Benign Oscillation within Minimal Invariant Subspaces at the Edge of Stability

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this work, we provide a fine-grained analysis of the training dynamics of weight matrices with a large learning rate η, commonly used in machine learning practice for improved empirical performance. This regime is also known as the edge of stability, where sharpness hovers around $2/\eta$, and the training loss oscillates yet decreases over long timescales. Within this regime, we observe an intriguing phenomenon: the oscillations in the training loss are artifacts of the oscillations of *only* a few leading singular values of the weight matrices within a small invariant subspace. Theoretically, we analyze this behavior based on a simplified deep matrix factorization problem, showing that this oscillation behavior closely follows that of its nonlinear counterparts. We provably show that for η within a specific range, the oscillations occur within a 2-period fixed orbit of the singular values, while the singular vectors remain invariant across all iterations. We extensively corroborate our theory with empirical justifications, namely in that (i) deep linear and nonlinear networks share many properties in their learning dynamics and (ii) our model captures the nuances that occur at the edge of stability which other models do not, providing deeper insights into this phenomenon.

1 Introduction

Deep neural networks have demonstrated remarkable performance across various applications [1]. Despite being heavily overparameterized, deep learning models generalize effectively well in practice, seemingly contradicting traditional statistical learning theory [2, 3]. Over the past decade, there has been an abundance of research devoted to understanding this phenomenon, with a key revelation being the implicit bias inherent in the optimizer used to train the network towards "simple" solutions [4–7]. For example, a line of work has shown that gradient descent (GD) learns simple functions [8, 9], while others suggest that GD exhibits a bias towards low-rank solutions [6, 10, 11].

More recently, there has been increasing interest in understanding how the learning rate plays a role in the learning dynamics [12–17]. One important observation within this line of research is that large learning rates improve both training efficiency and generalization [12, 13]. From an optimization perspective, the effect of large learning rates can be categorized into two related behaviors: (i) "edge of stability", where the sharpness of the network continually rises and then hovers just near $2/\eta$, where $\eta > 0$ is the learning rate [16]; (ii) "benign oscillation", where oscillations in the training loss have been shown to improve generalization compared to those with small learning rates [13]. The main hypothesis behind the benefits of large learning rates is that a large learning rate can potentially drive networks out of sharper minima to land in flatter minima within highly non-convex landscapes. It is a popular belief that among all possible minima, the flattest minima are directly correlated with better generalization [18–21]. Due to the profound implications of these phenomena, many works have been dedicated to understanding when and why they occur. However, many of the existing

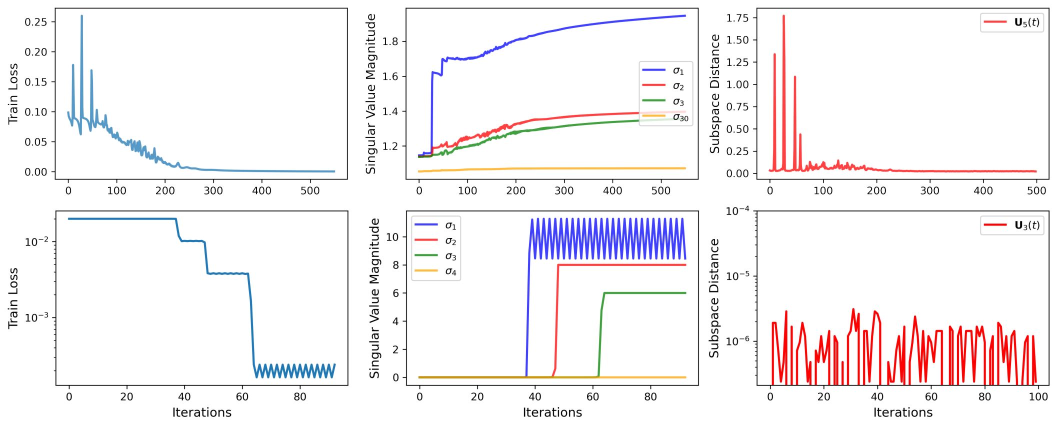


Figure 1: Similarities in the learning behaviors between deep nonlinear and linear networks. Top row: dynamics for the penultimate layer of a MLP. Bottow row: dynamics of the last layer of a DLN. Both networks show that the oscillations in the training loss are a consequence of movements in the dominant singular values, while the singular vectors remain approximately invariant across time.

works are often based on minimalistic examples such as scalar functions [22], which do not fully capture the complex behaviors exhibited by practical networks.

On the other hand, while Cohen et al. [16] demonstrated the prevalence of the edge of stability in many different settings, there were a few caveats—for example, in shallow or wide networks, or on simple datasets, sharpness does not quite rise to $2/\eta$ [23]. Some existing works that analyze the edge of stability construct simpler functions to mimic the behaviors of progressive sharpening and the edge of stability, but fail to capture these subtle nuances. Thus, the current theoretical understanding of this phenomenon is still far from satisfactory.

In this work, we analyze the effect of large learning rates for solving the deep matrix factorization problem. Interestingly, we observe that this problem captures both the nuances of the edge of stability while mimicking the behaviors of nonlinear networks when trained with large learning rates. We illustrate this claim in Figure 1, where we highlight a few similarities between deep linear and nonlinear networks. First, we observe that the oscillations in the training loss of both networks are heavily influenced by the magnitude of the dominant singular values of the weight matrices. Second, despite the oscillations, the consecutive weight updates seemingly occur only within invariant subspaces. These observations suggest that (i) the training dynamics of both networks largely occur within minimal subspaces and (ii) deep linear networks (DLNs) serve as viable surrogates for analyzing nonlinear networks, as previously done in the literature [24, 21, 11, 25, 6].

Our Contributions. Through our analyses, we make the following key contributions:

- **Characterization of GD Dynamics within Invariant Subspaces.** We precisely characterize the GD dynamics of each weight matrix of deep linear networks in contrast to existing works that use gradient flow [26, 25] or do not fully characterize the dynamics [10]. We show that, regardless of the magnitude of the learning rate, the singular vectors of the DLN remain invariant.
- **Benign Oscillations in Singular Values.** Using our characterization of the dynamics, we rigorously show that within a range of learning rates η, oscillations in DLNs occur within the singular value space of a period-2 orbit fixed point, depending upon the magnitude of the target singular value. We also show that the remaining singular values stay constant from initialization throughout all iterations despite having large learning rates, explaining the behavior in Figure 1.

We extensively support our analyses with empirical results and demonstrate the connection between DLNs and nonlinear networks at the edge of stability and its oscillations, offering deeper insights compared to existing works that have primarily focused on simpler functions.

Related Works. We briefly survey a few related works to highlight their differences, and provide a detailed discussion in Appendix A. DLNs are often used as prototypes to study the behaviors of nonlinear networks [27, 25, 24, 28]. The most relevant literature on DLNs are those by Yaras et al. [10, 29] and Kwon et al. [11], who reveal that the weight updates of deep networks occur within an invariant subspace. Our work differs from that of Yaras et al. [10] in that we fully capture the learning dynamics of DLNs throughout the entire GD process. While Kwon et al. [11] observe invariant weight updates, they use this observation for model compression and do not study the learning dynamics with large learning rates. Regarding the edge of stability, the most relevant works are those that analyze scalar functions to demonstrate that the edge of stability occurs on such functions, which have a non-zero third-order derivative and satisfy certain regularity conditions [23, 12, 22]. However, as mentioned previously, these works do not capture the more complicated models that we consider in this work.

Notation and Organization. We denote vectors with bold lower-case letters (e.g., $\mathbf{x}$) and matrices with bold upper-case letters (e.g., $\mathbf{X}$). We use $\mathbf{I}_n$ to denote an identity matrix of size $n \in \mathbb{N}$. We use $[L]$ to denote the set $\{1, 2, \dots, L\}$. We use the notation $\sigma_i(\mathbf{A})$ to denote the i-th singular value of the matrix $\mathbf{A}$. This paper is organized as follows. In Section 2.1, we set the stage by presenting the deep matrix factorization problem. In Section 3, we discuss our theory related to simplicity biases in deep linear networks and their behaviors at the edge of stability. Lastly, we corroborate our results with experiments in Section 4.

2 Background

2.1 Deep Matrix Factorization

We consider the deep matrix factorization problem, where the objective is to model a low-rank matrix $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$ with $\operatorname{rank}(\mathbf{M}^\star) = r$ via a DLN parameterized by a set of parameters $\mathbf{\Theta} = \left\{\mathbf{W}_\ell \in \mathbb{R}^{d\times d}\right\}_{\ell=1}^{L}$, which can be estimated by solving

$$\underset{\mathbf{\Theta}}{\operatorname{argmin}}\, f(\mathbf{\Theta}; \mathbf{M}^\star) \coloneqq \frac{1}{2}\| \underbrace{\mathbf{W}_L \cdot \ldots \cdot \mathbf{W}_1}_{=:\mathbf{W}_{L:1}} - \mathbf{M}^\star\|_{\mathsf{F}}^2, \tag{1}$$

where we adopt the abbreviation $\mathbf{W}_{j:i} = \mathbf{W}_j \cdot \ldots \cdot \mathbf{W}_i$ to denote the end-to-end DLN and is identity when $j < i$. We assume that each weight matrix has dimensions $\mathbf{W}_\ell \in \mathbb{R}^{d\times d}$ to observe the effects of overparameterization.

To obtain the desired solution, for every iteration $t \geq 0$, we update each weight matrix $\mathbf{W}_\ell \in \mathbb{R}^{d\times d}$ using GD with iterations given by

$$\mathbf{W}_\ell(t) = \mathbf{W}_\ell(t-1) - \eta \cdot \nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}(t-1)), \quad \forall \ell \in [L], \tag{2}$$

where $\eta > 0$ is the learning rate and $\nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}(t))$ is the gradient of $f(\mathbf{\Theta})$ with respect to the l-th weight matrix at the t-th GD iterate. We consider a particular initialization for each weight matrix:

$$\mathbf{W}_L(0) = \mathbf{0}, \qquad \mathbf{W}_\ell(0) = \alpha \mathbf{I}_d, \quad \forall \ell \in [L-1], \tag{3}$$

where $\alpha \in [0, 1]$ is a small constant. This particular choice of initialization was also considered in the work by Varre et al. [30], albeit for two-layer networks. We observe that this initialization induces a particular simplicity bias over other initializations, which we discuss in the following sections.

2.2 Edge of Stability and Benign Oscillation

In this section, we briefly define the edge of stability and the benign oscillation phenomenon.

Definition 1 (Sharpness)**.** *Given a loss function $g(\theta)$, the sharpness is defined to $S(\theta) := \|\nabla_\theta^2 g(\theta)\|_2$, which is the maximum eigenvalue of the Hessian of the loss.*

Classical optimization theory (descent lemma for GD) states that training via gradient descent is stable only when the sharpness is bounded by $2/\eta$ [17]. However, for overparameterized deep networks, the descent lemma does not predict optimization dynamics, giving rise to a phenomenon called the "edge of stability", which we formally define below.

Definition 2 (Edge of Stability [16])**.** *During training, the sharpness of the loss $S(\theta)$ continues to grow until it reaches $2/\eta$, and then it ceases to increase and hovers around $2/\eta$. During this process, the training loss behaves non-monotonically over short timescales, yet consistently decreases over long timescales.*

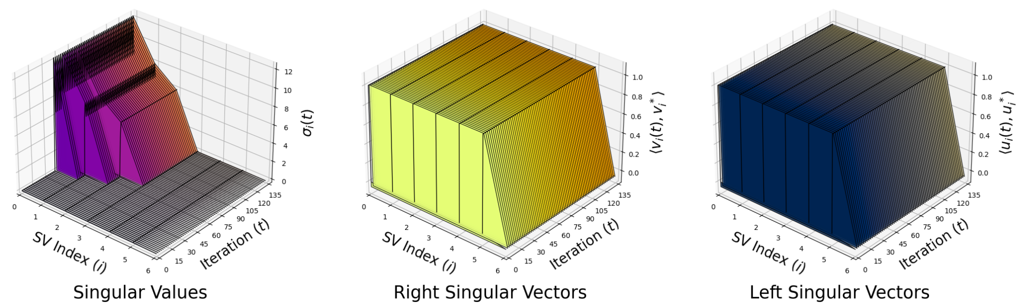


Figure 2: Illustrations of the singular vector and value evolution of the end-to-end DLN. The singular vectors of the network remain static across all iterations, despite all weight parameters being updated. The first two singular values undergo oscillations due to the large learning rate.

The increasing of the sharpness throughout training refers to a stage termed "progressive sharpening". Once the sharpness is above $2/\eta$, descent lemma suggests that the loss should no longer decrease. Despite this, the loss continues to decrease in deep networks, though non-monotonically.

Definition 3 (Benign Oscillation at the Edge of Stability [13])**.** *For a highly non-convex landscape, the implicit bias of GD at the edge of stability ensures that the sharpness achieved is upper bounded by $2/\eta$. This property of GD helps escaping sharper basins in the loss, where $S(\theta) > 2/\eta$ through oscillations and settles for minima in which the sharpness is roughly $2/\eta$.*

We term this oscillation as "benign" as it has the property to escape sharper landscapes as EOS upper-bounds the sharpness by $2/\eta$. Since the sharpness is $2/\eta$, for larger learning rates, we settle for flatter minima, which is seemingly believed to be beneficial for generalization.

3 Theoretical Results

In this section, we present our theoretical results discussing the simplicity biases inherent in GD for learning DLNs, as well as characterize the behavior of DLNs at the edge of stability.

3.1 Simplicity Biases in Deep Linear Networks

Our first result proves that, with the initialization stated in Equation (3), the weight matrices of the DLN possess low-dimensional structures, while their singular vectors remain static for all GD iterations $t \geq 1$.

Theorem 1 (Singular Vector Invariance)**.** *Let $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$ be a rank-r matrix with SVD $\mathbf{M}^\star = \mathbf{U}^\star\mathbf{\Sigma}^\star\mathbf{V}^{\star\top}$. Suppose we run GD (2) with learning rate η and with the initialization in Equation (3). Then, each weight matrix $\mathbf{W}_\ell(t) \in \mathbb{R}^{d\times d}$ has the following decomposition for all $t \geq 1$:*

$$\mathbf{W}_L(t) = \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}, \qquad \mathbf{W}_\ell(t) = \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t) & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top}, \quad \forall \ell \in [L-1], \quad (4)$$

where

$$\widetilde{\mathbf{\Sigma}}_L(t) = \widetilde{\mathbf{\Sigma}}_L(t-1) - \eta \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t-1)\cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t-1) - \mathbf{\Sigma}_r^\star\right)\cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t-1)$$
$$\widetilde{\mathbf{\Sigma}}(t) = \widetilde{\mathbf{\Sigma}}(t-1)\cdot\left(\mathbf{I}_r - \eta\cdot\widetilde{\mathbf{\Sigma}}_L(t-1)\cdot\left(\widetilde{\mathbf{\Sigma}}_L(t-1)\cdot\widetilde{\mathbf{\Sigma}}^{L-1}(t-1) - \mathbf{\Sigma}_r^\star\right)\cdot\widetilde{\mathbf{\Sigma}}^{L-3}(t-1)\right),$$

where $\widetilde{\mathbf{\Sigma}}_L(t), \widetilde{\mathbf{\Sigma}}(t) \in \mathbb{R}^{r\times r}$ is a diagonal matrix with $\widetilde{\mathbf{\Sigma}}_L(1) = \eta\alpha^{L-1}\cdot\mathbf{\Sigma}_r^\star$ and $\widetilde{\mathbf{\Sigma}}(1) = \alpha\mathbf{I}_r$.

