Meta-Learning Provably Learns Adaptable Foundation Models

Jacob L. Block* Sundararajan Srinivasan* Liam Collins*

Aryan Mokhtari* Sanjay Shakkottai*

Abstract

The power of *foundation models* (FMs) lies in their capacity to learn highly expressive representations that can be adapted to a broad spectrum of tasks. However, these pretrained models require additional training stages to become effective for downstream applications. In the multi-task setting, prior works have shown empirically that specific meta-learning approaches for preparing a model for future adaption through parameter-efficient fine-tuning (PEFT) can outperform standard retraining methods, but the mechanism of the benefits of meta-learning has been largely unexplored. We introduce a framework for generic PEFT-based meta-learning to learn a model that can be easily adapted to unseen tasks. For linear models using LoRA, we show that standard retraining is provably suboptimal for finding an adaptable set of parameters and provide strict performance guarantees for our proposed method. We then verify these theoretical insights through extensive experiments on synthetic data as well as real NLP tasks using large language models. We observe significant performance benefits using a simple implementation of our proposed meta-learning scheme during retraining relative to the conventional approach.

*Chandra Family Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA. {jblock@utexas.edu, sundararajans@utexas.edu, liamc@utexas.edu, mokhtari@austin.utexas.edu, sanjay.shakkottai@utexas.edu}

1 Introduction

Foundation Models (FMs) learn rich representations that are useful for a variety of downstream tasks. The first stage of FM training is referred to as *pretraining*, where a combination of massive public, propriety, and synthetic sources of data is used to learn a general-purpose model from scratch [DCLT19; BMRSKDN+20; AAAAAB+24; Rad+21]. However, due to the enormous cost of training state-of-the-art models on such large datasets, pretraining is largely infeasible for most. Thus, the most popular and viable way to utilize FMs for individual tasks is to take a pretrained model and adapt it for a specific task.

We consider the problem of adapting a pretrained FM to a collection of related tasks of interest. We refer to this process as *retraining*, where given a number of tasks with many samples, our goal is to recover a model that learns the task structure and can be quickly adapted to future tasks with limited samples. In other works this stage has been referred to pre-finetuning [AGSCZG21] or supervised fine-tuning [DYLLXLWYZZ24]. After retraining, we adapt the model to a new task with few samples in what we denote the *fine-tuning* stage. In this last stage, we typically employ parameter efficient fine-tuning (PEFT) methods – training heuristics which sacrifice learning expressiveness for improved computational efficiency [HSWALWWC21; LL21]. Ultimately, the purpose of retraining is to prepare the model for efficient future adaptation, and the effectiveness of a retraining method is measured by the model's performance on the fine-tuning task.

Standard approaches to retraining involve fitting the model to the aggregation of the different retraining tasks. While this seems reasonable and has been empirically successful [KMKSTCH20; RSRLNMZLL20], it does not leverage knowledge of the downstream fine-tuning procedure to cater the retrained model to perform well after such adaptation. Rather, it retrains the model to minimize the average loss across the retraining tasks regardless of the PEFT method to be employed later. Thus, there is no assurance the recovered solution is indeed adaptable to future unseen tasks relative to other possible retraining solutions.

Meta-learning is natural framework to address this issue, as it explicitly aims to learn adaptable models, typically in low-resource, few-shot settings using gradient-based adaptations [FAL17; LC18]. The success of meta-learning algorithms is largely attributed to their ability to learn useful representations, as even model-agnostic gradient-based meta-learning algorithms like MAML [FAL17] and Reptile [NAS18] have been shown to implicitly learn representations in linear settings [CMOS22; SZKA20]. Recent works have shown empirical benefits to retraining using specific PEFT-based meta-learning methods [HSP22; HJ22; BAWLM22; GMM22; HMMF23], but theoretical guarantees showcasing these gains have not been established.

Contributions. In this work, we study a general framework for PEFT-based meta-learning during retraining. Overall, *our aim is to show that meta-learning provably outperforms standard retraining methods, not that our framework is optimal amongst other meta-learning variants.* We focus on the Low-Rank Adaptation (LoRA) [HSWALWWC21] PEFT method and consider multiple linear regression tasks where each ground truth regressor is a rank-k perturbation of a common matrix $\boldsymbol{A}^* \in \mathbb{R}^{d\times d}$. Given a set of T tasks, our goal is to recover $\boldsymbol{A}^*$ so that we can easily fine-tune to a new, unseen task by learning a low-rank perturbation using LoRA. With this in mind, we show the following:

- We prove that standard retraining (which does not leverage any meta-learning scheme) even in the infinite sample case fails to recover parameters which are low-rank adaptable, as the recovered model is not low-rank away from $\boldsymbol{A}^*$ and consequently the test model (Proposition

3). Applying low-rank adaptations then completely fails, as for large number of tasks, the population risk on the test task scales as $\Omega\left((d-r)k^2\right)$, where d is the ambient dimension, $k \ll d$ is the ground truth adapter rank, and $r \geq k$ is the rank used for fine-tuning (Proposition 4). Further, fine-tuning with a very large rank to account for the discrepancy to the test task defeats the purpose of PEFT and results in squared prediction error which grows as $\mathcal{O}\left(\frac{kTd}{n}\right)$ with high probability, in the regime where $k(T+1) < d$ (Remarks 1, 2). Thus, standard retraining performs *worse* when given access to more tasks.

- For the meta-learning framework we study, we guarantee that any minimizer of the meta-learning loss in the infinite sample case is indeed low-rank adaptable to unseen tasks (Theorem 5). Further, we show that if there are at least three retraining tasks, the ground truth parameters are the unique global minima up to orthogonal symmetry (Theorem 8). As a result, LoRA fine-tuning is effective in adapting to the test task and with high probability achieves squared prediction error which grows as $\mathcal{O}\left(\frac{kd}{n}\right)$ (Corollary 10). In contrast to standard retraining, we achieve the optimal rate which does not include any dependence on T.
- We prove in the infinite sample case, every second-order stationary point of our meta-learning loss when applied to two retraining tasks is in fact globally optimal (Theorem 11). In this case there are no spurious local minima of our meta-learning loss and optimality is completely determined by second-order information. Thus, local optimization methods like perturbed gradient descent can efficiently find global minima.

To the author's knowledge, these are the first results showing that PEFT-based meta-learning provably outperforms standard retraining methods in any setting. The proofs of our results are presented in Appendix B.

To verify our theoretical insights, we compare the performance of the standard retraining and LoRA-based meta-learning objectives for synthetic multi-output linear regression and shallow neural network regression tasks. We show clear improvements using LoRA-based meta-learning for all data generation parameter settings. Then, we apply a simple implementation of our general LoRA-based meta-learning framework to the RoBERTa [LOGDJCLLZS19] large language model (LLM) on the ConvAI2 [Din+19] NLP dataset. Again, we show improvements using the LoRA-based meta-learning relative to standard retraining.

1.1 Related Work

Meta-learning is a framework for learning models that can be rapidly adapted to unseen tasks by leveraging access to prior tasks during training. For example, Model-Agnostic Meta-Learning (MAML) [FAL17] is a popular method that aims to find a model that can be adapted to a new task after a small number of steps of gradient descent on the new task's loss function. Other works have proposed methods specific to low-dimensional linear models and have shown strong theoretical results and connections between meta-learning and representation learning [CMOS22; TJNO21; SZKA20].

In the case of FMs, specific meta-learning approaches for incorporating PEFT-based adaption have been proposed. Hong and Jang [HJ22], Bansal, Alzubi, Wang, Lee, and McCallum [BAWLM22], and Gheini, Ma, and May [GMM22] applied meta-learning with architecture adaptations that inject task-specific trainable layers within the FM architecture. Hou, Salazar, and Polovets [HSP22] combined architecture adaptations with parameter perturbations similar to LoRA. They considered a complicated meta-learning loss that updates the adapters and FM weights over different splits of the data and showed empirical gains over standard retraining and other gradient-based MAML-style algorithms. Aghajanyan, Gupta, Shrivastava, Chen, Zettlemoyer, and Gupta [AGSCZG21] similarly proposed a multi-task objective that trains an FM on different tasks simultaneously to encourage learning a universally applicable representation. It forces the FM to learn a shared data representation but allows for task-specific prediction heads. Overall, each of these works proposed some kind of meta-learning or multi-task objective and showed empirical gains over standard retraining strategies. However, their experimental results motivate a deeper theoretical exploration of when standard retraining is insufficient relative to meta-learning approaches, how many tasks are needed to learn a rich representation, and how to best adapt to tasks unseen in the training stage.

