A Dynamical View on Optimization Algorithms of Overparameterized Neural Networks

Zhiqi Bu **Shiyun Xu** **Kan Chen**
Department of Applied Mathematics and Computational Science
University of Pennsylvania

Abstract

When equipped with efficient optimization algorithms, the over-parameterized neural networks have demonstrated high level of performance even though the loss function is non-convex and non-smooth. While many works have been focusing on understanding the loss dynamics by training neural networks with the gradient descent (GD), in this work, we consider a broad class of optimization algorithms that are commonly used in practice. For example, we show from a dynamical system perspective that the Heavy Ball (HB) method can converge to global minimum on mean squared error (MSE) at a linear rate (similar to GD); however, the Nesterov accelerated gradient descent (NAG) may only converge to global minimum sublinearly.

Our results rely on the connection between neural tangent kernel (NTK) and finitely-wide over-parameterized neural networks with ReLU activation, which leads to analyzing the limiting ordinary differential equations (ODE) for optimization algorithms. We show that, optimizing the non-convex loss over the weights corresponds to optimizing some strongly convex loss over the prediction error. As a consequence, we can leverage the classical convex optimization theory to understand the convergence behavior of neural networks. We believe our approach can also be extended to other optimization algorithms and network architectures.

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

1 Introduction

Neural Tangent Kernel (NTK) [32] has taken a huge step in understanding the behaviors of over-parameterized neural networks: it has been shown that the training process of a neural network can be characterized by a kernel matrix. Although the NTK matrix is randomly initialized and time-dependent, it in fact stays close to a constant limiting kernel matrix, on which the analysis of neural networks renders much easier. Leveraging NTK, researches have extensively studied how different aspects affect the convergence of training neural networks. For example, weight initialization with large variance can accelerate the convergence but worsen the generalization ability of neural networks [19, 61]. It has been analyzed in [20] that a two-layer fully-connected neural network (FCNN) with ReLU activation provably and globally converges to zero training loss. Later the results are enriched by extending the global convergence to multi-layer (deep) FCNN, convolutional neural networks (CNN), recurrent neural networks (RNN) and residual neural networks (ResNet) [20, 3, 7, 4]. Especially, the comprehensive result in [3] covers all above-mentioned network architectures, as well as different losses such as mean square error (MSE) and cross-entropy. Specifically, the over-parameterized neural networks enjoy an exponentially decaying MSE and a polynomially decaying cross-entropy. Other examples studied the convergence with NTK under different input distribution [20, 3], activation functions [18] and normalization layers [22].

There are also a wealth of recent literature on the generalization properties of the wide neural networks based on NTK. For example, using a data-dependent Rademacher complexity measure, a generalization bound independent of network size for a two-layer, ReLU activated, FCNN can be obtained [43, 6, 2]. Several lines of work [11, 42, 17] gave spectrally-normalized margin-based generalization bound to explain the nice generalization phenomenon of over-parameterized networks. The others [23, 42, 39] derived the generalization bound from the view of PAC-Bayes and compression

approaches. More examples include the generalization bound for high dimension data [1, 26] and with weight decay [36].

While many works have been focused on the convergence theory of neural network architectures, one would argue that the efficient optimization of a neural network is as important as the design of neural networks. Nevertheless, to our knowledge, this is the first paper to go beyond GD (and SGD) and analyze the global convergence of neural networks from a more general optimization algorithm perspective. In this work, we employ some state-of-the-art optimizers and analyze their convergence behavior under the NTK regime. A related work is [37], where authors also study GD with momentum under the NTK regime but only empirically.

We seek to answer the following questions:

- Can different optimization algorithms beside GD provably find global minimum in the non-convex optimization of neural networks?
- Can we leverage the classic convex optimization theory to explain the behavior (e.g. acceleration and rate) of optimization algorithms when training the neural networks?

Intriguingly, the known results of global and linear convergence of GD (and SGD) on MSE of neural networks is somewhat surprising, since the losses are non-convex with respect to weights. Note that the traditional convex optimization theory only claims $O(1/t)$ rate of GD, even for convex losses, and we only achieve the linear convergence rate when the loss is strongly convex. This phenomenon suggests a close relationship between the non-convex optimization in neural networks and some convex optimization problems that we will elaborate in this paper. We establish such connection rigorously which allows us to conveniently employ the classic convex optimization theory at low cost, thus bridging both worlds to inspire new insights into the training of neural networks.

Once we view the evolution of neural networks during training as an ordinary differential equation (ODE), a.k.a. the gradient flow, there are plenty of fruitful results in the long history of convex optimization [47, 13, 44, 51, 14, 54]. We can consider different optimization algorithms, e.g. the Heavy Ball method (HB) [48], the Nesterov accelerated gradient descent (NAG) [41], subgradient descent, Newton's method, GD with multiple momentums [46] and so on. Each optimization algorithm has a corresponding limiting ODE (see HB ODE [48] and NAG ODE [57]), which is equivalent to the discrete optimization algorithm with an infintely small step size. To analyze such ODEs, we can apply the Gronwall's inequality [28, 12] for GD and the Lyapunov function or energy [34, 60, 49] for higher order ODE (which is incurred by employing the momentum terms [50, 52]).

Our contribution is two-fold: we show that the non-convex weight dynamics has the same form as a strongly-convex error dynamics, where Lyapunov functions are applicable; we prove that HB also enjoys global linear convergence with a rate faster than GD, and NAG may converge at a sublinear rate yet require more width than HB and GD.

2 Preliminaries

In this section, we introduce the NTK approach to analyze the convergence behavior of any neural network from a dynamical system perspective. Particularly, we warm ourselves up with some known results of training a two-layer neural network [20], using the MSE loss and the GD.

To start with, we do not specify the neural network architecture (e.g. type of layers, activation functions, depth, width, etc.) and demonstrate our approach on an *arbitrary* neural network. Given a training set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ where $\boldsymbol{x}_i \in \mathbb{R}^p$, we denote the weights $\boldsymbol{w}_r \in \mathbb{R}^p$ as the weight vectors in the first hidden layer connecting to the r-th neuron, $\boldsymbol{W}$ as the union $\{\boldsymbol{w}_r\}$ and $\boldsymbol{a}$ as the set of weights in all the other layers. We write $f(\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{x}_i)$ as the neural network output. We aim to minimize the MSE loss:

$$L(\boldsymbol{W}, \boldsymbol{a}) = \frac{1}{2}\sum_{i=1}^{n} \left(f(\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{x}_i) - y_i\right)^2$$

Taking the same route as in [20], we focus on optimizing $\boldsymbol{W}$ with $\boldsymbol{a}$ fixed at initialization[1]. Applying the simplest gradient descent with a step size η, we have:

$$\boldsymbol{w}_r(k+1) = \boldsymbol{w}_r(k) - \eta \frac{\partial L(\boldsymbol{W}(k), \boldsymbol{a})}{\partial \boldsymbol{w}_r(k)}$$

Since GD is a discretization of its corresponding ordinary differential equation (known as the gradient flow), we analyze such ODE directly as an equivalent form of GD with an infinitesimal step size. The gradient flows of different optimization algorithms are dynamical systems that are much amenable to analyze and to understand. To be specific, GD has a gradient flow as

$$\frac{d\boldsymbol{w}_r(t)}{dt} = -\frac{\partial L(\boldsymbol{W}(t), \boldsymbol{a})}{\partial \boldsymbol{w}_r(t)} \tag{2.1}$$

Remark 2.1. *Different optimization algorithms may have the same limiting ODE: it has been shown in [29] that the forward Euler discretization of* (2.1) *gives GD,*

[1] In Section 5.3 we extend to training all layers simultaneously, including the deep ones. We remark that training only the first layer can find the global minimum of loss.

while the backward Euler discretization gives the proximal point algorithm [45]. Another example relating NAG and HB can be found in Section 3.

Simple chain rules give the following dynamics,

$$\frac{d\boldsymbol{w}_r(t)}{dt} = -\frac{\partial L}{\partial \boldsymbol{f}(t)}\frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r(t)} = -(\boldsymbol{f}-\boldsymbol{y})\frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r(t)} \quad (2.2)$$

$$\frac{d\boldsymbol{f}(t)}{dt} = \sum_r \frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r(t)}\frac{d\boldsymbol{w}_r(t)}{dt} = -\boldsymbol{H}(t)(\boldsymbol{f}-\boldsymbol{y}) \quad (2.3)$$

$$\dot{\Delta}(t) = -\boldsymbol{H}(t)\Delta(t) \quad (2.4)$$

which we refer to as the *weight dynamics*, the *prediction dynamics* and the *error dynamics*, respectively. Here we denote the error of the prediction $\Delta = \boldsymbol{f} - \boldsymbol{y} \in \mathbb{R}^n$ and the $\mathbb{R}^{n\times n}$ NTK matrix as

$$\boldsymbol{H}(t) := \sum_r \frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r(t)}\left(\frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r(t)}\right)^\top$$

which is the sum of outer products.