Remarks. Due to space limitations, we defer the proof to Appendix C.1. By using this particular initialization, Theorem 1 proves that (i) the singular vectors of each weight matrix remain static throughout the course of learning and *exactly* align with those of the target matrix $\mathbf{M}^\star$; and (ii) the residual singular values (i.e. the $d-r$ singular values) remain constant throughout all GD

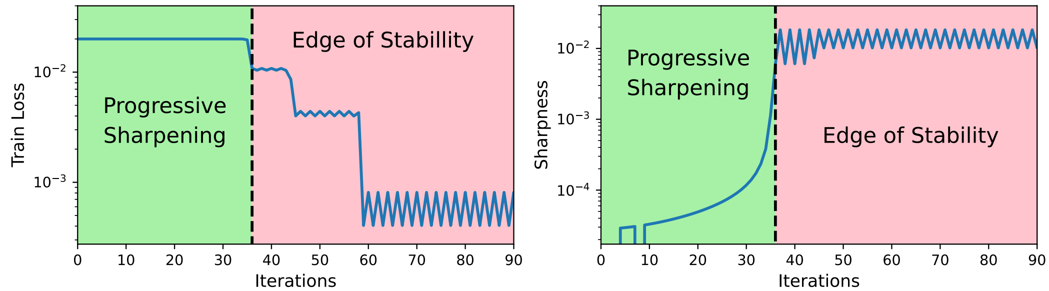


Figure 3: Depiction of the two phases of learning in the deep matrix factorization problem. Upon escaping the first saddle point, we enter the edge of stability regime, where the sharpness hovers just above $2/\eta$.

iterations, regardless of the learning rate (upto divergence). Looping back to Figure 1, these points provide insights to why only a few singular values contribute to the oscillations – only a few singular subspaces are updated while the rest remain close to initialization (and are invariant). Interestingly, Theorem 1 also shows that despite being overparameterized, the end-to-end DLN is *exactly* a low-rank matrix. To see the point more clearly, notice that we can write the end-to-end DLN as

$$\mathbf{W}_{L:1}(t) = \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \ldots \cdot \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t) & \mathbf{0} \\ \mathbf{0} & \alpha \mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top} = \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}.$$

Thus, $\mathbf{W}_{L:1}(t)$ is exactly a rank-r matrix, where r is the rank of $\mathbf{M}^\star$. We empirically corroborate our theory in Figure 2, where we show that indeed the singular vectors immediately align with those of the target's singular vectors.

Furthermore, by the singular vector invariance property, notice that we can rewrite the loss as

$$\frac{1}{2} \|\mathbf{W}_{L:1}(t) - \mathbf{M}^\star\|_{\mathsf{F}}^2 = \frac{1}{2} \| \underbrace{\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t)}_{=:\mathbf{\Sigma}_{L:1}(t)} - \mathbf{\Sigma}^\star\|_{\mathsf{F}}^2 = \frac{1}{2} \sum_{i=1}^{d} \left(\sigma_i(\mathbf{W}_{L:1}(t)) - \sigma_i^\star\right)^2, \quad (5)$$

where we use the notation $\sigma_i^\star = \sigma_i(\mathbf{M}^\star)$ for simplicity. Thus, we can simplify the loss in terms of the singular values alone. This loss is also separable – we can consider a single index i one at a time. This observation will become useful in the next section for analyzing the edge of stability.

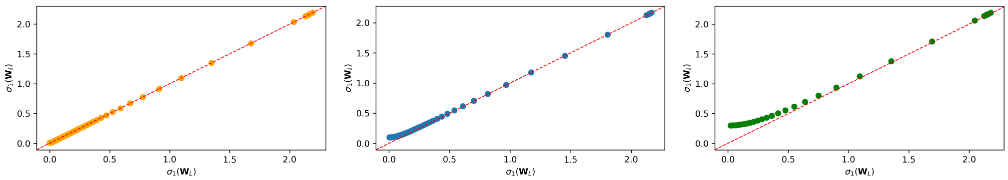
3.2 Edge of Stability in Deep Linear Networks

Generally, the learning dynamics of deep networks with a large learning rate undergo two phases: (i) progressive sharpening and (ii) edge of stability. For DLNs, we observe the same two phases, which we describe in more detail below:

1. (Saddle Escape & Progressive Sharpening). Recall that we use a small initialization scale $\alpha \in [0, 1]$ to initialize the weight matrices. This induces a saddle-to-saddle training dynamic [31], where the singular values are incrementally learned one at a time [11, 32, 25]. We observe that the escape of the first saddle point corresponds to the progressive sharpening stage, where the sharpness of the Hessian continually rises.
2. (Edge of Stability). Upon escaping the first saddle point, we enter the edge of stability regime, where the sharpness hovers slightly above or below $2/\eta$. Within this regime, the oscillations in the singular values begin to occur, which corresponds to oscillations in the training loss. We observe that the oscillations may occur within a 2-period fixed orbit.

Both of these stages are depicted in Figure 3. Our objective is to rigorously analyze the behavior at the edge of stability. To do so, throughout the rest of this section, by Theorem 1, we will consider the following loss in terms of the singular values:

$$\mathcal{L}(\theta^i) = \frac{1}{2} \left(\prod_{j=1}^{L} w_\ell^i - \sigma_i^\star \right)^2, \quad (6)$$



$\alpha = 0.01$ $\alpha = 0.10$ $\alpha = 0.30$

Figure 4: Observing the balancedness between the singular value initialized to 0 and a singular value initialized to α. The scattered points are successive GD iterations (going left to right). For a larger value of α, the initial gap between the two values is larger, but quickly gets closer over more GD iterations.

where we denote $w_\ell^i(t) = \sigma_i(\mathbf{W}_\ell(t))$ and $\theta^i = \left\{w_\ell^i\right\}_{\ell=1}^L$. Equipped with $\mathcal{L}(\cdot)$, we present a result stating the sharpness of the loss in terms of the singular values.

Lemma 2 (Informal)**.** *Consider the objective function $\mathcal{L}(\cdot)$ in Equation (6). Suppose we run GD in (2) with initialization $w_L^i(0) = 0$ and $w_\ell^i(0) = \alpha$, $\forall \ell \in [L-1]$. Then at convergence, the Hessian is a rank-1 matrix and the sharpness is given by $\mathcal{L}''(\theta^i) = 2\sigma_i^{\star\left(2-\frac{2}{L}\right)}$.*

We prove this in Lemma 2 in Appendix C. Now, before we proceed, we will first state a conjecture that we use for the main result.

Conjecture 1. *Suppose we run GD in (2) with learning rate $\eta = \frac{1}{\sigma_i^{\star\left(2-\frac{2}{L}\right)}}$ and the initialization in Equation (3). Then, as $t \to \infty$,*

$$|w_L^i(t) - w_\ell^i(t)| \to 0 \quad \forall i \in [r], \;\; \forall \ell \in [L-1].$$

Recall that by our initialization scheme, $w_L^i(0) = 0$ and $w_\ell^i(0) = \alpha$ for all $\ell \in [L-1]$. Thus, except for $w_L^i(t)$, all of the other singular values across all weight matrices remain balanced throughout all iterations of GD.[1] Conjecture 1 states that if we pick a learning rate roughly equal to 2 divided by the sharpness at the minima[2], then throughout the course of learning, $w_L^i(t)$ becomes increasingly balanced and equal to the rest of the singular values $\{w_\ell^i(t)\}$. We provide evidence to support this conjecture in Figure 4 and note that this has been rigorously proved for two-layer scalar networks [33]. Notice that at initialization, the gap is exactly α. Thus, in Figure 4, we observe that for larger values of α, the balancing quickly occurs, whereas for smaller values of α, the balancing is immediate.

Theorem 2 (Periodic Orbit at the Edge of Stability)**.** *Consider the objective function $\mathcal{L}(\cdot)$ in Equation (6), where $\sigma_i^\star$ is a singular value of a symmetric target matrix $\mathbf{M}^\star$. Let $\mathrm{GD}_\eta(\cdot)$ denote one GD step with learning rate η:*

$$\mathrm{GD}_\eta(w_\ell^i(t)) \coloneqq w_\ell^i(t+1) = w_\ell^i(t) - \eta \cdot \nabla_{w_\ell^i} \mathcal{L}(\theta^i(t)),$$

and define $s \coloneqq \sigma_i^{\star\frac{1}{L}}$. Then, under Conjecture 1, for any $\epsilon > 0$ and any point $w_\ell^i(t) \in [s-\epsilon, s]$, there exists a learning rate $\frac{2}{\mathcal{L}''(s)} < \eta < \frac{2}{\mathcal{L}''(s) - \epsilon\mathcal{L}'''(s)}$ such that $\mathrm{GD}_\eta(\mathrm{GD}_\eta(w_\ell^i(t))) = w_\ell^i(t)$.

Sketch of the Proof. We briefly outline the sketch of the proof here and defer the details to Appendix C. Since our goal is to demonstrate the edge of stability for deep matrix factorization, we first compute the Hessian of the simplified loss in Equation (6) via Lemma 2 in manuscript. Then, we establish the connection of the Hessian (as well as sharpness) between the simplified loss (6) and the original deep matrix factorization loss (Equation 1) through Lemma 1 in Appendix. Upon establishing this connection, in Lemma 2 in Appendix, we prove that GD achieves the smallest sharpness value amongst all minima (which is computed to be $2\sigma_i^{\star 2-\frac{2}{L}}$). Finally, we prove the occurrence of the edge of stability in the loss Equation (6) by proving existence of 2-period orbit oscillation in Theorem 2.

[1] Based on the scalar loss, the derivative with respect to each singular value is the same. Hence, by starting from the same initialization, they remain balanced.

[2] As opposed to quadratic loss for which using $\eta = \frac{2}{\|\nabla^2 g(\theta)\|}$ cause the iterates to diverge and blow up.

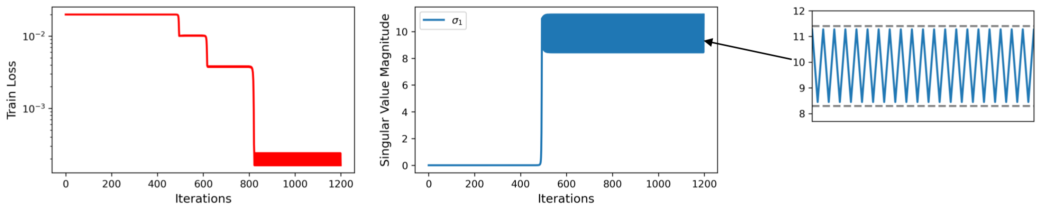


Figure 5: Close-up representation of the oscillation in the singular value as a 2-period fixed orbit. For a specific value of η, the singular value oscillates between only two values indicating a period-2 orbit.

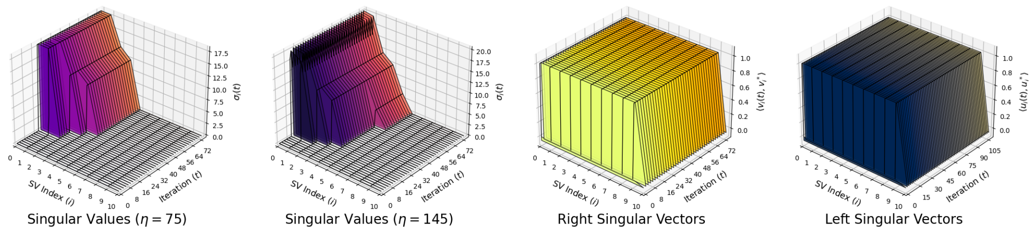


Figure 6: Depiction of the singular values and singular vectors of the end-to-end matrix throughout the course of learning for different learning rates η. For both learning rates, the singular vectors remain static and align with those of the target matrix.

Remarks. Theorem 2 shows that for any $w_\ell^i(t)$ within an ϵ-distance from the local minima s, there exists a learning rate such that the singular value is a fixed point under consecutive iterations of GD even when $\eta > \frac{2}{\mathcal{L}''(s)}$. This theorem proves that the loss do not blow up for a $\eta > \frac{2}{\mathcal{L}''(s)}$ (as opposed to what descent lemma for GD predicts), but is oscillating in a 2-period orbit. Hence, this theorem shows that Edge of Stability is achieved for the loss equation 6 and hence also achieved in the original Deep matrix factorization loss equation 1 due to the equivalance of the Hessian. In Figure 5, we provide a closer look at the periodicity of the first singular value of the end-to-end DLN. To establish Theorem 2, we used two assumptions. The first comes from Conjecture 1 to consider that the unbalanced singular value at initialization will become balanced with the rest of them. The second assumption comes from the symmetric structure of $\mathbf{M}^\star$, which was needed to connect the Hessian of the singular value loss to the overall loss, as outlined in the sketch proof. However, this is simply an artifact of the analysis—the results (including Theorem 1), as we consider non-symmetric matrices throughout all of our experiments.

Furthermore, note that Theorem 2 establishes the periodicity of the oscillations for the loss function $\mathcal{L}(\cdot)$, considering only a specific singular value $\sigma_i^\star$. If we pick a learning rate η that falls within the specified range for both, say $\sigma_1^\star$ and $\sigma_2^\star$, we will observe periodic orbits for both singular values. However, if we select a learning rate to induce oscillations in $\sigma_3^\star$, it may be too large for $\sigma_1^\star$, potentially leading to chaos and causing the overall loss to diverge. In Figure 6 and Figure 14 (Appendix), note that weights with a larger value of sharpness will have a high amplitude in oscillations. Based on the computed sharpness value, this implies that larger singular values have higher sharpness values and, hence, have higher oscillations.

Lastly, we briefly discuss the relationship between ours and existing results. There exists a large literature of work that focus on studying oscillations and chaos in dynamical systems [34–36]. For example, Chen et al. [37] analyzed the various phases of oscillation: catapult, periodic, and chaotic phases for GD in a quadratic regression problem with increasing learning rate. Chen et al. [23] analyzed the period-2 orbit for oscillations for a family of scalar functions. Our work focuses on orbits in deep linear networks, provided by the key insight in that the singular vectors remain invariant.

4 Experimental Results

This section is organized as follows. Firstly, in Section 4.1, we present additional experimental results to support our theory on DLNs. Secondly, in Section 4.2, we present results on the edge of stability in nonlinear networks and their relationship to DLNs.

4.1 Simplicity Biases and Oscillations in Deep Linear Networks

Simplicity Bias. In this section, we present additional synthetic results on the simplicity biases inherent in GD for learning DLNs. Here, the objective is to showcase the validity of Theorem 1 for both small and large learning rates. To this end, we generate a low-rank matrix $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$, where $d = 100$ with rank $r = 5$ and consider the deep matrix factorization problem. We initialize with scale $\alpha = 0.01$ and run GD on each of the factors with learning rates $\eta = 75, 145$. In Figure 6, we display the singular values throughout the course of learning and the angle between the target's singular vectors and those of the end-to-end DLN for both learning rates. As stated in Theorem 1, the singular vectors in both cases exactly align with each other, despite the learning rates. Furthermore, the residual singular values are exactly 0 throughout the course of learning. For $\eta = 145$, we observe oscillations in the first singular value as we enter the edge of stability due to the large learning rate.

Edge of Stability. Within the edge of stability regime, we observe that the range of oscillations is highly on the learning rate η. To this end, we perform an experiment where we vary the learning rate η and compute the amplitude of the oscillations under the same experimental setup as above, but with a target matrix rank of $r = 3$. In Figure 7, we show that as η increases, the oscillation in the singular value starts increasing progressively. When $\eta \in (145, 162)$, the range of oscillation increases only in the first singular value, while the other singular values do not show any oscillation. For $\eta > 165$, oscillations occur in the first two singular values and progressively increase with η, while the rest of the singular values remain constant. From Figure 6, we observe that as the oscillations occur for the singular values sequentially, while the singular vectors stay aligned throughout.