Lastly, although we focus on LoRA, different PEFT methods have been proposed, including variants of LoRA [LWYMWCC24; DPHZ23; ZCBHCCZ23] and architecture adaptations [HGJMDGAG19] among others. Further, recent works have analyzed the theoretical aspects of LoRA in the fine-tuning stage [JLR24; ZL23], but they explored orthogonal directions to the analysis of LoRA-based meta-learning during retraining. Extended discussion of these prior works is in Appendix A.

Notation. We use bold capital letters for matrices and bold lowercase letters for vectors. $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ refers to the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. $\boldsymbol{I}_d$ refers to the $d \times d$ identity matrix. $\|\cdot\|_F$ refers to the Frobenius norm. S_d refers to the set of $d \times d$ symmetric matrices, and S_d^+ is the set of $d \times d$ positive semi-definite matrices. O_d refers to the set of $d \times d$ orthogonal matrices. $[n]$ refers to the set $\{1, \ldots, n\}$. For a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, $\text{im}(\boldsymbol{X})$ and $\ker(\boldsymbol{X})$ refer to the image and kernel of $\boldsymbol{X}$, while $\text{vec}(\boldsymbol{X}) \in \mathbb{R}^{mn}$ denotes the column-wise vectorization of $\boldsymbol{X}$. For subspaces $\boldsymbol{M}, \boldsymbol{N}$, $\dim(\boldsymbol{M})$ refers to the dimension of $\boldsymbol{M}$ and $\boldsymbol{M} + \boldsymbol{N} = \{\boldsymbol{x} + \boldsymbol{y} | \boldsymbol{x} \in \boldsymbol{M}, \boldsymbol{y} \in \boldsymbol{N}\}$. If $\boldsymbol{M} \cap \boldsymbol{N} = \{\boldsymbol{0}\}$, we write the direct sum $\boldsymbol{M} \oplus \boldsymbol{N}$.

2 Retraining and Fine-Tuning Schemes

We briefly recap the optimization process for standard retraining of an FM across multiple tasks followed by fine-tuning on a downstream task. We then introduce a general framework for PEFT-based meta-learning which adjusts the retraining phase to incorporate insights from fine-tuning.

2.1 Standard Retraining Then Fine-Tuning

Consider a collection of T tasks of interest $\mathcal{T} = \{\mathcal{T}_t\}_{t=1}^T$ where each task $\mathcal{T}_t$ is drawn from task distribution $\mathcal{D}$ and consists of n_t labeled examples $\mathcal{T}_t = \{(\boldsymbol{x}_{t,j}, \boldsymbol{y}_{t,j})\}_{j=1}^{n_t}$, where $(\boldsymbol{x}_{t,j}, \boldsymbol{y}_{t,j})$ are i.i.d. from the t_{th} task's data distribution $\mathcal{D}_{\mathcal{T}_t}$. Without loss of generality we assume consistent dimensions across tasks, so $\boldsymbol{x}_{t,j} \in \mathbb{R}^{d_x}$, $\boldsymbol{y}_{t,j} \in \mathbb{R}^{d_y}$ for all $t \in [T], j \in [n_t]$. Let $\boldsymbol{X}_t \in \mathbb{R}^{d_x \times n_t}$ and $\boldsymbol{Y}_t \in \mathbb{R}^{d_y \times n_t}$ denote the concatenation of $\boldsymbol{x}_{t,j}$ and $\boldsymbol{y}_{t,j}$ for $j = 1, \ldots, n_t$ respectively. Consider a model $\Phi(\cdot\,; \boldsymbol{W}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ parameterized by weights $\boldsymbol{W}$ that maps feature vectors to predicted labels. We abuse notation and write $\Phi(\boldsymbol{X}_t\,; \boldsymbol{W})$ to denote the concatenation of $\Phi(\boldsymbol{x}_{t,j}\,; \boldsymbol{W})$ for $j = 1, \ldots, n_t$. Typically $\boldsymbol{W} = (\boldsymbol{W}_1, \ldots, \boldsymbol{W}_m)$ is a list of matrices where $\boldsymbol{W}_i \in \mathbb{R}^{d \times d}$ parameterize the layers of a neural network. We assume each $\boldsymbol{W}_i$ is square for convenience.

Retraining Phase. Given a loss function $\mathcal{L}$, standard retraining attempts to minimize the aggregated loss over a collection of training tasks [LOGDJCLLZS19; BMRSKDN+20]. This amounts to solving

$$\hat{\boldsymbol{W}}_{\mathrm{SR}} = \min_{\boldsymbol{W}} \sum_{t=1}^{T} \mathcal{L}\left(\Phi(\boldsymbol{X}_t; \boldsymbol{W}), \boldsymbol{Y}_t\right), \tag{1}$$

where SR stands for Standard Retraining. The above optimization problem seeks a set of universal parameters that define a unique mapping function capable of translating inputs to outputs across all tasks involved in the retraining phase. We denote the corresponding model as $\Phi(\,\cdot\,; \hat{\boldsymbol{W}}_{\mathrm{SR}})$.

Fine-Tuning Phase. In the subsequent fine-tuning, PEFT is often used to refine either the retrained weights $\hat{\boldsymbol{W}}_{\mathrm{SR}}$, the model's feature map Φ, or both to fit a downstream task with fewer labeled samples. More precisely, consider a downstream task $\mathcal{T}_{T+1}$ drawn from the same distribution $\mathcal{D}$ where $\mathcal{T}_{T+1} = \{(\boldsymbol{x}_{T+1,j}, \boldsymbol{y}_{T+1,j})\}_{j=1}^{n_{T+1}}$. To fit the model to task $\mathcal{T}_{T+1}$ we fix $\boldsymbol{W} = \hat{\boldsymbol{W}}_{\mathrm{SR}}$ in the original parameterization and fine-tune the mapping $\Phi(\,\cdot\,; \hat{\boldsymbol{W}}_{\mathrm{SR}})$ using additional parameters $\boldsymbol{\theta}$. For example, $\boldsymbol{\theta}$ could parameterize trainable perturbations of $\hat{\boldsymbol{W}}_{\mathrm{SR}}$ or new trainable layers inserted into the architecture of the retrained model [HSWALWWC21; LWYMWCC24; AGSCZG21]. We denote the fine-tuned model's mapping as $\Phi_{\mathrm{FT}}(\,\cdot\,; \hat{\boldsymbol{W}}_{\mathrm{SR}}, \boldsymbol{\theta}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ and again abuse notation by writing $\Phi_{\mathrm{FT}}(\boldsymbol{X}_{T+1}; \hat{\boldsymbol{W}}_{\mathrm{SR}}, \boldsymbol{\theta})$ to denote the concatenation of $\Phi_{\mathrm{FT}}(\boldsymbol{x}_{T+1,j}; \hat{\boldsymbol{W}}_{\mathrm{SR}}, \boldsymbol{\theta})$ for $j = 1, \ldots, n_{T+1}$. During the *fine-tuning stage*, the goal is to find the optimal additional parameters, $\boldsymbol{\theta}$, that minimize the loss for the downstream task $\mathcal{T}_{T+1}$, solving

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\Phi_{\mathrm{FT}}(\boldsymbol{X}_{T+1}\,; \hat{\boldsymbol{W}}_{\mathrm{SR}}, \boldsymbol{\theta}), \boldsymbol{Y}_{T+1}). \tag{2}$$

