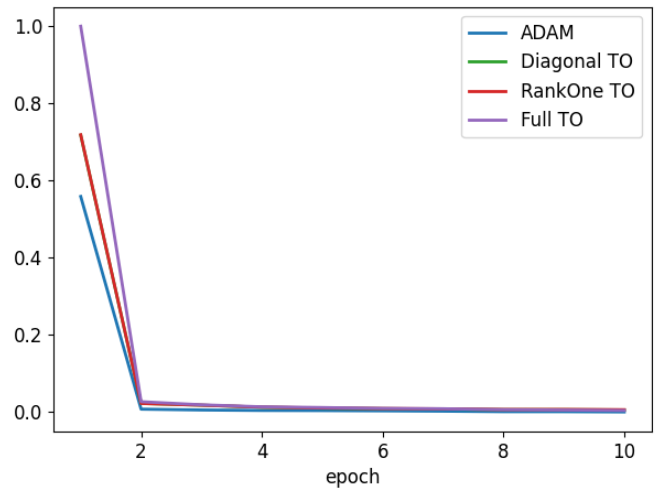


(c) CovType

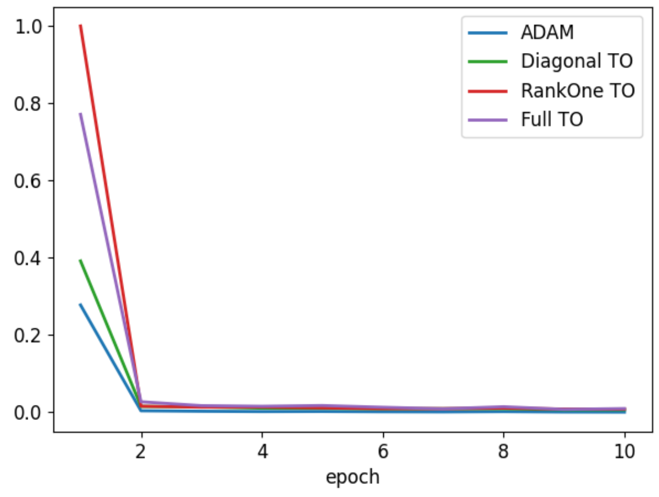


(d) MNIST

Figure 4: Standard deviation of training loss for logistic regression tasks with $\lambda = 0$. Standardization is based on minimum=1.31×10^{-3} and maximum=4.86×10^{-2} in Figure (a), minimum=2.78×10^{-4} and maximum=1.36×10^{-2} in Figure (b), minimum=1.00×10^{-2} and maximum=2.11×10^{-2} in Figure (c), and minimum=1.48×10^{-4} and maximum=6.48×10^{-3} in Figure (d).



(a) FFN-CovType



(b) FFN-KDD

Figure 5: Training loss curves for non-convex tasks. Standardization is based on minimum=1.48 and maximum=1.53 in the left figure, and minimum=0.32 and maximum=0.35 in the right figure.