The key observation of training the over-parameterized neural networks is that $\boldsymbol{W}(t)$ stays very close to its initialization $\boldsymbol{W}(0)$, even though the loss may change largely. This phenomenon is well-known as 'lazy training' [16, 32]. As a consequence, the neural network f is almost linear in $\boldsymbol{W}$ and the kernel $\boldsymbol{H}(t)$ behaves almost time-independently: $\lim_{m\to\infty}\boldsymbol{H}(0) \approx \boldsymbol{H}(0) \approx \boldsymbol{H}(t)$ (c.f. [20, Remark 3.1]).

Interestingly, suppose we define a pseudo-loss $\hat{L}(t) := \frac{1}{2}\Delta^\top \boldsymbol{H}\Delta$ and notice that $L = \frac{1}{2}\Delta^\top\Delta$, then optimizing the *non-convex* loss L over $\boldsymbol{w}_r$ leads to an error dynamics (2.4), as if we were actually optimizing a *strongly-convex* loss $\hat{L}$ over Δ with the same dynamical system as (2.1):

$$\frac{d\Delta(t)}{dt} = -\frac{\partial \hat{L}(\boldsymbol{W}(t), \boldsymbol{a})}{\partial \Delta(t)}.$$

The matrix ODE (2.4) with a constant and positive $\boldsymbol{H}$ has a solution converging to 0 at linear rate, as the classical theory on optimizing a strongly convex loss indicates. In other words, optimizing the non-convex L for over-parameterized neural networks converges faster than optimizing convex losses and reaches the convergence speed of optimizing strongly convex losses. Therefore it is essential to show that $\boldsymbol{H}$ is positive with the smallest eigenvalue bounded away from 0 at all time. We formalize this claim by quoting the results for the two-layer neural network of the following form,

$$f(\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{x}) = \frac{1}{\sqrt{m}}\sum_{r=1}^m a_r \sigma\left(\boldsymbol{w}_r^\top \boldsymbol{x}\right)$$

with $\sigma(\cdot)$ being the ReLU activation function. Now we quote an important fact that justifies our main theorem.

Fact 2.2 (Assumption 3.1 and Theorem 3.1 in [20])**.** *Define matrix $\boldsymbol{H}^\infty \in \mathbb{R}^{n\times n}$ with*

$$\begin{aligned}(\boldsymbol{H}^\infty)_{ij} =& \mathbb{E}_{\boldsymbol{w}_r\sim N(\boldsymbol{0},\boldsymbol{I})}\left[\boldsymbol{x}_i^\top \boldsymbol{x}_j \mathbb{I}\left\{\boldsymbol{w}_r^\top \boldsymbol{x}_i \ge 0, \boldsymbol{w}_r^\top \boldsymbol{x}_j \ge 0\right\}\right] \\ =& \left(\frac{1}{2} - \frac{\arccos(\boldsymbol{x}_i^\top \boldsymbol{x}_j)}{2\pi}\right)(\boldsymbol{x}_i^\top \boldsymbol{x}_j)\end{aligned}$$

and define $\lambda_0 := \lambda_{\min}(\boldsymbol{H}^\infty)$. Suppose for any $i \ne j$, $\boldsymbol{x}_i \nparallel \boldsymbol{x}_j$, then $\lambda_0 > 0$.

Here $\boldsymbol{H}^\infty$ is the limiting form at the initialization, i.e, $\boldsymbol{H}^\infty = \lim_{m\to\infty}\boldsymbol{H}(0)$. In [20], the authors establish that, for sufficiently wide hidden layer and under some data distributional assumptions, GD converges to zero training loss exponentially fast. Formally,

Theorem 1 (Theorem 3.2 and Lemma 3.2 in [20])**.** *Suppose $\forall i, \|\boldsymbol{x}_i\|_2 = 1$, $|y_i| < C$ for some constant C, and only the hidden layer weights $\{\boldsymbol{w}_r\}$ are optimized by GD. If we set the width of the hidden layer $m = \Omega(n^6/\lambda_0^4\delta^3)$ and we i.i.d. initialize $\boldsymbol{w}_r \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), a_r \sim \mathrm{unif}\{-1, 1\}$ for $r \in [m]$, then with high probability at least $1-\delta$ over the initialization, we have*

$$\lambda_{\min}(\boldsymbol{H}(t)) > \frac{1}{2}\lambda_{\min}(\lim_m \boldsymbol{H}(0)) := \frac{1}{2}\lambda_{\min}(\boldsymbol{H}^\infty) := \frac{\lambda_0}{2}$$

with $\boldsymbol{H}^\infty$ defined in Fact 2.2. Furthermore, we have the linear convergence

$$L(t) \le \exp(-\lambda_0 t)\, L(0).$$

We note that the NTK matrix is the Gram matrix induced by the ReLU activation:

$$\boldsymbol{H}_{ij}(t) = \sum_{r=1}^m \frac{\partial f_i(t)}{\partial \boldsymbol{w}_r}\left(\frac{\partial f_j(t)}{\partial \boldsymbol{w}_r}\right)^\top \quad (2.5)$$

$$= \frac{1}{m}\boldsymbol{x}_i^\top \boldsymbol{x}_j \sum_{r=1}^m \mathbb{I}(\boldsymbol{w}_r^\top \boldsymbol{x}_i \ge 0, \boldsymbol{w}_r^\top \boldsymbol{x}_j \ge 0) \quad (2.6)$$

We remark that the framework of Theorem 1 has been extended to training multiple layers simultaneously [20] (see also Section 5.3). The analysis has recently been generalized to different network architectures and losses and we now complement this line of research by extending to different optimization algorithms. To present the simplest proof, we only analyze the continuous gradient flows and we believe our approach can be easily extended to discrete time analysis.

3 Heavy Ball with Friction System

Our first result concerns the GD with momentum, or the Heavy Ball (HB) method [48]:

$$\begin{aligned}\boldsymbol{w}_r(k+1) =& \boldsymbol{w}_r(k) - \eta\frac{\partial L(\boldsymbol{W}(k), \boldsymbol{a})}{\partial \boldsymbol{w}_r(k)} \\ &+ \beta(\boldsymbol{w}_r(k) - \boldsymbol{w}_r(k-1))\end{aligned} \quad (3.1)$$

or equivalently

$$\begin{aligned} \boldsymbol{w}_r(k+1) &= \boldsymbol{w}_r(k) + \eta \boldsymbol{v}(k) \\ \boldsymbol{v}(k) &= -\frac{\partial L(\boldsymbol{W}(k), \boldsymbol{a})}{\partial \boldsymbol{w}_r(k)} + \beta \boldsymbol{v}(k-1) \end{aligned} \tag{3.2}$$

where η is the step size and the momentum term $\beta \in [0, 1]$. The corresponding gradient flow is known as the Heavy Ball with Friction (HBF) system. This is a non-linear dissipative dynamical system, originally proposed by [48] and heavily studied in [9, 24, 15, 8, 5, 60]: with $b > 0$

$$\ddot{\boldsymbol{w}}_r(t) + b\dot{\boldsymbol{w}}_r(t) + \frac{\partial L(\boldsymbol{W}(t), \boldsymbol{a})}{\partial \boldsymbol{w}_r(t)} = 0. \tag{3.3}$$

As shown later in Section 3.1, the error dynamics is

$$\ddot{\Delta}(t) + b\dot{\Delta}(t) + \frac{\partial \hat{L}}{\partial \Delta(t)} \overset{\text{a.s.}}{=} 0. \tag{3.4}$$

We note that other optimization algorithms may also correspond to the HBF system (3.3),(3.4): for example NAG-SC in [60][2], though NAG-SC and HB can be distinguished using high resolution ODE in [53].