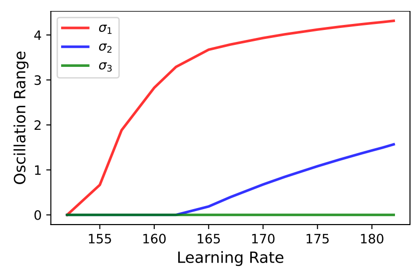


Figure 7: Range of oscillations of the singular values of end-to-end DLN.

While the edge of stability phenomenon persists across a wide range of deep network architectures and datasets, there are specific cases in which this phenomenon does not quite occur. For example, Cohen et al. [16] state that one of these caveats is that for specific networks (shallow) or simple datasets, "the sharpness does not rise *that* much". We observe that this is exactly the case for the DLN, which we demonstrate in Figure 8. In Figure 8, the dashed line represents $2/\eta$, and clearly, the sharpness value plateaus far below this value, despite the training loss going to zero. At a high level, our theory predicts this phenomenon. By Lemma 2, it is given by $2\sigma_i^{\star\left(2-\frac{2}{L}\right)}$, and so if this value is less than $2/\eta$, the DLN will not enter the edge of stability. This result provides deeper insights into when the edge of stability occurs. For specific learning rates, the phenomenon is not invoked.

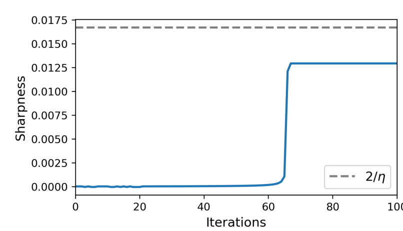
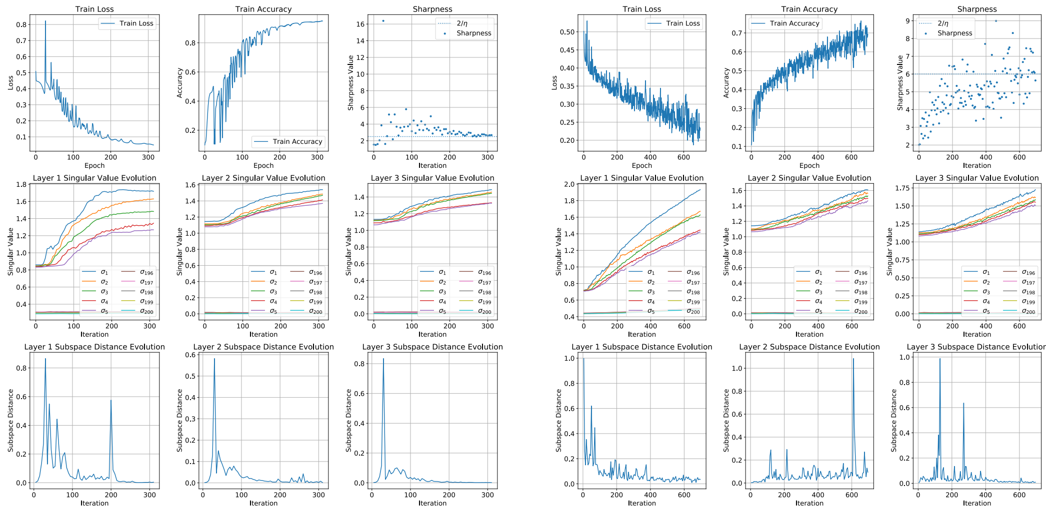


Figure 8: Sharpness of the DLN over GD iterations.

4.2 Benign Oscillations in Deep Nonlinear Networks

In this section, we bridge the connection between our observations in DLNs with deep neural networks with non-linear activation layers. To this end, we consider a four layer feed-forward neural network (i.e., MLP) with hidden layer size in each unit of 200 with ReLU activations. We use this deep network to classify images on $20k$-subsampled CIFAR-10 [38] and MNIST [39] datasets. For the



MNIST Dataset with 4-Layer MLP   CIFAR-10 Dataset with 4-Layer MLP

Figure 9: Prevalence of oscillatory behaviors and singular vector invariance in 4-layer networks with ReLU activations.

loss function, we use the MSE loss[3] by converting the ground-truth labels into one-hot vectors:

$$L(\mathbf{W}_4, \mathbf{W}_3, \mathbf{W}_2, \mathbf{W}_1) = \|\mathbf{Y} - \mathbf{W}_4\rho(\mathbf{W}_3\rho(\mathbf{W}_2\rho(\mathbf{W}_1\mathbf{X})))\|_{\mathsf{F}}^2, \tag{7}$$

where $\rho(\cdot)$ is the ReLU function, $\mathbf{X}$ and $\mathbf{Y}$ are the data and labels stacked as matrices, respectively[4]. For both networks, we intentionally choose a large learning rate to provoke oscillations at edge of stability. For each experiment, we plot the training loss, the sharpness, the singular values of $\mathbf{W}_3, \mathbf{W}_2, \mathbf{W}_1$ and the subspace distance for the left singular vector for each layer across successive iterations, which is defined as:

$$\text{Subspace Distance} = r - \|\mathbf{U}_r(\mathbf{W}(t))^\top \mathbf{U}_r(\mathbf{W}(t+1))\|_{\mathsf{F}}^2. \tag{8}$$

The subspace distance characterizes the stationarity of the singular vectors with respect to time.

In Figure 9, we observe that for both the MNIST and CIFAR-10 datasets, the training loss and accuracy demonstrate significant benign oscillatory behavior, and the sharpness value hovers around $2/\eta$. This indicates that gradient descent is operating at the edge of stability. Similar to DLNs, damped oscillations occur in the top 5 singular values, while the last 5 singular values remain the same as they were at initialization. Overall, these results suggest that the behavior of nonlinear networks at the edge of stability is well captured by linear networks, with two exceptions: (i) damped oscillations occur in the singular values for nonlinear networks, as opposed to free oscillations in linear networks; and (ii) the singular vector subspace shows momentary spiking in nonlinear networks, whereas it remains zero throughout in linear networks. The primary reason for these differences is the ReLU activation function, which nonetheless provides valuable insights into this phenomenon.

5 Conclusion

In this work, we unveiled an intriguing phenomenon: in the edge of stability regime, oscillations in the training loss are largely an artifact of oscillations occurring within a minimal invariant subspace. We analyze this phenomenon by focusing on the deep matrix factorization problem, demonstrating that deep linear networks exhibit very similar behaviors to their nonlinear counterparts. We showed that oscillations in linear networks may occur as a 2-period fixed orbit depending on the learning rate. We provided extensive empirical results corroborating our theory and connecting our results on linear networks to those on nonlinear networks.

[3] Sharpness for cross entropy loss drops down to zero at the end of training [16].

[4] Here, we ignore the bias terms of the network for simplicity in exposition.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[2] Behnam Neyshabur, Srinadh Bhojanapalli, David Mcallester, and Nati Srebro. Exploring generalization in deep learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[3] K. Kawaguchi, Y. Bengio, and L. Kaelbling. *Generalization in Deep Learning*, page 112–148. Cambridge University Press, December 2022.

[4] Gal Vardi. On the implicit bias in deep-learning algorithms. *arXiv preprint arXiv:2208.12591*, 2022.

[5] Zhiyuan Li, Yuping Luo, and Kaifeng Lyu. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. In *International Conference on Learning Representations*, 2021.

[6] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[7] Behnam Neyshabur. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.

[8] Daniel Kunin, Atsushi Yamamura, Chao Ma, and Surya Ganguli. The asymmetric maximum margin bias of quasi-homogeneous neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.

[9] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018.

[10] Can Yaras, Peng Wang, Wei Hu, Zhihui Zhu, Laura Balzano, and Qing Qu. The law of parsimony in gradient descent for learning deep linear networks. *arXiv preprint arXiv:2306.01154*, 2023.

[11] Soo Min Kwon, Zekai Zhang, Dogyoon Song, Laura Balzano, and Qing Qu. Efficient low-dimensional compression of overparameterized models. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 1009–1017. PMLR, 02–04 May 2024.

[12] Yuqing Wang, Zhenghao Xu, Tuo Zhao, and Molei Tao. Good regularity creates large learning rate implicit biases: edge of stability, balancing, and catapult. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023.

[13] Miao Lu, Beining Wu, Xiaodong Yang, and Difan Zou. Benign oscillation of stochastic gradient descent with large learning rate. In *The Twelfth International Conference on Learning Representations*, 2024.

[14] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[15] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[16] Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021.

[17] Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-stabilization: The implicit bias of gradient descent at the edge of stability. In *The Eleventh International Conference on Learning Representations*, 2023.

[18] Lijun Ding, Dmitriy Drusvyatskiy, Maryam Fazel, and Zaid Harchaoui. Flat minima generalize for low-rank matrix recovery. *arXiv preprint arXiv:2203.03756*, 2023.

[19] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

[20] Rotem Mulayoff and Tomer Michaeli. Unique properties of flat minima in deep networks. In *International conference on machine learning*, pages 7108–7118. PMLR, 2020.

[21] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International conference on machine learning*, pages 244–253. PMLR, 2018.

[22] Xingyu Zhu, Zixuan Wang, Xiang Wang, Mo Zhou, and Rong Ge. Understanding edge-of-stability training dynamics with a minimalist example. In *The Eleventh International Conference on Learning Representations*, 2023.

[23] Lei Chen and Joan Bruna. Beyond the edge of stability via two-step gradient updates, 2023.

[24] Peng Wang, Xiao Li, Can Yaras, Zhihui Zhu, Laura Balzano, Wei Hu, and Qing Qu. Understanding deep representation learning via layerwise feature compression and discrimination. *arXiv preprint arXiv:2311.02960*, 2024.

[25] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[26] Liu Ziyin, Botao Li, and Xiangming Meng. Exact solutions of a deep linear network. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24446–24458. Curran Associates, Inc., 2022.

[27] Niladri S. Chatterji and Philip M. Long. Deep linear networks can benignly overfit when shallow ones do. *Journal of Machine Learning Research*, 24(117):1–39, 2023.

[28] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29820–29834. Curran Associates, Inc., 2021.

[29] Can Yaras, Peng Wang, Laura Balzano, and Qing Qu. Compressible dynamics in deep overparameterized low-rank learning & adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

[30] Aditya Vardhan Varre, Maria-Luiza Vladarean, Loucas PILLAUD-VIVIEN, and Nicolas Flammarion. On the spectral bias of two-layer linear networks. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 64380–64414. Curran Associates, Inc., 2023.

[31] Arthur Jacot, François Ged, Berfin Şimşek, Clément Hongler, and Franck Gabriel. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity. *arXiv preprint arXiv:2106.15933*, 2022.

[32] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization. In *International Conference on Learning Representations*, 2020.

[33] Yuqing Wang, Minshuo Chen, Tuo Zhao, and Molei Tao. Large learning rate tames homogeneity: Convergence and balancing effect. *arXiv preprint arXiv:2110.03677*, 2021.

[34] Welington De Melo and Sebastian Van Strien. *One-dimensional dynamics*, volume 25. Springer Science & Business Media, 2012.

[35] Robert Devaney. *An introduction to chaotic dynamical systems*. CRC press, 2018.

[36] Andrzej Lasota and Michael C Mackey. *Chaos, fractals, and noise: stochastic aspects of dynamics*, volume 97. Springer Science & Business Media, 2013.

[37] Xuxing Chen, Krishna Balasubramanian, Promit Ghosal, and Bhavya Kumar Agrawalla. From stability to chaos: Analyzing gradient descent dynamics in quadratic regression. *Transactions on Machine Learning Research*, 2024.

[38] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[39] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[40] Atish Agarwala, Fabian Pedregosa, and Jeffrey Pennington. Second-order regression models exhibit progressive sharpening to the edge of stability. *arXiv preprint arXiv:2210.04860*, 2022.

[41] Sanjeev Arora, Zhiyuan Li, and Abhishek Panigrahi. Understanding gradient descent on the edge of stability in deep learning. In *International Conference on Machine Learning*, pages 948–1024. PMLR, 2022.

[42] Kaifeng Lyu, Zhiyuan Li, and Sanjeev Arora. Understanding the generalization benefit of normalization layers: Sharpness reduction. *Advances in Neural Information Processing Systems*, 35:34689–34708, 2022.

[43] Kwangjun Ahn, Jingzhao Zhang, and Suvrit Sra. Understanding the unstable convergence of gradient descent. In *International Conference on Machine Learning*, pages 247–257. PMLR, 2022.

[44] Zixuan Wang, Zhouzi Li, and Jian Li. Analyzing sharpness along gd trajectory: Progressive sharpening and edge of stability. *Advances in Neural Information Processing Systems*, 35:9983–9994, 2022.

[45] Mathieu Even, Scott Pesme, Suriya Gunasekar, and Nicolas Flammarion. (s) gd over diagonal linear networks: Implicit bias, large stepsizes and edge of stability. *Advances in Neural Information Processing Systems*, 36, 2024.

[46] Jingfeng Wu, Vladimir Braverman, and Jason D Lee. Implicit bias of gradient descent for logistic regression at the edge of stability. *Advances in Neural Information Processing Systems*, 36, 2024.

[47] Xitong Zhang, Ismail R Alkhouri, and Rongrong Wang. Structure-preserving network compression via low-rank induced training through linear layers composition. *arXiv preprint arXiv:2405.03089*, 2024.

[48] Hung-Hsu Chou, Carsten Gieshoff, Johannes Maly, and Holger Rauhut. Gradient descent for deep matrix factorization: Dynamics and implicit bias towards low rank. *Applied and Computational Harmonic Analysis*, 68:101595, 2024.

[49] Libin Zhu, Chaoyue Liu, Adityanarayanan Radhakrishnan, and Mikhail Belkin. Catapults in sgd: spikes in the training loss and their impact on generalization through feature learning. *arXiv preprint arXiv:2306.04815*, 2023.

[50] Elan Rosenfeld and Andrej Risteski. Outliers with opposing signals have an outsized effect on neural network optimization. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024.

Appendix

This Appendix is organized as follows. In Section A, we survey, summarize, and highlight the differences between our work and the related literature. In Section B, we provide additional experiments, namely (i) experiments with different initialization of the DLN; (ii) more experiments at the edge of stability in DLNs; (iii) more experiments at the edge of stability in MLPs. In Section C, we present the deferred proofs in detail. Lastly, in Section D, we provide additional results based on our theory, which may be of independent interest.

For the experiments on nonlinear networks, we use an A40 NVIDIA GPU, and otherwise run experiments a MacBook Pro with M2 Pro Chip.

A Related Work

Implicit Bias at the Edge of Stability. Due to the important practical implications of the edge of stability, there has been an explosion of research dedicated to understanding this phenomenon and its implicit regularization properties. Here, we survey a few of these works. Damian et al. [17] explained edge of stability through a mechanism called "self-stabilization", where they showed that during the momentary divergence of the iterates along the sharpest eigenvector direction of the Hessian, the iterates also move along the negative direction of the gradient of the curvature, which leads to stabilizing the sharpness to $2/\eta$. Agarwala et al. [40] proved that second-order regression models (the simplest class of models after the linearized NTK model) demonstrate progressive sharpening of the NTK eigenvalue towards a slightly different value than $2/\eta$. Arora et al. [41] mathematically analyzed the edge of stability, where they showed that the GD updates evolve along some deterministic flow on the manifold of the minima. Lyu et al. [42] showed that the normalization layers had an important role in the edge of stability – they showed that these layers encouraged GD to reduce the sharpness of the loss surface and enter the EOS regime. Ahn et al. [43] established the phenomenon in two-layer networks and find phase transitions for step-sizes in which networks fail to learn "threshold" neurons. Wang et al. [44] also analyze a two-layer network, but provide a theoretical proof for the change in sharpness across four different phases. [45] analyzed the edge of stability in diagonal linear networks and found that oscillations occur on the sparse support of the vectors. Lastly, Wu et al. [46] analyzed the convergence at the edge of stability for constant step size GD for logistic regression on linearly separable data.