In particular, when LoRA is used for fine-tuning, the model is adapted to task $\mathcal{T}_{T+1}$ by fixing the architecture and the retrained weights $\hat{\boldsymbol{W}}_{\mathrm{SR}}$ and only training low-rank perturbations for each of the matrices $\hat{\boldsymbol{W}}_{\mathrm{SR},1}, \ldots, \hat{\boldsymbol{W}}_{\mathrm{SR},m}$. For rank-$r$ adaptations, we parameterize $\boldsymbol{\theta} = ((\boldsymbol{Q}_1, \boldsymbol{V}_1), \ldots, (\boldsymbol{Q}_m, \boldsymbol{V}_m))$, where $\boldsymbol{Q}_i, \boldsymbol{V}_i \in \mathbb{R}^{d \times r}$ are the factors of the low-rank adaptation of the ith matrix in $\hat{\boldsymbol{W}}_{\mathrm{SR}}$. The fine-tuned model is just the original model where the ith weight matrix $\boldsymbol{W}_i$ is now perturbed to be $\boldsymbol{W}_i + \boldsymbol{Q}_i \boldsymbol{V}_i^{\top}$. For $\boldsymbol{Q}, \boldsymbol{V} \in \left(\mathbb{R}^{d \times r}\right)^m$, define the LoRA loss

$$\begin{aligned} &\mathcal{L}_{\mathrm{LoRA}}\left(\boldsymbol{Q}, \boldsymbol{V}\,; \boldsymbol{W}\right) = \\ &\qquad \mathcal{L}\left(\Phi\left(\boldsymbol{X}_{T+1}\,; \left(\boldsymbol{W}_i + \boldsymbol{Q}_i \boldsymbol{V}_i^{\top}\right)_{i=1}^{m}\right), \boldsymbol{Y}_{T+1}\right). \end{aligned} \tag{3}$$

The LoRA fine-tuning optimization problem is then

$$\min_{\boldsymbol{Q}, \boldsymbol{V}} \mathcal{L}_{\mathrm{LoRA}}(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{W}}_{\mathrm{SR}}). \tag{4}$$

This pipeline seems reasonable as we first fit the model to the aggregation of the retraining tasks which we hope will promote learning the general structure of the tasks drawn from $\mathcal{D}$. However, nothing about standard retraining promotes learning an adaptable solution relative to other candidate solutions that fit the retraining tasks. Next, we introduce a general meta-learning framework which explicitly incorporates the adaption mechanism during retraining.

2.2 PEFT-Based Meta-Learning

Since the ultimate goal of retraining is to perform well on an unseen downstream task, we study a general PEFT-based meta-learning (PEFT-ML) objective that explicitly fits weights and adapter parameters to the training tasks. Rather than training a single model on the aggregation of the retraining tasks, we instead incorporate the adapters during the retraining process and learn adapted models for each task. Let $\boldsymbol{\theta}^{(t)}$ be the set of adapter parameters for the t_{th} training task $\mathcal{T}_t$. The PEFT-ML objective searches for a single set of base weights $\hat{\boldsymbol{W}}_{\text{Meta}}$ such that for all $t \in [T]$, the t_{th} adapted model $\Phi_{\text{FT}}(\,\cdot\,; \hat{\boldsymbol{W}}_{\text{Meta}}, \boldsymbol{\theta}^{(t)})$ minimizes the loss over the training task $\mathcal{T}_t$. More precisely, we define the proposed PEFT-ML objective as

$$\hat{\boldsymbol{W}}_{\text{Meta}} = \min_{\boldsymbol{W}} \sum_{t=1}^{T} \mathcal{L}_t(\boldsymbol{W}), \tag{5}$$

where $\mathcal{L}_t(\boldsymbol{W}) = \min_{\boldsymbol{\theta}^{(t)}} \mathcal{L}\left(\Phi_{\text{FT}}\left(\boldsymbol{X}_t\,; \boldsymbol{W}, \boldsymbol{\theta}^{(t)}\right), \boldsymbol{Y}_t\right)$. When we use LoRA as the adaptation method, we define $\boldsymbol{Q}^{(t)}, \boldsymbol{V}^{(t)} \in \left(\mathbb{R}^{d\times r}\right)^m$ as the list of factors of the low-rank adapter $\boldsymbol{Q}_i^{(t)}(\boldsymbol{V}_i^{(t)})^\top$ applied to the ith weight matrix for the t_{th} task. Then the inner objective $\mathcal{L}_t(\boldsymbol{W})$ reduces to

$$\min_{\boldsymbol{Q}^{(t)}, \boldsymbol{V}^{(t)}} \mathcal{L}_{\text{LoRA}}\left(\boldsymbol{Q}^{(t)}, \boldsymbol{V}^{(t)}\,; \boldsymbol{W}\right). \tag{6}$$

In this case, we refer to the objective function as LoRA-ML. This proposed optimization problem is designed to replace the standard retraining objective in (1). After solving (5) we recover base parameters $\hat{\boldsymbol{W}}_{\text{Meta}}$ that are explicitly designed to be adaptable to downstream tasks drawn from the same distribution as those seen in retraining. To perform finetuning, we then run the exact same minimization in (2) but using retrained weights $\hat{\boldsymbol{W}}_{\text{Meta}}$ instead of $\hat{\boldsymbol{W}}_{\text{SR}}$.

3 Main Results

To establish our theoretical results, we consider $T \geq 1$ multi-output linear regression retraining tasks $\{\mathcal{T}_t\}_{t=1}^{T}$ and one downstream test task $\mathcal{T}_{T+1}$, where the ground-truth regressor for each task is a low-rank perturbation of a common shared matrix. Precisely, for each $t \in [T+1]$ we assume task $\mathcal{T}_t$ is independently drawn from distribution $\mathcal{D}_{\boldsymbol{A}^*}$ which associated with some fixed matrix $\boldsymbol{A}^* \in \mathbb{R}^{d\times d}$ and intrinsic adaptation rank $k \ll d$. Then, task $\mathcal{T}_t$ is equipped with n_t samples $(\boldsymbol{x}_{t,j}, \boldsymbol{y}_{t,j}) \in \mathbb{R}^d \times \mathbb{R}^d$ for $j = 1, \ldots, n_t$ which are related by the noisy linear transformation parameterized by $\boldsymbol{A}^* + \boldsymbol{R}_t^*$, where $\text{rank}(\boldsymbol{R}_t^*) = k$. Formally,

$$\begin{aligned} &\mathcal{T}_t \sim \mathcal{D}_{\boldsymbol{A}^*} : \\ &\quad \mathcal{T}_t = \{\boldsymbol{A}^* + \boldsymbol{R}_t^*, \{\boldsymbol{x}_{t,j}, \boldsymbol{y}_{t,j}\}_{j=1}^{n_t}\} \ \text{ s.t. } \ \text{rank}(\boldsymbol{R}_t^*) = k \end{aligned}$$

where $\boldsymbol{R}_t^* \in \mathbb{R}^{d\times d}$ and $\boldsymbol{x}_{t,j}, \boldsymbol{y}_{t,j} \in \mathbb{R}^d$ are generated as follows. Consider rank-k factor $\boldsymbol{U}_t^* \in \mathbb{R}^{d\times k}$ where $\text{vec}(\boldsymbol{U}_t^*) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{dk})$, input vectors $\boldsymbol{x}_{t,j}$ which satisfy $\mathbb{E}\left[\boldsymbol{x}_{t,j}\right] = \boldsymbol{0}$ and $\mathbb{E}\left[\boldsymbol{x}_{t,j}\boldsymbol{x}_{t,j}^\top\right] = \sigma_x^2 \boldsymbol{I}_d$, and independently generated noise terms $\boldsymbol{\epsilon}_{t,j} \sim \mathcal{N}(\boldsymbol{0}, \sigma_\epsilon^2 \boldsymbol{I}_d)$. Then,

$$\boldsymbol{R}_t^* = \boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top} \qquad \boldsymbol{y}_{t,j} = (\boldsymbol{A}^* + \boldsymbol{R}_t^*)\boldsymbol{x}_{t,j} + \boldsymbol{\epsilon}_{t,j}$$

The above generative model defines the input-output relationships for each task as similar linear models, differing from each other only by a low-rank perturbation.

