Efficient Low-Dimensional Compression of Overparameterized Models

Soo Min Kwon* University of Michigan

Zekai Zhang* Tsinghua University

Dogyoon Song University of Michigan

Laura Balzano University of Michigan

Qing Qu University of Michigan

Abstract

In this work, we present a novel approach for compressing overparameterized models, developed through studying their learning dynamics. We observe that for many deep models, updates to the weight matrices occur within a low-dimensional invariant subspace. For deep linear models, we demonstrate that their principal components are fitted incrementally within a small subspace, and use these insights to propose a compression algorithm for deep linear networks that involve decreasing the width of their intermediate layers. We empirically evaluate the effectiveness of our compression technique on matrix recovery problems. Remarkably, by using an initialization that exploits the structure of the problem, we observe that our compressed network converges faster than the original network, consistently yielding smaller recovery errors. We substantiate this observation by developing a theory focused on deep matrix factorization. Finally, we empirically demonstrate how our compressed model has the potential to improve the utility of deep nonlinear models. Overall, our algorithm improves the training efficiency by more than 2×, without compromising generalization.

1 INTRODUCTION

Overparameterization has proven to be a powerful modeling approach for solving various problems in machine learning and signal processing (LeCun et al., 2015; Haderlein et al., 2021; Zou et al., 2021). In the literature, it has been observed that overparameterized models yield solutions with superior generalization capabilities. This phenomenon has demonstrated prevalence across a wide range of problems, along with a broad class of model architectures, and has far-reaching implications, including the acceleration of convergence (Arora et al., 2018; Liu and Belkin, 2020) and the improvement of sample complexity (Arora et al., 2019; Sun et al., 2022). For example, Arora et al. (2019) illustrated the advantages of deep linear models within the context of low-rank matrix recovery. They showed that deeper models promoted low-rank solutions as a function of depth, consequently decreasing the sample complexity compared to classical approaches like nuclear norm minimization (Recht et al., 2010) and alternating gradient or alternating minimization (Chi et al., 2019; Jain et al., 2013). Outside of matrix recovery, there exists an abundance of research showcasing the benefits of overparameterized models under many different settings (Tripuraneni et al., 2021; Dar et al., 2021; Chang et al., 2020; Ma and Fattahi, 2022).

Nevertheless, the benefits of overparameterization come with a cost; they require extensive computational resources to train. The number of parameters to estimate rapidly increases with the signal dimension, hindering the use of overparameterized models for large-scale problems. To tackle this issue, researchers have begun to study the learning dynamics of these models, seeking opportunities for compression (Li et al., 2023; Yaras et al., 2022; Idelbayev and Carreira-Perpiñán, 2020; Cheng et al., 2018). For instance, recent works have shown that assuming the updates of the weight matrices in nonlinear networks have a low-dimensional structure can significantly reduce the training overhead with only a small tradeoff in accuracy (Hu et al., 2022; Wang et al., 2023; Horváth et al., 2023). How-

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

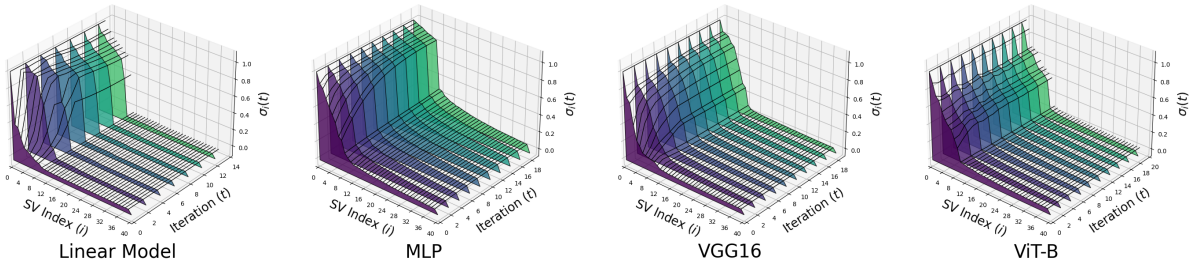


Figure 1: **Prevalence of low-dimensional weight updates across various networks.** The plots depict the singular values of the weight updates from initialization for the penultimate layer for different types of (nonlinear) architectures: deep linear network (DLN), multi-layer perception (MLP), VGG (Simonyan and Zisserman, 2015), and ViT-B (Dosovitskiy et al., 2021). The first two networks are trained on MNIST, while the latter are trained on CIFAR-10. This result shows a prevalent phenomenon across linear and nonlinear networks – gradient descent only updates a small portion of the singular values, while the others remain small and almost unchanged.

ever, these results are limited to only specific network architectures under certain settings. Our first observation is that low-rank weight matrices arise rather universally across a wide range of network architectures when trained with gradient descent (GD). We demonstrate this in Figure 1, where we plot the singular values of the weight updates for the penultimate weight matrix across various networks, showing that the training largely occurs within a low-dimensional subspace (see Appendix A.4 for a discussion). Although this suggests that it is possible to compress the overparameterized weight matrices, it is not immediately clear how one could tackle this task. In addition, therein lies the question of whether one could construct compressed networks without compromising the performance of their wider counterparts.

In this work, we take a step toward developing a principled approach for compressing overparameterized models. For overparameterized deep linear networks (DLNs), by carefully exploring the learning dynamics, we show strong evidence for several interesting low-rank properties that occur during the training process. Firstly, we demonstrate that the singular subspaces of the DLN are fitted incrementally (Li et al., 2021; Jacot et al., 2021; Chou et al., 2023), but *only* within a small invariant subspace across a wide range of matrix recovery problems. Secondly, we leverage this observation to propose a simple, yet highly effective method for compressing DLNs that involves decreasing the width of the intermediate layers of the original wide DLN. The main takeaway of our method is the following:

> *When properly initialized, the compressed DLN attains better solutions than the wide counterpart throughout all iterations of GD.*

By capitalizing on the property of incremental learning, we rigorously substantiate this finding on the deep matrix factorization problem as an illustrative example. We highlight that our approach can provide insights into how one can compress overparameterized weights without increasing recovery error. Below, we outline some of our key contributions.

- **Fast Convergence and Efficient Training.** We demonstrate that our compressed network attains a lower recovery error than the overparameterized network throughout all iterations of GD. As a result, we achieve (1) faster convergence in fewer GD iterations and (2) further speed up training by estimating fewer parameters, all while enjoying the benefits of overparameterized networks.

- **Benefits of Incremental Learning.** We empirically demonstrate the prevalence of incremental learning (also commonly referred to as sequential learning), wherein the singular subspaces of a DLN are fitted one at a time (see Figure 2). We establish that this phenomenon occurs in several canonical matrix recovery problems and leverage these insights to rigorously establish the superiority of our compressed DLN over wide DLNs.

- **Compression of Deep Nonlinear Networks.** We demonstrate how to leverage our findings in deep linear models to accelerate the training of deep nonlinear networks. By overparameterizing the penultimate layer of these deep networks and employing our compression technique, we can reduce memory complexity and training time while achieving similar (or even better) test accuracy.

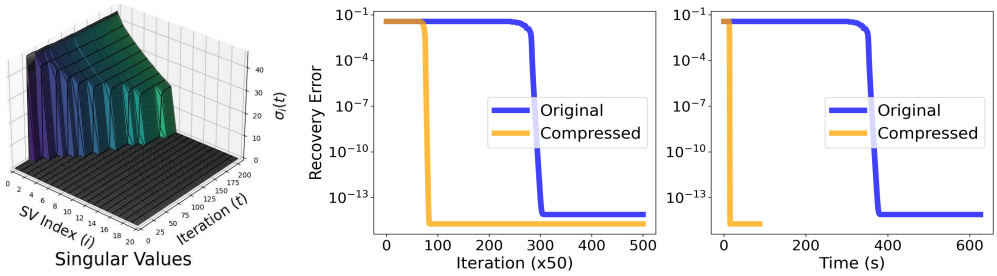


Figure 2: **Motivating the benefits of our compressed DLN while showcasing the incremental learning phenomenon.** Left: Plot of the change in singular values of the end-to-end DLN for matrix completion with $r = 10$; this hints that we can perform low-rank training in a small subspace without having to overparameterize. Middle & Right: Recovery error for the original and compressed DLN across iterations and time, respectively.

2 EFFICIENT NETWORK COMPRESSION

In this section, we present our problem formulation along with our corresponding compression algorithm. For clarity in notation, we provide a full list of symbols and their descriptions in Table 2 of the Appendix.

2.1 A Basic Problem Setup

We motivate our study based upon the low-rank matrix recovery problem, where the goal is to estimate a ground truth low-rank matrix $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ from measurements $\boldsymbol{y} = \mathcal{A}(\boldsymbol{M}^*) \in \mathbb{R}^m$, where we assume that the rank r of $\boldsymbol{M}^*$ is much smaller than its ambient dimension ($r \ll d$). Here, $\mathcal{A}(\cdot) : \mathbb{R}^{d\times d} \to \mathbb{R}^m$ is a linear operator and m denotes the number of measurements.

Moreover, we consider *deep* low-rank matrix recovery by modeling $\boldsymbol{M}^*$ via a DLN parameterized by a sequence[1] $\boldsymbol{\Theta} = (\boldsymbol{W}_l \in \mathbb{R}^{d\times d})_{l=1}^L$, which can be estimated by solving the following least-squares problem:

$$\widehat{\boldsymbol{\Theta}} \in \arg\min_{\boldsymbol{\Theta}} \underbrace{\frac{1}{2}\left\|\mathcal{A}(\boldsymbol{W}_{L:1} - \boldsymbol{M}^*)\right\|_2^2}_{=:\,\ell_{\mathcal{A}}(\boldsymbol{\Theta};\,\boldsymbol{M}^*)}, \tag{1}$$

where we abbreviate $\boldsymbol{W}_{L:1} := \boldsymbol{W}_L \cdot \ldots \cdot \boldsymbol{W}_1$ for exposition. This problem frequently arises in many signal processing settings; if $\mathcal{A} = \mathrm{Id}$, the identity map, it simplifies to *deep matrix factorization*; if $\mathcal{A} = \mathcal{P}_{\boldsymbol{\Omega}}$, the projection operator onto a subset of its entries defined by $\boldsymbol{\Omega}$, it is *deep matrix completion*.

It is well-known that GD with small initialization has an implicit bias towards low-rank solutions (Gidel et al., 2019; Arora et al., 2019). To obtain the desired low-rank solution, for every iteration $t \geq 0$, we update each weight matrix $\boldsymbol{W}_l$ using GD given by

$$\boldsymbol{W}_l(t) = \boldsymbol{W}_l(t-1) - \eta \cdot \nabla_{\boldsymbol{W}_l} \ell_{\mathcal{A}}(\boldsymbol{\Theta}(t-1)), \tag{2}$$

$\forall l \in [L]$, where $\eta > 0$ is the learning rate and $\nabla_{\boldsymbol{W}_l} \ell_{\mathcal{A}}(\boldsymbol{\Theta}(t))$ is the gradient of $\ell_{\mathcal{A}}(\boldsymbol{\Theta})$ with respect to the l-th weight matrix at the t-th GD iterate. Additionally, we initialize the weights to be orthogonal matrices scaled by a small constant $\epsilon > 0$:

$$\boldsymbol{W}_l(0)^\top \boldsymbol{W}_l(0) = \boldsymbol{W}_l(0)\boldsymbol{W}_l(0)^\top = \epsilon \boldsymbol{I}, \tag{3}$$

$\forall l \in [L]$. For the case in which the weight matrices are not square, we can initialize them to be ϵ-scaled semi-orthogonal weight matrices that satisfy

$$\boldsymbol{W}_l(0)^\top \boldsymbol{W}_l(0) = \epsilon \boldsymbol{I} \quad \text{or} \quad \boldsymbol{W}_l(0)\boldsymbol{W}_l(0)^\top = \epsilon \boldsymbol{I}, \tag{4}$$

which depends on the shape of $\boldsymbol{W}_l$. It has been proven that such an initialization leads to favorable convergence properties when training DLNs (Yaras et al., 2023; Pennington et al., 2018; Chen et al., 2018), which we adopt throughout this paper for analysis.[2]

2.2 Efficient Low-Rank Network Compression Methods

While overparameterization offers benefits such as decreased sample complexity and improved generalization, they come at the cost of a large increase in computational complexity. However, as we observed in Figures 1 and 2, overparameterized networks manifest effective low-dimensionality in their training dynamics across various learning tasks, which could be potentially exploited for an efficient training strategy. To this end, instead of overparameterizing each layer of the DLN, we consider a compressed DLN parameterized by $\widetilde{\boldsymbol{\Theta}} := \left(\widetilde{\boldsymbol{W}}_1, \cdots, \widetilde{\boldsymbol{W}}_L\right)$, where $\widetilde{\boldsymbol{W}}_L \in \mathbb{R}^{d\times \hat{r}}$,

[1] In general, both the target matrix $\boldsymbol{M}^*$ and the layer parameter matrices $\boldsymbol{W}_l \in \mathbb{R}^{d_l \times d_{l-1}}$ can have arbitrary shapes with any $d_1, \ldots, d_{L-1}$. Here, we assumed square shapes for simplicity in exposition.

[2] Nonetheless, our experiments show that our method is effective even with small *random* uniform initialization.

$\widetilde{\boldsymbol{W}}_1 \in \mathbb{R}^{\hat{r}\times d}$, and $\widetilde{\boldsymbol{W}}_l \in \mathbb{R}^{\hat{r}\times\hat{r}}$ for $2 \le l \le L-1$, and $\hat{r}$ is any positive integer such that $\hat{r} \ge r = \operatorname{rank}(\boldsymbol{M}^*)$. The end-to-end matrix product at GD iterate t with this parameterization is given by

$$\widetilde{\boldsymbol{W}}_{L:1}(t) = \widetilde{\boldsymbol{W}}_L(t) \cdot \ldots \cdot \widetilde{\boldsymbol{W}}_1(t) \in \mathbb{R}^{d\times d}. \tag{5}$$

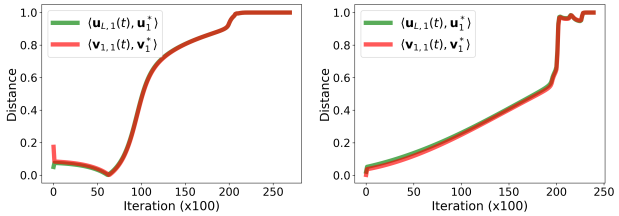
Note that we do not assume knowledge of r, but only require $\hat{r} \ge r$. This reduces the total number of parameters from Ld^2 into $2d\hat{r} + (L-2)\cdot\hat{r}^2$, which is a substantial reduction for sufficiently small $\hat{r}$. This compressed DLN is largely motivated by Figure 2. If $\hat{r} \ge r$, then we can "prune" out the portion of the DLN that corresponds to the singular values that remain close to zero. However, truncating singular values before we are sufficiently close to the minimizer would at best cause slower convergence and at worst lead to sub-optimal solutions. We show that a particular initialization of $\widetilde{\boldsymbol{\Theta}}$ can mitigate these issues and remarkably outperform the wide DLN.