In particular, we study the case as in [60, Equation (7)] and [55], when $b = \sqrt{2\lambda_0}$, i.e. twice the strongly convexity of $\hat{L}$:

$$\ddot{\boldsymbol{w}}_r(t) + \sqrt{2\lambda_0}\dot{\boldsymbol{w}}_r(t) + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}(\boldsymbol{f} - \boldsymbol{y}) = 0 \tag{3.5}$$

Our choice of parameter b leads to a global linear convergence to zero training loss, without requiring Lipschitz gradients of $\hat{L}$. For other choices of parameters together with the Lipschitz condition of $\hat{L}$, HB can enjoy linear converge locally [48, 38] and globally [41, 25, 59, 55, 10].

To solve a second order ODE requires initial conditions on $\boldsymbol{w}_r$ and $\dot{\boldsymbol{w}}_r$, which we assume as $\dot{\boldsymbol{w}}_r(0) = 0$ without loss of generality. Now we state the our main theorem under MSE loss.

Theorem 2. *Suppose we set $b = \sqrt{2\lambda_0}$ and only the hidden layer weights $\{\boldsymbol{w}_r\}$ are optimized by HB. If we set the width of the hidden layer $m = \Omega\left(\frac{n^6}{\delta^3 \lambda_0^4}\right)$ and we i.i.d. initialize $\boldsymbol{w}_r \sim \mathcal{N}(\boldsymbol{0}, \mathbf{I}), a_r \sim \text{unif}\{-1, 1\}$ for $r \in [m]$, then with high probability at least $1 - \delta$ over the initialization, we have*

$$L(t) \leq \frac{4}{\lambda_0} \exp\left(-\sqrt{\lambda_0/2} \cdot t\right) \hat{L}(0)$$

[2]We consider the following NAG-SC: assume L is $\sqrt{\lambda_0/2}$-strongly convex and $b = \sqrt{2\lambda_0}$,

$$\begin{aligned} \boldsymbol{v}(k+1) &= \boldsymbol{w}_r(k) - \eta \frac{\partial L}{\partial \boldsymbol{w}_r(k)} \\ \boldsymbol{w}_r(k+1) &= \boldsymbol{v}(k+1) + \frac{1 - \sqrt{\lambda_0 \eta/2}}{1 + \sqrt{\lambda_0 \eta/2}} (\boldsymbol{v}(k+1) - \boldsymbol{v}(k)) \end{aligned}$$

We notice that indeed $\sqrt{\lambda_0/2} > \lambda_0$, suggesting a boost in the linear convergence rate of HB when compared to GD, as we observe in Figure 1. To prove this, we claim that $\lambda_0 < 1/2$ as $\text{tr}(\boldsymbol{H}^\infty) = n/2 = \sum_i \lambda_i > n\lambda_0$. Another observation is that, to guarantee the linear convergence, HB requires the same order of width as GD in Theorem 1.

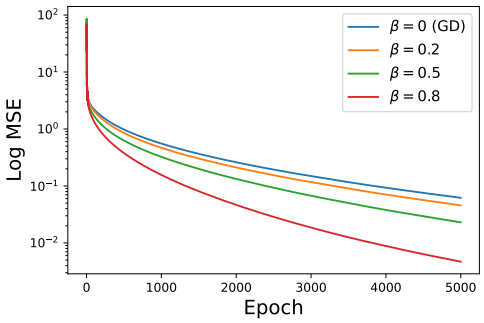

Figure 1: HB shows linear convergence with larger convergence rate than GD. Experiment details in Appendix D.1.

In the following section, we prove our theorem by employing the Lyapunov function.

3.1 Proof of Theorem 2

First, we use the chain rule to characterize the prediction dynamics of $\boldsymbol{f}$,

$$\begin{aligned} \dot{f}_i(t) &= \sum_{r \in [m]} \frac{\partial f_i}{\partial \boldsymbol{w}_r} \dot{\boldsymbol{w}}_r \\ \ddot{f}_i(t) &= \sum_{r,l \in [m]} \dot{\boldsymbol{w}}_r{}^\top \frac{\partial^2 f_i}{\partial \boldsymbol{w}_r \partial \boldsymbol{w}_l} \dot{\boldsymbol{w}}_l + \sum_{r \in [m]} \frac{\partial f_i}{\partial \boldsymbol{w}_r} \ddot{\boldsymbol{w}}_r \\ &\overset{\text{a.s.}}{=} \sum_{r \in [m]} \frac{\partial f_i}{\partial \boldsymbol{w}_r} \ddot{\boldsymbol{w}}_r \end{aligned}$$

where the last equality follows from a key observation that, with $\delta(\cdot)$ denoting the Dirac Delta function,

$$\begin{aligned} \frac{\partial f_i}{\partial \boldsymbol{w}_r} &= \frac{1}{\sqrt{m}} a_r \boldsymbol{x}_i \mathbb{I}(\boldsymbol{w}_r^\top \boldsymbol{x}_i > 0), \\ \frac{\partial^2 f_i}{\partial \boldsymbol{w}_r^2} &= \frac{1}{\sqrt{m}} a_r \boldsymbol{x}_i \boldsymbol{x}_i^\top \delta(\boldsymbol{w}_r^\top \boldsymbol{x}_i) \overset{\text{a.s.}}{=} \boldsymbol{0}, \\ \frac{\partial^2 f_i}{\partial \boldsymbol{w}_r \partial \boldsymbol{w}_l} &= \boldsymbol{0}, \quad \text{for } l \neq r. \end{aligned} \tag{3.6}$$

Multiplying $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}$ to (3.5) and sum over r, we obtain the prediction dynamics as

$$\ddot{\boldsymbol{f}}(t) + \sqrt{2\lambda_0}\dot{\boldsymbol{f}}(t) + \boldsymbol{H}(t)(\boldsymbol{f} - \boldsymbol{y}) \overset{\text{a.s.}}{=} 0$$

and consequently, the dynamics of the error is

$$\ddot{\Delta}(t) + \sqrt{2\lambda_0}\dot{\Delta}(t) + \boldsymbol{H}(t)\Delta(t) \overset{\text{a.s.}}{=} 0 \tag{3.7}$$

or in an analogous form to (3.5),

$$\ddot{\Delta}(t)+\sqrt{2\lambda_0}\dot{\Delta}(t)+\frac{\partial \hat{L}}{\partial \Delta(t)} \overset{\text{a.s.}}{=} 0$$

In what follows, we drop the *a.s.* and focus on the ordinary differential equations. To establish the linear convergence in MSE, we need to guarantee $\boldsymbol{H}(t)$ is positive definite with $\lambda_{\min}(\boldsymbol{H}(t)) \geq \lambda_0/2$. In other words, the pseudo loss $\hat{L}$ is $\frac{\lambda_0}{2}$-strongly convex. We start with $t=0$, by showing that for wide enough neural networks, $\boldsymbol{H}(0)$ has positive smallest eigenvalue with high probability.

Lemma 3.1 (Lemma 3.1 in [20]). *If* $m = \Omega\left(\frac{n^2}{\lambda_0^2}\log\left(\frac{n^2}{\delta}\right)\right)$, *then we have* $\|\boldsymbol{H}(0)-\boldsymbol{H}^{\infty}\|_2 \leq \frac{\lambda_0}{4}$ *and* $\lambda_{\min}(\boldsymbol{H}(0)) \geq \frac{3}{4}\lambda_0$ *with probability of at least* $1-\delta$.

Next, we introduce a lemma that shows for any t, if $\boldsymbol{w}_r(t)$ is close to $\boldsymbol{w}_r(0)$, then $\boldsymbol{H}(t)$ is close to $\boldsymbol{H}(0)$. Together with Lemma 3.1, $\lambda_{\min}(\boldsymbol{H}(t))$ always has a positive smallest eigenvalue. In words, the lazy training leads to the positive definiteness.

Lemma 3.2 (Lemma 3.2 in [20]). *If* $\boldsymbol{w}_r$ *are i.i.d. generated from* $\mathcal{N}(\boldsymbol{0},\mathbf{I})$ *for* $r\in[m]$, *and* $\|\boldsymbol{w}_r(0)-\boldsymbol{w}_r\|_2 \leq \frac{c\delta\lambda_0}{n^2} =: R$ *for some small positive constant* c, *then the following holds with probability at least* $1-\delta$ *we have* $\|\boldsymbol{H}(t)-\boldsymbol{H}(0)\|_2 < \frac{\lambda_0}{4}$ *and* $\lambda_{\min}(\boldsymbol{H}(t)) > \frac{\lambda_0}{2}$.