Remark 1. *For convenience, we require a mild sense of task diversity and assume that the aggregated columns from all $\boldsymbol{U}_t^*$, $t \in [T+1]$, form a linearly independent set, i.e.* $\dim\left(\text{im}(\boldsymbol{U}_1^*) \oplus \cdots \oplus \text{im}(\boldsymbol{U}_{T+1}^*)\right) = k(T+1)$. *Thus, we implicitly require that the ambient dimension $d > k(T+1)$. Then for $d > k(T+1)$, the generation process of each $\boldsymbol{U}_t^*$ ensures that this assumption then holds with probability 1.*

The learner uses the linear model $\Phi(\boldsymbol{x};\boldsymbol{A}) = \boldsymbol{A}\boldsymbol{x}$ for $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ and retrains on tasks $\mathcal{T}_1,\ldots,\mathcal{T}_T$ with the ultimate goal of efficient adaption to $\mathcal{T}_{T+1}$ using LoRA. Ideally, we hope to recover parameter value $\hat{\boldsymbol{A}} = \boldsymbol{A}^*$ in the retraining phase so that the fine-tuned model $\Phi_{\mathrm{FT}}(\boldsymbol{x};\hat{\boldsymbol{A}},\boldsymbol{Q},\boldsymbol{V}) = (\hat{\boldsymbol{A}}+\boldsymbol{Q}\boldsymbol{V}^\top)\boldsymbol{x}$ can fit the data distribution of any downstream task also drawn from $\mathcal{D}$ with a proper rank-k adapter $\boldsymbol{Q}\boldsymbol{V}^\top$. We define the finite-sample loss function for task t as

$$\mathcal{L}_t^{n_t}(\boldsymbol{A}) = \frac{1}{2n_t}\sum_{j=1}^{n_t}\|\boldsymbol{y}_{t,j} - \boldsymbol{A}\boldsymbol{x}_{t,j}\|_2^2\,, \tag{7}$$

and we define $\mathcal{L}_t^*(\boldsymbol{A})$ as the shifted and scaled infinite sample loss, i.e.,

$$\begin{aligned}\mathcal{L}_t^*(\boldsymbol{A}) &= \frac{1}{\sigma_x^2}\left(\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\left[\mathcal{L}_t^{n_t}(\boldsymbol{A})\right] - \frac{\sigma_\epsilon^2}{2}\right)\\ &= \frac{1}{2}\left\|\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{A}\right\|_F^2.\end{aligned} \tag{8}$$

We consider the setting where for the retraining tasks $t \le T$ we have large n_t, but for the test task n_{T+1} is small. This reflects practical scenarios where we have access to large retraining datasets compared to the low-resource fine-tuning task. Thus, we assume access to the infinite sample loss functions $\mathcal{L}_t^*$ for the retraining tasks $t \le T$. Then, for ease of notation, define $n = n_{T+1}$ as the number of test task samples. We ultimately aim to use LoRA to fit the finite-sample test task loss $\mathcal{L}_{T+1}^n$ efficiently in n.

Given a learned representation $\hat{\boldsymbol{A}} \in \mathbb{R}^{d\times d}$ from retraining, the fine-tuning problem using LoRA with rank r reduces to

$$\min_{\boldsymbol{Q},\boldsymbol{V}\in\mathbb{R}^{d\times r}} \mathcal{L}_{T+1}^n(\hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top) \tag{9}$$

Since $\boldsymbol{Q}\boldsymbol{V}^\top$ can parameterize any rank-r matrix, (9) is a specific parametrization for what is commonly known as reduced rank regression [Ize75]. It is clear that to even realize the optimal regressor $\boldsymbol{A}^* - \hat{\boldsymbol{A}} + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top}$ for $\boldsymbol{Q}\boldsymbol{V}^\top$, we need $r \ge \operatorname{rank}(\boldsymbol{A}^* - \hat{\boldsymbol{A}} + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top})$. Further, results in reduced rank regression and matrix sensing in general reveal the importance of $\operatorname{rank}(\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \hat{\boldsymbol{A}})$ in terms of the hardness of minimizing $\mathcal{L}_{T+1}^n(\hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top)$.

Lemma 1 (Bunea, She, and Wegkamp, 2011)**.** *Consider $\hat{\boldsymbol{A}} \in \mathbb{R}^{d\times d}$ and let $r = \operatorname{rank}(\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \hat{\boldsymbol{A}})$. Let $\boldsymbol{Q}^*, \boldsymbol{V}^* \in \mathbb{R}^{d\times r}$ minimize $\mathcal{L}_{T+1}^n(\hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top)$ over all rank-r factors $\boldsymbol{Q},\boldsymbol{V} \in \mathbb{R}^{d\times r}$ and let $\boldsymbol{X}_{T+1} = [\boldsymbol{x}_{T+1,1},\ldots,\boldsymbol{x}_{T+1,n}]$ be the matrix whose columns are the test task input samples. Denote the matrix of prediction errors $\boldsymbol{E} = (\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top})\boldsymbol{X}_{T+1} - (\hat{\boldsymbol{A}} + \boldsymbol{Q}^*\boldsymbol{V}^{*\top})\boldsymbol{X}_{T+1}$. Then $\forall\gamma > 0$,*

$$\mathbb{P}\left(\frac{1}{n}\|\boldsymbol{E}\|_F^2 \le \frac{24(1+\gamma)^2\sigma_\epsilon^2 rd}{n}\,\middle|\, \boldsymbol{X}_{T+1}\right) \ge 1 - e^{-\gamma^2 d} \tag{10}$$

In general, the squared prediction error scales linearly with rd. This matches the information-theoretic lower bound to learn rd number of parameters and is further minimax optimal over all rank-r matrices when the eigenvalues of $\boldsymbol{X}_{T+1}^\top\boldsymbol{X}_{T+1}$ are uniformly bounded so that a restricted isometry condition is satisfied [RT11]. Thus, a larger rank of $\boldsymbol{A}^* - \hat{\boldsymbol{A}} + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top}$ inflates the fine-tuning prediction error, as we hope to recover $\hat{\boldsymbol{A}} = \boldsymbol{A}^*$ so that $\operatorname{rank}(\boldsymbol{A}^* - \hat{\boldsymbol{A}} + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top}) = k$. With this in mind, we compare the standard retraining (1) and LoRA-ML (6) objectives.

3.1 Standard Retraining Then Fine-Tuning

Consider standard retraining then fine-tuning as a candidate for ultimately minimizing (9). The learner first finds a single matrix $\hat{\boldsymbol{A}}_{\text{SR}}$ that minimizes the sum of losses $\sum_{t=1}^{T} \mathcal{L}_t^*$:

$$\hat{\boldsymbol{A}}_{\text{SR}} = \arg\min_{\boldsymbol{A}} \frac{1}{2} \sum_{t=1}^{T} \left\| \boldsymbol{A}^* + \boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top} - \boldsymbol{A} \right\|_F^2 . \tag{11}$$

Then when given test task $\mathcal{T}_{T+1}$, the learner solves $\min_{\boldsymbol{Q},\boldsymbol{V} \in \mathbb{R}^{d\times r}} \mathcal{L}_{T+1}^n(\hat{\boldsymbol{A}}_{\text{SR}} + \boldsymbol{Q}\boldsymbol{V}^\top)$. However, this strategy suffers substantial loss in both the retraining and fine-tuning stages. Notice the loss in (11) is convex and quadratic in $\boldsymbol{A}$, so simple first-order optimality shows that

$$\hat{\boldsymbol{A}}_{\text{SR}} = \boldsymbol{A}^* + \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top} . \tag{12}$$

Thus, $\hat{\boldsymbol{A}}_{\text{SR}}$ recovers $\boldsymbol{A}^*$ added to the average of the retraining ground truth adaptations $\boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top}$. However, $\hat{\boldsymbol{A}}_{\text{SR}}$ performs poorly on all of the retraining tasks, as standard retraining is unable to disentangle the common structure $\boldsymbol{A}^*$ from the task-specific adapters $\boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top}$.