For initialization, we choose a small constant $\epsilon > 0$ and let $\widetilde{\boldsymbol{W}}_l(0) = \epsilon \cdot \boldsymbol{I}_{\hat{r}}$ for $2 \le l \le L-1$; additionally, we initialize the left and right most factors $\widetilde{\boldsymbol{W}}_L(0)$ and $\widetilde{\boldsymbol{W}}_1(0)$ by extracting the top-$\hat{r}$ singular vectors of the surrogate matrix

$$\boldsymbol{M}^{\text{surr}} := \mathcal{A}^\dagger\mathcal{A}(\boldsymbol{M}^*) = \frac{1}{m}\sum_{i=1}^{m} y_i \boldsymbol{A}_i, \tag{6}$$

where $y_i = \langle \boldsymbol{A}_i, \boldsymbol{M}^*\rangle$, and scaling them by ϵ. For a random operator $\mathcal{A}(\cdot)$, the singular subspaces of $\boldsymbol{M}^{\text{surr}}$ are known to closely approximate those of $\boldsymbol{M}^*$ with high probability (Chi et al., 2019). Thus, we expect $\widetilde{\boldsymbol{W}}_L(0)$ and $\widetilde{\boldsymbol{W}}_1(0)$ to be roughly close to the singular subspaces of $\boldsymbol{M}^*$. To update each weight matrix, we use a learning rate $\eta > 0$ to update the intermediate layers, and a rate $\alpha\cdot\eta$ with $\alpha > 0$ to update $\widetilde{\boldsymbol{W}}_L(t)$ and $\widetilde{\boldsymbol{W}}_1(t)$. We observe that using a scale $\alpha > 1$ to update the left and rightmost factors often accelerates convergence. The complete algorithm is summarized in Algorithm 1.

To provide an intuition for the use of spectral initialization, we conducted an experiment where we analyzed the trajectories of the factors $\boldsymbol{W}_L(t)$ and $\boldsymbol{W}_1(t)$ of the original DLN starting from an orthogonal initialization. In Figure 3, we show that these factors ultimately align with the singular vectors of the target matrix $\boldsymbol{M}^*$. Interestingly, this observation is similar to that of Stöger and Soltanolkotabi (2021), where they note that the initial iterations of GD for two-layer matrix factorization resemble the power method. We find a similar result, but it applies to the left and rightmost factors of the DLN. Thus, initializing these factors close to the target singular vectors could accelerate convergence even when compressing the DLN.



Matrix Factorization Matrix Completion

Figure 3: **Motivating the use of spectral initialization for DLNs when starting from an orthogonal initialization.** These plots measure the similarity between the first principal component of $\boldsymbol{W}_L(t)$ and $\boldsymbol{U}^*$ (and respectively $\boldsymbol{W}_1(t)$ and $\boldsymbol{V}^*$). This result shows that these DLN factors fit the left and right singular vectors of the target matrix $\boldsymbol{M}^*$.

2.3 Extension to Compression in Linear Layers of Nonlinear Networks

In this section, we explore our network compression idea to improve the training efficiency and generalization capabilities of deep nonlinear networks. Traditionally, the success of deep neural networks is attributed to the power of overparameterization, which is often achieved by increasing the width of weight matrices keeping the depth fixed. Interestingly, recent work by Huh et al. (2023) demonstrated that increasing the *depth* of each weight matrix by *adding linear layers* also improves generalization, across a wide range of datasets for classification problems. More concretely, consider a simple deep network parameterized by $\boldsymbol{\Theta} = (\boldsymbol{W}_l)_{l=1}^L$, which represents a map[3] $\psi_{\boldsymbol{\Theta}} : \boldsymbol{x} \mapsto \boldsymbol{W}_L\rho\left(\boldsymbol{W}_{L-1}\ldots\rho(\boldsymbol{W}_1\boldsymbol{x})\right)$, where $\rho(\cdot)$ is a predetermined nonlinear activation function such as ReLU. Huh et al. (2023) showed that adding linear layers results in a more favorable deep network, namely, $\psi'_{\boldsymbol{\Theta}} : \boldsymbol{x} \mapsto \boldsymbol{W}_{L+2}\rho\left(\boldsymbol{W}_{L+1}\boldsymbol{W}_L\boldsymbol{W}_{L-1}\ldots\rho(\boldsymbol{W}_1\boldsymbol{x})\right)$. Observe that this amounts to overparameterizing the penultimate layer of the deep nonlinear network using a 3-layer DLN $\boldsymbol{W}_{L+1:L-1} = \boldsymbol{W}_{L+1}\boldsymbol{W}_L\boldsymbol{W}_{L-1}$.

Motivated by this, we explore improving the performance of deep nonlinear networks across a variety of architectures (e.g., MLP and ViT) by (1) initially overparameterizing their penultimate layer with a DLN, followed by (2) compressing the DLN using our proposed technique. This modification is anticipated to enjoy the advantages of overparameterization through DLNs, while mitigating the increase in computational demands. However, we note that determining initial subspaces for $\widetilde{\boldsymbol{W}}_L(0)$ and $\widetilde{\boldsymbol{W}}_l(0)$ in this setting is not as straightforward as in the low-rank matrix recovery context. As a first step towards applying compres-

[3]Here, we omit the bias of the network for simplicity.

Algorithm 1 Compressed DLNs (C-DLNs) for Learning Low-Dimensional Models

Require: loss function $\ell(\cdot)$; $\boldsymbol{y} \in \mathbb{R}^m$; $\epsilon, \eta, \alpha \in \mathbb{R}_+$; $\hat{r}, L, T \in \mathbb{N}$

1: $\widetilde{\boldsymbol{W}}_l(0) \leftarrow \epsilon \cdot \boldsymbol{I}_{\hat{r}}, \quad 2 \le l \le L-1$ ▷ Initialize intermediate weight parameters

2: $\boldsymbol{U}_{\hat{r}}, \boldsymbol{V}_{\hat{r}} \leftarrow \texttt{SVD}(\boldsymbol{M}^{\text{surr}})$ ▷ Compute $\boldsymbol{M}^{\text{surr}}$ via Equation (6) and take top-$\hat{r}$ SVD

3: $\widetilde{\boldsymbol{W}}_L = \epsilon \cdot \boldsymbol{U}_{\hat{r}}, \quad \widetilde{\boldsymbol{W}}_1 = \epsilon \cdot \boldsymbol{V}_{\hat{r}}$ ▷ Initialize outer weight parameters

4: **for** $t = 1, \dots, T$ **do** ▷ GD update for T iterations

$$\widetilde{\boldsymbol{W}}_L(t+1) \leftarrow \widetilde{\boldsymbol{W}}_L(t) - \alpha\eta \cdot \nabla_{\widetilde{\boldsymbol{W}}_L} \ell\big(\widetilde{\boldsymbol{\Theta}}(t)\big),$$
$$\widetilde{\boldsymbol{W}}_l(t+1) \leftarrow \widetilde{\boldsymbol{W}}_l(t) - \eta \cdot \nabla_{\widetilde{\boldsymbol{W}}_l} \ell\big(\widetilde{\boldsymbol{\Theta}}(t)\big),$$
$$\widetilde{\boldsymbol{W}}_1(t+1) = \widetilde{\boldsymbol{W}}_1(t) - \alpha\eta \cdot \nabla_{\widetilde{\boldsymbol{W}}_1} \ell\big(\widetilde{\boldsymbol{\Theta}}(t)\big),$$

5: **end for**

6: Return $\widetilde{\boldsymbol{W}}_{L:1} = \widetilde{\boldsymbol{W}}_L(T) \cdot \ldots \cdot \widetilde{\boldsymbol{W}}_1(T)$ ▷ Output of compressed DLN

sion techniques to deep nonlinear networks, we propose initializing the subspaces by using the singular subspaces of the cross-correlation matrix, $\boldsymbol{M}^{\text{corr}} := \boldsymbol{Y}\boldsymbol{X}^\top \in \mathbb{R}^{d_y \times d_x}$ where applicable and otherwise using random subspaces. When the number of classes d_y is smaller than the feature dimension d_x, this matrix $\boldsymbol{M}^{\text{corr}}$ is low-rank with $\text{rank}\,(\boldsymbol{M}^{\text{corr}}) \le d_y \ll d_x$, permitting the choice $\hat{r} \ge d_y$.

3 THEORETICAL RESULTS

In this section, we present a theory elucidating the superior performance of our compressed DLN for the deep matrix factorization case. Our analysis aims to explain the benefits of the spectral initialization and the incremental learning phenomenon in achieving accelerated convergence (in iteration complexity).

3.1 The Benefits of Spectral Initialization

Let $\boldsymbol{M}^* \in \mathbb{R}^{d \times d}$ be a matrix of rank r and $\boldsymbol{M}^* = \boldsymbol{U}^*\boldsymbol{\Sigma}^*\boldsymbol{V}^{*\top}$ be a singular value decomposition (SVD) of $\boldsymbol{M}^*$. We consider the deep matrix factorization setting, where $\mathcal{A} = \text{Id}$, i.e., we have full observation of $\boldsymbol{M}^*$. Here, $\boldsymbol{M}^{\text{surr}} = \boldsymbol{M}^*$ and so Algorithm 1 initializes $\widetilde{\boldsymbol{W}}_L(0) = \epsilon \cdot \boldsymbol{U}^*_{\hat{r}}$ and $\widetilde{\boldsymbol{W}}_1(0) = \epsilon \cdot \boldsymbol{V}^{*\top}_{\hat{r}}$, the top-$\hat{r}$ singular subspaces of $\boldsymbol{M}^*$ themselves. This leads to the compressed deep matrix factorization problem

$$\min_{\widetilde{\boldsymbol{\Theta}}} \ell(\widetilde{\boldsymbol{\Theta}}(t)) = \frac{1}{2}\|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\text{F}}^2, \tag{7}$$

where the intermediate layers are initialized to $\widetilde{\boldsymbol{W}}_l(0) = \epsilon \cdot \boldsymbol{I}_{\hat{r}}$ for $2 \le l \le L-1$. In Figure 3, we showed that the left and rightmost factors of the original DLN align with the left and right singular vectors of the target matrix $\boldsymbol{M}^*$ more closely throughout GD. This observation implies that our particular choice of initialization could accelerate the training process. We substantiate this observation by proving that this initialization has two advantages: (1) the compressed DLN has a low-dimensional structure and (2) the compressed DLN incurs a lower recovery error at initialization than the original DLN.

Theorem 1. *Let $\boldsymbol{M}^* \in \mathbb{R}^{d \times d}$ be a matrix of rank r and $\boldsymbol{M}^* = \boldsymbol{U}^*\boldsymbol{\Sigma}^*\boldsymbol{V}^{*\top}$ be a SVD of $\boldsymbol{M}^*$. Suppose we run Algorithm 1 with $\alpha = 1$ to update all weights $\left(\widetilde{\boldsymbol{W}}_l\right)_{l=1}^L$ of Equation (7), where $\mathcal{A} = Id$. Then, the end-to-end compressed DLN possesses low-dimensional structures, in the sense that for all $t \ge 1$, $\widetilde{\boldsymbol{W}}_{L:1}(t)$ admits the following decomposition:*

$$\widetilde{\boldsymbol{W}}_{L:1}(t) = \boldsymbol{U}^*_{\hat{r}} \begin{bmatrix} \boldsymbol{\Lambda}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \beta(t)^L \cdot \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} \boldsymbol{V}^{*\top}_{\hat{r}}, \tag{8}$$

where $\boldsymbol{\Lambda}(t) \in \mathbb{R}^{r \times r}$ is a diagonal matrix with entries $\lambda_i(t)^L$, where

$$\lambda_i(t) = \lambda_i(t-1) \cdot \big(1 - \eta \cdot (\lambda_i(t-1)^L - \sigma_i^*) \cdot \lambda_i(t-1)^{L-2}\big),$$

for $1 \le i \le r$, with $\lambda_i(0) = \epsilon$ and σ_i^* is the i-th diagonal entry of $\boldsymbol{\Sigma}^*$ and

$$\beta(t) = \beta(t-1) \cdot \Big(1 - \eta \cdot \beta(t-1)^{2(L-1)}\Big),$$

with $\beta(0) = \epsilon$.

Remarks. We defer all proofs to Appendix B. By using our initialization technique as outlined in Algorithm 1, Theorem 1 allows us to characterize the GD updates of the compressed DLN, which also constitutes a valid SVD. This also shows that the compressed DLN directly finds low-rank solutions of rank r, as the last $\hat{r} - r$ singular values is a decreasing function where $\beta(t) \le \epsilon^L$ for a small constant $\epsilon > 0$.

We also note a couple of points regarding the relationship between Theorem 1 and existing theory. The

main difference of our result from that of Yaras et al. (2023) is that, through the use of spectral initialization, we are able to characterize the dynamics of *all* of the singular values, whereas Yaras et al. (2023) only characterizes the behavior of $d-2r$ singular values but does not specify the behavior of the remaining $2r$ singular values. The result by Arora et al. (2019) has a similar flavor — if we consider infinitesimal steps of their gradient flow result regarding the behavior of the singular values, we can recover the discrete steps as outlined in Theorem 1. With Theorem 1 in place, an immediate consequence is that the compressed DLN exhibits a lower recovery error than the original DLN at the initialization $t = 0$, as outlined by Corollary 1.

Corollary 1. *Let $\boldsymbol{W}_{L:1}(0)$ denote the original DLN at $t = 0$ initialized with orthogonal weights according to Equation (3) and $\widetilde{\boldsymbol{W}}_{L:1}(0)$ denote the compressed DLN at $t = 0$. Then, we have*

$$\|\boldsymbol{W}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \|\widetilde{\boldsymbol{W}}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2. \tag{9}$$

This result can be established by applying Theorem 1 at $t = 0$. Next, we demonstrate that the inequality in Equation (9) holds for all $t \geq 0$, which involves leveraging the incremental learning phenomenon along with an analysis using gradient flow.

3.2 The Benefits of Incremental Learning

In this section, we establish that the inequality in Equation (9) holds for all GD iterations $t \geq 0$ for the deep matrix factorization case. To prove such a result, we assume that both the original DLN and the compressed DLN undergo incremental learning, in the sense that the singular values of both networks and their respective singular vectors are fitted sequentially. This assumption is stated formally in Assumption 1.