The next lemma gives two important facts given that $\lambda_{\min}(\boldsymbol{H}(s))$ for previous time $s\leq t$: the loss decays exponentially and weights stay close to their initialization at the current time t. In other words, the positive definiteness indicates the convergence, which further indicates the lazy training. We emphasize that Lemma 3.3 is specific to the choice of optimization algorithms and hence the proof is much different than its analog in [20, Lemma 3.3] for GD.

Lemma 3.3. *Assume* $0\leq s\leq t$ *and* $\lambda_{\min}(\boldsymbol{H}(s)) \geq \frac{\lambda_0}{2}$. *Then we have* $L(t) \leq \exp\left(-\sqrt{\lambda_0/2}t\right)\frac{4\hat{L}(0)}{\lambda_0}$ *and* $\|\boldsymbol{w}_r(t)-\boldsymbol{w}_r(0)\|_2 \leq \sqrt{\frac{32n\hat{L}(0)}{9m\alpha^6}} =: R'$.

Proof of Lemma 3.3. Borrowing the idea of [60, 55], we define the Lyapunov function or Lyapunov energy as

$$\begin{aligned}V(t) :=& \hat{L}+\frac{1}{2}\left\|\sqrt{\frac{\lambda_0}{2}}\Delta(t)+\dot{\Delta}(t)\right\|^2\\ =&\frac{1}{2}\Delta(t)^{\top}\boldsymbol{H}(t)\Delta(t)+\frac{1}{2}\left\|\sqrt{\frac{\lambda_0}{2}}\Delta(t)+\dot{\Delta}(t)\right\|^2.\end{aligned}$$

The Lyapunov function represents the total energy of the system and always decreases along the trajectory of the training dynamics since, as we will later show, $\dot{V}(t)<0$. Here we simplify the notation by denoting the dependence on t in the subscript and use $\alpha := b/2 = \sqrt{\lambda_0/2}$. We derive by the chain rule,

$$\begin{aligned}\dot{V}(t) =& \dot{\Delta}_t^{\top}\boldsymbol{H}(t)\Delta_t+\frac{1}{2}\Delta_t^{\top}\dot{\boldsymbol{H}}(t)\Delta_t\\ &+\left\langle \alpha\dot{\Delta}_t+\ddot{\Delta}_t, \alpha\Delta_t+\dot{\Delta}_t\right\rangle.\end{aligned}$$

Notice that by (2.6), we have $\dot{\boldsymbol{H}}(t)\overset{\text{a.s.}}{=}0$. Substituting the error dynamics (3.7) for $\ddot{\Delta}_t$, we have

$$\begin{aligned}\dot{V}(t) =& \left\langle \boldsymbol{H}\Delta_t, \dot{\Delta}_t\right\rangle+\left\langle -\alpha\dot{\Delta}_t-\boldsymbol{H}\Delta_t, \alpha\Delta_t+\dot{\Delta}_t\right\rangle\\ =&-\alpha\left\langle \boldsymbol{H}\Delta_t,\Delta_t\right\rangle-\alpha^2\left\langle\dot{\Delta}_t,\Delta_t\right\rangle-\alpha\left\langle\dot{\Delta}_t,\dot{\Delta}_t\right\rangle\end{aligned}$$

Using $\lambda_{\min}(\boldsymbol{H})\geq\lambda_0/2=\alpha^2$, we get

$$\begin{aligned}\left\langle \boldsymbol{H}\Delta_t,\Delta_t\right\rangle \geq& \frac{1}{2}\left\langle \boldsymbol{H}\Delta_t,\Delta_t\right\rangle+\frac{\alpha^2}{2}\langle\Delta_t,\Delta_t\rangle\\ =&\hat{L}(t)+\frac{\alpha^2}{2}\langle\Delta_t,\Delta_t\rangle\end{aligned}$$

and hence we have

$$\begin{aligned}\dot{V}(t) =& -\alpha\hat{L}(t)-\frac{\alpha^3}{2}\langle\Delta_t,\Delta_t\rangle\\ &-\alpha^2\left\langle\dot{\Delta}_t,\Delta_t\right\rangle-\alpha\left\langle\dot{\Delta}_t,\dot{\Delta}_t\right\rangle\\ <&-\alpha\left(\hat{L}(t)+\frac{1}{2}\left\|\alpha\Delta_t+\dot{\Delta}_t\right\|^2\right)=-\alpha V(t)\end{aligned}$$

where in the last inequality we throw away $-\frac{\alpha}{2}\left\langle\dot{\Delta}_t,\dot{\Delta}_t\right\rangle$. Clearly $\dot{V}(t)<0$ for all t. For this first order scalar ODE, we apply the Gronwall's inequality to derive

$$V(t)<e^{-\alpha t}V(0)$$

and we obtain

$$\begin{aligned}\hat{L}(t)\leq V(t) &< e^{-\alpha t}V(0)\\ &=e^{-\alpha t}\left(\frac{1}{2}\Delta(0)^{\top}\boldsymbol{H}(0)\Delta(0)+\frac{\alpha^2}{2}\|\Delta(0)\|^2\right).\end{aligned}$$

Again using $\lambda_{\min}(\boldsymbol{H}(0))\geq\alpha^2$, we have

$$\begin{aligned}\hat{L}(t) &\leq \exp\left(-\alpha\cdot t\right)\left(\frac{1}{2}\Delta(0)^{\top}\boldsymbol{H}(0)\Delta(0)+\hat{L}(0)\right)\\ &=2\exp\left(-\alpha\cdot t\right)\hat{L}(0)\end{aligned}$$

and

$$L(t)\leq\frac{2}{\alpha^2}\exp\left(-\alpha t\right)\hat{L}(0).$$

In words, the prediction $\boldsymbol{f}(t)\to\mathbf{y}$ exponentially fast, with a convergence factor $\alpha=\sqrt{\lambda_0/2}$.

Now we move on to show that $\boldsymbol{w}_r(t)$ stays close to $\boldsymbol{w}_r(0)$. Multiplying $e^{bt}=e^{2\alpha t}$ to the weight dynamics (3.5), we have

$$\frac{d}{dt}\left(e^{2\alpha t}\dot{\boldsymbol{w}}_r\right)=-\frac{1}{\sqrt{m}}e^{2\alpha t}a_r\sum_i(f_i-y_i)\boldsymbol{x}_i\mathbb{I}(\boldsymbol{w}_r^{\top}\boldsymbol{x}_i\geq 0)$$

which gives a close-form solution

$$\dot{\boldsymbol{w}}_r = -e^{-2\alpha t}\int_0^t \frac{1}{\sqrt{m}} e^{2\alpha s} a_r \sum_i (f_i - y_i)\boldsymbol{x}_i \mathbb{I}(\boldsymbol{w}_r^\top \boldsymbol{x}_i \geq 0) ds$$

whose norm satisfies

$$\begin{aligned}
\|\dot{\boldsymbol{w}}_r\| &\leq e^{-2\alpha t}\frac{1}{\sqrt{m}}\int_0^t e^{2\alpha s}\sum_i |f_i(s) - y_i| ds \\
&\leq e^{-2\alpha t}\sqrt{\frac{n}{m}}\int_0^t e^{2\alpha s}\|f(s) - \boldsymbol{y}\|_2 ds \\
&= \sqrt{\frac{n}{m}}\int_0^t e^{2\alpha(s-t)}\sqrt{L(s)} ds \\
&\leq \sqrt{\frac{2n\hat{L}(0)}{m\alpha^2}}\int_0^t e^{\frac{3}{2}\alpha s - 2\alpha t} ds \\
&\leq \sqrt{\frac{8n\hat{L}(0)}{9m\alpha^4}} e^{-\frac{\alpha}{2}t}
\end{aligned}$$

Finally by Cauchy Schwarz inequality, we bound the weight distance from initialization,

$$\|\boldsymbol{w}_r(t) - \boldsymbol{w}_r(0)\|_2 \leq \int_0^t \|\dot{\boldsymbol{w}}_r(s)\|_2 ds < \sqrt{\frac{32n\hat{L}(0)}{9m\alpha^6}}$$

□

We quote the next lemma to show that $R'(t) < R$ indicates that for all $t > 0$, the conditions in Lemma 3.2 and Lemma 3.3 hold. This lemma is closely related to [20, Lemma 3.4] and the proof is given in Appendix C.