Theorem 2. *Let $\boldsymbol{U}^* = (\boldsymbol{U}_1^*, \dots, \boldsymbol{U}_T^*)$. Then,*

$$\mathbb{E}_{\boldsymbol{U}^*} \left[\sum_{t=1}^{T} \mathcal{L}_t(\hat{\boldsymbol{A}}_{SR}) \right] = (T + 1 - \frac{2}{T}) kd(d+1) = \Omega\left(Tkd^2\right)$$

Thus, $\hat{\boldsymbol{A}}_{\text{SR}}$ suffers significant loss on the retraining tasks when averaged over the generation process of ground truth parameters $\boldsymbol{U}^*$. Further, $\hat{\boldsymbol{A}}_{\text{SR}}$ is not low-rank adaptable to the test task. Crucially, the intrinsic dimension of the test task is $\text{rank}(\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^* \boldsymbol{U}_{T+1}^{*\top} - \hat{\boldsymbol{A}}_{\text{SR}}) = k(T+1)$, so an adaptation rank of $k(T+1)$ is required to even achieve the ground truth test task parameters.

Proposition 3. *If test task fine-tuning rank $r < k(T+1)$, then $\mathcal{L}_{T+1}^*(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{A}}_{SR}) > 0$ for all $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d\times r}$.*

Even though the test task parameters are only rank-k away from $\boldsymbol{A}^*$, standard retraining fails to exploit this structure and inflates the necessary rank to $k(T+1)$. Thus, standard retraining actually recovers worse representations as the number of tasks T grows. In this case, failing to fine-tune with large enough rank causes significant loss.

Proposition 4. *For a large number of retraining tasks T and test task fine-tuning rank $r < k(T+1)$, $\mathcal{L}_{T+1}^*(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{A}}_{SR}) = \Omega\left((d-r)k^2\right)$ for all $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d\times r}$.*

When the test task fine-tuning rank r is under-specified relative to the required rank $k(T+1)$, the squared error between the recovered parameter for the test task and the ground truth grows like $(d-r)k^2$ for large T.

The above propositions demonstrate the cost of under-specifying the fine-tuning rank relative to the large intrinsic dimension of the test task which results from standard retraining. On the other hand, applying the necessarily large fine-tuning rank $r = k(T+1)$ both defeats the purpose of low-rank adaptation as a PEFT method and still incurs large prediction error when fine-tuning with limited samples.

Remark 2. *Consider the finite-sample loss (9) using $\hat{\boldsymbol{A}}_{SR}$ adapted with LoRA using rank $r = k(T+1)$. This can achieve optimal population risk but suffers in the finite-sample setting. Using Lemma 1, we can only hope to achieve squared prediction error of order $\mathcal{O}(\frac{kTd}{n})$ when fine-tuning, much larger than the optimal rate $\mathcal{O}\left(\frac{kd}{n}\right)$ if we had in fact recovered the ground truth $\boldsymbol{A}^*$ during retraining.*

Thus, **standard retraining recovers parameters that cannot be efficiently low-rank adapted to any relevant task**. In contrast, our analysis of using LoRA-ML during retraining shows much improved performance.

3.2 LoRA-Meta-Learning

Consider applying (6) to this problem instance. We introduce low-rank adapters during the retraining phase to model the different training tasks and search for a value of $\boldsymbol{A}$ such that for all $\mathcal{T}_t$, the loss $\mathcal{L}_t^*$ after running LoRA on $\mathcal{T}_t$ is minimized. This promotes values of $\boldsymbol{A}$ that can be easily adapted to unseen tasks downstream. We use the LoRA-ML loss but with symmetric low-rank adapters $\boldsymbol{U}_t\boldsymbol{U}_t^\top$ for the t_{th} task $\mathcal{T}_t$ in retraining. We still use asymmetric adapters for fine-tuning on the test task with loss $\mathcal{L}_{T+1}^n$. The LoRA-ML loss given access to infinite sample task losses $\mathcal{L}_t^*$ is then

$$\mathcal{L}_{\text{Meta}}(\boldsymbol{A}) = \sum_{t=1}^{T} \min_{\boldsymbol{U}_t} \mathcal{L}_t^*(\boldsymbol{A} + \boldsymbol{U}_t\boldsymbol{U}_t^\top). \tag{13}$$

Define the concatenation of each $\boldsymbol{U}_t$ as $\boldsymbol{U} = (\boldsymbol{U}_1, \ldots, \boldsymbol{U}_T) \in \left(\mathbb{R}^{d\times k}\right)^T$. Then minimizing (13) is equivalent to solving $\min_{\boldsymbol{A},\boldsymbol{U}} \mathcal{L}^*(\boldsymbol{A}, \boldsymbol{U})$ where

$$\mathcal{L}^*(\boldsymbol{A}, \boldsymbol{U}) = \frac{1}{2}\sum_{t=1}^{T} \left\| \boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{A} - \boldsymbol{U}_t\boldsymbol{U}_t^\top \right\|_F^2. \tag{14}$$

We have seen that standard retraining does not recover an optimal solution, but it is unclear what the global minima of this new objective function are and if they can be easily found. Note that by fixing $\boldsymbol{A}$, (14) is T independent symmetric matrix factorization problems, and by fixing $\boldsymbol{U}$, (14) is a convex quadratic problem over $\boldsymbol{A}$. Despite these well-understood sub-problems, joint minimization over $\boldsymbol{A}$ and $\boldsymbol{U}$ presents challenging variable interactions that complicate the analysis. Nevertheless, we employ a careful landscape analysis of (14) to address these questions.

3.2.1 Landscape of Global Minima of (14)

We first show that the objective is well-posed, i.e., minimization of $\mathcal{L}$ leads to an adaptable solution.

Theorem 5. *If $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, then $\hat{\boldsymbol{A}} = \boldsymbol{A}^* + \boldsymbol{C}$ where $\operatorname{rank}(\boldsymbol{C}) \leq 2k$*

Any point is a global minimum of (14) if and only if it achieves zero loss. Theorem 5 guarantees that the values of $\boldsymbol{A}$ that induce global minima of (14) are at most rank-$2k$ away from the ground truth parameter $\boldsymbol{A}^*$. Then, the remaining intrinsic dimension of the test task is just $3k \ll d$.

Corollary 6. *If $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, then there exists a rank-$3k$ adapter $\boldsymbol{Q}\boldsymbol{V}^\top$ such that $\mathcal{L}_{T+1}^*(\boldsymbol{Q}, \boldsymbol{V}; \hat{\boldsymbol{A}}) = 0$.*

Since the sufficient LoRA rank for fine-tuning is just $3k$, we realize a much improved fine-sample prediction error.

Corollary 7. *Let $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ and let $\boldsymbol{Q}^*, \boldsymbol{V}^* \in \mathbb{R}^{d \times 3k}$ minimize $\mathcal{L}^n_{T+1}(\hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top)$ over all $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d \times 3k}$. Then, $\hat{\boldsymbol{A}} + \boldsymbol{Q}^*\boldsymbol{V}^{*\top}$ satisfies Lemma 1 with $r = 3k$.*

Thus, retraining with LoRA-ML leads to squared prediction error on the task task which grows asymptotically as $\mathcal{O}\left(\frac{kd}{n}\right)$. Although the unnecessary factor of T incurred by standard retraining is avoided when using LoRA-ML, the rate still contains an additional factor of 3 over the ideal case when $r = k$ since $\boldsymbol{A}^*$ is not guaranteed to be recovered exactly. However, this minor discrepancy is mitigated when the number of tasks satisfies $T \geq 3$. In this case, exact recovery of the ground truth parameter $\boldsymbol{A}^*$ is possible.