Assumption 1. *Let $\boldsymbol{W}_{L:1}(t) \in \mathbb{R}^{d\times d}$ denote the end-to-end weight matrix at GD iterate t with respect to Equation (1) with $\mathcal{A} = Id$ and $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ be the target matrix with rank r. Then, GD follows an incremental learning procedure in the sense that there exist small constants $c_{val}, c_{vec} \in [0, 1]$ and a sequence of time points $t_1 \leq t_2 \leq \ldots \leq t_r \in \mathbb{R}$ such that*

$$(\sigma_i(\boldsymbol{W}_{L:1}(t)) - \sigma_i(\boldsymbol{M}^*))^2 \leq c_{val}, \tag{10}$$

$$\langle \boldsymbol{u}_i(t), \boldsymbol{u}_i^* \rangle \geq 1 - c_{vec}, \tag{11}$$

$$\langle \boldsymbol{v}_i(t), \boldsymbol{v}_i^* \rangle \geq 1 - c_{vec}, \tag{12}$$

for all $t > t_i$ and $\lim_{\epsilon\to 0} \sigma_i(\boldsymbol{W}_{L:1}(t)) = 0$ for all $t < t_i$, where $\epsilon > 0$ is the initialization scale.

Assumption 1 states that there exists a sequence of time points where the i-th principal components of both networks are fitted to a certain precision, while the singular values associated with the remaining principal components remain close to the initialization scale. This assumption has been widely studied and adopted for the two-layer case (Jin et al., 2023; Jiang et al., 2023b), with some results extending to the deep matrix factorization case (Li et al., 2021; Jacot et al., 2021; Chou et al., 2023; Gidel et al., 2019). In Figure 4, we further show the occurrence of this phenomenon and demonstrate its validity with extensive experimental results in Appendix A.1. In Appendix A.1, we also show that generally, we have $c_{\text{vec}} = c_{\text{val}} = 0$, which will play a role in our proofs as well. While we initially assume this phenomenon for deep matrix factorization, our extensive experiments also confirm that Assumption 1 holds for low-rank matrix sensing and completion.

By using this phenomenon, we can analyze the singular values of the original and compressed DLN one at a time, and show that the singular values of the compressed network are fitted more quickly than those of the original network, leading to faster convergence in terms of iteration complexity. Our main result is stated in Theorem 2.

Theorem 2. *Let $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ be a rank-r matrix and let $\hat{r} \in \mathbb{N}$ such that $\hat{r} \geq r$. Suppose that we run gradient flow with respect to the original DLN in Equation (1) and with respect to the compressed network defined in Equation (7) with $\mathcal{A} = Id$. Then, if Assumption 1 holds such that $c_{vec} = 0$, we have that $\forall t \geq 0$,*

$$\|\boldsymbol{W}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2. \tag{13}$$

Remarks. The proof relies on analyzing the evolution of singular values of $\boldsymbol{W}_{L:1}$ and $\widetilde{\boldsymbol{W}}_{L:1}$ using gradient flow. Through this analysis, we demonstrate that the singular values of the compressed network are fitted more quickly than those of the original DLN throughout all iterations of GD, thereby highlighting the benefits of spectral initialization and our compression technique. We remark that there is a slight discrepancy between our algorithm and the analysis. Our algorithm employs discrete gradient steps to update the weight matrices. However, it is well-established that using differential equations (and hence gradient flow) for theoretical analysis has a rich history, and it is known that discrete gradient steps approximate the gradient flow trajectories as long as the step size is sufficiently small (Helmke and Moore, 1996; Arora et al., 2018).

4 EXPERIMENTS

This section is organized as follows. In Section 4.1, we present results for solving low-rank matrix recov-

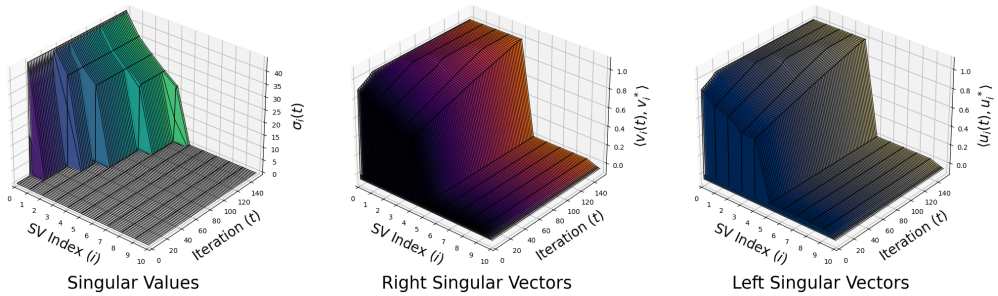


Figure 4: **Occurence of the incremental learning phenomenon in deep matrix factorization.** We observe that the first $r = 5$ singular values are fitted incrementally, along with their respective singular subspaces, corroborating Assumption 1.

ery problems on both synthetic and real data. In Section 4.2, we show that adding linear layers to a deep network indeed improves generalization, and provide results on compressing the linear layers using our proposed method.

4.1 Matrix Recovery Problems

Throughout all of the experiments in this section, we use a DLN of depth $L = 3$ and small initialization scale $\epsilon = 10^{-3}$. To quantatively measure the performance between the original and compressed DLN, we use the recovery error defined as

$$\text{Recovery Error} = \|\widehat{\boldsymbol{W}} - \boldsymbol{M}^*\|_{\mathsf{F}},$$

where $\widehat{\boldsymbol{W}}$ is an estimate of the target matrix.

Deep Matrix Factorization. For deep matrix factorization, we synthetically generate a data matrix $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ with $d = 100$ and rank $r = 10$. We use the left and right singular vectors of $\boldsymbol{M}^*$ for $\widetilde{\boldsymbol{W}}_L(0) \in \mathcal{O}^{d\times\hat{r}}$ and $\widetilde{\boldsymbol{W}}_1(0) \in \mathcal{O}^{\hat{r}\times d}$, where we choose $\hat{r} = 20$ as our upper bound on the rank r. We run GD with $\eta = 10$ for the learning rate and $\alpha = 5$ for the scale. In Figure 5, we can observe that the compressed network maintains a lower recovery error throughout all iterations of GD, corroborating Theorem 2. As a result, this leads to two advantages: (1) a reduction in the number of iterations to converge to a specific threshold when using the compressed DLN, and (2) a further reduction in time complexity, as each iteration of the compressed DLN is much faster than that of the original DLN.

Deep Matrix Completion. We present our results for deep matrix completion and present results for deep matrix sensing in Appendix A.2. Our goal is to show that choosing the singular subspaces of the

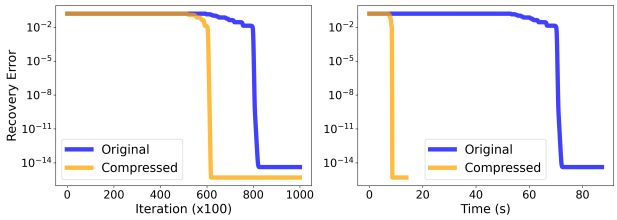
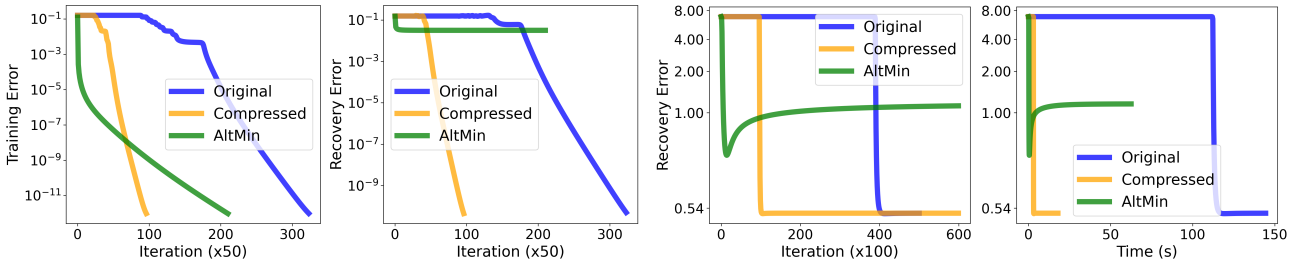


Figure 5: **Empirical results on deep linear matrix factorization.** Left: Shows that our compressed network achieves a lower recovery error than the original network, corroborating our theory. Right: Demonstrates the speed up over the original network.

surrogate matrix in Equation (6) serve as good initial points for $\widetilde{\boldsymbol{W}}_L(0)$ and $\widetilde{\boldsymbol{W}}_1(0)$. We also compare the performance of our compressed network to AltMin (Jain et al., 2013), which involves alternatingly minimizing over just the factor matrices to construct $\widehat{\boldsymbol{W}} = \widetilde{\boldsymbol{W}}_L\widetilde{\boldsymbol{W}}_1$. Firstly, we compare these algorithms using synthetic data, where we follow the setup in deep matrix factorization. For the observation set $\boldsymbol{\Omega}$, we consider the "missing completely at random" (MCAR) setting, where each entry of $\boldsymbol{\Omega}$ is Bernoulli with probability p. We choose $p = 0.3$ so that roughly 30% of the observations are observed. We run GD with learning rate $\eta = 10$ and $\alpha = 5$. In Figure 6, we observe the same trends as seen in the deep matrix factorization case, where our compressed DLN consistently exhibits lower recovery error than the original DLN. Additionally, it is evident that AltMin fails to recover the underlying matrix completely, as the rank is overspecified. We observe that while the training loss goes to zero, the recovery error does not decrease, as there are insufficient measurements for recovery using this parameterization. To efficiently use AltMin (or both networks for the $L = 2$ case), one would need to obtain a more accurate estimate of the rank $\hat{r}$ for recovery.



Results on Synthetic Data | Results on MovieLens Data

Figure 6: **Results on matrix completion with synthetic and real data.** Left: Results on matrix completion with only 30% observed entries for recovering a rank $r = 10$ matrix. Right: Results on MovieLens dataset with $\hat{r} = 10$. For both experiments, we observe that our compressed network achieves faster convergence than the original network and also outperforms overparameterized AltMin (Jain et al., 2013).

Next, we compare these algorithms on the MovieLens 100K dataset (Harper and Konstan, 2015). Since the MovieLens dataset is not precisely a low-rank matrix, this experiment also serves to demonstrate the performance of DLNs on approximately low-rank matrix completion. We randomly choose 80% of the samples to train the network and test on the remaining 20%. For the hyperparameters, we choose $\hat{r} = 10$, $\eta = 0.5$, and $\alpha = 5$. As depicted in Figure 6, we find that our compressed network achieves the same recovery error as the original DLN in less than $5\times$ the time. While AltMin initially finds solutions with lower recovery error at a faster rate, its recovery error eventually plateaus, whereas both DLNs can find solutions with overall lower recovery error.

4.2 Applications of DLNs for Deep Nonlinear Networks

To demonstrate the application of our compressed DLN on deep nonlinear networks, we consider the setting in Section 2.3, where we overparameterize the penultimate layer using a DLN. We consider this additional overparameterization using two network architectures: MLPs and ViTs. For MLPs, we train two MLPs: (1) one MLP with 3 hidden layers and one linear penultimate layer and (2) one MLP with 3 hidden layers and one 3-layer DLN as the penultimate layer. Mathematically, this amounts to the compressed network

$$\psi_{\widetilde{\boldsymbol{\Theta}}}(\boldsymbol{x}) = \boldsymbol{W}_6\rho(\underbrace{\widetilde{\boldsymbol{W}}_5\widetilde{\boldsymbol{W}}_4\widetilde{\boldsymbol{W}}_3}_{\widetilde{\boldsymbol{W}}_{5:3}}(\rho(\boldsymbol{W}_2\rho(\boldsymbol{W}_1\boldsymbol{x}))).$$

For ViT, we consider a smaller variant of ViT-base, which consists of 6 alternating layers of multi-headed self-attention and MLP blocks. We set the token dimension as 512 and the MLP dimension as 3072 and compress last linear layer in the 6-th MLP block, which can also be seen as the penultimate layer of the ViT. Here, our objective is two-fold: (1) to demonstrate that overparameterizing the penultimate layer has better generalization capabilities and (2) to show that the compressed networks have similar (or better) performance while significantly reducing the training time.

We illustrate this by training the MLPs on the FashionMNIST dataset and training the ViTs on the CIFAR-10 dataset. For $\hat{r}$, we choose $\hat{r} = 4d_y$, where d_y denotes the number of classes in the dataset. For the MLPs, we run GD with $\eta = 5 \times 10^{-3}$ and $\alpha = 1$ and used $\eta = 1 \times 10^{-4}$, $\alpha = 1$ with cosine annealing for the ViTs. Both models were tested 5 times starting from random initialization to account for variability. In Table 1, we observe that our compressed network can achieve the highest accuracy for MLPs on average, with very competitive results for ViTs. However, the compressed network takes significantly less time to train while storing fewer parameters. Overall, these results highlight the effectiveness of our approach, demonstrating that one can reduce runtime and memory without sacrificing the performance of deeper and wider models.

5 RELATED WORK

Deep Linear Networks. Despite their simplicity, deep linear networks have been widely adopted for theoretical analysis, as it has been observed that they share similar behavioral characteristics as their nonlinear counterparts (Saxe et al., 2014). Some examples include the study of their optimization landscape (Kawaguchi, 2016; Lin et al., 2021; Eftekhari, 2020; Chatterji and Long, 2023) and the study of understanding feature representation in deep networks (Yaras et al., 2022; Zhu et al., 2021; Papyan et al., 2020; Jiang et al., 2023a; Li et al., 2023; Yaras et al., 2022; Wang et al., 2022; Zhou et al., 2022a,b). Our work highlights that these deep linear models are

Method		Test Accuracy (%)	Time	MACs	Memory (Train)	# of Parameters
MLP	Original	89.87 ± 0.216	$\underline{83.02 \pm 20.87}$ sec	$\underline{2.370 \times 10^7}$	$\underline{25.80}$ MB	$\underline{1.850}$ M
	DLN	$\underline{90.20 \pm 0.040}$	95.82 ± 12.29 sec	3.940×10^7	35.90 MB	3.080 M
	C-DLN	$\mathbf{90.28 \pm 0.104}$	$\mathbf{54.85 \pm 2.950}$ sec	$\mathbf{1.670 \times 10^7}$	**21.60** MB	**1.300** M
ViT	Original	84.72 ± 0.120	$\underline{32.80 \pm 2.786}$ min	$\underline{2.100 \times 10^{10}}$	$\underline{7.110}$ GB	$\underline{25.20}$ M
	DLN	$\mathbf{84.90 \pm 0.230}$	45.80 ± 2.638 min	3.640×10^{10}	7.811 GB	44.10 M
	C-DLN	$\underline{84.89 \pm 0.187}$	$\mathbf{31.40 \pm 1.855}$ min	$\mathbf{1.980 \times 10^{10}}$	**7.090** GB	**23.80** M

Table 1: **Quantitative results for deep nonlinear networks.** DLN and C-DLN denote networks that are overparameterized using their respective models. The reported time is the amount of time taken for the models to achieve 99% test accuracy. Note that the reduction in memory and MACs is not substantial as we only consider compressing the penultimate layer. Best results are in bold and second best results are underlined.

not only useful analytical tools but also powerful models for solving low-rank matrix recovery tasks. This observation is built upon some of the work done by Arora et al. (2019), where they show that deeper models tend towards more accurate solutions for these matrix recovery problems in settings where the number of observations are very limited. The most relevant works to this study are those conducted by Yaras et al. (2023) and Khodak et al. (2021). We were unaware of the work by Khodak et al. (2021) at the time of writing this paper, where they explore the advantages of spectral initialization and weight decay for deep nonlinear networks. Their methods involve assuming a Gaussian prior on the weights of deep nonlinear networks, which they then apply spectral initialization with weight decay to train the weights of such networks. However, their methods are not directly applicable to the problems addressed in this paper, as a Gaussian prior alone is insufficient to realize the benefits of the compressed network as demonstrated in this study. This point is further emphasized by our theory – the spectral initialization step must incorporate some function of the data for it to be meaningful. The study by Yaras et al. (2023) investigates the learning dynamics of DLNs starting from orthogonal initialization in the deep matrix factorization case. Their theory includes an additional weight decay parameter, which can also be extended to our theory.