Edge of Stability in Toy Functions. To analyze the edge of stability in slightly simpler settings, many works have constructed scalar functions to analyze the prevalence of this phenomenon. For example, Chen et al. [23] studied a certain class of scalar functions and identified conditions in which the function enters the edge of stability through a two-step convergence analysis. Wang et al. [12] showed that the edge of stability occurs in specific scalar functions, which satisfies certain regularity conditions and developed a global convergence theory for a family of non-convex functions without globally Lipschitz continuous gradients. Lastly, Zhu et al. [22] analyzed local oscillatory behaviors for 4-layer scalar networks with balanced initialization. Overall, all of these works showed that the necessary condition for the edge of stability to occur is that the second derivative of the loss function is non-zero, even though they assumed simple scalar functions. Our work takes one step further to analyze the prevalence of the edge of stability in DLNs. Although our loss simplifies to a loss in terms of the singular values, they precisely characterize the dynamics of the DLNs for the deep matrix factorization problem.

Deep Linear Networks. Over the past decade, many existing works have analyzed the learning dynamics of DLNs as a surrogate for deep nonlinear networks to study the effects of depth and implicit regularization [25, 21, 6, 47]. Generally, these works focus on unveiling the dynamics of a phenomenon called "incremental learning", where small initialization scales induce a greedy singular value learning approach [11, 32, 25], analyzing the learning dynamics via gradient flow [25, 48, 6], or showing that the DLN is biased towards low-rank solution [29, 6, 11], amongst others. However, these works do not consider the occurence of the edge of stability in such networks. On the other hand, while works such as those by Yaras et al. [29] and Kwon et al. [11] have similar observations in that the weight updates occur within an invariant subspace, they do not analyze the edge of stability regime.

Difference with related works on GD oscillation Recently, [49] *empirically* found that in SGD, catapults occur in a low-dimensional subspace spanned by the top eigenvectors of the tangent kernel. In our paper, we theoretically analyze this oscillatory phenomenon for Gradient Descent in deep linear networks. Our theoretical analysis and empirical findings further justify the observations in their paper. [50] found that oscillations occur on groups of opposing signals in the training data, which constitute the loss. These opposing signals have features high in magnitude. Our work further supports and justifies this observation. We observe that features with large strengths (which are the singular values $\sigma_1 > \sigma_2 > ...\sigma_r$) demonstrate an increasing tendency for oscillations in their corresponding singular loss (Figure 14). This is because the sharpness achieved by GD on each singular value loss is $\sigma_i^{2-\frac{2}{L}}$, and higher sharpness demonstrates large oscillations for a fixed learning rate.

B Additional Experiments

In this section, we provide additional results to supplement those in the main paper.

B.1 Choice of Initialization

To analyze DLNs, we considered a particular initialization that was also similarly considered in the literature:

$$\mathbf{W}_L(0) = \mathbf{0}, \qquad \mathbf{W}_\ell(0) = \alpha \mathbf{I}_d, \quad \forall \ell \in [L-1], \tag{9}$$

where $\alpha \in [0,1]$ is a small constant. In this section, we investigate the edge of stability regime, where we consider α-scaled orthogonal matrices instead:

$$\mathbf{W}_\ell = \alpha \mathbf{P}_\ell \in \mathbb{R}^{d\times d}, \quad \text{where } \mathbf{P}_\ell^\top \mathbf{P}_\ell = \mathbf{I}_d.$$

To this end, we consider the deep matrix factorization problem with a target matrix $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$, where $d = 100$, $r = 5$, and $\alpha = 0.01$. We use GD with a large learning rate $\eta = 160$ to update the weight matrices. In Figure 10, we display plots of the singular values and vectors throughout the course of GD. Here, we observe that oscillations in both the singular values and vectors occur, whereas with the initialization we consider, oscillations only occur on the singular values. Thus, the analysis in this case becomes difficult, and does not directly align with the observations in Section 4.

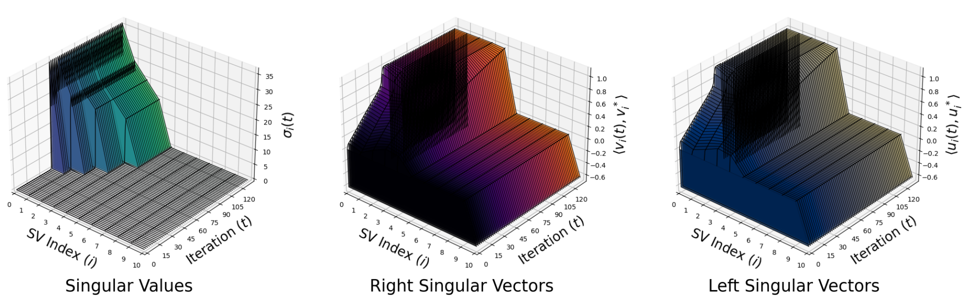
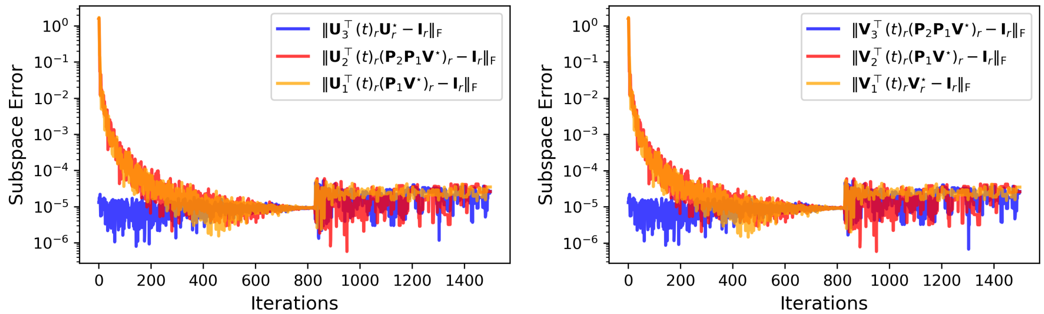


Figure 10: Demonstrating the prevalence of the edge of stability and their oscillations in DLNs with balanced orthogonal initialization. Here, we observe oscillations in both the singular values and vectors.

Next, we investigate the possibility of extending our analysis to the case in which we initialize with one zero and the rest orthogonal matrices:

$$\mathbf{W}_L(0) = \mathbf{0}, \qquad \mathbf{W}_\ell(0) = \alpha \mathbf{P}_\ell, \quad \forall \ell \in [L-1]. \tag{10}$$

For this case, we observe an interesting simplicity bias as well, where after some GD iteration T, the decomposition in Theorem 1 similarly holds, but with different singular vectors for the intermediate matrices. We formally present this as a conjecture in Conjecture 2.



Left Singular Vectors Right Singular Vectors

Figure 11: Empirically verifying Conjecture 2 by showing that after some GD iterations, the singular vectors of the intermediate matrices align, displaying singular vector invariance.

Conjecture 2 (Invariance in Orthogonally Initialized DLNs.). *Suppose $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$ be a rank-r matrix with SVD $\mathbf{M}^\star = \mathbf{U}^\star\mathbf{\Sigma}^\star\mathbf{V}^{\star\top}$. Let $\mathbf{W}_\ell(t) \in \mathbb{R}^{d\times d}$ denote the ℓ-th weight matrix at GD (2) iterate t. Then, after some $t \geq T$, each weight matrix admits the following decomposition:*

$$\mathbf{W}_L(t) = \mathbf{U}^\star \begin{bmatrix} \mathbf{\Sigma}_L(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \left[\left(\prod_{i=L-1}^{1} \mathbf{P}_i\right)\mathbf{V}^\star\right]^\top, \tag{11}$$

$$\mathbf{W}_\ell(t) = \left[\left(\prod_{i=l}^{1} \mathbf{P}_i\right)\mathbf{V}^\star\right] \begin{bmatrix} \mathbf{\Sigma}_\ell(t) & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{I}_{d-r} \end{bmatrix} \left[\left(\prod_{i=l-1}^{1} \mathbf{P}_i\right)\mathbf{V}^\star\right]^\top, \quad \forall \ell \in [2, L-1], \tag{12}$$

$$\mathbf{W}_1(t) = \mathbf{P}_1\mathbf{V}^\star \begin{bmatrix} \mathbf{\Sigma}_1(t) & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top}, \tag{13}$$

where $\mathbf{W}_L(0) = \mathbf{0}$ and $\mathbf{W}_\ell(0) = \alpha\mathbf{P}_l$, $\forall \ell \in [L-1]$.

We empirically verify Conjecture 2 in Figure 11, where we compute the distance between the predicted left and right singular vectors in Conjecture 2 and the singular vectors of the weight matrices across GD. We observe that while the distance is large at initialization, the distance quickly goes to zero after a few iterations, verifying the conjecture. Furthermore, we illustrate in Figures 12 and 13, that even for this initialization, the oscillations only occur in the singular value space. Thus, it is possible to relax our initialization assumptions, but this requires a slightly more delicate analysis.

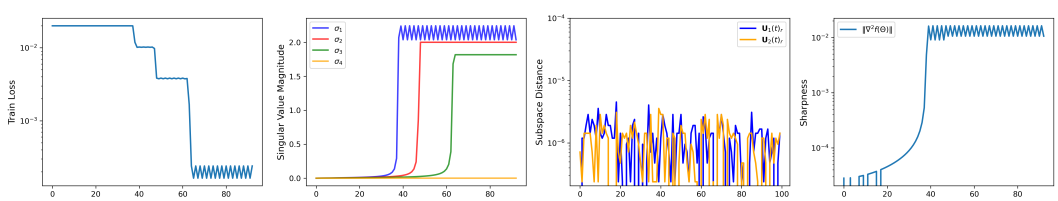


Figure 12: Demonstrating the edge of stability phenomenon, where the initialization is orthogonal rather than identity with learning rate $\eta = 160$.

B.2 More Experiments on Deep Linear Networks

In this section, we provide more experimental results on the edge of stability in DLNs. Specifically, in Figure 14, we provide plots on how the oscillations change as a function of the learning rate η. As we increase the learning rate, which corresponds to the columns from top to bottom, we can see that the oscillations occur in the top singular value, and then progressively occurs in the second singular value. For a learning rate of $\eta = 92$, we observe slight oscillations in the third singular value, but there is overall chaos in the learning dynamics. This is predicted by our analysis in Theorem 2 – the learning rate is out of the specified range and hence the orbit no longer occurs. These figures were

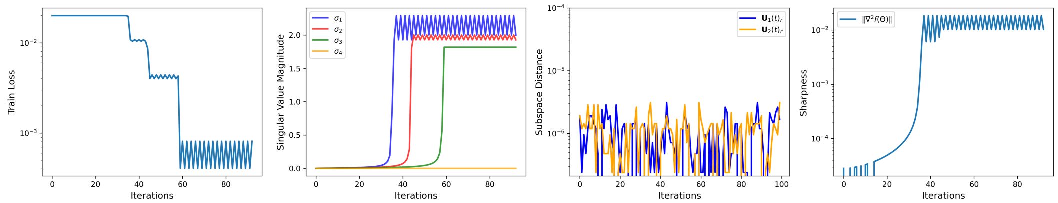
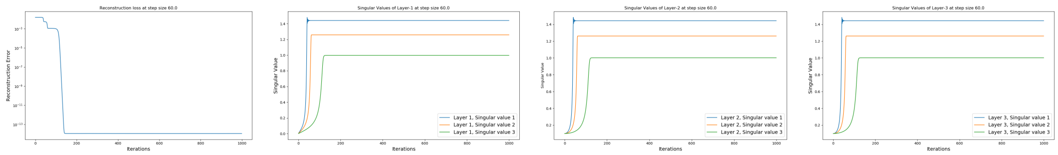
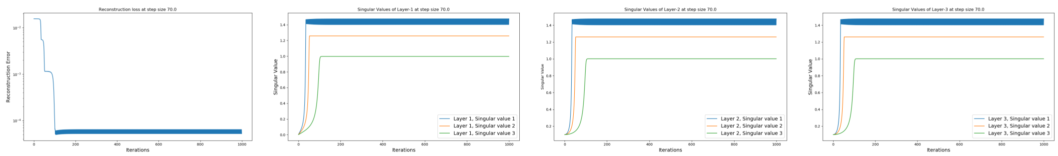


Figure 13: Demonstrating the edge of stability phenomenon, where the initialization is orthogonal rather than identity with learning rate $\eta = 172$.

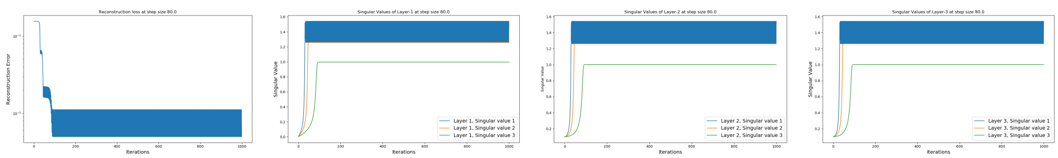
generated using normal random initialization with scale $\alpha = 0.1$ and a target matrix with size $d = 50$ and rank $r = 3$. We use random initialization to demonstrate that our observations hold without making the assumptions on initialization.