Theorem 8. *For any $T \geq 3$, if $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ then $\hat{\boldsymbol{A}} = \boldsymbol{A}^*$ and $\boldsymbol{U}_t\boldsymbol{U}_t^\top = \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}$ for all $t \in [T]$*

This guarantees that the ground truth parameters are the unique global minimum up to orthogonal symmetry when there are three or more tasks, regardless of the ambient dimension or the number of columns k. This result is surprising, as most theoretical results for multi-task learning require higher task diversity, typically where the number of tasks T is required to be larger than the effective task dimension k [DHKLL21; CMOS22]. However, we establish this uniqueness result for the absolute condition $T \geq 3$. As a result, we only need a rank-k adaptation to realize the test task.

Corollary 9. *For $T \geq 3$, if $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, then there exists $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d \times k}$ such that $\mathcal{L}^*_{T+1}(\boldsymbol{Q}, \boldsymbol{V}; \hat{\boldsymbol{A}}) = 0$.*

We then achieve the desired fine-sample prediction error.

Corollary 10. *For $T \geq 3$, let $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ and let $\boldsymbol{Q}^*, \boldsymbol{V}^* \in \mathbb{R}^{d \times k}$ minimize $\mathcal{L}^n_{T+1}(\hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top)$ over all $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d \times k}$. Then, $\hat{\boldsymbol{A}} + \boldsymbol{Q}^*\boldsymbol{V}^{*\top}$ satisfies Lemma 1 with $r = k$.*

Note that the condition $T \geq 3$ is necessary to establish Theorem 8, as if there are only two tasks we can construct ground truth parameters such that the induced loss $\mathcal{L}^*$ has infinite solutions. See Appendix E.1 for an example.

Summary. These results show that all global minima of the LoRA-ML objective are low-rank adaptable to the downstream task and achieve finite-sample test task prediction error which grows as $\mathcal{O}\left(\frac{kd}{n}\right)$. Crucially, this avoids the factor of T incurred by standard retraining. Further, if $T \geq 3$, minimizing the LoRA-ML objective guarantees recovery of the ground truth parameters.

3.2.2 Algorithms for Minimizing (14)

As shown above, minimizing the LoRA-ML objective (14) leads to recovery of the ground truth parameters, with a small rank-$2k$ error term when $T = 2$. We prove that this minimization problem can always be solved by local optimization methods when there are two retraining tasks.

Theorem 11. *If $T = 2$, then $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ if and only if $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ is a second order stationary point (SOSP) of $\mathcal{L}^*$.*

Thus, when $T = 2$ local optimization algorithms for finding SOSPs, such as perturbed gradient descent and cubic-regularized Newton method, can efficiently minimize the meta-learning objective.

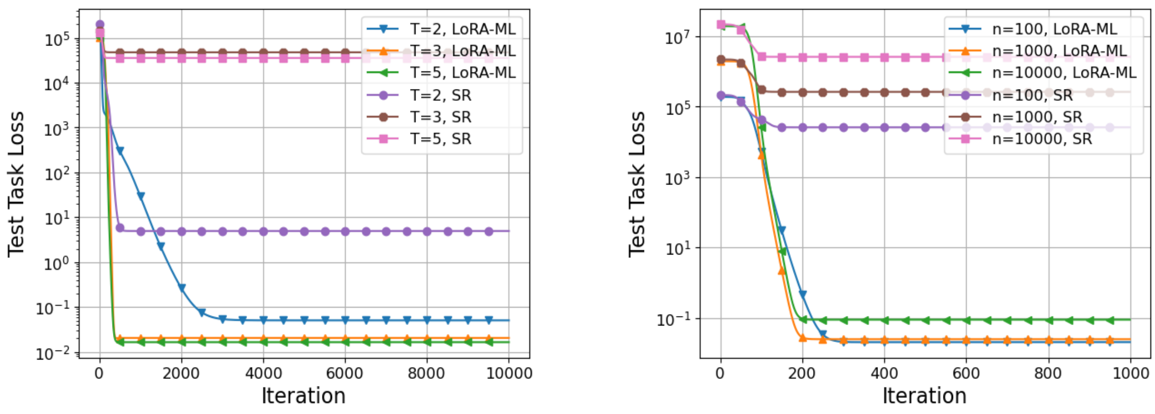
Surprisingly, when there are three or more tasks, numerical experiments (see Appendix E.2) show that adversarially picking $\boldsymbol{U}_t^*$ can result in specific instantiations of (14) with spurious local minima. In the next section, we perform extensive numerical experiments for various values of T which show that these spurious minima are almost never found in practice and vanilla gradient descent is sufficient to minimize (14).

4 Experiments

We first test our model on synthetic regression tasks. We consider data of the form $\boldsymbol{y}_{t,j} = \Phi(\boldsymbol{x}_{t,j}; \boldsymbol{A}^* + \boldsymbol{R}_t^*) + \boldsymbol{\epsilon}_{t,j}$ for task t and $j \in [n_t]$. We generate $\boldsymbol{x}_{t,j}$, $\boldsymbol{\epsilon}_{t,j}$, and $\boldsymbol{R}_t^*$ just as in Section 3. For the linear experiments, we set $\Phi(\boldsymbol{x}_{t,j}; \boldsymbol{A}^* + \boldsymbol{R}_t^*) = (\boldsymbol{A}^* + \boldsymbol{R}_t^*)\boldsymbol{x}_{t,j}$, and for the shallow network experiments we set $\Phi(\boldsymbol{x}_{t,j}; \boldsymbol{A}^* + \boldsymbol{R}_t^*) = \boldsymbol{c}^{*\top} s(\boldsymbol{A}^* + \boldsymbol{R}_t^*)\boldsymbol{x}_{t,j}$, where $s(\cdot)$ is the element-wise sigmoid function and $\boldsymbol{c}^* \in \mathbb{R}^d$ is an additional parameter shared across tasks. Although $\boldsymbol{c}^*$ is a parameter of the network, we suppress its notation as it does not require task-specific adaptation according to the data model. $\boldsymbol{A}^*$ (and $\boldsymbol{c}^*$ if applicable) are constructed by sampling each entry as an i.i.d. $\mathcal{N}(0, 1)$ random variable. We define parameters N, n such that $n_t = N$ for all $t \leq T$ and $n_{T+1} = n$. Setting $\mathcal{L}$ to be the mean squared error loss, we apply simple optimization methods to the standard retraining (1) and the LoRA-ML (5) objectives. After recovering $\hat{\boldsymbol{A}}$ (and $\hat{\boldsymbol{c}}$ if applicable) during retraining, we apply the low-rank adaptation $\boldsymbol{Q}\boldsymbol{V}^\top$ only to $\hat{\boldsymbol{A}}$ when fine-tuning to the test task. In each experiment we vary a single hyperparameter from a fixed set of values and plot the prediction error between the recovered model and the ground truth model $\frac{1}{n}\left\|\Phi(\boldsymbol{X}_{T+1}; \hat{\boldsymbol{A}} + \boldsymbol{Q}\boldsymbol{V}^\top) - \Phi(\boldsymbol{X}_{T+1}; \boldsymbol{A}^* + \boldsymbol{R}_{T+1}^*)\right\|_F^2$, averaged over 10 trials. See Appendix C for hyperparameter details and further ablations for both experimental settings.

4.1 Linear Model

For the linear experiments, we use gradient descent to optimize the loss in each training stage. When $T = 2$, we use a rank-$3k$ adaptation during fine-tuning to account for the inexact recovery explained in Theorem 5, and otherwise use a rank-k adaptation.