Lemma 3.4. *If $R' < R$, then we have $\lambda_{\min}(\boldsymbol{H}(t)) \geq \frac{\lambda_0}{2}$, for all $r \in [m]$, $\|\boldsymbol{w}_r(t) - \boldsymbol{w}_r(0)\|_2 \leq R'$ and $L(t) \leq \frac{4}{\lambda_0}\exp\left(-\sqrt{\lambda_0/2}t\right)\hat{L}(0)$.*

Finally we study the width requirement for $R' < R$ to hold true, i.e. we need $\sqrt{\frac{32n\hat{L}(0)}{9m\alpha^6}} < O(\frac{\delta\lambda_0}{n^2})$ which is equivalent to $m = \Omega(\frac{n^6}{\delta^3\lambda_0^4})$, shown in Appendix A.1.

4 Nesterov Accelerated Gradient Descent

In this section, we analyze the dynamics of the generalized Nesterov Accelerated Gradient (NAG) descent as follows,

$$\begin{aligned}
\boldsymbol{v}(k+1) &= \boldsymbol{w}_r(k) - \eta\frac{\partial L(\boldsymbol{W}(k), \boldsymbol{a})}{\partial \boldsymbol{w}_r(k)} \\
\boldsymbol{w}_r(k+1) &= \boldsymbol{v}(k+1) + \frac{k-1}{k+\gamma-1}(\boldsymbol{v}(k+1) - \boldsymbol{v}(k))
\end{aligned} \tag{4.1}$$

where η is step size. We remark that the NAG (4.1) has a time-dependent momentum coefficient $\frac{k-1}{k+\gamma-1}$ while in practice, e.g. in [58] (adopted in Pytorch and Tensorflow) and in Footnote 2, a time-independent momentum is commonly used and may lead to linear convergence. It has been given in [57] that the corresponding gradient flow as

$$\ddot{\boldsymbol{w}}_r(t) + \frac{\gamma}{t}\dot{\boldsymbol{w}}_r(t) + \frac{\partial L(\boldsymbol{W}(t), \boldsymbol{a})}{\partial \boldsymbol{w}_r(t)} = 0 \tag{4.2}$$

with initial conditions $\dot{\boldsymbol{w}}_r(0) = 0$. It follows that the prediction dynamics and the error dynamics are

$$\ddot{\boldsymbol{f}}(t) + \frac{\gamma}{t}\dot{\boldsymbol{f}}(t) + \boldsymbol{H}(t)(\boldsymbol{f} - \boldsymbol{y}) \overset{\text{a.s.}}{=} 0 \tag{4.3}$$

$$\ddot{\Delta}(t) + \frac{\gamma}{t}\dot{\Delta}(t) + \boldsymbol{H}(t)\Delta(t) \overset{\text{a.s.}}{=} 0 \tag{4.4}$$

with the same NTK matrix $\boldsymbol{H}(t)$ as defined in (2.6) Again, using $\hat{L}(t)$, we have an error dynamics that is analogous to (4.2),

$$\ddot{\Delta}(t) + \frac{\gamma}{t}\dot{\Delta}(t) + \frac{\partial \hat{L}}{\partial \Delta(t)} \overset{\text{a.s.}}{=} 0 \tag{4.5}$$

Theorem 3. *Suppose we set $4 < \alpha \leq \frac{2\gamma}{3}$ and $\gamma > 6$ and only the hidden layer weights $\{\boldsymbol{w}_r\}$ are optimized by NAG. If we set the width of the hidden layer $m = \Omega\left(\frac{n^{5\alpha/2-4}}{\delta^{3\alpha/2-3}\lambda_0^{3\alpha/2-2}}\right)$ and we i.i.d. initialize $\boldsymbol{w}_r \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), a_r \sim \text{unif}\{-1, 1\}$ for $r \in [m]$, then with high probability at least $1 - \delta$ over the initialization, we have*

$$L(t) \leq A(\alpha, \gamma, \lambda_0)t^{-\alpha}L(0) \tag{4.6}$$

where $A(\alpha, \gamma, \lambda_0) \in \mathbb{R}$ is defined in (4.7) and only depends on α, γ and λ_0.

We pause here to discuss the choice of γ in (4.1). In [57], the 'magic constant' γ has been extensively studied. When $\gamma \geq 3$, the convergence rate is shown to be $O(t^{-\frac{2\gamma}{3}})$. When $\gamma < 3$, there exist counter-examples that fail the desired $O(1/t^2)$ convergence rate. As we can visualize in Figure 3, γ provides a trade-off between the size of neural network and the efficiency of optimization. In addition, we remark that we only need $\gamma > 3$ to derive the convergence but we assume $\gamma > 6$ only to guarantee the lazy training $\boldsymbol{w}_r(t) \approx \boldsymbol{w}_r(0)$.

From Theorem 3, NAG only converges at polynomial rate, in contrast to the linear convergence of HB and GD. This can be visualized as the linear pattern in Figure 2 (note the GD pattern is concave). Therefore, in the long run when t is sufficiently large, NAG may be outperformed by GD. In addition, NAG and HB are well-known to have non-monotone loss dynamics, which is different than GD.

To prove Theorem 3, we use the same framework as in Section 3: given Lemma 3.1 and Lemma 3.2, we will prove Lemma 4.1 as an analogy to Lemma 3.3, but customized for NAG.

Lemma 4.1. *Assume $0 \leq s \leq t$ and $\lambda_{\min}(\boldsymbol{H}(s)) \geq \frac{\lambda_0}{2}$, then we have $L(t) \leq \frac{C(\alpha,\gamma)}{t^\alpha(\lambda_0/2)^{\frac{\alpha}{2}}}L(0)$ for $2 \leq \alpha \leq \frac{2}{3}\gamma$ and*

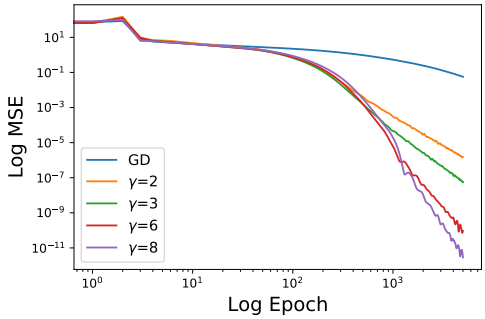

Figure 2: NAG shows polynomial decay in loss but converges faster than GD. Experiment details in Appendix D.2.

some $C(\alpha,\gamma)$ only depending on α and γ. Furthermore, if $\alpha>4$, we have

$$\begin{aligned}&\|\boldsymbol{w}_r(t)-\boldsymbol{w}_r(0)\|_2\le R'(\epsilon)\\:=&\frac{2\epsilon^{2-\alpha/2}}{\alpha-4}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}(\gamma-\alpha/2+1)^2}}+\sqrt{2L(0)}\epsilon.\end{aligned}$$

Proof of Lemma 4.1. We define the Lyapunov function as in [57]

$$V(t;\alpha,\gamma):=t^\alpha\hat{L}(t)+\frac{(2\gamma-\alpha)^2t^{\alpha-2}}{8}\left\|\Delta_t+\frac{2t}{2\gamma-\alpha}\dot{\Delta}_t\right\|^2$$

and use the same loss L and pseudo-loss $\hat{L}$ as before.

For $\gamma>3$ and $2\le\alpha\le\frac{2\gamma}{3}$, we apply from [57, Theorem 8]: with some $C(\alpha,\gamma)$ only depending on α and γ,

$$\hat{L}(t)\le\frac{C(\alpha,\gamma)}{t^\alpha(\lambda_0/2)^{\frac{\alpha-2}{2}}}L(0):=\frac{A(\alpha,\gamma,\lambda_0)}{t^\alpha(\lambda_0/2)^{-1}}L(0).\quad(4.7)$$

Leveraging $\lambda_{\min}(\boldsymbol{H})\ge\frac{\lambda_0}{2}$, we have $\hat{L}(t)\ge\frac{\lambda_0}{2}L(t)$ and

$$L(t)\le\frac{2}{\lambda_0}\hat{L}(t)\le\frac{C(\alpha,\gamma)}{t^\alpha(\lambda_0/2)^{\frac{\alpha}{2}}}L(0).$$

As for the lazy training, since the weight dynamics is

$$\ddot{\boldsymbol{w}}_r(t)+\frac{\gamma}{t}\dot{\boldsymbol{w}}_r(t)+\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{w}_r}(\boldsymbol{f}-\boldsymbol{y})=0.$$