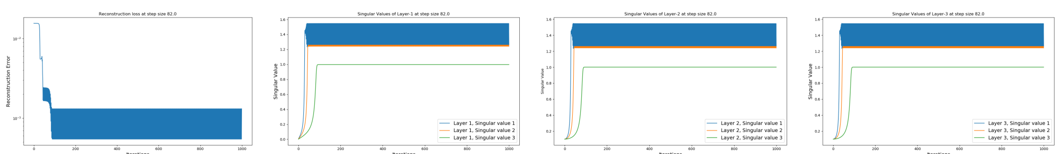
Train Loss ($\eta = 60$) Layer 1 σ_i ($\eta = 60$) Layer 2 σ_i ($\eta = 60$) Layer 3 σ_i ($\eta = 60$)



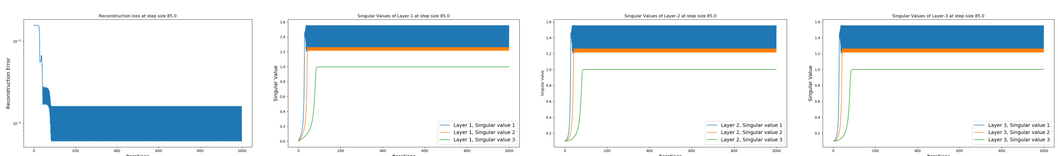
Train Loss ($\eta = 70$) Layer 1 σ_i ($\eta = 70$) Layer 2 σ_i ($\eta = 70$) Layer 3 σ_i ($\eta = 70$)



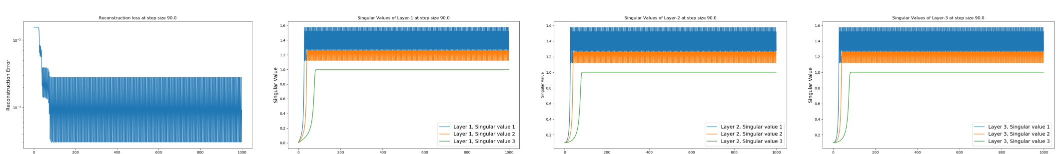
Train Loss ($\eta = 80$) Layer 1 σ_i ($\eta = 80$) Layer 2 σ_i ($\eta = 80$) Layer 3 σ_i ($\eta = 80$)



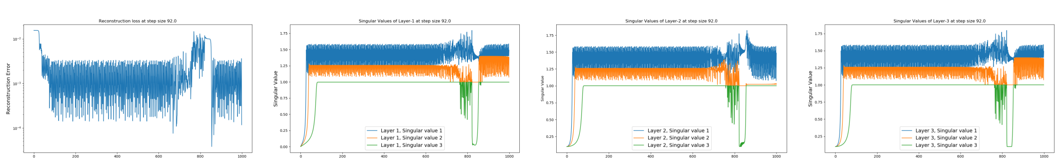
Train Loss ($\eta = 82$) Layer 1 σ_i ($\eta = 82$) Layer 2 σ_i ($\eta = 82$) Layer 3 σ_i ($\eta = 82$)



Train Loss ($\eta = 85$) Layer 1 σ_i ($\eta = 85$) Layer 2 σ_i ($\eta = 85$) Layer 3 σ_i ($\eta = 85$)



Train Loss ($\eta = 90$) Layer 1 σ_i ($\eta = 90$) Layer 2 σ_i ($\eta = 90$) Layer 3 σ_i ($\eta = 90$)

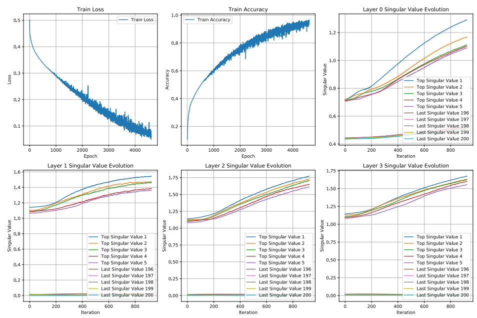


Train Loss ($\eta = 92$) Layer 1 σ_i ($\eta = 92$) Layer 2 σ_i ($\eta = 92$) Layer 3 σ_i ($\eta = 92$)

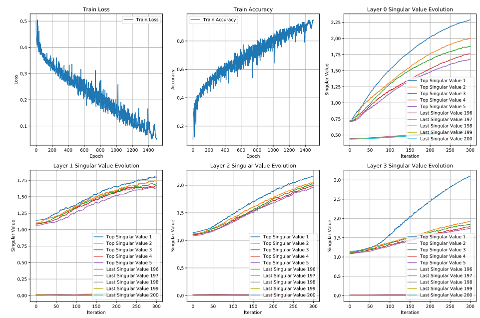
Figure 14: Depiction of the edge of stability progressively occuring on each singular value depending on the learning rate η.

B.3 More Experiments on Deep Nonlinear Networks

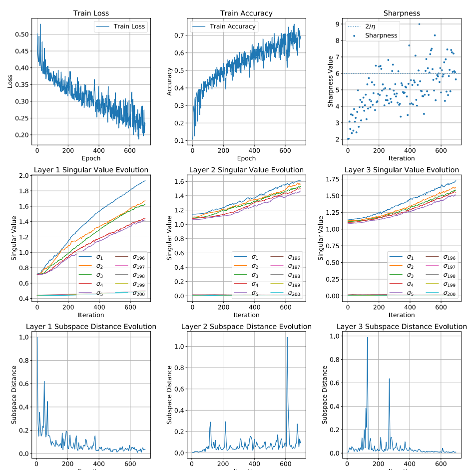
In this section, we consider a 4-layer MLP and demonstrate the prevalence of the edge of stability with subsets of the MNIST and CIFAR-10 datasets for varying values of η. The network architecture is the same as the one considered in the main text in Section 4.2.



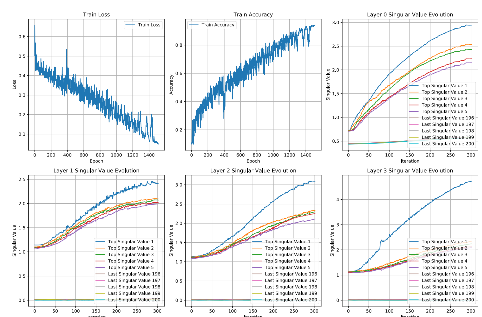
$\eta = 0.1$



$\eta = 0.8$

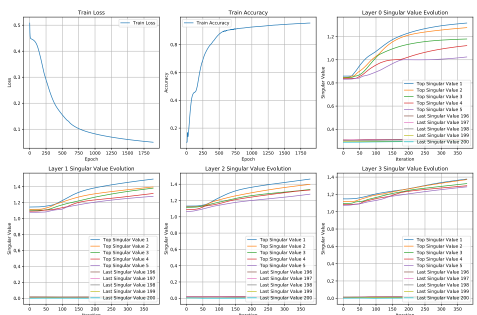


$\eta = 1.0$

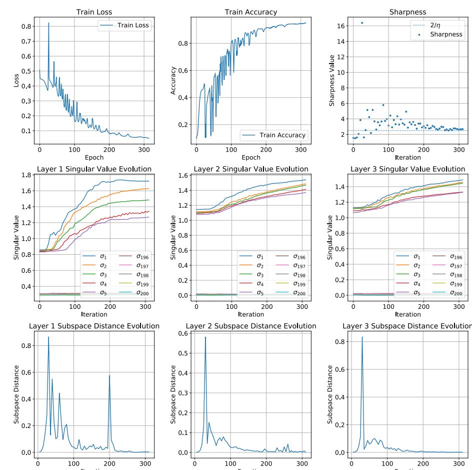


$\eta = 1.5$

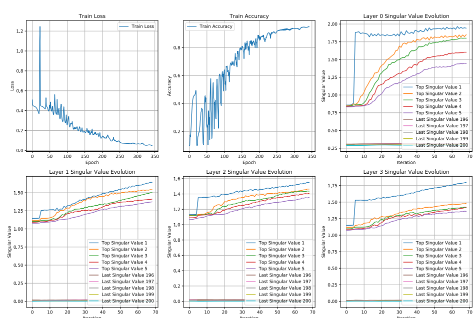
Figure 15: Oscillations in singular values of layers in 4 layer MLP with ReLU activations trained on CIFAR-10 dataset (20k) at various learning rates.



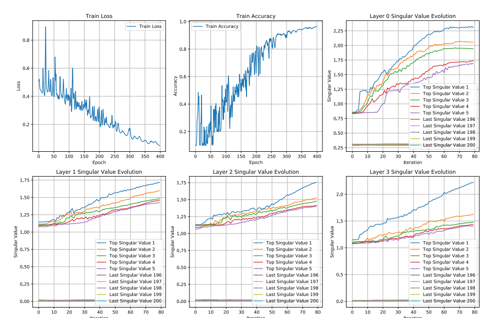
$\eta = 0.1$



$\eta = 0.8$

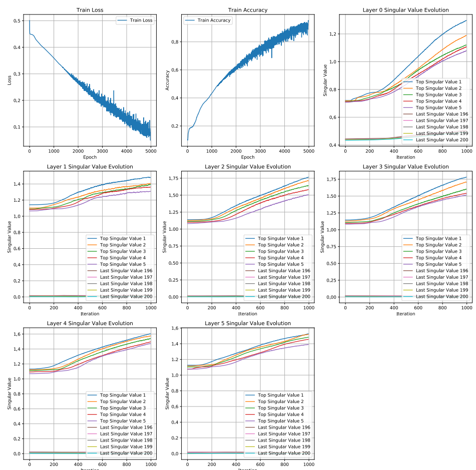


$\eta = 1.0$

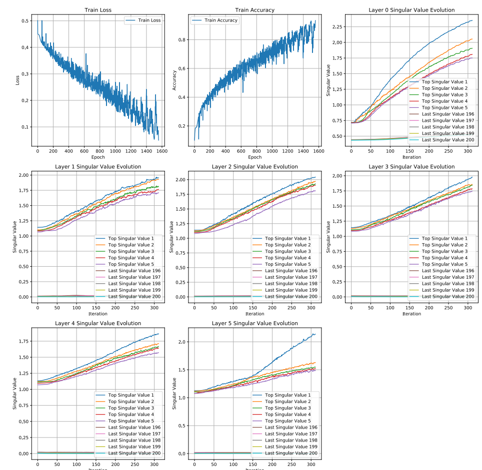


$\eta = 1.5$

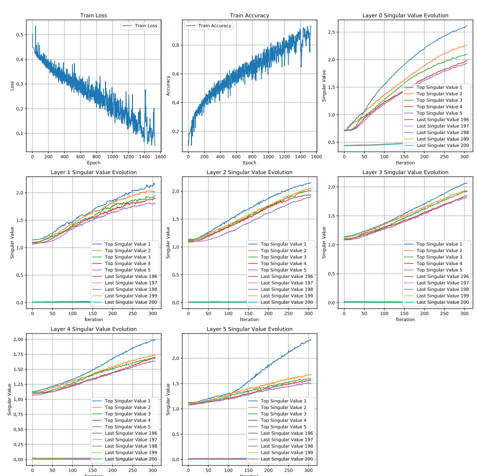
Figure 16: Oscillations in singular values of layers in 4 layer MLP with ReLU activations trained on MNIST dataset (20k) at various learning rates.



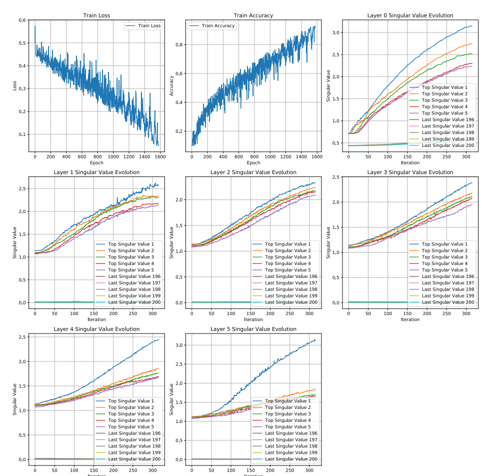
$\eta = 0.1$



$\eta = 0.8$

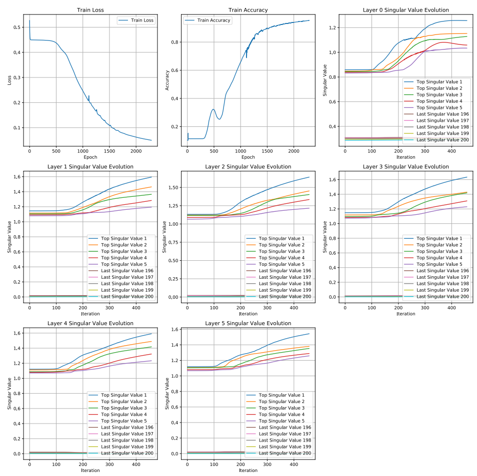


$\eta = 1.0$

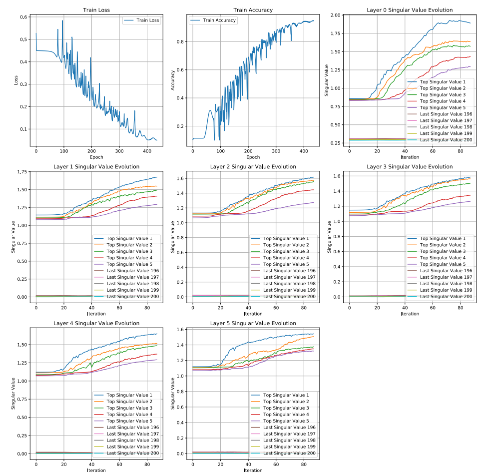


$\eta = 1.5$

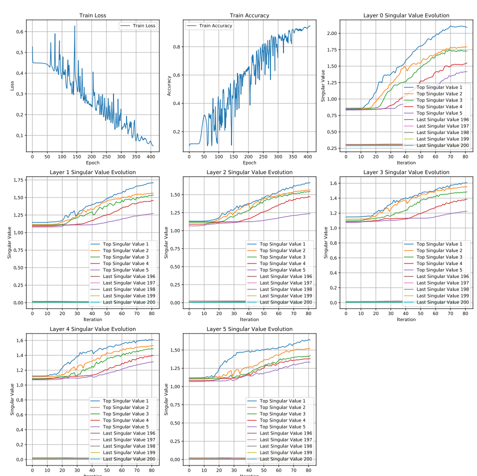
Figure 17: Oscillations in singular values of layers in 6 layer MLP with ReLU activations trained on CIFAR-10 dataset (20k) at various learning rates.



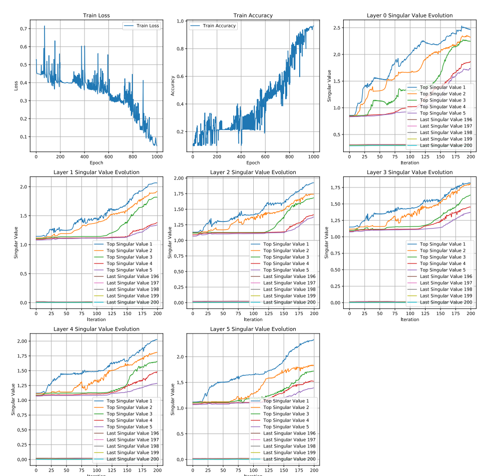
$\eta = 0.1$



$\eta = 0.8$



$\eta = 1.0$



$\eta = 1.5$

Figure 18: Oscillations in singular values of layers in 6 layer MLP with ReLU activations trained on MNIST dataset (20k) at various learning rates.

C Deferred Proofs

In this section, we provide detailed proofs of the theory presented in the main paper. This section is split into two: (i) proofs for the simplicity biases in DLNs and (ii) proofs for the edge of stability.

C.1 Simplicity Biases in Deep Linear Networks

Theorem 1. *Suppose $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$ be a rank-r matrix with SVD $\mathbf{M}^\star = \mathbf{U}^\star\mathbf{\Sigma}^\star\mathbf{V}^{\star\top}$. Let $\mathbf{W}_\ell(t) \in \mathbb{R}^{d\times d}$ denote the ℓ-th weight matrix at GD iterate t. Then, each weight matrix has the following decomposition for all $t \geq 1$:*

$$\mathbf{W}_L(t) = \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}, \qquad \mathbf{W}_\ell(t) = \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t) & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top}, \quad \forall \ell \in [L-1], \tag{14}$$

where

$$\widetilde{\mathbf{\Sigma}}_L(t) = \widetilde{\mathbf{\Sigma}}_L(t-1) - \eta \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t-1) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t-1) - \mathbf{\Sigma}^\star_r\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t-1)$$
$$\widetilde{\mathbf{\Sigma}}(t) = \widetilde{\mathbf{\Sigma}}(t-1) \cdot \left(\mathbf{I}_r - \eta \cdot \widetilde{\mathbf{\Sigma}}_L(t-1) \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t-1) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t-1) - \mathbf{\Sigma}^\star_r\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-3}(t-1)\right),$$

where $\widetilde{\mathbf{\Sigma}}_L(t), \widetilde{\mathbf{\Sigma}}(t) \in \mathbb{R}^{r\times r}$ is a diagonal matrix with $\widetilde{\mathbf{\Sigma}}_L(1) = \eta\alpha^{L-1} \cdot \mathbf{\Sigma}^\star_r$ and $\widetilde{\mathbf{\Sigma}}(1) = \alpha\mathbf{I}_r$.

Proof. We will prove using mathematical induction.