(a) Varying number of retraining tasks T

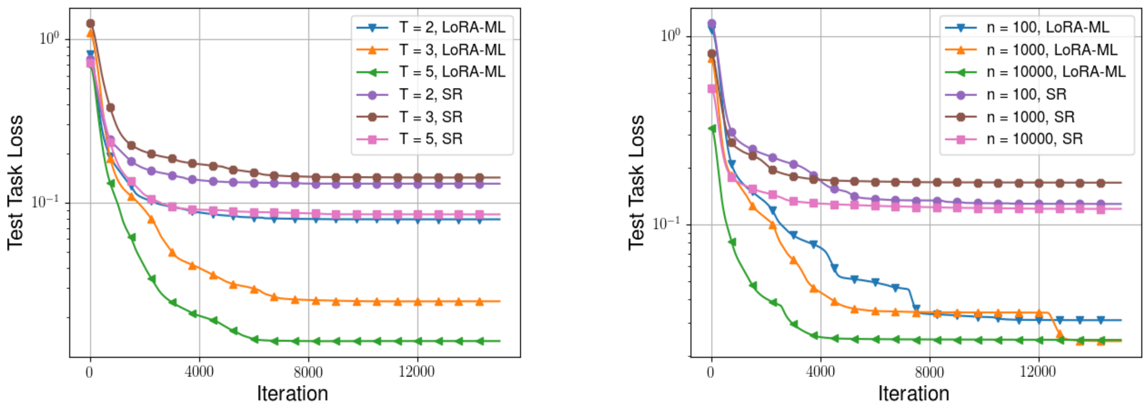
(b) Varying number of test task samples n

Figure 1: Linear model fine-tuning performance

Figures 1a and 1b show that LoRA-ML retraining significantly outperforms standard retraining (SR), for all data settings. When varying T beyond 2, we observe that applying gradient descent to the LoRA-ML objective is sufficient to achieve global minimization and recover an adaptable solution. Thus, even though there may exist spurious local minimizers, we do not encounter them in practice. Further, the fine-tuning performance after standard retraining worsens with larger T. This is supported by our theory in Section 3.1 which shows that as T increases, standard retraining recovers worse solutions that leave a larger intrinsic dimension for the fine-tuning stage.

4.2 Shallow Neural Network

For the shallow network, we use the AdamW optimizer [LH19] and apply rank-k adapters in all experiments.



(a) Varying number of retraining tasks T

(b) Varying number of test task samples n

Figure 2: Shallow network fine-tuning performance

Figures 2a and 2b again show that retraining using the LoRA-ML objective leads to much better fine-tuning performance relative to standard retraining. Figure 2a shows that the LoRA-ML objective effectively exploits the number of tasks, since fine-tuning performance improves as T increases. Further, we observe in Figure 2b that fine-tuning with any number of samples after standard retraining cannot even recover the performance of fine-tuning with just 100 samples after LoRA-ML retraining.

4.3 LLM Experiments

To test the LoRA-ML objective on real data, we use the pretrained RoBERTa-Large language model on the ConvAI2 dataset. ConvAI2 consists of conversations between two personas, where each persona is associated with a short list of factual information that informs their responses. We model learning the dialogue continuations of each individual persona as a different task, where we aim to select the correct conversation continuation from a set of candidates given the conversation history.

We perform retraining using $T = 10$ tasks and select the model from the epoch with the best average accuracy on the heldout samples. We then fine-tune to 10 test tasks individually. We run 5 trials and report the median accuracy on the heldout data from the best performing epoch for each task in the tables below. All training was done on a single NVIDIA A40 GPU, and we report our training hyperparameters in Appendix D.

Table 1: Test task prediction accuracies using rank-8 and rank-16 fine-tuning adaptations

(a) Test task prediction accuracies using rank-8 fine-tuning adaptations

Algorithm	**Task 1**	**Task 2**	**Task 3**	**Task 4**	**Task 5**	**Task 6**	**Task 7**	**Task 8**	**Task 9**	**Task 10**	**Average**
SR	43.75	40.00	43.48	41.94	41.03	37.23	42.73	43.20	41.13	40.76	41.52
LoRA-ML-8	**50.00**	**50.00**	**47.82**	**48.39**	**46.15**	**41.49**	**44.55**	**44.00**	**42.55**	**42.68**	**45.76**

(b) Test task prediction accuracies using rank-16 fine-tuning adaptations

Algorithm	**Task 1**	**Task 2**	**Task 3**	**Task 4**	**Task 5**	**Task 6**	**Task 7**	**Task 8**	**Task 9**	**Task 10**	**Average**
SR	43.75	43.33	39.13	38.71	39.74	35.11	38.18	39.20	39.72	38.85	39.57
LoRA-ML-8	**50.0**	**53.33**	**50.0**	**50.0**	**48.72**	**42.55**	**45.45**	**44.80**	**45.39**	**44.59**	**47.48**
LoRA-ML-16	43.75	33.33	36.96	40.32	43.59	39.36	42.73	41.60	40.43	40.13	40.22

Table 1a shows test task fine-tuning performance using rank-8 LoRA after each retraining method. The SR row denotes the model recovered using standard retraining, while LoRA-ML-8 denotes the model recovered by LoRA-ML using rank-8 adapters. Then, as suggested by Theorem 5, we test if we can improve performance by increasing the LoRA rank during fine-tuning relative to the rank of the adapters in retraining with LoRA-ML. Table 1b shows fine-tuning performance using rank-16 LoRA after each of standard retraining, LoRA-ML with rank-8 adapters, and LoRA-ML with rank-16 adapters. We observe that LoRA-ML consistently outperforms standard retraining. Additionally, we observe that increasing the fine-tuning rank relative to the rank used in retraining may confer performance benefits, as even though the LoRA-ML-16 model is more expressive than LoRA-ML-8, restricting the adapter rank during retraining may act as a form of regularization.

5 Conclusion

We introduced the PEFT-ML objective function for retraining an FM on a collection of tasks to prepare the model for subsequent downstream fine-tuning. We provide theoretical results demonstrating strict performance gaps between standard retraining and the PEFT-ML objective using LoRA (LoRA-ML). Empirically, a basic implementation of the LoRA-ML objective outperforms standard retraining for adapting to unseen downstream tasks. Future avenues include extending our theoretical analysis to more general adapters and different model architectures.

Acknowledgments

This work was supported in part by NSF Grants 2019844, 2107037, and 2112471, ONR Grant N00014-19-1-2566, the Machine Learning Lab (MLL) at UT Austin, the NSF AI Institute for Foundations of Machine Learning (IFML), and Qualcomm through the Wireless Networking and Communications Group (WNCG) Industrial Affiliates Program.

References

[AAAAAB+24] M. Abdin, J. Aneja, H. Awadalla, A. Awadallah, A. A. Awan, N. Bach, et al. *Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone*. 2024. arXiv: 2404.14219 [cs.CL]. URL: https://arxiv.org/abs/2404.14219 (page 2).

[AGSCZG21] A. Aghajanyan, A. Gupta, A. Shrivastava, X. Chen, L. Zettlemoyer, and S. Gupta. "Muppet: Massive Multi-task Representations with Pre-Finetuning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 5799–5811. DOI: 10.18653/v1/2021.emnlp-main.468. URL: https://aclanthology.org/2021.emnlp-main.468 (pages 2, 4, 5).

[BAWLM22] T. Bansal, S. Alzubi, T. Wang, J.-Y. Lee, and A. McCallum. "Meta-Adapters: Parameter Efficient Few-shot Fine-tuning through Meta-Learning". In: *First Conference on Automated Machine Learning (Main Track)*. 2022. URL: https://openreview.net/forum?id=BCGNf-prLg5 (pages 2, 3).

[BMRSKDN+20] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (pages 2, 4).

[BSW11] F. Bunea, Y. She, and M. H. Wegkamp. "OPTIMAL SELECTION OF REDUCED RANK ESTIMATORS OF HIGH-DIMENSIONAL MATRICES". *The Annals of Statistics* 39.2 (2011), pp. 1282–1309. ISSN: 00905364, 21688966. URL: http://www.jstor.org/stable/29783674 (visited on 01/14/2025) (page 7).

[CMOS22] L. Collins, A. Mokhtari, S. Oh, and S. Shakkottai. "MAML and ANIL Provably Learn Representations". In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, July 2022, pp. 4238–4310. URL: https://proceedings.mlr.press/v162/collins22a.html (pages 2, 3, 10).

[DPHZ23] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. "QLoRA: Efficient Finetuning of Quantized LLMs". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=OUIFPHEgJU (pages 4, 18).

[DCLT19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, and T. Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–

4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423 (page 2).