Multiplying both sides with t^γ, we obtain

$$\begin{aligned}&\frac{d}{dt}(t^\gamma\dot{\boldsymbol{w}}_r(t))=-t^\gamma\frac{\partial\boldsymbol{f}}{\partial\boldsymbol{w}_r}(\boldsymbol{f}-\boldsymbol{y})\\=&-\frac{1}{\sqrt{m}}t^\gamma a_r\sum_i(f_i-y_i)\boldsymbol{x}_i\mathbb{I}(\boldsymbol{w}_r^\top\boldsymbol{x}_i\ge0)\end{aligned}$$

which gives a close-form solution

$$\dot{\boldsymbol{w}}_r=-\frac{1}{t^\gamma}\int_0^t\frac{1}{\sqrt{m}}s^\gamma a_r\sum_i(f_i-y_i)\boldsymbol{x}_i\mathbb{I}(\boldsymbol{w}_r^\top\boldsymbol{x}_i\ge0)ds$$

whose norm satisfies

$$\begin{aligned}\|\dot{\boldsymbol{w}}_r\|_2&\le\frac{1}{t^\gamma\sqrt{m}}\int_0^t s^\gamma\sum_i|f_i(s)-y_i|ds\\&\le\frac{1}{t^\gamma}\sqrt{\frac{n}{m}}\int_0^t s^\gamma\|\boldsymbol{f}(s)-\boldsymbol{y}\|_2ds\\&=\frac{1}{t^\gamma}\sqrt{\frac{n}{m}}\int_0^t s^\gamma\sqrt{L(s)}ds\\&\le\frac{1}{t^\gamma}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}}}\int_0^t s^{\gamma-\alpha/2}ds\\&=t^{1-\alpha/2}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}(\gamma-\alpha/2+1)^2}}.\end{aligned}$$

Next we bound the weight distance. For any $0<\epsilon<t$, we break the integral into two pieces.

$$\begin{aligned}\|\boldsymbol{w}_r(t)-\boldsymbol{w}_r(0)\|_2&\le\int_0^t\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds\\&=\int_\epsilon^t\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds+\int_0^\epsilon\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds\\&\le\frac{2\epsilon^{2-\alpha/2}}{\alpha-4}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}(\gamma-\alpha/2+1)^2}}\\&+\int_0^\epsilon\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds\end{aligned}$$

Now we need to bound $\|\dot{\boldsymbol{w}}_r(s)\|_2$ in $\int_0^\epsilon\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds$ with a time-independent upper bound, different than the previous $O(t^{1-\alpha/2})$ one. We achieve this goal by analyzing another Lyapunov function from [49]

$$\mathcal{E}(t)=L(\boldsymbol{W}(t),\boldsymbol{a})+\frac{1}{2}\sum_r\dot{\boldsymbol{w}}_r(t)^\top\dot{\boldsymbol{w}}_r(t)$$

By simple differentiation, we see $\mathcal{E}$ is decreasing in t: $\dot{\mathcal{E}}(t)=-\sum_r\frac{\gamma}{t}\|\dot{\boldsymbol{w}}_r(t)\|_2^2\le0$. This implies that

$$\frac{1}{2}\sum_r\dot{\boldsymbol{w}}_r(t)^\top\dot{\boldsymbol{w}}_r(t)\le\mathcal{E}(t)\le\mathcal{E}(0)=L(0).$$

Hence we obtain from $\|\dot{\boldsymbol{w}}_r(s)\|^2\le\sum_r\|\dot{\boldsymbol{w}}_r(s)\|^2$ that

$$\int_0^\epsilon\|\dot{\boldsymbol{w}}_r(s)\|_2\,ds\le\sqrt{2L(0)}\epsilon.$$

Therefore, for sufficiently small ϵ and sufficiently large m, we have

$$\begin{aligned}&\|\boldsymbol{w}_r(t)-\boldsymbol{w}_r(0)\|_2\le R'(\epsilon)\\:=&\frac{2\epsilon^{2-\alpha/2}}{\alpha-4}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}(\gamma-\alpha/2+1)^2}}+\sqrt{2L(0)}\epsilon\end{aligned}$$

□

Lastly, we give Lemma 4.2 in analogy to Lemma 3.4, in order to show that if $R'(\epsilon)<R$, then the conditions in Lemma 3.1, Lemma 3.2 hold.

Lemma 4.2. *If $R' < R$, then we have $\lambda_{\min}(\boldsymbol{H}(t)) \geq \frac{\lambda_0}{2}$, for all $r \in [m]$, $\|w_r(t) - w_r(0)\|_2 \leq R'$ and $L(t) \leq t^{-\alpha} C(\alpha, \gamma)(\lambda_0/2)^{-\alpha/2} L(0)$, for $4 < \alpha \leq \frac{2\gamma}{3}$ and $\gamma > 6$.*

The width requirement for $R' < R$ is equivalent to $\frac{2\epsilon^{2-\alpha/2}}{\alpha-4}\sqrt{\frac{nC(\alpha,\gamma)L(0)}{m(\lambda_0/2)^{\frac{\alpha}{2}}(\gamma-\alpha/2+1)^2}} + \sqrt{2L(0)}\epsilon < O(\frac{\delta\lambda_0}{n^2})$ for a sufficiently small fixed $\epsilon > 0$. We show the details in Appendix A.2 that it suffices to use $m = \Omega\left(\frac{n^{5\alpha/2-4}}{\delta^{3\alpha/2-3}\lambda_0^{3\alpha/2-2}}\right)$. We note this width lower bound is larger than the width required by GD in [20] and our HB analysis, suggesting that smaller γ or α may be preferred for NAG (see Figure 3), since the width requirement is increasing in α.

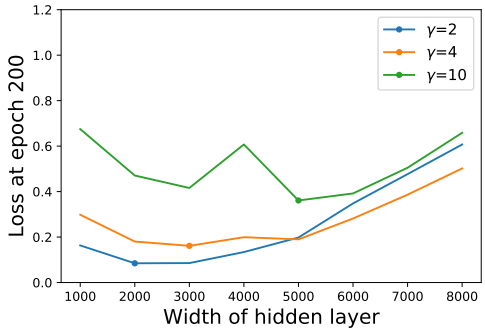

Figure 3: NAG may require wider layer to converge at optimal rate. The dots represent minimum MSE for each case. The loss is averaged of 10 runs. Experiment details in Appendix D.2.

5 Notable Extensions

The limiting ODEs of different optimization algorithms give rise to a general framework of analyzing the strongly-convex pseudo-loss $\hat{L}$. We further extend the analysis in three major directions: higher order optimization algorithms, higher order ODEs and deeper neural networks.

5.1 Higher order momentum

We consider the convergence behavior of higher order ODE, resulting from using multiple momentums in GD, as an extension of HB.

$$\boldsymbol{w}_r(k+1) = \boldsymbol{w}_r(k) - \eta \frac{\partial L(\boldsymbol{W}(k), \boldsymbol{a})}{\partial \boldsymbol{w}_r(k)} + \sum_{j=[J]} \beta_j (\boldsymbol{w}_r(k-j+1) - \boldsymbol{w}_r(k-j))$$

which reduces to HB when $J = 1$ and to GD when $J = 0$. Consequently, the gradient flow is a $(J+1)$-th order ODE,

$$\frac{d^{J+1}}{dt^{J+1}}\boldsymbol{w}_r(t) + \sum_{j\in[J]} b_j \frac{d^j}{dt^j}\boldsymbol{w}_r(t) + \frac{\partial L(\boldsymbol{W}(t), \boldsymbol{a})}{\partial \boldsymbol{w}_r(t)} = 0.$$

By similar argument as in (3.6), we have $\frac{\partial^j \boldsymbol{f}}{\partial \boldsymbol{W}^j} \overset{\text{a.s.}}{=} 0$ for $j > 1$, hence the error dynamics follows the same form as the weight dynamics, except L is replaced by $\hat{L}$,

$$\frac{d^{J+1}}{dt^{J+1}}\Delta(t) + \sum_{j\in[J]} b_j \frac{d^j}{dt^j}\Delta(t) + \frac{\partial \hat{L}}{\partial \Delta(t)} = 0. \quad (5.1)$$

To directly analyze this high order ODE is difficult. For the special case of $J = 2$, a second momentum has been shown to further accelerate the convergence than the first momentum [46], when the loss is a positive quadratic form. Empirically, we observe that, although L is not convex nor quadratic, employing the second momentum converges slightly faster as well.