Low-Rank Training and Adaptation. Low-rank training refers to modeling the weight updates of networks (whether shallow or deep) as a product of low-rank matrices, rather than updating the entire matrix itself. By updating the low-rank matrices, one can reduce training costs and improve generalization by exploiting its intrinsic structure. This method has a long and rich history, dating back to the Burer-Monteiro factorization (Burer and Monteiro, 2003) and including efficient implementations of alternating minimization (Ma et al., 2023; Chi et al., 2019). Our work can be viewed as an improvement upon these techniques, as deeper models are more favorable for modeling low-rank matrices. There is also a body of work related to low-rank adaptation, which generally involves low-rank fine-tuning of large models (Hu et al., 2022; Wang et al., 2023; Zhao et al., 2023; Lialin et al., 2023; Zhai et al., 2023). For instance, Hu et al. (2022) proposed Low-Rank Adaptation (LoRA), which fine-tunes large language models (LLMs) by assuming that the weight updates have a low-dimensional structure and modeling them as a product of two matrices. This method has shown to significantly reduce memory complexity with only a minimal tradeoff in test accuracy. There also have been attempts at extending LoRA by training low-rank matrices from scratch, as such techniques not only reduce the training costs but also leads to possible performance gain by restricting the training dynamics to a low-dimensional manifold (Wang et al., 2023; Zhao et al., 2023; Lialin et al., 2023; Zhai et al., 2023). We believe that our work can improve the utility of algorithms of these algorithms, where we can model the weight updates using a DLN. For example, it would be an interesting direction to explore whether we can model the weight updates in LoRA using our compressed DLN to improve its expressive power.

6 CONCLUSION

In this work, we proposed an efficient technique to compress overparameterized networks by studying the learning dynamics of DLNs. Our method involved reducing the width of the intermediate layers along with a spectral initialization scheme that improved convergence compared to its wider counterpart. We theoretically argued the benefits of our network while showcasing their results on matrix factorization and completion with applications on nonlinear networks for classification tasks. We believe that our work opens doors to many exciting questions: the theoretical analysis of the learning trajectory for deep matrix sensing, and the practical application of further improving low-rank training for deep nonlinear networks.

Acknowledgments

SMK, DS, and QQ acknowledge support from NSF CAREER CCF-2143904, NSF CCF-2212066, NSF CCF-2212326, and NSF IIS 2312842, an AWS AI Award, a gift grant from KLA, and MICDE Catalyst Grant. SMK and LB acknowledge support from DoE award DE-SC0022186, ARO YIP W911NF1910027, and NSF CAREER CCF-1845076. Results presented in this paper were obtained using CloudBank, which is supported by the NSF under Award #1925001. SMK would also like to thank Can Yaras for fruitful discussions. ZZ would like to thank Zaicun Li for insights and justifications regarding mathematical proofs.

References

Arora, S., Cohen, N., and Hazan, E. (2018). On the optimization of deep networks: Implicit acceleration by overparameterization. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 244–253. PMLR.

Arora, S., Cohen, N., Hu, W., and Luo, Y. (2019). Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Burer, S. and Monteiro, R. (2003). A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming, Series B*, 95:329–357.

Chang, X., Li, Y., Oymak, S., and Thrampoulidis, C. (2020). Provable benefits of overparameterization in model compression: From double descent to pruning neural networks. In *AAAI Conference on Artificial Intelligence*.

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

Chen, M., Pennington, J., and Schoenholz, S. (2018). Dynamical isometry and a mean field theory of RNNs: Gating enables signal propagation in recurrent neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 873–882. PMLR.

Cheng, Y., Wang, D., Zhou, P., and Zhang, T. (2018). Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 35(1):126–136.

Chi, Y., Lu, Y. M., and Chen, Y. (2019). Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269.

Chou, H.-H., Gieshoff, C., Maly, J., and Rauhut, H. (2023). Gradient descent for deep matrix factorization: Dynamics and implicit bias towards low rank. *Applied and Computational Harmonic Analysis*, page 101595.

Dar, Y., Muthukumar, V., and Baraniuk, R. G. (2021). A farewell to the bias-variance tradeoff? An overview of the theory of overparameterized machine learning. *arXiv preprint arXiv:2109.02355*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Eftekhari, A. (2020). Training linear neural networks: Non-local convergence and complexity results. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 2836–2847. PMLR.

Gidel, G., Bach, F., and Lacoste-Julien, S. (2019). Implicit regularization of discrete gradient dynamics in linear neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Haderlein, J. F., Mareels, I. M. Y., Peterson, A. D. H., Eskikand, P. Z., Burkitt, A. N., and Grayden, D. B. (2021). On the benefit of overparameterization in state reconstruction. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 1580–1585.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4).

Helmke, U. and Moore, J. (1996). Optimization and dynamical systems. *Proceedings of the IEEE*, 84(6).

Horváth, S., Laskaridis, S., Rajput, S., and Wang, H. (2023). Maestro: Uncovering low-rank structures via trainable decomposition. *arXiv preprint arXiv:2308.14929*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Huh, M., Mobahi, H., Zhang, R., Cheung, B., Agrawal, P., and Isola, P. (2023). The low-rank simplicity bias in deep networks. *Transactions on Machine Learning Research*.

Idelbayev, Y. and Carreira-Perpiñán, M. A. (2020). Low-rank compression of neural nets: Learning the rank of each layer. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8046–8056.

Jacot, A., Ged, F., Şimşek, B., Hongler, C., and Gabriel, F. (2021). Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity. *arXiv preprint arXiv:2106.15933*.

Jain, P., Netrapalli, P., and Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, page 665–674, New York, NY, USA.

Jiang, J., Zhou, J., Wang, P., Qu, Q., Mixon, D., You, C., and Zhu, Z. (2023a). Generalized neural collapse for a large number of classes. *arXiv preprint arXiv:2310.05351*.

Jiang, L., Chen, Y., and Ding, L. (2023b). Algorithmic regularization in model-free overparametrized asymmetric matrix factorization. *SIAM Journal on Mathematics of Data Science*, 5(3):723–744.

Jin, J., Li, Z., Lyu, K., Du, S. S., and Lee, J. D. (2023). Understanding incremental learning of gradient descent: A fine-grained analysis of matrix sensing. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15200–15238. PMLR.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Khodak, M., Tenenholtz, N. A., Mackey, L., and Fusi, N. (2021). Initialization and regularization of factorized neural layers. In *International Conference on Learning Representations*.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.

Li, X., Liu, S., Zhou, J., Lu, X., Fernandez-Granda, C., Zhu, Z., and Qu, Q. (2023). Principled and efficient transfer learning of deep models via neural collapse. *arXiv preprint arXiv:2212.12206*.

Li, Z., Luo, Y., and Lyu, K. (2021). Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. In *International Conference on Learning Representations*.

Lialin, V., Shivagunde, N., Muckatira, S., and Rumshisky, A. (2023). Stack more layers differently: High-rank training through low-rank updates. *arXiv preprint arXiv:2307.05695*.

Lin, D., Sun, R., and Zhang, Z. (2021). Faster directional convergence of linear neural networks under spherically symmetric data. In *Advances in Neural Information Processing Systems*.

Liu, C. and Belkin, M. (2020). Accelerating SGD with momentum for over-parameterized learning. In *International Conference on Learning Representations*.

Ma, C., Xu, X., Tong, T., and Chi, Y. (2023). Provably accelerating ill-conditioned low-rank estimation via scaled gradient descent, even with overparameterization. *arXiv preprint arXiv:2310.06159*.

Ma, J. and Fattahi, S. (2022). Blessing of depth in linear regression: Deeper models have flatter landscape around the true solution. In *Advances in Neural Information Processing Systems*, volume 35, pages 34334–34346. Curran Associates, Inc.

Papyan, V., Han, X., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences of the United States of America*, 117:24652 – 24663.

Pennington, J., Schoenholz, S., and Ganguli, S. (2018). The emergence of spectral universality in deep networks. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1924–1932. PMLR.

Recht, B., Fazel, M., and Parrilo, P. A. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *2nd International Conference on Learning Representations, ICLR*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Stöger, D. and Soltanolkotabi, M. (2021). Small random initialization is akin to spectral learning: Optimization and generalization guarantees for overparameterized low-rank matrix reconstruction. In *Advances in Neural Information Processing Systems*, volume 34, pages 23831–23843. Curran Associates, Inc.

Sun, Y.-L., Narang, A., Gulluk, H. I., Oymak, S., and Fazel, M. (2022). Towards sample-efficient overparameterized meta-learning. In *Neural Information Processing Systems*.

Teschl, G. (2012). Ordinary differential equations and dynamical systems.

Tripuraneni, N., Adlam, B., and Pennington, J. (2021). Overparameterization improves robustness to covariate shift in high dimensions. In *Advances in Neural Information Processing Systems*, volume 34, pages 13883–13897. Curran Associates, Inc.

Wang, H., Agarwal, S., U-chupala, P., Tanaka, Y., Xing, E., and Papailiopoulos, D. (2023). Cuttlefish: Low-rank model training without all the tuning.

Wang, P., Liu, H., Yaras, C., Balzano, L., and Qu, Q. (2022). Linear convergence analysis of neural collapse with unconstrained features. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*.

Yaras, C., Wang, P., Hu, W., Zhu, Z., Balzano, L., and Qu, Q. (2023). The law of parsimony in gradient descent for learning deep linear networks. *arXiv preprint arXiv:2306.01154*.

Yaras, C., Wang, P., Zhu, Z., Balzano, L., and Qu, Q. (2022). Neural collapse with normalized features: A geometric analysis over the riemannian manifold. In *Advances in Neural Information Processing Systems*, volume 35, pages 11547–11560. Curran Associates, Inc.

Zhai, Y., Tong, S., Li, X., Cai, M., Qu, Q., Lee, Y. J., and Ma, Y. (2023). Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Conference on Parsimony and Learning (Proceedings Track)*.

Zhao, J., Zhang, Y., Chen, B., Schaefer, F. T., and Anandkumar, A. (2023). Incremental low-rank learning. In *Workshop on Efficient Systems for Foundation Models @ ICML2023*.

Zhou, J., Li, X., Ding, T., You, C., Qu, Q., and Zhu, Z. (2022a). On the optimization landscape of neural collapse under mse loss: Global optimality with unconstrained features. In *International Conference on Machine Learning*, pages 27179–27202. PMLR.

Zhou, J., You, C., Li, X., Liu, K., Liu, S., Qu, Q., and Zhu, Z. (2022b). Are all losses created equal: A neural collapse perspective. *Advances in Neural Information Processing Systems*, 35:31697–31710.

Zhu, Z., Ding, T., Zhou, J., Li, X., You, C., Sulam, J., and Qu, Q. (2021). A geometric analysis of neural collapse with unconstrained features. In *Advances in Neural Information Processing Systems*.

Zou, D., Wu, J., Braverman, V., Gu, Q., Foster, D. P., and Kakade, S. (2021). The benefits of implicit regularization from sgd in least squares problems. In *Advances in Neural Information Processing Systems*, volume 34, pages 5456–5468. Curran Associates, Inc.

7 Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [**Yes**/No/Not Applicable]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [**Yes**/No/Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [**Yes**/No/Not Applicable]
   (b) Complete proofs of all theoretical results. [**Yes**/No/Not Applicable]
   (c) Clear explanations of any assumptions. [**Yes**/No/Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [**Yes**/No/Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [**Yes**/No/Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [**Yes**/No/Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [**Yes**/No/Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [**Yes**/No/Not Applicable]
   (b) The license information of the assets, if applicable. [Yes/No/**Not Applicable**]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes/No/**Not Applicable**]
   (d) Information about consent from data providers/curators. [Yes/No/**Not Applicable**]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes/No/**Not Applicable**]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Yes/No/**Not Applicable**]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/**Not Applicable**]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/**Not Applicable**]

Supplementary Materials

A ADDITIONAL RESULTS

In this section, we present additional experimental results to supplement those presented in the main text. These results include extensive plots for the validity of Assumption 1, results for deep matrix sensing, ablation studies, and a discussion on the prevalence of low-rank updates for nonlinear networks. All experiments were run using either a CPU with processor 3.0 GHz Intel Xeon Gold 6154 or a NVIDIA V100 GPU. The code to reproduce our results is available at https://github.com/soominkwon/comp-deep-nets. For clarity in notation throughout the Appendix, we provide a table of notation in Table 2.

Notation	Definition
$L \in \mathbb{N}$	Depth of the DLN
$\hat{r} \in \mathbb{N}$	Estimated rank for the compressed DLN
$\epsilon \in \mathbb{R}_+$	Weight initialization scale
$\eta \in \mathbb{R}_+$	Learning rate
$\alpha \in \mathbb{R}_+$	Discrepant learning rate scale
$\boldsymbol{W}_l \in \mathbb{R}^{d \times d}$	l-th weight matrix of original DLN
$\widetilde{\boldsymbol{W}}_l \in \mathbb{R}^{\hat{r} \times \hat{r}}$	l-th weight matrix of compressed DLN

Table 2: Summary of the notation used throughout this work.