Base Case. For the base case, we will show that the decomposition holds for each weight matrix at $t = 1$. The gradient of $f(\mathbf{\Theta})$ with respect to $\mathbf{W}_\ell$ is

$$\nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}) = \mathbf{W}^\top_{L:\ell+1} \cdot (\mathbf{W}_{L:1} - \mathbf{M}^\star) \cdot \mathbf{W}^\top_{\ell-1:1}.$$

For $\mathbf{W}_L(1)$, we have

$$\begin{aligned}\mathbf{W}_L(1) &= \mathbf{W}_L(0) - \eta \cdot \nabla_{\mathbf{W}_L} f(\mathbf{\Theta}(0)) \\ &= \mathbf{W}_L(0) - \eta \cdot (\mathbf{W}_{L:1}(0) - \mathbf{M}^\star) \cdot \mathbf{W}^\top_{L-1:1}(0) \\ &= \eta\alpha^{L-1}\mathbf{M}^\star \\ &= \mathbf{U}^\star \cdot \left(\eta\alpha^{L-1} \cdot \mathbf{\Sigma}^\star\right) \cdot \mathbf{V}^{\star\top} \\ &= \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}.\end{aligned}$$

Then, for each $\mathbf{W}_\ell(1)$ in $\ell \in [L-1]$, we have

$$\begin{aligned}\mathbf{W}_\ell(1) &= \mathbf{W}_\ell(0) - \eta \cdot \nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}(0)) \\ &= \alpha\mathbf{I}_d,\end{aligned}$$

where the last equality follows from the fact that $\mathbf{W}_L(0) = \mathbf{0}$. Finally, we have

$$\mathbf{W}_\ell(1) = \alpha\mathbf{V}^\star\mathbf{V}^{\star\top} = \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(1) & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top}, \quad \forall \ell \in [L-1].$$

Inductive Step. By the inductive hypothesis, suppose that the decomposition holds. Then, notice that we can simplify the end-to-end weight matrix to

$$\mathbf{W}_{L:1}(t) = \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top},$$

for which we can simplify the gradients to

$$\begin{aligned}\nabla_{\mathbf{W}_L} f(\mathbf{\Theta}(t)) &= \left(\mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}^\star_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}\right) \cdot \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}^{L-1}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top} \\ &= \mathbf{U}^\star \begin{bmatrix} \left(\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}^\star_r\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top},\end{aligned}$$

for the last layer matrix, and similarly,

$$\nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}(t)) = \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}_r^\star\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-2}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}, \quad \ell \in [L-1],$$

for all other layer matrices. Thus, for the next GD iteration, we have

$$\begin{aligned} \mathbf{W}_L(t+1) &= \mathbf{W}_L(t) - \eta \cdot \nabla_{\mathbf{W}_L}(\mathbf{\Theta}(t)) \\ &= \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t) - \eta \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}_r^\star\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top} \\ &= \mathbf{U}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}_L(t+1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{V}^{\star\top}. \end{aligned}$$

Similarly, we have

$$\begin{aligned} \mathbf{W}_\ell(t+1) &= \mathbf{W}_\ell(t) - \eta \cdot \nabla_{\mathbf{W}_\ell}(\mathbf{\Theta}(t)) \\ &= \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t) - \eta \cdot \widetilde{\mathbf{\Sigma}}_L(t) \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}_r^\star\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-2}(t) & \mathbf{0} \\ \mathbf{0} & \alpha \mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top} \\ &= \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t) \cdot \left(\mathbf{I}_r - \eta \cdot \widetilde{\mathbf{\Sigma}}_L(t) \cdot \left(\widetilde{\mathbf{\Sigma}}_L(t) \cdot \widetilde{\mathbf{\Sigma}}^{L-1}(t) - \mathbf{\Sigma}_r^\star\right) \cdot \widetilde{\mathbf{\Sigma}}^{L-3}(t)\right) & \mathbf{0} \\ \mathbf{0} & \alpha \mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top} \\ &= \mathbf{V}^\star \begin{bmatrix} \widetilde{\mathbf{\Sigma}}(t+1) & \mathbf{0} \\ \mathbf{0} & \alpha \mathbf{I}_{d-r} \end{bmatrix} \mathbf{V}^{\star\top}, \end{aligned}$$

for all $\ell \in [L-1]$. This concludes the proof.

□

C.2 Edge of Stability in Deep Linear Networks

Throughout this section, for simplicity in notation, we denote $\sigma_i = \sigma_i^\star$, and this is clarified where necessary. Here, we give a brief overview of the proofs provided in this section.

In Lemma 1, we establish the relation of the Hessian of the original deep matrix factorization loss and loss on the singular value. Our Lemma shows that the eigenvalues of the Hessian of the deep matrix factorization when trained with GD are given as $2\sigma_i^{2-\frac{2}{L}}$ and the rest $N^4L^2 - r$ eigenvalues are zero. Here, we establish that under assumption that target matrix is symmetric, analyzing the eigenvalues of the Hessian of the singular values is sufficient. Then, in Lemma 2, we derive that the sharpness achieved by GD on the singular value loss is $2\sigma_i^{2-\frac{2}{L}}$. This is in fact the minimum value of sharpness achieved among all global minima points. Finally, in Theorem 2, we prove that edge of stability can be observed in the singular value loss by showing the existence of a 2-period orbit oscillations for a learning rate occurring in the edge of stability.

Lemma 1. *Consider running GD on the loss defined in Equation (1) with a symmetric matrix $\mathbf{M}^\star \in \mathbb{R}^{d\times d}$. Then, the eigenvalues of the Hessian with respect to the end-to-end DLN of Equation (1) are equivalent to those of the loss given by*

$$\mathcal{L}(\theta^i) = \frac{1}{2}\left(\prod_{j=1}^{L} w_\ell^i - \sigma_i\right)^2, \quad \forall i \in [d].$$

Proof. We can express the objective function for deep matrix factorization in a vectorized form:

$$f(\mathbf{\Theta}) \coloneqq \frac{1}{2}\|\mathbf{W}_{L:1} - \mathbf{M}^\star\|_{\mathsf{F}}^2 = \frac{1}{2}\|\mathrm{vec}(\mathbf{W}_{L:1}) - \mathrm{vec}(\mathbf{M}^\star)\|_2^2.$$

Then, each block of the Hessian $\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta}) \in \mathbb{R}^{d^2L\times d^2L}$ is given as

$$\left[\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta})\right]_{\ell,m} = \nabla_{\mathrm{vec}(\mathbf{W}_\ell)} f(\mathbf{\Theta}) \nabla^\top_{\mathrm{vec}(\mathbf{W}_m)} f(\mathbf{\Theta}) \in \mathbb{R}^{d^2\times d^2}.$$

By the vectorization trick, each vectorized layer matrix has an SVD of the form $\mathrm{vec}(\mathbf{W}_\ell) = \mathrm{vec}(\mathbf{U}_\ell \mathbf{\Sigma}_\ell \mathbf{V}_\ell^\top) = (\mathbf{V}_\ell \otimes \mathbf{U}_\ell) \cdot \mathrm{vec}(\mathbf{\Sigma}_\ell)$. Then, by Theorem 1, notice that we have

$$\nabla_{\mathrm{vec}(\mathbf{W}_\ell)} f(\mathbf{\Theta}(t)) = (\mathbf{V}_\ell \otimes \mathbf{U}_\ell) \cdot \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta}(t)).$$

Now, each block of the Hessian is given by

$$\begin{aligned}\nabla_{\mathrm{vec}(\mathbf{W}_\ell)} f(\mathbf{\Theta}) \nabla^\top_{\mathrm{vec}(\mathbf{W}_m)} f(\mathbf{\Theta}) &= \nabla^\top_{\mathrm{vec}(\mathbf{W}_m)} \cdot (\mathbf{V}_\ell \otimes \mathbf{U}_\ell) \cdot \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta}) \\ &= (\mathbf{V}_\ell \otimes \mathbf{U}_\ell) \cdot \nabla^\top_{\mathrm{vec}(\mathbf{W}_m)} f(\mathbf{\Theta}) \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta}) \\ &= (\mathbf{V}_\ell \otimes \mathbf{U}_\ell) \cdot \nabla^\top_{\mathrm{vec}(\mathbf{\Sigma}_m)} \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta}) \cdot (\mathbf{V}_m \otimes \mathbf{U}_m)^\top,\end{aligned}$$

where we applied the invariance property in the last line. Notice that the curvature of the Hessian of the loss with respect to the original weight matrices simply depend on the curvature of the loss with respect to the singular values.

Now, let w_ℓ^i denote the i-th singular value entry of $\mathbf{\Sigma}_\ell$. Let us define $\mathbf{w}_\ell \in \mathbb{R}^d$ as a vector containing all of the diagonal elements of $\mathbf{\Sigma}_\ell$,

$$\mathbf{w}_\ell = [w_\ell^1 \; w_\ell^2 \; \ldots \; w_\ell^d]^\top.$$

Note that $\mathbf{\Sigma}_\ell = diag(\mathbf{w}_\ell)$, so, $\nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta}) = \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta}) \otimes \mathbf{e}_1$. This is because vectorizing $\mathbf{\Sigma}_\ell$ pads additional d zeroes. So taking the second derivative, gives us the he relationship between $\nabla^\top_{\mathrm{vec}(\mathbf{\Sigma}_m)} f(\mathbf{\Theta}) \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta})$ and $\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})$ which is given as:

$$\underbrace{\nabla^\top_{\mathrm{vec}(\mathbf{\Sigma}_m)} f(\mathbf{\Theta}) \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta})}_{\mathbb{R}^{d^2 \times d^2}} = \underbrace{\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})}_{\mathbb{R}^{d \times d}} \otimes (\mathbf{e}_1 \mathbf{e}_1^\top),$$

where $\mathbf{e}_1 \in \mathbb{R}^d$ is the first elementary basis vector. This result also states that the non-zeros eigenvalues of $\nabla^\top_{\mathrm{vec}(\mathbf{\Sigma}_m)} f(\mathbf{\Theta}) \nabla_{\mathrm{vec}(\mathbf{\Sigma}_\ell)} f(\mathbf{\Theta})$ are the same as those of $\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})$. Then, notice that $\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})$ can be computed as

$$\left[\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})\right]_{i,j} = \frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i}$$

For $i = j$ and $i > r$,

$$\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i}\right) = \left(\prod_{k \neq l} w_k^i\right) \left(\prod_{k \neq m} w_k^i\right) + \left(\prod_{k} w_k^i - \sigma_i\right) \left(\prod_{k \neq l, k \neq m} w_k\right)$$

So, if either $l \neq L$ and $m \neq L$, then $\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i}\right) = 0$ since $w_L^i = 0$, for all i. This makes $\nabla^\top_{\mathbf{w}_m} f(\mathbf{\Theta}) \nabla_{\mathbf{w}_\ell} f(\mathbf{\Theta})$ to be a diagonal matrix with rank r. Hence, the overall Hessian for deep matrix factorization is given by

$$\begin{aligned}\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta}) &= \left[\nabla_{\mathrm{vec}(\mathbf{W}_\ell)} \nabla_{\mathrm{vec}(\mathbf{W}_m)^\top} f(\mathbf{\Theta})\right]_{l,m=1,2,\ldots,L} \\ &= \left[(\mathbf{V}_\ell \otimes \mathbf{U_l}) \nabla_{\mathrm{vec}(\mathbf{\Sigma_m})^\top} \nabla_{\mathrm{vec}(\mathbf{\Sigma_l})} f(\mathbf{\Theta}(t)) (\mathbf{V}_m \otimes \mathbf{U}_m)^\top\right]_{l,m=1,2,\ldots,L} \\ &= \left[(\mathbf{V}_\ell \otimes \mathbf{U_l}) \left(\nabla_{\mathbf{W}_m} \nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}(t)) \otimes (\mathbf{e}_1 \mathbf{e}_1^\top)\right) (\mathbf{V}_m \otimes \mathbf{U}_m)^\top\right]_{l,m=1,2,\ldots,L} \\ &= \left[(\mathbf{V}_\ell \otimes \mathbf{U_l}) \left(\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i}\right)_{i,j} \otimes (\mathbf{e}_1 \mathbf{e}_1^\top)\right) (\mathbf{V}_m \otimes \mathbf{U}_m)^\top\right]_{l,m=1,2,\ldots,L}\end{aligned}$$

Now, since $\mathbf{M}$ is a symmetric matrix, we have $\mathbf{U}_\ell = \mathbf{V}_\ell$ and $\mathbf{U}_m = \mathbf{V}_m$, so the Hessian is simplified to:

$$\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta}) = \left[(\mathbf{V}_\ell \otimes \mathbf{V}_\ell) \left(\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i}\right)_{i,j} \otimes (\mathbf{e}_1 \mathbf{e}_1^\top)\right) (\mathbf{V}_m \otimes \mathbf{V}_m)^\top\right]_{l,m=1,2,\ldots,L}$$

In Lemma 2, we calculated $\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^i \partial w_m^i}\right)_{l,m}$, which is a matrix representing the second-order partial derivatives of the loss function L with respect to the weights w_l^i and w_m^i.

At convergence for gradient descent (GD), this matrix was found to be rank 1 with eigenvalue $2\sigma_i^{2-\frac{2}{L}}$. This means that at convergence, the Hessian matrix $\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^i \partial w_m^i}\right)_{l,m}$ has only one non-zero eigenvalue $2\sigma_i^{2-\frac{2}{L}}$, indicating that it is a rank 1 matrix. Let us denote

$$H_1 = \left[\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^i \partial w_m^j}\right)_{l,m=1,.,L}\right]_{i,j=1,..,n}$$

$$= \begin{pmatrix} \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^1 \partial w_m^1}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^1 \partial w_m^2}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^1 \partial w_m^3}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^1 \partial w_m^n}\right) \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^2 \partial w_m^1}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^2 \partial w_m^2}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^2 \partial w_m^3}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^2 \partial w_m^N}\right) \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^3 \partial w_m^1}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^3 \partial w_m^2}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^3 \partial w_m^3}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^3 \partial w_m^n}\right) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^n \partial w_m^1}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^n \partial w_m^2}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^n \partial w_m^3}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_l^n \partial w_m^n}\right) \end{pmatrix}$$

$$= \begin{pmatrix} H_1(1,1) & H_1(1,2) & H_1(1,3) & \cdots & H_1(1,L) \\ H_1(2,1) & H_1(2,2) & H_1(2,3) & \cdots & H_1(2,L) \\ H_1(3,1) & H_1(3,2) & H_1(3,3) & \cdots & H_1(3,L) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ H_2(L,1) & H_2(L,2) & H_2(L,3) & \cdots & H_2(L,L) \end{pmatrix}$$

and also denote

$$H_2 = \left[\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^i \partial w_m^j}\right)_{i,j=1,..,n}\right]_{l,m=1,.,L}$$

$$= \begin{pmatrix} \left(\frac{\partial^2 \mathcal{L}}{\partial w_1^i \partial w_1^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_1^i \partial w_2^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_1^i \partial w_3^j}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_1^i \partial w_n^j}\right) \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_2^i \partial w_1^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_2^i \partial w_2^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_2^i \partial w_3^j}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_2^i \partial w_L^j}\right) \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_3^i \partial w_1^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_3^i \partial w_2^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_3^i \partial w_3^j}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_3^i \partial w_n^j}\right) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \left(\frac{\partial^2 \mathcal{L}}{\partial w_L^i \partial w_1^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_L^i \partial w_2^j}\right) & \left(\frac{\partial^2 \mathcal{L}}{\partial w_L^i \partial w_3^j}\right) & \cdots & \left(\frac{\partial^2 \mathcal{L}}{\partial w_L^i \partial w_L^j}\right) \end{pmatrix}$$

$$= \begin{pmatrix} H_2(1,1) & H_2(1,2) & H_2(1,3) & \cdots & H_2(1,L) \\ H_2(2,1) & H_2(2,2) & H_2(2,3) & \cdots & H_2(2,L) \\ H_2(3,1) & H_2(3,2) & H_2(3,3) & \cdots & H_2(3,L) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ H_2(L,1) & H_2(L,2) & H_2(L,3) & \cdots & H_2(L,L) \end{pmatrix}$$

Note that H_1 and H_2 are related by a permutation matrix, since the hessian is obtained in each case, are after rearranging the variables, the eigenvalues of H_1 and H_2 are the same.