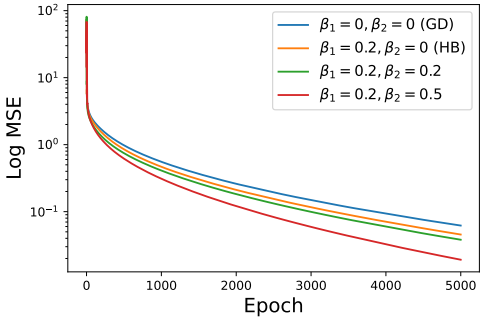

Figure 4: Higher momentum method allows faster convergence rate than GD and HB. Experiment details in Appendix D.3.

5.2 Newton's method

By the definition of Newton-Ralphson method, we have

$$\boldsymbol{w}_r(k+1) = \boldsymbol{w}_r(k) - \nabla^2 L(\boldsymbol{W}(k), \boldsymbol{a})^{-1} \nabla L(\boldsymbol{W}(k), \boldsymbol{a})$$

We observe by the chain rule (c.f. (3.6)) that

$$\frac{\partial^2 L}{\partial \boldsymbol{w}_r^2} = \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r} + \frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{w}_r^2}(\boldsymbol{f} - \boldsymbol{y}) \overset{\text{a.s.}}{=} \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r} \in \mathbb{R}^{p\times p}.$$

The corresponding gradient flow is then

$$\dot{\boldsymbol{w}}_r = -\left(\frac{\partial^2 L}{\partial \boldsymbol{w}_r^2}\right)^{-1} \frac{\partial L}{\partial \boldsymbol{w}_r} = -\left(\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^{-1} \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top (\boldsymbol{f} - \boldsymbol{y})$$

The error dynamics can be obtained by left multiplying $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}$ and sum over r:

$$\dot{\Delta}(t) \overset{\text{a.s.}}{=} -\sum_r \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\left((\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r})^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^{-1}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \Delta(t)$$

We can define the pseudo-loss

$$\hat{L} = \frac{1}{2}\Delta^\top\Big(\sum_r \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\left((\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r})^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^{-1}(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r})^\top\Big)\Delta$$

so that the error dynamic is simply $\dot{\Delta} = -\frac{\partial \hat{L}}{\partial \Delta}$.

After some linear algebra calculation, we claim that the minimum eigenvalue of $\sum_r \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\left(\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^{-1}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top$ is lower bounded by $\frac{\lambda_{\min}(\boldsymbol{H}(t))}{\max_r \lambda_{\max}\left(\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)}$. Finally, using the fact that

$$\lambda_{\max}\left(\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right) = \lambda_{\max}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top\right)$$
$$\leq \mathrm{tr}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}_r}\right)^\top\right) = \sum_{i=1}^n \frac{1}{m}\mathbb{I}\left(\boldsymbol{w}_r(t)^\top \boldsymbol{x}_i \geq 0\right) \leq \frac{n}{m},$$

and assuming that $\lambda_{\min}(\boldsymbol{H}) \geq \lambda_0$, we obtain the convergence factor as $\frac{\lambda_0 m}{n}$. This shows that the convergence of Newton's method is significantly faster than that of HB (rate $\sqrt{\lambda_0/2}$) and GD (rate λ_0), as a result of $m \gg n$. A rigorous theorem of convergence of the Newton's method, including the width requirement and the positivity of $\boldsymbol{H}$ at all t, will be investigated in the future work.

5.3 Multi-layer Training

Though we do not dive into the detailed proof of extending our main theorem to multi-layer neural network, we still provide sketch proof direction as follows.

On high level, we can write the training dynamics for the deep fully-connected neural networks as

$$\boldsymbol{x}_r^{(l)} = \frac{1}{\sqrt{m}}\sigma\left((\boldsymbol{w}_r^{(l)})^\top \boldsymbol{x}^{(l-1)}\right)$$
$$f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{a}) = \boldsymbol{a}^\top \boldsymbol{x}^{\mathcal{L}}$$

in which $l \in [\mathcal{L}-1], \boldsymbol{x}^{(0)} = \boldsymbol{x}$, $\boldsymbol{x}_r^{(l)}$ is the r-th element in l-th layer, $\boldsymbol{W} = \{\boldsymbol{w}_r^{(1)}, \cdots, \boldsymbol{w}_r^{(\mathcal{L}-1)}, \boldsymbol{a}\}$. Similar to (2.2)-(2.4), we again have the HB and NAG limiting ODEs

$$\ddot{\Delta}(t) + b(t)\dot{\Delta}(t) + \boldsymbol{H}(t)\Delta(t) = 0$$

where the NTK is $\sum_l \boldsymbol{H}_l(t) + \boldsymbol{H}_a(t)$ with $\boldsymbol{H}_l(t) = \sum_r \frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r^{(l)}(t)}\left(\frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{w}_r^{(l)}(t)}\right)^\top$ and $\boldsymbol{H}_a(t) = \frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{a}(t)}\left(\frac{\partial \boldsymbol{f}(t)}{\partial \boldsymbol{a}(t)}\right)^\top$. Clearly $\boldsymbol{H}_l$ and $\boldsymbol{H}_a$ are positive semi-definite. Therefore it suffices to show there exists at least one $l \in [\mathcal{L}-1]$ that is positive definite. We note that [19] has shown $\boldsymbol{H}_{\mathcal{L}-1}$ is positive definite for GD and we believe similar analysis applies to other optimization algorithms such as HB and NAG.

6 Discussion

In this paper, we extend the convergence analysis of over-parameterized neural networks to different accelerated optimization algorithms, including the Heavy Ball method (HB) and the Nesterov accelerated gradient descent (NAG). Our analysis is based on the neural tangent kernel (NTK) which characterizes the training dynamics as ODEs known as the gradient flows. We observe from (3.6) that, for *piecewise linear* activation functions (e.g. ReLU, Leaky ReLU [40] and maxout [27, 30]), the weight dynamics takes the same form as the error dynamics (for example, (3.3) and (3.4)), with the only difference lying in the losses. In particular, the loss in the error dynamics is strongly-convex, leading to linear convergence of HB and GD, and polynomial decay or linear convergence of NAG (depending on whether the momentum is time-dependent or not). We emphasize that by constructing the strongly-convex loss $\hat{L}$, we can easily borrow the rich results from the convex optimization world to analyze the convergence of neural networks on non-convex loss. We remark that instead of using the Gronwall inequality, as in the case of GD [20], we use the Lyapunov functions, which are common and traditional tools in solving ODEs with convex losses. In fact, the Gronwall inequality does not work on second order ODE.

A major extension of our single-layer training process is to training all layers in deep fully-connected neural networks, which has been well-established for GD in [3, 19]. Since the main focus of this work is on the optimization algorithms, we do not over-complicate our proof by exploring deeper learning (including two-layer training) but leave the discussion in Section 5.3. We also emphasize that the error dynamics of GD/HB/NAG, i.e. the limiting ODEs, are indeed compatible to arbitrary neural networks such as CNN and ResNet. To extend our analysis only requires a different route to guarantee $\lambda_{\min}(\boldsymbol{H}(t)) > 0$.

We pause to discuss the width requirement in NTK regime. Unfortunately, existing works generally require unrealistic width (see the summary in [63]): n^{24} is required in [3] and n^8 in [63], both comparable to our $\frac{n^6}{\lambda_0(n)^4}$. Hence we do not attempt to reduce our width with any complicated method. Nevertheless, all our experiments use discrete gradient descents and realistic widths, ranging from 10^3 to 10^4 hidden neurons. This is detailed in Appendix D.

Another important future direction is to study the adaptive optimizers such as AdaGrad[21], AdaDelta[62], RMSprop[31], Adam[33], DIN[5] and their variants with momentums and mini-batches. These optimizers are expected to converge faster than those analyzed in this work, but they may require a different framework to analyze. Notably, optimization algorithms with adaptive learning rate may correspond to a system of ODEs [56], instead of a single ODE.

References

[1] Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. "High-dimensional dynamics of generalization error in neural networks". In: *Neural Networks* (2020).

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. "Learning and generalization in overparameterized neural networks, going beyond two layers". In: *Advances in neural information processing systems*. 2019, pp. 6158–6169.

[3] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. "A convergence theory for deep learning via over-parameterization". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 242–252.

[4] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. "On the convergence rate of training recurrent neural networks". In: *Advances in neural information processing systems*. 2019, pp. 6676–6688.