A.1 Experimental Results on Incremental Learning

Here, we aim to present more empirical results to validate our observation of the incremental learning phenomenon. To that end, we synthetically generate a target matrix $\boldsymbol{M}^* \in \mathbb{R}^{d \times d}$ with $d = 100$ and rank $r = 5$ and $r = 10$ with $\hat{r} = 2r$. We consider deep matrix factorization, sensing, and completion, where our goal is to fit the target matrix $\boldsymbol{M}^*$ using a DLN of $L = 3$ with initialization scale $\epsilon = 10^{-3}$. To train the weights, we run GD with step size $\eta = 10$ and $\alpha = 5, 2, 5$ for matrix factorization, sensing, and completion, respectively. For deep matrix sensing, each sensing matrix $\boldsymbol{A}_i$ was filled with i.i.d. Gaussian entries $\mathcal{N}(0, 1)$, for all $i \in [m]$, where $m = 2000$. For matrix completion, we consider the MCAR setting, with 20% observed entries. The same setup was also used to generate Figure 4. The results are displayed in Figures 7, 8 9, 10, 11. The same setup was also used to generate Figure 4. In Figure 12, we display a two-dimensional plot of the change in singular values. These plots demonstrate that throughout deep matrix factorization and sensing problems, the principal components of the DLN are fitted incrementally, starting from the largest principal component to the next. The two-dimensional plot in Figure 12 illustrates this more precisely. We can observe that the learning of the singular values occurs one at a time, while the other singular values remain close to their initial values until the previous one is adjusted to its target singular value. These findings serve to support our assumptions and observations as outlined in Assumption 1.

Furthermore, recall that in our definition of Assumption 1, there exist constants $c_{\text{val}}, c_{\text{vec}} \in [0, 1]$ that define the precision with which the singular values and vectors fit the target values, respectively. We perform a study on deep matrix factorization to demonstrate that both c_{val} and c_{vec} are (very) close to 0. To this end, we synthetically generate a target matrix $\boldsymbol{M}^* \in \mathbb{R}^{d \times d}$ with $d = 100$ and rank $r = 3$ with learning rate $\eta = 5$. We measure the difference in singular values and vectors throughout the course of GD, and display our results in Figure 13.

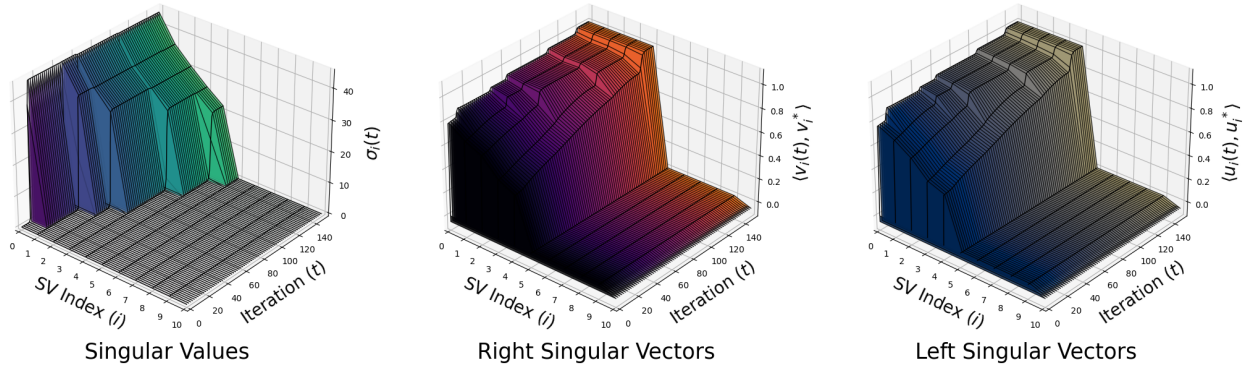


Figure 7: **Occurence of the incremental learning phenomenon in matrix completion.** We observe that the first $r = 5$ singular values are fitted incrementally, along with their respective singular subspaces.

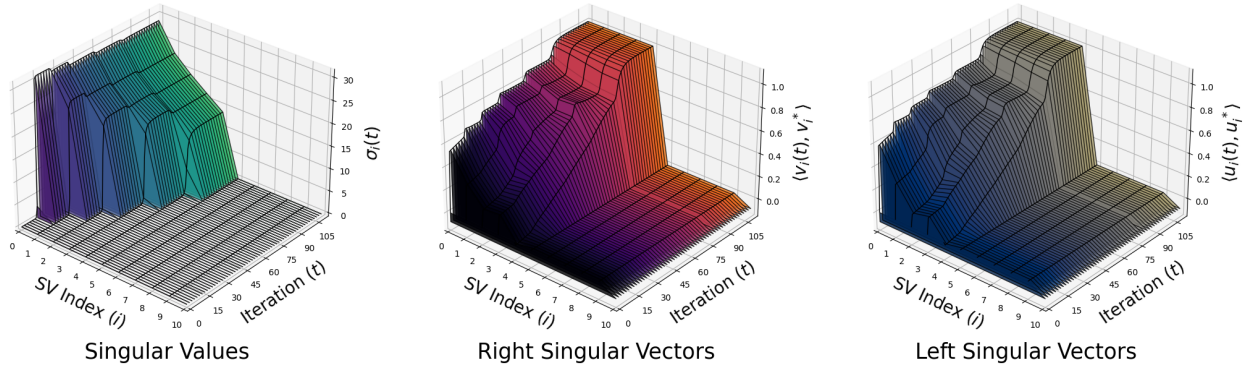


Figure 8: **Occurence of the incremental learning phenomenon in matrix sensing.** We observe that the first $r = 5$ singular values are fitted incrementally, along with their respective singular subspaces.

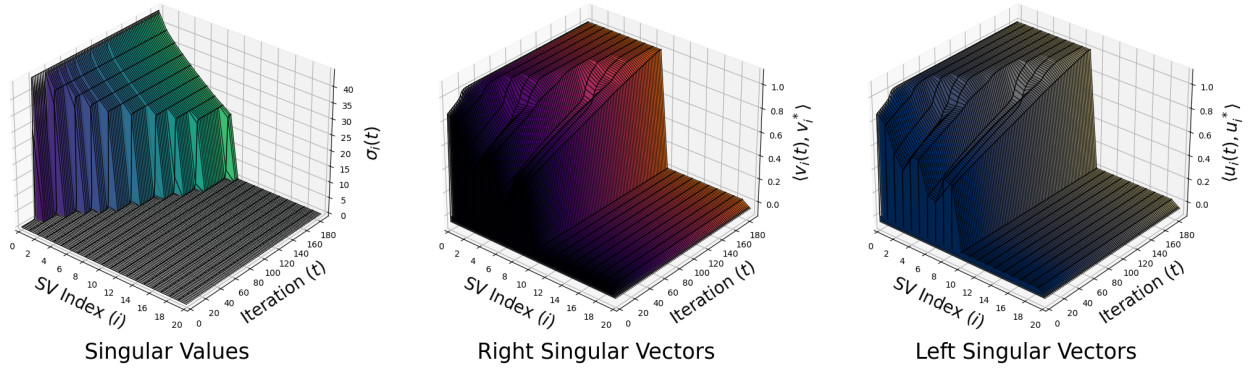


Figure 9: **Occurence of the incremental learning phenomenon in matrix factorization.** We observe that the first $r = 10$ singular values are fitted incrementally, along with their respective singular subspaces.

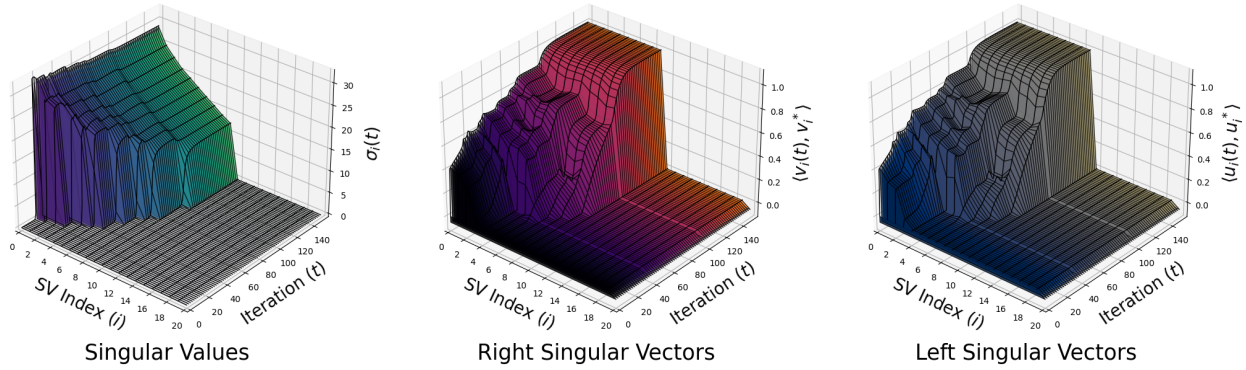


Figure 10: **Occurence of the incremental learning phenomenon in matrix sensing.** We observe that the first $r = 10$ singular values are fitted incrementally, along with their respective singular subspaces.

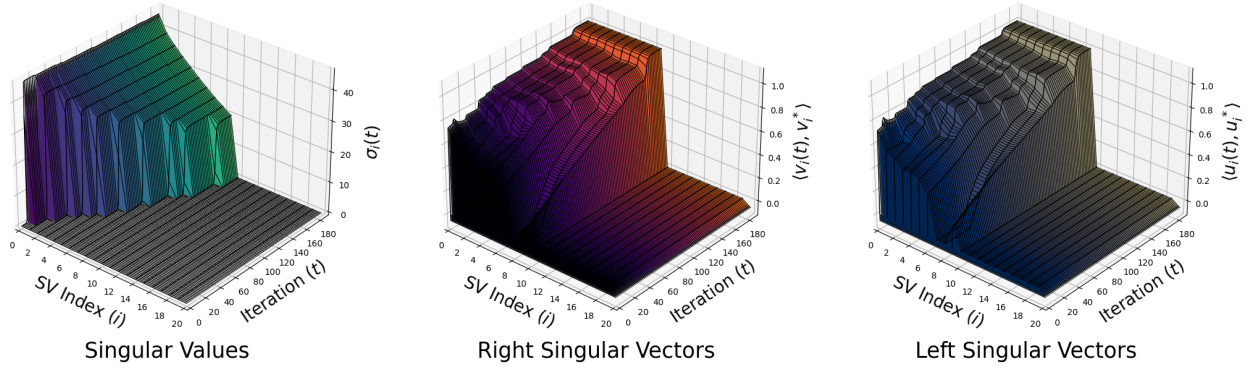


Figure 11: **Occurence of the incremental learning phenomenon in matrix completion.** We observe that the first $r = 10$ singular values are fitted incrementally, along with their respective singular subspaces.

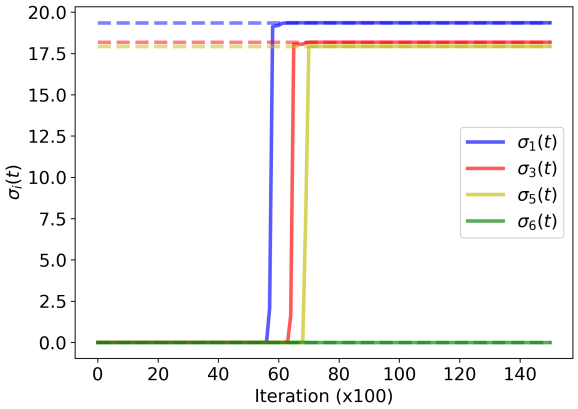


Figure 12: **Occurence of the incremental learning phenomenon in matrix factorization, with a more close up view on its singular values.** Since the underlying matrix is rank $r = 5$, the sixth singular value and onwards stay (almost) unchanged from initialization. The dotted lines represent the magnitude of the target singular value (σ_i^*).

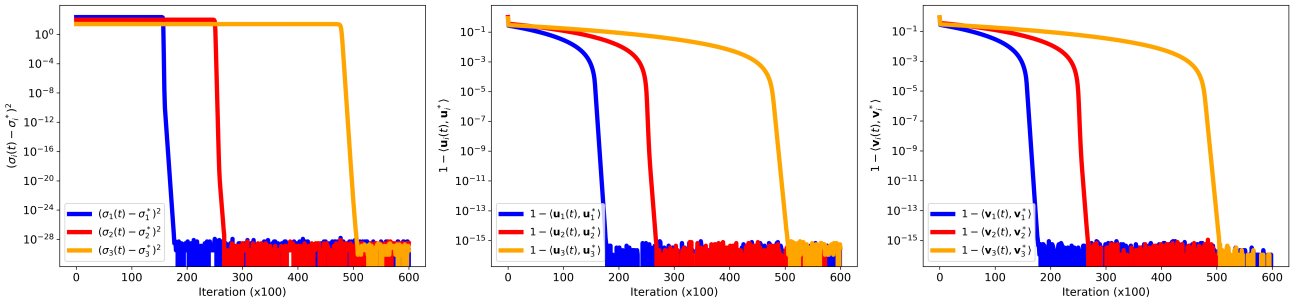
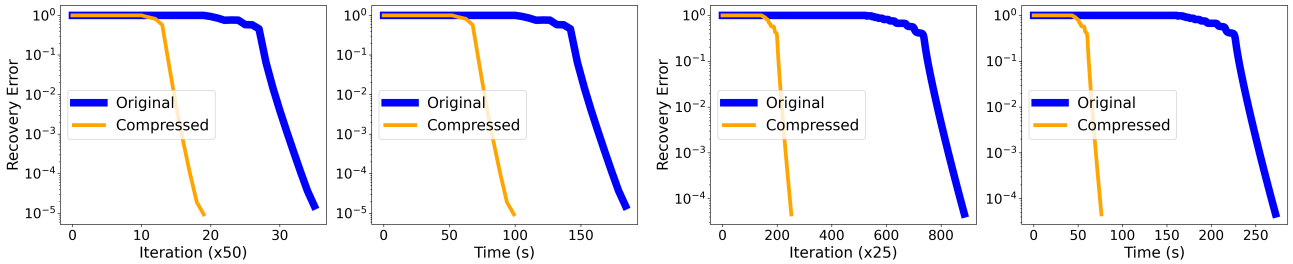


Figure 13: **Experiments observing the values of c_{val} and c_{vec} throughout the course of GD.** These plots demonstrate that after t_i, which indicate the time in which the i-th principal components are learned in Assumption 1, the values c_{val} and c_{vec} are very close to 0 (smaller than approximately 10^{-15}).



$d = 100,\ r = 5,\ m = 2000$ $\qquad$ $d = 200,\ r = 10,\ m = 8000$

Figure 14: **Results on deep matrix sensing with random Gaussian measurements.** We observe the same phenomena in the matrix sensing setting similar to Figures 5 and 6, where the compressed network consistently outperforms the original network.