Next, in lemma-2, we obtained the diagonal blocks of H_1 , i.e, $H_1(i,i)$ which was rank 1 and had eigenvalue to be $2s_i^{2-\frac{2}{L}}$. And the off-diagonal blocks $H_1(i,j) = \mathbf{0}$.

So, this makes, H_1 to be a block diagonal matrix with eigenvalues $\|H_1(1,1)\|_2 = 2s_1^{2-\frac{2}{L}}, \|H_1(2,2)\|_2 = 2s_2^{2-\frac{2}{L}}, .., \|H_1(r,r)\|_2 = 2s_r^{2-\frac{2}{L}}$ (which are the only eigenvalue of each block).

For H_2, at convergence for GD, all the blocks are same, $H_2(1,1) = H_2(1,2) = ...H_2(L,L)$ and each such block is diagonal with rank r. So, the overall rank of the block matrix H_2 is still r (as repitition of the block matrix merely increases the number of zero eigenvalues but keeps the non-zero eigenvalues the same).

Now, establishing the connection between the block matrix whose eigenvalues we derived and the hessian of the original loss, we are left with the last step. The Hessian of the original loss:

$$\begin{aligned}
\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta}) &= \left[\nabla_{\mathrm{vec}(\mathbf{W}_\ell)} \nabla_{\mathrm{vec}(\mathbf{W}_m)^\top} f(\mathbf{\Theta})\right]_{l,m=1,2,\ldots,L} \\
&= \left[(\mathbf{V}_\ell \otimes \mathbf{V}_\ell) \left(\left(\frac{\partial^2 \mathcal{L}}{\partial w_l^j \partial w_m^i} \right)_{i,j} \otimes (\mathbf{e}_1 \mathbf{e}_1^\top) \right) (\mathbf{V}_m \otimes \mathbf{V}_m)^\top \right]_{l,m=1,2,\ldots,L} \\
&= \left[(\mathbf{V}_\ell \otimes \mathbf{V}_\ell) \left(H_2(i,j) \otimes (\mathbf{e}_1 \mathbf{e}_1^\top)\right) (\mathbf{V}_m \otimes \mathbf{V}_m)^\top \right]_{l,m=1,2,\ldots,L}
\end{aligned}$$

Since, we already showed that $H_2(i,j)$ is a rank r matrix with eigenvalues $s_i^{2-\frac{2}{L}}, i-1,2,...r.$, note that $H_2(i,j) \otimes (\mathbf{e}_1 \mathbf{e}_1^\top)$ also has the same eigenvalues and rank. Now, we observe that every block matrix in $\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta})$ has the same eigenvalues. This is because:

1. Multiplication by orthogonal matrices $(\mathbf{V}_m \otimes \mathbf{V}_m)$ and $(\mathbf{V}_\ell \otimes \mathbf{V}_\ell)$ does not change the rank or the eigenvalues of the matrix.
2. Each block has the same set of orthogonal matrices multiplied on both sides (due to the symmetric assumption). So, the eigenvalues and rank of $\nabla^2_{\mathbf{\Theta}} f(\mathbf{\Theta})$ and H_2 are the same.

With this, we show that that the eigenvalues of the Hessian for GD for the deep matrix factorization loss at convergence are $2s_i^{2-\frac{2}{L}}, i = 1,2,...r.$ ☐

Lemma 2. *Consider the i^{th} singular value loss on L variables $\mathcal{L}(w_L^i, ..w_2^i, w_1^i) = \frac{1}{2}(w_L^i \prod_{l=2}^L w_\ell^i - \sigma_i)^2$, then for Gradient descent on the loss with initialization $(w_L^i(0), ..w_2^i(0), w_1^i(0)) = (0, \alpha, \alpha.., \alpha)$, prior to GD oscillations would converge to a point where the sharpness achieved is given as $\|\nabla^2 \mathcal{L}\|_2 = 2s_i^{2-\frac{2}{L}}$. Furthermore, the sharpness of the final point achieved by Gradient Flow is larger provably.*

Proof. For sake of notation and easy proof writing, we will slightly alter the notation to $w_L^i = x$, $w_{L-1}^i = y_1$, $w_{L-2}^i = y_2$,.., $w_1^i = y_N$, $\sigma_i = s$ and $L = N+1$. Since, the loss is wrt $N+1$ variables, we will start by calculating the Hessian matrix $\nabla^2 L$ which will be $(N+1) \times (N+1)$ symmetric matrix. So, given $\mathcal{L}(x, y_1, .., y_N) = \frac{1}{2}(x \prod_{j=1}^N y_j - s)^2$,

First Derivatives:

$$\frac{\partial L}{\partial x} = \left(x \prod_{j=1}^N y_j - s \right) \cdot \prod_{i=1}^N y_i, \quad \frac{\partial L}{\partial y_i} = \left(x \prod_{j=1}^N y_j - s \right) \cdot x \prod_{j \neq i}^N y_j \quad \text{for all } i = 1, 2.., N$$

Second Derivatives:

$$\nabla_x^2 \mathcal{L} = \left(\prod_{i=1}^N y_i \right)^2, \quad \nabla_{y_j}^2 \mathcal{L} = \left(x \prod_{i \neq j}^N y_i \right)^2,$$

$$\nabla_x \nabla_{y_i} \mathcal{L} = \nabla_{y_i} \nabla_x \mathcal{L} = \left(x \prod_{j=1}^{N} y_j - s \right) \prod_{j \neq i}^{N} y_j + x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \quad \text{for all } i = 1, 2.., N$$

$$\nabla_{y_i} \nabla_{y_j} \mathcal{L} = x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k + x \left(x \prod_{j=1}^{N} y_j - s \right) \prod_{k \neq i, k \neq j} y_k \quad \text{for all } i = 1, 2.., N \text{ and } j = 1, 2.., N$$

648 Calculating the elementwise Hessian, the $(N+1) \times (N+1)$ can be written as a block matrix
649 structure:

$$\nabla^2 \mathcal{L}(x, y_1, .., y_N) = \left[\begin{array}{c|c} \nabla_x^2 \mathcal{L} & \nabla_x \nabla_{y_i} \mathcal{L}_{i=1,2,..N} \\ \hline (\nabla_x \nabla_{y_i} \mathcal{L}_{i=1,2,..N})^\top & (\nabla_{y_i} \nabla_{y_j} \mathcal{L})_{i,j=1,2,..N} \end{array} \right]$$

650 where $(\nabla_{y_i} \nabla_{y_j} \mathcal{L})_{i,j=1,2,..N}$ is an $N \times N$ matrix with the ij^{th} element being $\nabla_{y_i} \nabla_{y_j} \mathcal{L}$.

651 $\nabla_x \nabla_{y_i} \mathcal{L}$ is a $(1 \times N)$ vector with the i^{th} element being $\nabla_x \nabla_{y_i} \mathcal{L}$.

652 Putting the expressions for the second order derivatives at the minima $(x \prod_{j=1}^{N} y_j - s) = 0$, we get:

$$\nabla^2 \mathcal{L}_{(x \prod_{j=1}^{N} y_j = s)}(x, y_1, .., y_N) = \left[\begin{array}{c|c} (\prod_{i=1}^{N} y_i)^2 & \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right]_j \\ \hline \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right]_j & \left[x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k \right]_{i,j} \end{array} \right] =: \mathbf{H}$$

653 Since, due to same initiaization, all y_i's are same throughout. Note that due to repetition, the matrix
654 $\mathbf{H}$ can be represented by sum of few rank-1 outer-products as follows:

$$\mathbf{H} = (\prod_{i=1}^{N} y_i)^2 \mathbf{e_1} \mathbf{e_1}^\top + \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right]_j \mathbf{e_2} \mathbf{e_1}^\top + \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right]_j \mathbf{e_1} \mathbf{e_2}^\top + x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k \mathbf{e_2} \mathbf{e_2}^\top$$

655 where $\mathbf{e_1} = [1, 0, 0, ...0]^\top$ and $\mathbf{e_2} = \frac{1}{N}[0, 1, 1, ...1]^\top$.

656 So, it is easy to observe that the span of the eigenvector of $\mathbf{H}$ will be $span(\mathbf{e_1}, \mathbf{e_2})$. Say eigenvector v
657 of $\mathbf{H}$ is written as $v = a\mathbf{e_1} + b\mathbf{e_2}$ with $a^2 + b^2 = 1$, since eigenvector has unit norm. Then we have:

$$\begin{aligned}
\mathbf{H}(a\mathbf{e_1} + b\mathbf{e_2}) &= (\prod_{i=1}^{N} y_i)^2 a\mathbf{e_1} + ax \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \mathbf{e_2} + bx \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \mathbf{e_2} + bx^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k \mathbf{e_1} \\
&= ((\prod_{i=1}^{N} y_i)^2 a + b \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right]) \mathbf{e_1} + (a \left[x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \right] + bx^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k) \mathbf{e_2}
\end{aligned}$$

658 which shows that H maps $span(\mathbf{e_1}, \mathbf{e_2})$ to itself. We used the fact that $\mathbf{e_1}^\top \mathbf{e_2} = 0$ and $\mathbf{e_1}^\top \mathbf{e_1} =$
659 $\mathbf{e_2}^\top \mathbf{e_2} = 1$. Now, by definition of eigenvector:

$$\mathbf{H}(a\mathbf{e_1} + b\mathbf{e_2}) = \lambda(a\mathbf{e_1} + b\mathbf{e_2})$$

660 So, using the above two equations, we get the linear system of equations as follows:

$$\begin{bmatrix} (\prod_{i=1}^{N} y_i)^2 - \lambda & x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \\ x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i & x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k - \lambda \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Since, $a^2 + b^2 = 1$, we can't have $a = 0, b = 0$, so it must hold that

$$det \begin{bmatrix} (\prod_{i=1}^{N} y_i)^2 - \lambda & x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i \\ x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i & x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k - \lambda \end{bmatrix} = 0$$

This gives us a quadratic equation on λ as follows:

$$\begin{aligned}
&((\prod_{i=1}^{N} y_i)^2 - \lambda)(x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k - \lambda) - (x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i)^2 = 0 \\
\implies &\lambda^2 - \lambda((\prod_{i=1}^{N} y_i)^2 + x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k) + (x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i)^2 - (x \prod_{j \neq i}^{N} y_j \prod_{i=1}^{N} y_i)^2 = 0 \\
\implies &\lambda(\lambda - (\prod_{i=1}^{N} y_i)^2 + x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k) = 0
\end{aligned}$$

Since, the matrix is rank-1 by repetition of values, the largest eigenvalue corresponds to

$$\begin{aligned}
\lambda(x, y_1, ..., y_N) &= (\prod_{i=1}^{N} y_i)^2 + x^2 \prod_{k \neq j}^{N} y_k \prod_{k \neq i}^{N} y_k \\
\implies \lambda(x, y_1, ..., y_N) &= (\prod_{i \neq j}^{N} y_i)^2 (x^2 + y_j^2)
\end{aligned}$$

The last line is due to the fact that $y_j = y_i$ due to the same initialization.

Now, to find the the solution $(x, y_1, ..., y_N)$ that gives the smallest value of λ subject to the constraint $x y_j \prod_{i \neq j}^{N} y_i = s$, we substitute y_j from the constraint:

$$\lambda(x, y_1, ..., y_N) = (\prod_{i \neq j}^{N} y_i)^2 (x^2 + \frac{s^2}{(\prod_{i \neq j}^{N} y_i)^2 x^2})$$

To make sure, that the minimum eigenvalue λ is reached for a choice of $(x, y_1, ..., y_N)$, we need to ensure that $\frac{\partial \lambda}{\partial x}$ and $\frac{\partial \lambda}{\partial y_i}$ for all $i = 1, 2, ..N$ equates to 0. Furthermore, the second derivative $\frac{\partial^2 \lambda}{\partial^2 x}$ and $\frac{\partial^2 \lambda}{\partial^2 y_i}$ for all i are strictly positive.

$$\begin{aligned}
\frac{\partial \lambda}{\partial x} &= (\prod_{i \neq j}^{N} y_i)(2x - \frac{2s^2}{(\prod_{i \neq j}^{N} y_i)^2 x^3}) = 0 \\
\implies x^2 &\prod_{i \neq j}^{N} y_i = s
\end{aligned}$$

This equality combined with constraint $x y_j \prod_{i \neq j}^{N} y_i = x \prod_{i=1}^{N} y_i = s$, This relation gives us the solution for each of $x = y_1 = y_2 = ... = y_N = s^{\frac{1}{N+1}}$, as all of the y_j are equivalent.

Furthermore, we see that:

$$\frac{\partial^2 \lambda}{\partial^2 x} = (\prod_{i \neq j}^{N} y_i)(2 + \frac{6s^2}{(\prod_{i\neq j}^{N} y_i)^2 x^4}) > 0$$

Hence, $x = y_1 = y_2 = ... = y_N = s^{\frac{1}{N+1}}$ is unique minima for $\lambda(x, y_1, ..., y_N)$.

Note that in equation for λ and the constraint, x and y_j are interchangable, implying that $\frac{\partial \lambda}{\partial y_j} = 0$ and $\frac{\partial^2 \lambda}{\partial^2 y_j} > 0$ at that particular solution $x = y_1 = y_2 = ... = y_N = s^{\frac{1}{N+1}}$, i.e. it is a unique minima for all x and y_j.

From Conjecture 1, we showed that due to the balancing effect of GD, the solution found by GD with large step-size (just before oscillation) is $x = y_1 = y_2 = .. = y = s^{\frac{1}{N+1}}$. So, GD indeed finds the flattest minima of the loss curve.

Putting this solution into the value of λ, we obtain:

$$\lambda(\hat{x}, \hat{y}_1, ..., \hat{y}_N) = 2s^{\frac{2N}{N+1}} = 2s^{2-\frac{1}{N+1}}$$

Reverting to the earlier notation, we obtain $\|\nabla^2 \mathcal{L}\| = 2s^{2-\frac{2}{L}}$.

□

Theorem 2 (Periodic Orbit at the Edge of Stability)**.** *Consider the objective function $\mathcal{L}(\cdot)$ in Equation (6), where $\sigma_i^\star$ is a singular value for a PSD target matrix $\mathbf{M}^\star$. Let $\mathrm{GD}_\eta(\cdot)$ denote one GD step with learning rate η:*

$$\mathrm{GD}_\eta(w_\ell^i f(\mathbf{\Theta}(t))) \coloneqq w_\ell^i(t+1) = w_\ell^i f(\mathbf{\Theta}(t)) - \eta \cdot \nabla_{w_\ell^i} \mathcal{L}(\theta^i(t)),$$

and define $s \coloneqq {\sigma_i^\star}^{\frac{1}{L}}$. Then, under Conjecture 1, for any $\epsilon > 0$ and any point $w_\ell^i f(\mathbf{\Theta}(t)) \in [s - \epsilon, s]$, there exists a learning rate $\frac{2}{\mathcal{L}''(s)} < \eta < \frac{2}{\mathcal{L}''(s) - \epsilon \mathcal{L}'''(s)}$ such that $w_\ell^i(t+2) = \mathrm{GD}_\eta(\mathrm{GD}_\eta(w_\ell^i f(\mathbf{\Theta}(t)))) = w_\ell^i f(\mathbf{\Theta}(t))$.