[5] F Alvarez et al. "A second-order gradient-like dissipative dynamical system with Hessian-driven damping.-Application to optimization and mechanics". In: *Journal de mathematiques pures et appliquees* 8.81 (2002), pp. 747–779.

[6] Sanjeev Arora et al. "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks". In: *arXiv preprint arXiv:1901.08584* (2019).

[7] Sanjeev Arora et al. "On exact computation with an infinitely wide neural net". In: *Advances in Neural Information Processing Systems*. 2019, pp. 8141–8150.

[8] Hedy Attouch and Felipe Alvarez. "The heavy ball with friction dynamical system for convex constrained minimization problems". In: *Optimization*. Springer, 2000, pp. 25–35.

[9] Hedy Attouch, Xavier Goudou, and Patrick Redont. "The heavy ball with friction method, I. The continuous dynamical system: global exploration of the local minima of a real-valued function by asymptotic analysis of a dissipative dynamical system". In: *Communications in Contemporary Mathematics* 2.01 (2000), pp. 1–34.

[10] Jean-François Aujol, Charles Dossal, and Aude Rondepierre. "Convergence rates of the Heavy-Ball method for quasi-strongly convex optimization". In: (2020).

[11] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. "Spectrally-normalized margin bounds for neural networks". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6240–6249.

[12] Richard Bellman et al. "The stability of solutions of linear differential equations". In: *Duke Mathematical Journal* 10.4 (1943), pp. 643–647.

[13] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[14] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[15] Alexandre Cabot, Hans Engler, and Sébastien Gadat. "On the long time behavior of second order differential equations with asymptotically small dissipation". In: *Transactions of the American Mathematical Society* 361.11 (2009), pp. 5983–6017.

[16] Lenaic Chizat, Edouard Oyallon, and Francis Bach. "On lazy training in differentiable programming". In: *Advances in Neural Information Processing Systems*. 2019, pp. 2937–2947.

[17] Moustapha Cisse et al. "Parseval networks: Improving robustness to adversarial examples". In: *arXiv preprint arXiv:1704.08847* (2017).

[18] Simon S Du and Jason D Lee. "On the power of over-parametrization in neural networks with quadratic activation". In: *arXiv preprint arXiv:1803.01206* (2018).

[19] Simon S Du et al. "Gradient Descent Finds Global Minima of Deep Neural Networks". In: ().

[20] Simon S Du et al. "Gradient descent provably optimizes over-parameterized neural networks". In: *arXiv preprint arXiv:1810.02054* (2018).

[21] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of machine learning research* 12.7 (2011).

[22] Yonatan Dukler, Quanquan Gu, and Guido Montúfar. "Optimization theory for relu neural networks trained with normalization layers". In: *arXiv preprint arXiv:2006.06878* (2020).

[23] Gintare Karolina Dziugaite and Daniel M Roy. "Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data". In: *arXiv preprint arXiv:1703.11008* (2017).

[24] Sébastien Gadat, Fabien Panloup, Sofiane Saadane, et al. "Stochastic heavy ball". In: *Electronic Journal of Statistics* 12.1 (2018), pp. 461–529.

[25] Euhanna Ghadimi, Hamid Reza Feyzmahdavian, and Mikael Johansson. "Global convergence of the heavy-ball method for convex optimization". In: *2015 European control conference (ECC)*. IEEE. 2015, pp. 310–315.

[26] Behrooz Ghorbani et al. "Linearized two-layers neural networks in high dimension". In: *arXiv preprint arXiv:1904.12191* (2019).

[27] Ian Goodfellow et al. "Maxout networks". In: *International conference on machine learning*. PMLR. 2013, pp. 1319–1327.

[28] Thomas Hakon Gronwall. "Note on the derivatives with respect to a parameter of the solutions of a system of differential equations". In: *Annals of Mathematics* (1919), pp. 292–296.

[29] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*. Vol. 31. Springer Science & Business Media, 2006.

[30] Kaiming He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[31] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent". In: ().

[32] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems*. 2018, pp. 8571–8580.

[33] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[34] Joseph La Salle and Solomon Lefschetz. *Stability by Liapunov's Direct Method with Applications by Joseph L Salle and Solomon Lefschetz*. Elsevier, 2012.

[35] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST handwritten digit database". In: (2010).

[36] Jaehoon Lee et al. "Finite versus infinite neural networks: an empirical study". In: *arXiv preprint arXiv:2007.15801* (2020).

[37] Jaehoon Lee et al. "Wide neural networks of any depth evolve as linear models under gradient descent". In: *Advances in neural information processing systems*. 2019, pp. 8572–8583.

[38] Laurent Lessard, Benjamin Recht, and Andrew Packard. "Analysis and design of optimization algorithms via integral quadratic constraints". In: *SIAM Journal on Optimization* 26.1 (2016), pp. 57–95.

[39] Yuanzhi Li and Yingyu Liang. "Learning overparameterized neural networks via stochastic gradient descent on structured data". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8157–8166.

[40] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[41] Yu Nesterov. "A method of solving a convex programming problem with convergence rate O (1/k2)". In: *Sov. Math. Dokl.* Vol. 27. 2.

[42] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. "A pac-bayesian approach to spectrally-normalized margin bounds for neural networks". In: *arXiv preprint arXiv:1707.09564* (2017).

[43] Behnam Neyshabur et al. "The role of over-parametrization in generalization of neural networks". In: *International Conference on Learning Representations*. 2018.

[44] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[45] Neal Parikh and Stephen Boyd. "Proximal algorithms". In: *Foundations and Trends in optimization* 1.3 (2014), pp. 127–239.

[46] Barak Pearlmutter. "Gradient descent: Second order momentum and saturating error". In: *Advances in neural information processing systems*. 1992, pp. 887–894.

[47] Boris T Polyak. "Introduction to optimization. optimization software". In: *Inc., Publications Division, New York* 1 (1987).

[48] Boris T Polyak. "Some methods of speeding up the convergence of iteration methods". In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), pp. 1–17.

[49] Boris Polyak and Pavel Shcherbakov. "Lyapunov functions: An optimization theory perspective". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 7456–7461.

[50] Ning Qian. "On the momentum term in gradient descent learning algorithms". In: *Neural networks* 12.1 (1999), pp. 145–151.

[51] Andrzej Ruszczynski. *Nonlinear optimization*. Princeton university press, 2011.

[52] Heinz Rutishauser. "Theory of gradient methods". In: *Refined iterative methods for computation of the solution and the eigenvalues of self-adjoint boundary value problems*. Springer, 1959, pp. 24–49.

[53] Bin Shi et al. "Understanding the acceleration phenomenon via high-resolution differential equations". In: *arXiv preprint arXiv:1810.08907* (2018).

[54] Naum Zuselevich Shor. *Minimization methods for non-differentiable functions*. Vol. 3. Springer Science & Business Media, 2012.

[55] Jonathan W Siegel. "Accelerated first-order methods: Differential equations and Lyapunov functions". In: *arXiv preprint arXiv:1903.05671* (2019).

[56] André Belotto da Silva and Maxime Gazeau. "A General System of Differential Equations to Model First-Order Adaptive Algorithms". In: *Journal of Machine Learning Research* 21.129 (2020), pp. 1–42.

[57] Weijie Su, Stephen Boyd, and Emmanuel Candes. "A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights". In: *Advances in neural information processing systems*. 2014, pp. 2510–2518.

[58] Ilya Sutskever et al. "On the importance of initialization and momentum in deep learning". In: *International conference on machine learning*. PMLR. 2013, pp. 1139–1147.

[59] Bryan Van Scoy, Randy A Freeman, and Kevin M Lynch. "The fastest known globally convergent first-order method for minimizing strongly convex functions". In: *IEEE Control Systems Letters* 2.1 (2017), pp. 49–54.

[60] Ashia C Wilson, Benjamin Recht, and Michael I Jordan. "A lyapunov analysis of momentum methods in optimization". In: *arXiv preprint arXiv:1611.02635* (2016).

[61] Lechao Xiao, Jeffrey Pennington, and Samuel S Schoenholz. "Disentangling Trainability and Generalization in Deep Neural Networks". In: ().

[62] Matthew D Zeiler. "Adadelta: an adaptive learning rate method". In: *arXiv preprint arXiv:1212.5701* (2012).

[63] Difan Zou and Quanquan Gu. "An Improved Analysis of Training Over-parameterized Deep Neural Networks". In: *Advances in neural information processing systems* (2019).