A.2 Results on Deep Matrix Sensing

In this section, we present the deferred results for deep matrix sensing. We consider modeling a low-rank matrix $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ with varying dimensions d and rank r. For the sensing operator, we use random Gaussian matrices $\boldsymbol{A}_i \in \mathbb{R}^{d\times d}$ with varying values of measurements m. As previously discussed, to initialize $\widetilde{\boldsymbol{W}}_L(0)$ and $\widetilde{\boldsymbol{W}}_1(0)$, we take the left and right singular subspaces of the surrogate matrix

$$\boldsymbol{S} = \mathcal{A}^\dagger \mathcal{A}(\boldsymbol{M}^*).$$

For the hyperparameters, we use the same setup as described in Section A.1. The results are displayed in Figure 14. In Figure 14, we observe that for deep matrix sensing, we observe the same phenomenon, where the compressed network consistently outperforms the original network in terms of recovery error for all iterations of GD. This empirically shows the benefits of spectral initialization and the use of DLNs.

A.3 Ablation Studies

In this section, we conduct several ablation studies to demonstrate (1) the effect of the learning rate scale α, (2) the performance of the compressed network with small *random* initialization, (3) the added time complexity of SVD, and (4) the effect of depth.

The Effect of α. In Section 2.2, we introduced the concept of a learning rate ratio (or scale), denoted as α. Recall that in Figure 3, we showed that the factor matrices $\widetilde{\boldsymbol{W}}_L$ and $\widetilde{\boldsymbol{W}}_1$ ultimately align the the singular subspaces of the target matrix. By increasing α (i.e., by choosing $\alpha > 1$), we can hope to fit these target subspaces more quickly, leading to accelerated convergece in terms of iterations. To this end, we peform an ablation study where we vary alpha from $\alpha \in [0.5, 10]$ in increments of 0.25, and report the recovery error on synthetic matrix completion. We present our results in Figure 15, where we observe for larger values of α significantly improves convergence. For values of $\alpha \leq 1$, we have slower convergence in terms of iterations than the original wide DLN,

but since each iteration is much faster, it still takes much less time to train the compressed DLN. However, for large values of α (e.g. $\alpha \gg 10$), the training becomes unstable and often diverges. In Figure 17, we demonstrate a case of this, where for large values of α, the compressed network overfits and incurs a large recovery error. For smaller values of α (e.g., $\alpha = 0.1$), the networks takes significantly longer to converge.

On the other hand, choosing a learning rate scale $\alpha > 1$ may like an unfair advantage for the compressed network. For a fairer comparison, we first fix $\alpha = 1$ and perform a grid search of 20 evenly spaced steps for $\eta \in [10, 100]$ and choose the value of η that converges the fastest (in terms of number of iterations) for a tolerance level of 10^{-12} for both networks. In Figure 16, we demonstrate that the original network still converges slower than the compressed network.

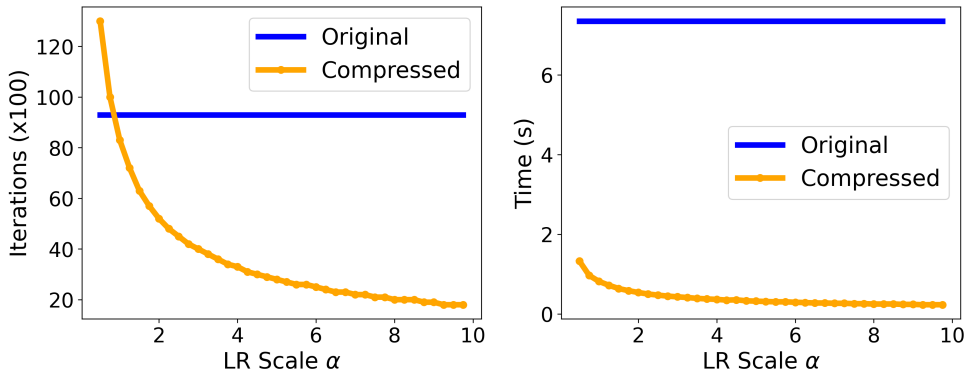


Figure 15: **Ablation study on the performance of the compressed DLN with varying values of α.** For small values of α, the compressed network takes more iterations to reach the convergence criteria, but since the time complexity of each iteration is smaller, it still takes much shorter time overall.

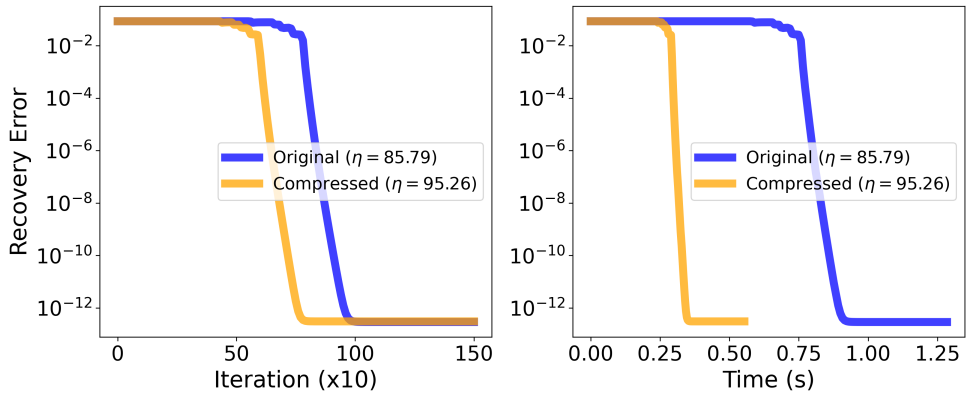


Figure 16: **Ablation study on the performance of the compressed DLN with $\alpha = 1$ and different learning rates η for both networks.** We perform a grid search of 20 evenly spaced steps for $\eta \in [10, 100]$ and choose the value of η that converges the fastest for a tolerance level of 10^{-12}. We demonstrate that the original network still converges slower than the compressed network.

The Use of Small Random Initialization. Thus far, all of our experiments and theory relied on the fact that the original network was initialized with orthogonal weights scaled by a small scale $\epsilon > 0$. In this section, we perform a study to show that our compressed network also outperforms the original network when the original network is initialized with random weights also scaled by ϵ. We consider deep matrix factorization and matrix completion, following the setting of Section 4.1. We showcase our results in Figure 18, where show that the same phenomenon persists in this setting as well.

Added Time Complexity of Taking SVD. Recall that our algorithm involves taking the SVD of a carefully constructed surrogate matrix to initialize the weights of the compressed network. In this study, we report the

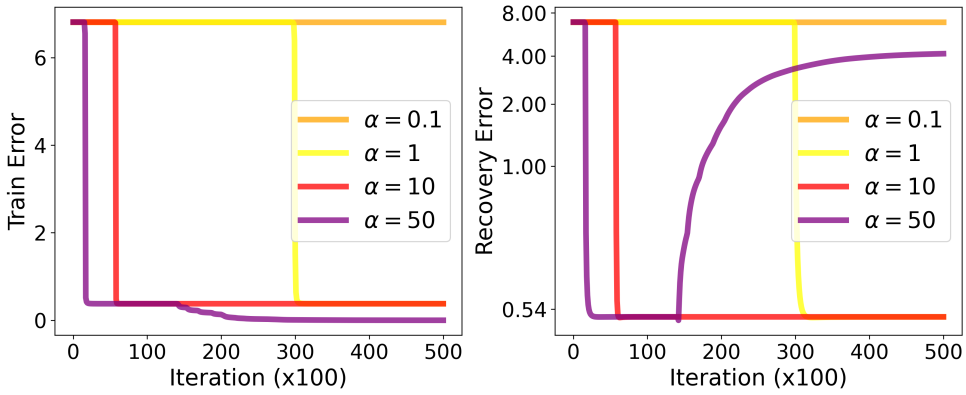
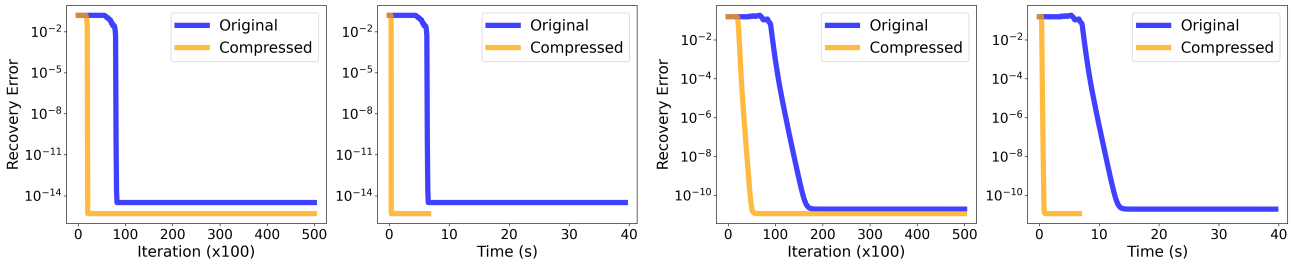


Figure 17: **Ablation study on α on the MovieLens dataset.** These plots demonstrate that for large values of α, the compressed network can overfit, whereas smaller values of α increases the time it takes for the model to converge.



Deep Matrix Factorization | Deep Matrix Completion

Figure 18: **Comparison of the performance of networks when the original network is initialized with near-zero random weights.** We observe that the compressed network still outperforms the original wide network in this setting for both deep matrix factorization and completion.

time (in seconds) it takes to train both the original network and the compressed network, including the time taken for the SVD step, for varying values of d (the dimensions of the target matrix). Our results are presented in Table 3. We observe that the time taken for the SVD is almost negligible compared to the training time for synthetic data. For real larger-scale experiments, while we anticipate that computing the SVD may take a longer time, it will also significantly reduce the training time, resulting in a smaller overall time.

Method	SVD	Training	Total
DLN ($d = 100$)	-	184.3	184.3
C-DLN ($d = 100$)	0.009	109.7	109.7
DLN ($d = 300$)	-	511.2	511.2
C-DLN ($d = 300$)	0.082	381.3	381.4

Table 3: **The time (measured in seconds) required to train both the compressed and original networks, accounting for the SVD step.** We note that the time taken to compute the SVD is nearly negligible.

The Effect of Depth. In this section, we make a quick note on the effect of depth on matrix completion. We synthetically generate data according to Section 4.1. Here, our objective is to answer whether the compressed network can still outperform the original network when the number of layers is small (e.g. $L = 2$). The work by Arora et al. Arora et al. (2019) suggests that the implicit regularization property is "stronger" as a function of depth, where deeper networks favor more low-rank solutions. In Figure 19, we illustrate that when the number of observed measurements is very limited and the number of layers is small, both networks seem to perform

considerably worse compared to the 3-layer case. While the training error is still smaller for the compressed DLN, the recovery error is slightly higher. This result implies that deeper networks are more favorable than shallow ones when the number of measurements is extremely limited, and the advantages of the compressed network become especially apparent when both networks can achieve a low recovery error.

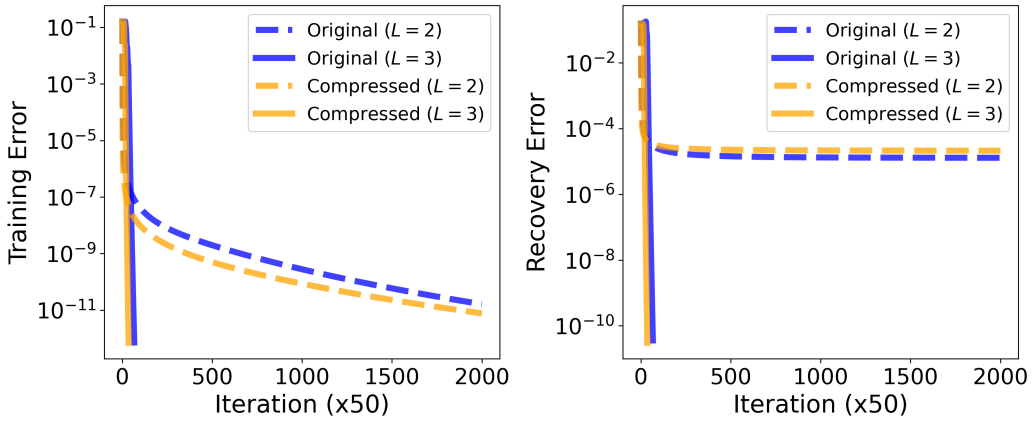


Figure 19: **The effect of depth on deep matrix completion with limited number of measurements.** We show that the compressed network only has advantages over the original network when the original network converges in terms of recovery (or test) error.

A.4 Additional Results and Details for Nonlinear Networks

Additional Results. In Figure 1, we plotted the singular values of the change in weight updates from initialization of the penultimate layer matrix (i.e., $\boldsymbol{W}(t) - \boldsymbol{W}(0)$ where $\boldsymbol{W}$ denotes the penultimate layer matrix) and showed that the changes happen within a low-dimensional subspace. In Figure 20, we plot the change in the top-r singular vectors. We use the subspace distance defined as

$$\text{Subspace Distance } = r - \|\boldsymbol{U}_r(t)^\top \boldsymbol{U}_r(t+1)\|_{\mathsf{F}}^2, \tag{14}$$

where $\boldsymbol{U}_r$ denotes the singular vectors of $\boldsymbol{W}$ corresponding to the top-r singular values. For this experiment, we choose $r = 10$ and initialize the networks as described in the training details (next subsection). Here, we observe that the subspace distance is very small throughout training all network architectures, which hints that the subspaces greatly overlap throughout all t (i.e., the training occurs within an invariant subspace).

In Figure 21, we present additional plots for experiments on deep nonlinear networks. Here, we depict the change in test accuracy over GD iterations for MLP and ViT trained on FashionMNIST and CIFAR-10, respectively. We display the test accuracies for the original (in green), original network whose penultimate layer is a DLN (Original + DLN in blue), and the original network whose penultimate layer is a compressed DLN (in orange) throughout GD and their respective training times in seconds. We observe that networks whose penultimate layer is modeled as a DLN exhibit the best performance in terms of test accuracy, but they take the longest time to train due to the additional overparameterization. By using our compression approach, we can reduce this training time by almost $2\times$, while achieving very similar test accuracy. Interestingly, Figure 21 shows that for ViT, we can outperform the other networks using our compressed DLN, while having the shortest training time.

Training Details. To plot Figure 1, we observe the change in singular values of the penultimate weight matrix for DLNs, MLPs, VGG (Simonyan and Zisserman, 2015), and ViT-B (Dosovitskiy et al., 2021). We train the DLN starting from orthogonal initialization using SGD with learning rate 0.01 and batch size 128. All of the nonlinear networks were initialized using random uniform initialization. For MLP, we train using the same parameters as the DLN. For VGG, we use batch size 128 with learning rate 0.05, weight decay parameter 5×10^{-4}, momentum 0.9 and step size scheduler with cosine annealing. Lastly, for ViT-B, we train using ADAM with a batch size of 512 and learning rate 10^{-5} with cosine annealing. The ViT-B architecture is the same as the one presented by Dosovitskiy et al. (2021).