Proof. Note that the loss on each singular value is $(w_L^i w_{L-1}^i .. w_1^i - s_i)^2$. Since due to balancedness property of GD, all the variables get coupled say to y^L: We take $L = N + 1$. Define the loss function as follows:

$$f(y) = (y^{N+1} - s)^2$$

First Derivative

The first derivative of f with respect to y is:

$$f'(y) = 2(N+1)y^N(y^{N+1} - s)$$

Second Derivative

The second derivative of f is given by:

$$f''(y) = 2(N+1)\left[Ny^{2N} + (N+1)y^{2N} - Nsy^N\right]$$

Evaluation at Local Minima

At the local minimum $\hat{y} = s^{\frac{1}{N+1}}$, the second derivative evaluates to:

$$f''(\hat{y}) = 2(N+1)^2 \hat{y}^{2N} = 2(N+1)^2 s^{\frac{2N}{N+1}}$$

Third Derivative

The third derivative of f is:

$$f'''(y) = 2(N+1)[3N(N+1)y^{2N-1}]$$

And evaluated at the local minimum $\hat{y}$:

$$f'''(\hat{y}) = 6(N+1)^2 N\hat{y}^{2N-1} = 6(N+1)^2 N s^{\frac{2N-1}{N+1}}$$

By inspection, we note that $f^{'''}(\hat{y}) > 0$ indicating that self-stabilization phenomenon may occur and iterates will not blow up even if $h > \frac{2}{f''(y)}$. Let $y_0 = \hat{y} - \epsilon$ ($\epsilon > 0$) be a point close to the minima $\hat{y}$ we want to prove that after two steps of gradient descent with learning rate $h > \frac{2}{f''(\hat{y})}$, it returns to the same point y_0. We do this using local Taylor series approximation. The proof strategy is motivated from the work in [23].

$$\begin{aligned}
&y_0 = \hat{y} - \epsilon,\\
&\quad f'(y_0) = f'(\hat{y}) - f^{''}(\hat{y})\epsilon^2 + \frac{1}{2} f^3(\hat{y})\epsilon^2 - \frac{1}{6} f^4 \epsilon^3 + \mathcal{O}(\epsilon^4),\\
&\quad = -f^{''}\epsilon + \frac{1}{2} f^3\epsilon^2 - \frac{1}{6} f^4\epsilon^3 + \mathcal{O}(\epsilon^4),\\
&\quad y_1 = y_0 - hf^{'}(y_0) = \hat{y} - \epsilon - h(-f^{''}\epsilon + \frac{1}{2} f^3\epsilon^2 - \frac{1}{6} f^4\epsilon^3) + \mathcal{O}(\epsilon^4),\\
&\quad f^{'}(y_1) = f^{''}(y_1 - \hat{y}) + \frac{1}{2} f^3(y_1 - \hat{y})^2 + \frac{1}{6} f^4(y_1 - \hat{y})^3 + \mathcal{O}(\epsilon^4),\\
&\quad y_2 = y_1 - hf^{'}(y_1),\\
&\quad \frac{y_2 - y_0}{h} = -(-f^{''}\epsilon + \frac{1}{2} f^3\epsilon^2 - \frac{1}{6} f^4\epsilon^3) - f^{''}(-\epsilon - h(-f^{''}\epsilon + \frac{1}{2} f^3\epsilon^2 - \frac{1}{6} f^4\epsilon^3))\\
&\quad - \frac{1}{2} f^3(-f^{''}\epsilon + \frac{1}{2} f^3\epsilon^2 - \frac{1}{6} f^4\epsilon^3) - \frac{1}{6} f^4(-\epsilon - h(-f^{''}\epsilon))^3 + \mathcal{O}(\epsilon^4)
\end{aligned}$$

When $h = \frac{2}{f''}$, we observe that

$$\frac{y_2 - y_0}{h} = (\frac{1}{2} h(f^{(3)})^2 - \frac{1}{3} f^4)\epsilon^3 + \mathcal{O}(\epsilon^4)$$

which is positive if $(\frac{1}{2} h(f^{(3)})^2 - \frac{1}{3} f^4) = \frac{1}{3f''}(3f^{(3)})^2 - f^{''} f^{(4)}) > 0$.

Furthermore, when $h = \frac{2}{f''(\hat{y}) - \epsilon f'''(\hat{y})}$, then $hf^{''} = 2 + 2\frac{f^3}{f''}\epsilon + \mathcal{O}(\epsilon^2)$, so

$$\frac{y_2 - y_0}{h} = -2f^3\epsilon^2 + \mathcal{O}(\epsilon^3),$$

which is negative since when ϵ is sufficiently small and we already have $f^3(\hat{y}) > 0$.

Note that the loss f is continous and $(N+1)$-times differentiable, so $y_2 - y_0$ is also continous. Now, as $y_2 - y_0$ is positive for $h = \frac{2}{f''}$ and negative for a larger learning rate $h = \frac{2}{f''(\hat{y}) - \epsilon f'''(\hat{y})}$. So there must exist $\frac{2}{f''(\hat{y})} < \eta < \frac{2}{f''(\hat{y}) - \epsilon f'''(\hat{y})}$, such that $y_2 = y_0$ by continuity.

To complete the theorem, we need to prove that $f^3(\hat{y}) > 0$ and $(3f^{(3)})^2 - f^{''} f^{(4)}) > 0$ at $y = \hat{y}$. To avoid computing the fourth order derivative of the loss (f^4), we will impose conditions on a reparaterized version of the loss.

Let $f(y) = (g(y) - s)^2$, then by definition we have

$$\begin{aligned}
f^{''}(y) &= 2(g(y) - s)g^{''}(y) + 2(g^{'}(y))^2\\
f^{'''}(y) &= 2(g(y) - s)g^3(y) + 6g^{''}(y)g^{''}(y)\\
f^4(y) &= 2(g(y) - s)g^{(4)}(y) + 6(g^{''}(y))^2 + 8g^{'}(y)g^{(3)}(y)
\end{aligned}$$

At minima, $y = \hat{y}$, $g(\hat{y}) = s$, where we have $f^{''}(y) = 2(g'(y))^2$, $f^{(3)}(y) = 6g^{''}(y)g^{'}(y)$ and $f^{(4)}(y) = 6(g^{''}(y))^2 + 8g^{'}(y)g^{(3)}(y)$. The earlier condition on $f^3(\hat{y}) \neq 0$ implies that $g^{'}(y) \neq 0$. And the condition which was $(3(f^{(3)})^2 - f^{''}f^{(4)}) > 0$ would imply that

$$\begin{aligned} &108(g''(y))^2(g'(y))^2 - 2(g'(y))^2\left(6(g''(y))^2 + 8g'(y)g^3(y)\right) > 0 \\ &\implies 6(g''(y))^2 > g'(y)g^3(y) \end{aligned}$$

For our case, $g(y) = y^{N+1}$, so $g(\hat{y}) \neq 0$ (fulfilling condition-1) and furthermore, $g'(y) = (N+1)y^N$, $g''(y) = N(N+1)y^{N-1}$ and $g'''(y) = N(N+1)(N-1)y^{N-2}$. Putting the above expression in the condition before we get

$$\begin{aligned} &6(N(N+1)y^{N-1})^2 > (N+1)y^N N(N+1)(N-1)y^{N-2} \\ &\implies 6(N(N+1))^2 - N(N+1)^2(N-1) > 0 \\ &\implies 5N + 1 > 0 \end{aligned}$$

which is indeed true for any $N > 1$. This means that we need $L > 2$, to observe period-2 orbit oscillation. This is because the second derivative of the loss is constant when $L = 1$, and any $\eta > \frac{2}{f''(y)}$ would make the loss blow up in that case. This completes the Lemma. ☐

D Auxiliary Results

In this section, we provide an additional auxiliary result that we are able to prove using our theory on singular vector invariance. In the literature, there is a popular notion that there is a correlation between the flatness of a minima and generalization. Here, we present a preliminary result that this may also be the case for DLNs, where flatness is measured by the trace of the Hessian. To do so, we first compute the trace of the Hessian with respect to the deep matrix factorization loss in Lemma 1.

Lemma 1. *Let $\mathbf{W}_{L:1}(t) \in \mathbb{R}^{d\times d}$ denote the end-to-end DLN at GD iterate t. Then, the trace of Hessian of Equation (1) with respect to $\mathbf{W}_{L:1}(t)$ is given by*

$$tr\left(\nabla^2_{\mathbf{W}_{L:1}(t)} f(\mathbf{\Theta}(t))\right) = \sum_{\ell=1}^{L} \|\mathbf{W}_{\ell-1:1}(t)\|_{\mathsf{F}}^2 \cdot \|\mathbf{W}_{L:\ell+1}(t)\|_{\mathsf{F}}^2. \tag{15}$$

Proof. We will use the following properties of the Kronecker product throughout this proof:

$$\begin{aligned} (\mathbf{A}\otimes\mathbf{B})^\top &= \mathbf{A}^\top \otimes \mathbf{B}^\top && \text{(Transpose Property)} \\ (\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\otimes\mathbf{D}) &= \mathbf{AC}\otimes\mathbf{BD} && \text{(Distributive Property)} \\ \mathrm{tr}(\mathbf{A}\otimes\mathbf{B}) &= \mathrm{tr}(\mathbf{A})\cdot\mathrm{tr}(\mathbf{B}) && \text{(Trace Property)} \end{aligned}$$

We can express the objective function for deep matrix factorization in a vectorized form:

$$f(\mathbf{\Theta}) \coloneqq \frac{1}{2}\|\mathbf{W}_{L:1} - \mathbf{M}^\star\|_{\mathsf{F}}^2 = \frac{1}{2}\|\mathrm{vec}(\mathbf{W}_{L:1}) - \mathrm{vec}(\mathbf{M}^\star)\|_2^2. \tag{16}$$

Then, notice that for any weight matrix $\mathbf{W}_\ell$, we can write

$$f(\mathbf{\Theta}) = \frac{1}{2}\|\mathrm{vec}(\mathbf{W}_{L:1}) - \mathrm{vec}(\mathbf{M}^\star)\|_2^2 = \frac{1}{2}\left\|(\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1})\cdot\mathrm{vec}(\mathbf{W}_\ell) - \mathrm{vec}(\mathbf{M}^\star)\right\|_2^2 \tag{17}$$

Let us define $\mathbf{Z} \coloneqq (\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1})$. The gradient of Equation (17) with respect to the vectorized weight matrix $\mathbf{W}_\ell$ is

$$\nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta}) = \mathbf{Z}^\top\mathbf{Z}\cdot\mathrm{vec}(\mathbf{W}_\ell) - \mathbf{Z}^\top\cdot\mathrm{vec}(\mathbf{M}^\star). \tag{18}$$

Then, notice that for the trace of the Hessian, we only need to consider the diagonal elements of the Hessian, which involves taking the gradient of $\nabla_{\mathbf{W}_\ell} f(\mathbf{\Theta})$ with respect to the vectorized $\mathbf{W}_\ell$:

$$\begin{aligned} \left[\nabla^2 f(\mathbf{\Theta})\right]_{\ell,\ell} &= \mathbf{Z}^\top\mathbf{Z} \\ &= (\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1})^\top(\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1}) \\ &= (\mathbf{W}_{L:\ell+1} \otimes \mathbf{W}_{\ell-1:1}^\top)\cdot(\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1}) && \text{(by Transpose Property)} \\ &= \mathbf{W}_{L:\ell+1}\mathbf{W}_{L:\ell+1}^\top \otimes \mathbf{W}_{\ell-1:1}^\top\mathbf{W}_{\ell-1:1}, && \text{(by Distributive Property)} \end{aligned}$$

where we denoted $\left[\nabla^2 f(\boldsymbol{\Theta})\right]_{\ell,\ell}$ as the ℓ-th diagonal element of the Hessian. Finally, the trace of the Hessian is

$$
\begin{aligned}
\operatorname{tr}\left(\nabla^2_{\mathbf{W}_{L:1}(t)} f(\boldsymbol{\Theta}(t))\right) &= \sum_{\ell=1}^{L} \operatorname{tr}\left(\left[\nabla^2 f(\boldsymbol{\Theta}(t))\right]_{\ell,\ell}\right) \\
&= \sum_{\ell=1}^{L} \operatorname{tr}\left(\mathbf{W}_{L:\ell+1}(t)\mathbf{W}^\top_{L:\ell+1}(t) \otimes \mathbf{W}^\top_{\ell-1:1}(t)\mathbf{W}_{\ell-1:1}(t)\right) \\
&= \sum_{\ell=1}^{L} \operatorname{tr}\left(\mathbf{W}_{L:\ell+1}(t)\mathbf{W}^\top_{L:\ell+1}(t)\right) \cdot \operatorname{tr}\left(\mathbf{W}^\top_{\ell-1:1}(t)\mathbf{W}_{\ell-1:1}(t)\right) \\
&\qquad \text{(by Trace Property)} \\
&= \sum_{\ell=1}^{L} \|\mathbf{W}_{\ell-1:1}(t)\|^2_{\mathsf{F}} \cdot \|\mathbf{W}_{L:\ell+1}(t)\|^2_{\mathsf{F}}.
\end{aligned}
$$

This concludes the proof. □

Then, suppose we solve the deep matrix factorization problem with initialization scale α. Notice that at GD iteration t, trace of the Hessian of the end-to-end DLN is given by

$$
\begin{aligned}
\operatorname{tr}\left(\nabla^2_{\boldsymbol{W}_{L:1}(t)} f(\boldsymbol{\Theta}(t))\right) = d\left[(d-r)\alpha^{2(L-1)} + \sum_{i=1}^{r} \sigma_i^{2\frac{(L-1)}{L}}(t)\right] + d\left[\sum_{i=1}^{r} \sigma_i^{2\frac{(L-1)}{L}}(t)\right] + \\
\sum_{l=2}^{L-1}\left[\sum_{i=1}^{r} \sigma_i^{2\frac{(l-1)}{L}}(t)\right]\left[(d-r)\alpha^{2(L-l-1)} + \sum_{i=1}^{r} \sigma_i^{2\frac{(L-l-1)}{L}}(t)\right]
\end{aligned}
$$

This also holds for any initialization of the DLN. Then, at convergence (i.e. when the gradient is zero), we can set $\sigma_i(t) = \sigma_i^\star$, and then the trace of the Hessian is only dependent on $\sigma_i^\star$ and α. Then, for smaller values of α, the DLN has a smaller trace of the Hessian at convergence. This result hints at that there may exists a bias towards flat solutions as measured by the trace of the Hessian, when starting from a smaller initialization scale. We leave further investigation of this phenomenon for future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, our introduction discusses the edge of stability in DLNs. Our main claims on this are both theoretically and empirically verified throughout the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, our analysis requires a particular initialization of the DLN, which we investigate more in the Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Yes, we provide proofs in the Appendix with corresponding theory and and assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Yes, we consider simple deep matrix factorization problem and the parameters necessary to reproduce the results. We will also make code available in the near future and have submitted code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release code upon finishing the blind peer review process.

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Yes we discuss the parameters of the networks and will release relevant code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not have randomness in the initialization and so it is not relevant here.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.

- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, we specify in the beginning of the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have read and followed the NeurIPS code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA] .

    Justification: We do not believe that there are any societal impacts regarding our work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not believe that there are such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, we consider MLPs and DLNs which we construct. The datasets are properly cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will release relevant code with documentation.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We did not conduct experiments with human subjects nor croudsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We did not conduct experiments with human subjects nor croudsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.