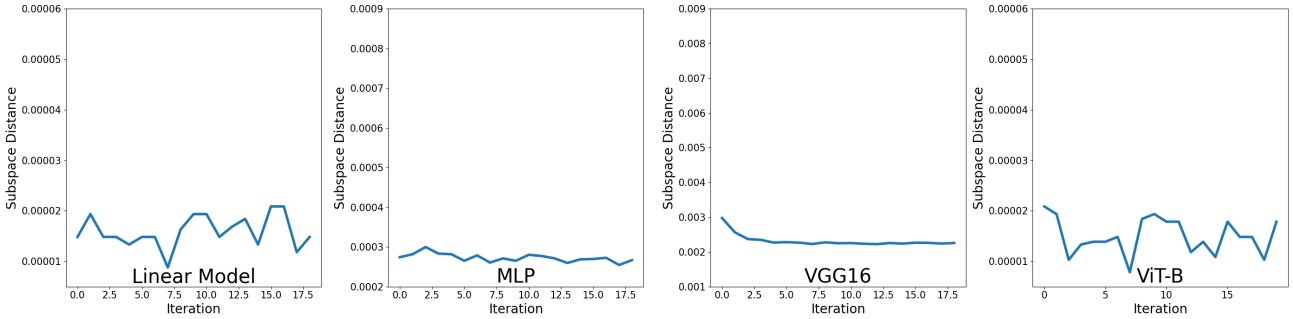
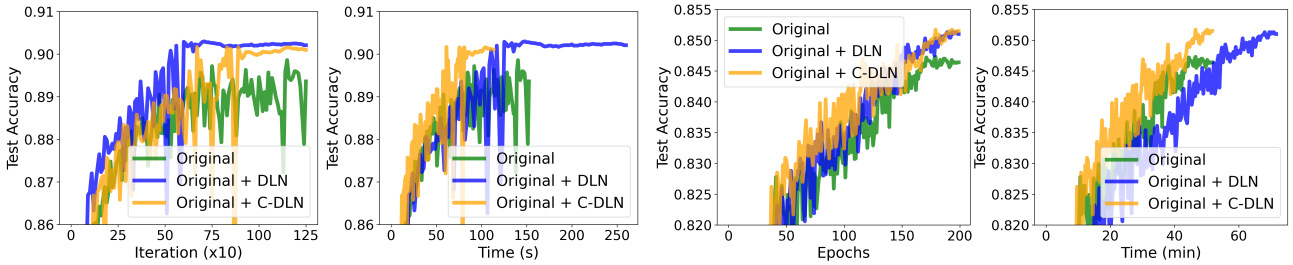


Figure 20: **Distance between consecutive singular subspaces of the penultimate weight matrices.** These plots show that the subspace distance defined in Equation (14) is small, hinting that the training happens within an invariant subspace.



Results on FashionMNIST

Results on CIFAR-10

Figure 21: **Results on classification tasks with nonlinear networks with overparameterized penultimate layers.** Left: Results with FashionMNIST dataset with MLP. Right: Results on CIFAR-10 dataset with ViT. For both experiments, we observe that the compressed networks (colored in orange) has similar test accuracy (if not better) than the original network (colored in blue) throughout training, while reducing run-time by almost $2\times$. The plot colored in green represents the network with a standard penultimate layer, which has lower test accuracy throughout all experiments.

B THEORETICAL RESULTS

B.1 Deferred Proofs for Spectral Initialization

To keep this section self-contained, we will first restate all of our results. Throughout this section, we will consider the compressed deep matrix factorization problem, which at GD iterate t is defined as

$$\ell(\widetilde{\boldsymbol{\Theta}}(t)) = \frac{1}{2}\|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2,$$

where $\widetilde{\boldsymbol{W}}_L(0) = \epsilon \cdot \boldsymbol{U}_{\hat{r}}^*$, $\widetilde{\boldsymbol{W}}_1(0) = \epsilon \cdot \boldsymbol{V}_{\hat{r}}^{*\top}$, and $\widetilde{\boldsymbol{W}}_l(0) = \epsilon \cdot \boldsymbol{I}_{\hat{r}}$ for $2 \le l \le L-1$, for some small initialization scale $\epsilon > 0$.

Theorem 1. *Let $\boldsymbol{M}^* \in \mathbb{R}^{d\times d}$ be a matrix of rank r and $\boldsymbol{M}^* = \boldsymbol{U}^*\boldsymbol{\Sigma}^*\boldsymbol{V}^{*\top}$ be a SVD of $\boldsymbol{M}^*$. Suppose we run Algorithm 1 with $\alpha = 1$ to update all weights $\left(\widetilde{\boldsymbol{W}}_l\right)_{l=1}^L$ of Equation (7), where $\mathcal{A} = Id$. Then, the end-to-end compressed DLN possesses low-dimensional structures, in the sense that for all $t \ge 1$, $\widetilde{\boldsymbol{W}}_{L:1}(t)$ admits the following decomposition:*

$$\widetilde{\boldsymbol{W}}_{L:1}(t) = \boldsymbol{U}_{\hat{r}}^* \begin{bmatrix} \boldsymbol{\Lambda}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \beta(t)^L \cdot \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} \boldsymbol{V}_{\hat{r}}^{*\top}, \tag{15}$$

where $\boldsymbol{\Lambda}(t) \in \mathbb{R}^{r\times r}$ is a diagonal matrix with entries $\lambda_i(t)^L$, where

$$\lambda_i(t) = \lambda_i(t-1) \cdot \left(1 - \eta \cdot (\lambda_i(t-1)^L - \sigma_i^*) \cdot \lambda_i(t-1)^{L-2}\right), \quad 1 \le i \le r, \tag{16}$$

with $\lambda_i(0) = \epsilon$ *and* σ_i^* *is the* i*-th diagonal entry of* $\mathbf{\Sigma}^*$ *and*

$$\beta(t) = \beta(t-1) \cdot \left(1 - \eta \cdot \beta(t-1)^{2(L-1)}\right), \tag{17}$$

with $\beta(0) = \epsilon$.

Proof. We will prove using mathematical induction.

Base Case. For the base case at $t = 0$, the decomposition holds immediately by the choice of initialization:

$$\begin{aligned}
\widetilde{\boldsymbol{W}}_{L:1}(0) &= \underbrace{\boldsymbol{U}_{\hat{r}}^* \cdot \epsilon \boldsymbol{I}_{\hat{r}}}_{\widetilde{\boldsymbol{W}}_L(0)} \cdot \underbrace{\epsilon^{(L-2)} \boldsymbol{I}_{\hat{r}}}_{\widetilde{\boldsymbol{W}}_{2:L-1}(0)} \cdot \underbrace{\epsilon \boldsymbol{I}_{\hat{r}} \cdot \boldsymbol{V}_{\hat{r}}^{*\top}}_{\widetilde{\boldsymbol{W}}_1(0)} \\
&= \boldsymbol{U}_{\hat{r}}^* \cdot \begin{bmatrix} \epsilon^L \boldsymbol{I}_r & \mathbf{0} \\ \mathbf{0} & \epsilon^L \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} \cdot \boldsymbol{V}_{\hat{r}}^{*\top} \\
&= \boldsymbol{U}_{\hat{r}}^* \cdot \begin{bmatrix} \mathbf{\Lambda}(0) & \mathbf{0} \\ \mathbf{0} & \beta(0)^L \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} \cdot \boldsymbol{V}_{\hat{r}}^{*\top}.
\end{aligned}$$

Inductive Hypothesis. Now, by the inductive hypothesis, suppose that the decomposition holds for some $t \geq 0$. Notice that by the rotational invariance of the Frobenius norm, we can rewrite the objective function into

$$\begin{aligned}
\ell(\widetilde{\mathbf{\Theta}}(t)) &= \frac{1}{2} \|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 && (18) \\
&= \frac{1}{2} \left\| \boldsymbol{U}_{\hat{r}}^* \begin{bmatrix} \mathbf{\Lambda}(t) & \mathbf{0} \\ \mathbf{0} & \beta(t)^L \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} \boldsymbol{V}_{\hat{r}}^{*\top} - \boldsymbol{U}_{\hat{r}}^* \begin{bmatrix} \mathbf{\Sigma}_r^* & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \boldsymbol{V}_{\hat{r}}^{*\top} \right\|_{\mathsf{F}}^2 && (19) \\
&= \frac{1}{2} \left\| \begin{bmatrix} \mathbf{\Lambda}(t) & \mathbf{0} \\ \mathbf{0} & \beta(t)^L \boldsymbol{I}_{\hat{r}-r} \end{bmatrix} - \begin{bmatrix} \mathbf{\Sigma}_r^* & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \right\|_{\mathsf{F}}^2 && (20)
\end{aligned}$$

where $\mathbf{\Sigma}_r^*$ is the leading r principal submatrix $\mathbf{\Sigma}^*$. Then, Equation (20) implies that updating the factor $\widetilde{\boldsymbol{W}}_L(t)$ is equivalent to updating a diagonal matrix of dimensions $\hat{r}$ and multiplying by $\boldsymbol{U}_{\hat{r}}^*$ (and respectively $\widetilde{\boldsymbol{W}}_1(t)$ with $\boldsymbol{V}_{\hat{r}}^{*\top}$). To this end, we will consider updating the diagonal entries to show that the decomposition holds for $t+1$.

Inductive Step. We will first consider the top-r components. Notice that the i-th diagonal entry $\forall i \in [r]$ of the l-th layer matrix $\forall l \in [L]$ are all scalars (and hence are all the same), and so the update steps for the i-th entry are given by

$$\begin{aligned}
\lambda_i(t+1) &= \lambda_i(t) - \eta \cdot \left(\lambda_i(t)^L - \sigma_i^*\right) \cdot \lambda_i(t)^{L-1} \\
&= \lambda_i(t) \cdot \left(1 - \eta \cdot (\lambda_i(t)^L - \sigma_i^*) \cdot \lambda(t)_i^{L-2}\right).
\end{aligned}$$

Thus, the i-th entry of $\mathbf{\Lambda}(t)$ is given by $\lambda_i(t)^L$ for $1 \leq i \leq r$. Similarly, the update steps for the last $\hat{r} - r$ elements are given by

$$\begin{aligned}
\beta(t+1) &= \beta(t) - \eta \cdot \left(\beta(t)^L \cdot \beta(t)^{L-1}\right) \\
&= \beta(t) \cdot \left(1 - \eta \cdot \beta(t)^{2(L-1)}\right),
\end{aligned}$$

which gives us $\beta(t)^L$ for the last $\hat{r} - r$ elements. This completes the proof. □

Corollary 1. *Let* $\boldsymbol{W}_{L:1}(0)$ *denote the original DLN at* $t = 0$ *initialized with orthogonal weights according to Equation (3) and* $\widetilde{\boldsymbol{W}}_{L:1}(0)$ *denote the compressed DLN at* $t = 0$. *Then, we have*

$$\|\boldsymbol{W}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \|\widetilde{\boldsymbol{W}}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2. \tag{21}$$

Proof. Let $\boldsymbol{W}_{L:1}(0) = \boldsymbol{U}_L \boldsymbol{\Sigma}_{L:1} \boldsymbol{V}_1^\top$ and $\boldsymbol{M}^* = \boldsymbol{U}^* \boldsymbol{\Sigma}^* \boldsymbol{V}^{*\top}$ denote the SVD of $\boldsymbol{W}_{L:1}(0)$ and $\boldsymbol{M}^*$, respectively. Since $\boldsymbol{W}_{L:1}(0)$ is a product of orthogonal matrices scaled by ϵ, its singular values are $\boldsymbol{\Sigma}_{L:1} = \epsilon^L \boldsymbol{I}_d$. Then, we have

$$\begin{aligned}
\|\boldsymbol{W}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 &\geq \|\boldsymbol{\Sigma}_{L:1} - \boldsymbol{\Sigma}^*\|_{\mathsf{F}}^2 && \text{(By Lemma 1)} \\
&= \|\epsilon^L \boldsymbol{I}_d - \boldsymbol{\Sigma}^*\|_{\mathsf{F}}^2 \\
&\geq \|\epsilon^L \boldsymbol{I}_{\hat{r}} - \boldsymbol{\Sigma}^*_{\hat{r}}\|_{\mathsf{F}}^2 && (\hat{r} \leq d) \\
&= \|\widetilde{\boldsymbol{W}}_{L:1}(0) - \boldsymbol{M}^*\|_{\mathsf{F}}^2. && \text{(Theorem 1 at } t = 0)
\end{aligned}$$

This proves the result. □

B.2 Deferred Proofs for Incremental Learning

In this section, we provide the deferred proofs from Section 3.2. Recall that in this analysis, we will consider running gradient flow over the original network in Equation (1) with $\mathcal{A} = Id$ and over the compressed network in Equation (7). Before we present our proof for our main result, we first restate a result from Arora et al. (2019), which characterizes change in singular values of the original network $\boldsymbol{W}_{L:1}(t)$.

Proposition 1 (Arora et al. (2019))**.** *Consider running gradient flow over the deep matrix factorization problem defined in Equation (1) with $\mathcal{A} = Id$ and $L \geq 2$. The end-to-end weight matrix can be expressed as*

$$\boldsymbol{W}_{L:1}(t) = \boldsymbol{U}(t) \boldsymbol{S}(t) \boldsymbol{V}^\top(t), \tag{22}$$

where $\boldsymbol{U}(t)$ and $\boldsymbol{V}(t)$ are orthonormal matrices and $\boldsymbol{S}(t)$ is a diagonal matrix. Denote $\sigma_i(t)$ as the i-th entry of the matrix $\boldsymbol{S}(t)$. Then, each $\sigma_i(t)$ for all $i = 1, \ldots, d$ has the following gradient flow trajectory:

$$\dot{\sigma}_i(t) = -L \cdot (\sigma_i^2(t))^{1-1/L} \cdot \langle \nabla_{\boldsymbol{W}_{L:1}} \ell(\boldsymbol{\Theta}(t)), \boldsymbol{u}_i(t) \boldsymbol{v}_i^\top(t) \rangle, \tag{23}$$

where $\boldsymbol{u}_i(t)$ and $\boldsymbol{u}_i(t)$ denotes the i-th column of $\boldsymbol{U}(t)$ and $\boldsymbol{V}(t)$ respectively and

$$\nabla_{\boldsymbol{W}_{L:1}} \ell(\boldsymbol{\Theta}) = \boldsymbol{W}_{L:1} - \boldsymbol{M}^*. \tag{24}$$

We would like to remark that in order to directly use this result, we need to assume that the factor matrices are balanced at initialization, i.e.,

$$\boldsymbol{W}_{l+1}^\top(0) \boldsymbol{W}_{l+1}(0) = \boldsymbol{W}_l(0) \boldsymbol{W}_l^\top(0), \quad l = 1, \ldots, L-1.$$

Since we are using orthogonal initialization scaled with a small constant $\epsilon > 0$, we immediately satisfy this condition. Equipped with this result, we introduce our main result.

Theorem 2. *Let $\boldsymbol{M}^* \in \mathbb{R}^{d \times d}$ be a rank-r matrix and such that $\hat{r} \in \mathbb{N}$ with $\hat{r} \geq r$. Suppose that we run gradient flow with respect to the original DLN in Equation (1) with $\mathcal{A} = Id$ and with respect to the compressed network defined in Equation (7). Then, if Assumption 1 holds such that $c_{vec} = 0$, we have that $\forall t \geq 0$,*

$$\|\boldsymbol{W}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2. \tag{25}$$

Proof. For clarity, we organize the proof into subsections.

Notation & Assumptions. By Assumption 1, let us define a sequence of time points $t_1^{\text{orig}} \leq t_2^{\text{orig}} \leq \ldots \leq t_r^{\text{orig}}$ and $t_1^{\text{comp}} \leq t_2^{\text{comp}} \leq \ldots \leq t_r^{\text{comp}}$ in which the i-th principal components are fitted up to some precision defined by $c_{\text{val}}^{\text{orig}}, c_{\text{vec}}^{\text{orig}}$ and $c_{\text{val}}^{\text{comp}}, c_{\text{vec}}^{\text{comp}}$ for both the original and compressed network, respectively. Recall that t_i denotes the time in which i-th principal components are fitted and when the $(i+1)$-th components start being learned. We assume that $c_{\text{vec}}^{\text{orig}} = 0$ and that for all $t < t_i$,

$$\sigma_{i+1}(t) = \widetilde{\sigma}_{i+1}(t) = \epsilon^L,$$

where $\sigma_i(t)$ and $\widetilde{\sigma}_i(t)$ denotes the i-th singular value of the original and compressed network, respectively. Notice that by Theorem 1, we have that $\widetilde{\boldsymbol{u}}_i(t) = \boldsymbol{u}_i^*$ and $\widetilde{\boldsymbol{v}}_i(t) = \boldsymbol{v}_i^*$ for all t, where $\widetilde{\boldsymbol{u}}_i(t)$ and $\widetilde{\boldsymbol{v}}_i(t)$ denote the i-th left and right singular vector of $\widetilde{\boldsymbol{W}}_{L:1}(t)$ respectively, and so $c_{\text{vec}}^{\text{comp}} = 0$ by definition. The second assumption states that the singular values are learned incrementally, and that the trailing $(i+1)$ to d singular values stay at initialization until all the first i singular values are learned.

Roadmap of Proof. Notice that by using Lemma 1, we can write the inequality in Equation (25) as

$$\|\boldsymbol{W}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \sum_{i=1}^{\hat{r}} (\sigma_i(t) - \sigma_i^*)^2 \underset{?}{\geq} \sum_{i=1}^{\hat{r}} (\widetilde{\sigma}_i(t) - \sigma_i^*)^2 = \|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2.$$

Then, to establish the result, we will show that

$$\sum_{i=1}^{\hat{r}} (\sigma_i(t) - \sigma_i^*)^2 \geq \sum_{i=1}^{\hat{r}} (\widetilde{\sigma}_i(t) - \sigma_i^*)^2.$$

To do this, we will first show that $t_i^{\text{comp}} \leq t_i^{\text{orig}}$ for all $i \in [r]$ using mathematical induction, as that would imply that the singular values of the compressed network are learned faster, leading to a lower recovery error. Then, we will show that the inequality also holds for all $j \in [r+1, \hat{r}]$.

Base Case. Let t_0^{orig} and t_0^{comp} denote the initial time points. For the base case, the initialization gives us $t_0^{\text{orig}} = t_0^{\text{comp}} = 0$.

Inductive Hypothesis. By the inductive hypothesis, suppose that $t_i^{\text{orig}} \geq t_i^{\text{comp}}$ such that the i-th principal components of the compressed network are fitted faster than the original network.

Inductive Step (Top-r Components). For the inductive step, we need to show $t_{i+1}^{\text{orig}} \geq t_{i+1}^{\text{comp}}$. We will do this by analyzing the dynamics of the singular values of both networks.

Notice that by the inductive hypothesis and by assumption, for all $t < t_i^{\text{comp}} \leq t_i^{\text{orig}}$, we have

$$\sigma_{i+1}(t) = \widetilde{\sigma}_{i+1}(t) = \epsilon^L. \tag{26}$$

Then, for all $t > t_i^{\text{orig}} \geq t_i^{\text{comp}}$, by Proposition 1, the gradient dynamics of $\sigma_{i+1}(t)$ are governed by

$$\begin{aligned}
\dot{\sigma}_{i+1}(t) &= -L \cdot (\sigma_{i+1}^2(t))^{1-1/L} \cdot \langle \nabla_{\boldsymbol{W}_{L:1}} \ell(\boldsymbol{\Theta}(t)), \boldsymbol{u}_{i+1}(t)\boldsymbol{v}_{i+1}^\top(t) \rangle, \\
&= -L \cdot (\sigma_{i+1}^2(t))^{1-1/L} \cdot (\boldsymbol{u}_{i+1}^\top(t)\boldsymbol{W}_{L:1}(t)\boldsymbol{v}_{i+1}(t) - \boldsymbol{u}_{i+1}^\top(t)\boldsymbol{M}^*\boldsymbol{v}_{i+1}(t)) \\
&= -L \cdot (\sigma_{i+1}^2(t))^{1-1/L} \cdot (\sigma_{i+1}(t) - \boldsymbol{u}_{i+1}^\top(t)\boldsymbol{M}^*\boldsymbol{v}_{i+1}(t)) \\
&\leq -L \cdot (\sigma_{i+1}^2(t))^{1-1/L} \cdot (\sigma_{i+1}(t) - \sigma_{i+1}^*),
\end{aligned}$$

where the last inequality follows from that $c_{\text{vec}}^{\text{orig}} = 0$ and so $\boldsymbol{u}_{i+1}^\top(t)$ and $\boldsymbol{v}_{i+1}^\top(t)$ are vectors that are orthogonal to $\text{span}\{\boldsymbol{u}_j^*,\, 1 \leq j \leq i\}$ and $\text{span}\{\boldsymbol{v}_j^*,\, 1 \leq j \leq i\}$, such that $\boldsymbol{u}_{i+1}^\top(t)\boldsymbol{M}^*\boldsymbol{v}_{i+1}(t)$ is at most σ_{i+1}^*.

Similarly, notice that for the compressed network, we have for times $t > t_i^{\text{comp}}$,

$$\begin{aligned}
\dot{\widetilde{\sigma}}_{i+1}(t) &= -L \cdot (\widetilde{\sigma}_{i+1}^2(t))^{1-1/L} \cdot \langle \nabla_{\widetilde{\boldsymbol{W}}_{L:1}} \ell(\widetilde{\boldsymbol{\Theta}}(t)), \boldsymbol{u}_{i+1}^* \boldsymbol{v}_{i+1}^{*\top} \rangle, && (27) \\
&= -L \cdot (\sigma_{i+1}^2(t))^{1-1/L} \cdot (\boldsymbol{u}_{i+1}^{*\top} \widetilde{\boldsymbol{W}}_{L:1}(t)\boldsymbol{v}_{i+1}^* - \boldsymbol{u}_{i+1}^{*\top}\boldsymbol{M}^*\boldsymbol{v}_{i+1}^*) && (28) \\
&= -L \cdot (\widetilde{\sigma}_{i+1}^2(t))^{1-1/L} \cdot (\widetilde{\sigma}_{i+1}(t) - \sigma_{i+1}^*). && (29)
\end{aligned}$$

Therefore by the comparison theorem for ODEs (Theorem 1.3 of Teschl (2012)) we have that

$$\epsilon^L \leq \sigma_{i+1}(t) \leq \widetilde{\sigma}_{i+1}(t) \leq \sigma_{i+1}^*, \qquad \forall t > t_i^{\text{orig}}, \tag{30}$$

Then, for all t in $t_i^{\text{comp}} \leq t \leq t_i^{\text{orig}}$, the singular values of the compressed network follow the dynamics outlined in Equation (27), whereas the singular value of the original network stays at initialization by assumption. Thus,

$$\epsilon^L = \sigma_{i+1}(t) \leq \widetilde{\sigma}_{i+1}(t) \leq \sigma_{i+1}^*. \tag{31}$$

Hence, by combining Equation (26), Equation (30), Equation (31), we have that for $t_{i+1}^{\text{orig}} \geq t_{i+1}^{\text{comp}}$ for all $i \in [r]$.

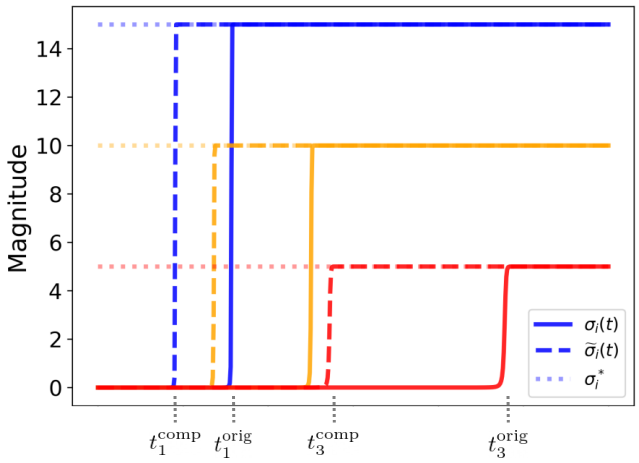


Figure 22: **Visual illustration of the proof technique for Theorem 2.** The proof involves showing that $t_i^{\text{comp}} \leq t_i^{\text{orig}}$, which amounts to showing that the singular values of the compressed network fits the target value faster than the original network.

Inductive Step (Residual Components). For the remaining residual singular values, notice that $\forall t > t_r^{\text{orig}}$, the dynamics of the residual singular values $\forall j \in [r+1, \hat{r}]$ are governed by

$$\begin{aligned}
\dot{\sigma}_j(t) &= -L \cdot (\sigma_j^2(t))^{1-1/L} \cdot \langle \nabla_{\boldsymbol{W}_{L:1}} \ell(\boldsymbol{\Theta}(t)), \boldsymbol{u}_j(t)\boldsymbol{v}_j^\top(t) \rangle, \\
&= -L \cdot (\sigma_j^2(t))^{1-1/L} \cdot (\boldsymbol{u}_j^\top(t)\boldsymbol{W}_{L:1}(t)\boldsymbol{v}_j(t) - \boldsymbol{u}_j^\top(t)\boldsymbol{M}^*\boldsymbol{v}_j(t)) \\
&= -L \cdot (\sigma_j^2(t))^{1-1/L} \cdot (\sigma_j(t) - \boldsymbol{u}_j^\top(t)\boldsymbol{M}^*\boldsymbol{v}_j(t)) \\
&= -L \cdot (\sigma_j^2(t))^{1-1/L} \cdot \sigma_j(t),
\end{aligned}$$

which is identical to that of the compressed network, and so $t_{i+1}^{\text{orig}} \geq t_{i+1}^{\text{comp}}$ for all $i \in [r+1, \hat{r}]$.

Thus,

$$(\sigma_i(t) - \sigma_i^*)^2 \geq (\widetilde{\sigma}_i(t) - \sigma_j^*)^2, \qquad \forall i \in [\hat{r}], \tag{32}$$

and so we have

$$\|\boldsymbol{W}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2 \geq \sum_{i=1}^{\hat{r}} (\sigma_i(t) - \sigma_i^*)^2 \geq \sum_{i=1}^{\hat{r}} (\widetilde{\sigma}_i(t) - \sigma_i^*)^2 = \|\widetilde{\boldsymbol{W}}_{L:1}(t) - \boldsymbol{M}^*\|_{\mathsf{F}}^2,$$

which completes the proof. □

Remarks. In Figure 22, we provide a visual illustration of the proof technique. Here, we can clearly see that $t_i^{\text{comp}} \leq t_i^{\text{orig}}$, which leads to the lower recovery error at the same time instance t. To rigorously show that $t_i^{\text{comp}} \leq t_i^{\text{orig}}$, we relied on using the gradient flow result stated in Proposition 1 and the incremental learning assumption presented in Assumption 1. We justified the use of Assumption 1 in the beginning of Appendix A.1, where we showed that the principal components of the DLN are fitted incrementally, and that generally, $c_{\text{vec}} = c_{\text{val}} = 0$ (see Figure 13).

B.3 Auxiliary Results

Lemma 1. *Consider two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{d_1 \times d_2}$ and their respective compact singular value decompositions (SVD) $\boldsymbol{A} = \boldsymbol{U}_A \boldsymbol{\Sigma}_A \boldsymbol{V}_A^\top$ and $\boldsymbol{B} = \boldsymbol{U}_B \boldsymbol{\Sigma}_B \boldsymbol{V}_B^\top$. We have that*

$$\|\boldsymbol{A} - \boldsymbol{B}\|_{\mathsf{F}}^2 \geq \|\boldsymbol{\Sigma}_A - \boldsymbol{\Sigma}_B\|_{\mathsf{F}}^2.$$

Proof. By using the definition of the Frobenius norm, we have

$$\begin{aligned}
\|\boldsymbol{A}-\boldsymbol{B}\|_{\mathsf{F}}^2 &= \|\boldsymbol{U}_A\boldsymbol{\Sigma}_A\boldsymbol{V}_A^\top - \boldsymbol{U}_B\boldsymbol{\Sigma}_B\boldsymbol{V}_B^\top\|_{\mathsf{F}}^2 \\
&= \operatorname{tr}(\boldsymbol{\Sigma}_A^2) - 2\operatorname{tr}(\boldsymbol{A}^\top\boldsymbol{B}) + \operatorname{tr}(\boldsymbol{\Sigma}_B^2) \\
&\geq \operatorname{tr}(\boldsymbol{\Sigma}_A^2) - 2\operatorname{tr}(\boldsymbol{\Sigma}_A\boldsymbol{\Sigma}_B) + \operatorname{tr}(\boldsymbol{\Sigma}_B^2) && \text{(Von-Neumann Inequality)} \\
&= \|\boldsymbol{\Sigma}_A - \boldsymbol{\Sigma}_B\|_{\mathsf{F}}^2.
\end{aligned}$$

This proves the desired result. □