[Din+19] E. Dinan et al. “The Second Conversational Intelligence Challenge (ConvAI2)”. *ArXiv* abs/1902.00098 (2019). URL: https://api.semanticscholar.org/CorpusID:59553505 (page 3).

[DYLLXLWYZZ24] G. Dong, H. Yuan, K. Lu, C. Li, M. Xue, D. Liu, W. Wang, Z. Yuan, C. Zhou, and J. Zhou. *How Abilities in Large Language Models are Affected by Supervised Fine-tuning Data Composition*. 2024. arXiv: 2310.05492 [cs.CL]. URL: https://arxiv.org/abs/2310.05492 (page 2).

[DHKLL21] S. S. Du, W. Hu, S. M. Kakade, J. D. Lee, and Q. Lei. “Few-Shot Learning via Learning the Representation, Provably”. In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=pW2Q2xLwIMD (page 10).

[FAL17] C. Finn, P. Abbeel, and S. Levine. “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by D. Precup and Y. W. Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, Aug. 2017, pp. 1126–1135. URL: https://proceedings.mlr.press/v70/finn17a.html (pages 2, 3).

[GMM22] M. Gheini, X. Ma, and J. May. *Know Where You’re Going: Meta-Learning for Parameter-Efficient Fine-Tuning*. 2022. arXiv: 2205.12453 [cs.CL] (pages 2, 3).

[HJ22] S. K. Hong and T. Y. Jang. “AMAL: Meta Knowledge-Driven Few-Shot Adapter Learning”. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Y. Goldberg, Z. Kozareva, and Y. Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 10381–10389. DOI: 10.18653/v1/2022.emnlp-main.709. URL: https://aclanthology.org/2022.emnlp-main.709 (pages 2, 3).

[HSP22] Z. Hou, J. Salazar, and G. Polovets. “Meta-Learning the Difference: Preparing Large Language Models for Efficient Adaptation”. *Transactions of the Association for Computational Linguistics* 10 (Nov. 2022), pp. 1249–1265. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00517. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00517/2059907/tacl_a_00517.pdf. URL: https://doi.org/10.1162/tacl%5C_a%5C_00517 (pages 2, 3).

[HGJMDGAG19] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. “Parameter-Efficient Transfer Learning for NLP”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 2790–2799. URL: https://proceedings.mlr.press/v97/houlsby19a.html (page 4).

[HSWALWWC21] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. “Lora: Low-rank adaptation of large language models”. *arXiv preprint arXiv:2106.09685* (2021) (pages 2, 5, 18).

[HMMF23] N. Z. Hu, E. Mitchell, C. D. Manning, and C. Finn. "Meta-Learning Online Adaptation of Language Models". In: *The 2023 Conference on Empirical Methods in Natural Language Processing*. 2023. URL: https://openreview.net/forum?id=jPrl18r4RA (page 2).

[Ize75] A. J. Izenman. "Reduced-rank regression for the multivariate linear model". *Journal of Multivariate Analysis* 5.2 (1975), pp. 248–264. ISSN: 0047-259X. DOI: https://doi.org/10.1016/0047-259X(75)90042-1. URL: https://www.sciencedirect.com/science/article/pii/0047259X75900421 (page 7).

[JLR24] U. Jang, J. D. Lee, and E. K. Ryu. *LoRA Training in the NTK Regime has No Spurious Local Minima*. 2024. arXiv: 2402.11867 [cs.LG] (pages 4, 18).

[KMKSTCH20] D. Khashabi, S. Min, T. Khot, A. Sabharwal, O. Tafjord, P. Clark, and H. Hajishirzi. "UNIFIEDQA: Crossing Format Boundaries with a Single QA System". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by T. Cohn, Y. He, and Y. Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 1896–1907. DOI: 10.18653/v1/2020.findings-emnlp.171. URL: https://aclanthology.org/2020.findings-emnlp.171 (page 2).

[LC18] Y. Lee and S. Choi. "Gradient-Based Meta-Learning with Learned Layerwise Metric and Subspace". In: *International Conference on Machine Learning*. 2018. URL: https://api.semanticscholar.org/CorpusID:3350728 (page 2).

[LL21] X. L. Li and P. Liang. "Prefix-Tuning: Optimizing Continuous Prompts for Generation". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by C. Zong, F. Xia, W. Li, and R. Navigli. Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597. DOI: 10.18653/v1/2021.acl-long.353. URL: https://aclanthology.org/2021.acl-long.353 (page 2).

[LWYMWCC24] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen. "DoRA: Weight-Decomposed Low-Rank Adaptation". *arXiv preprint arXiv:2402.09353* (2024) (pages 4, 5, 18).

[LOGDJCLLZS19] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. cite arxiv:1907.11692. 2019. URL: http://arxiv.org/abs/1907.11692 (pages 3, 4).

[LH19] I. Loshchilov and F. Hutter. "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7 (page 12).

[NAS18] A. Nichol, J. Achiam, and J. Schulman. "On First-Order Meta-Learning Algorithms". *ArXiv* abs/1803.02999 (2018). URL: https://api.semanticscholar.org/CorpusID:4587331 (page 2).

[Rad+21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *International Conference on Machine Learning*. 2021. URL: https://api.semanticscholar.org/CorpusID:231591445 (page 2).

[RSRLNMZLL20] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. "Exploring the limits of transfer learning with a unified text-to-text transformer". *Journal of machine learning research* 21.140 (2020), pp. 1–67 (page 2).

[RT11] A. Rohde and A. B. Tsybakov. "ESTIMATION OF HIGH-DIMENSIONAL LOW-RANK MATRICES". *The Annals of Statistics* 39.2 (2011), pp. 887–930. ISSN: 00905364, 21688966. URL: http://www.jstor.org/stable/29783661 (visited on 01/15/2025) (page 7).

[SZKA20] N. Saunshi, Y. Zhang, M. Khodak, and S. Arora. "A sample complexity separation between non-convex and convex meta-learning". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org, 2020 (pages 2, 3).

[TJNO21] K. K. Thekumparampil, P. Jain, P. Netrapalli, and S. Oh. "Statistically and Computationally Efficient Linear Meta-representation Learning". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 18487–18500. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/99e7e6ce097324aceb45f98299ceb621-Paper.pdf (page 3).

[ZL23] Y. Zeng and K. Lee. *The Expressive Power of Low-Rank Adaptation*. 2023. arXiv: 2310.17513 [cs.LG] (pages 4, 18).

[ZCBHCCZ23] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao. "Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=lq62uWRJjiY (pages 4, 18).

[ZQW20] Y. Zhang, Q. Qu, and J. Wright. *From Symmetry to Geometry: Tractable Nonconvex Problems*. July 2020 (page 24).

A Related Work on LoRA-Style PEFT

There is a vast amount of work in developing PEFT methods for FMs. The LoRA algorithm [HSWALWWC21] has established itself as a popular and successful PEFT strategy and has inspired various extensions such as QLoRA, DoRA, and others [DPHZ23; LWYMWCC24; ZCBHCCZ23]. These algorithms are heuristics for mimicking the full finetuning of an FM to a specific downstream task and have proven to be empirically successful in various settings. However, there is a lack of theoretical analysis on the adaptability of PFMs under LoRA-style adaptations, the ability to efficiently optimize LoRA-style objectives, and the kinds of solutions they recover. Some recent works have attempted to analyze different parts of these theoretical questions.

Convergence of LoRA. [JLR24] analyzes the optimization landscape for LoRA for the Neural Tangent Kernel regime. The authors show that LoRA finetuning converges in this setting as they prove that the objective function satisfies a strict saddle property, ensuring that there are no spurious local minima. However, this focuses on the actual ability of LoRA to converge to the optimal low-rank adapter given an FM, and does not consider the adaptability of the FM in the first place.

Expressivity of LoRA. [ZL23] derives the expressive power of LoRA as a function of model depth. This work shows that under some mild conditions, fully connected and transformer networks when respectively adapted with LoRA can closely approximate arbitrary smaller networks. They quantify the required LoRA rank to achieve this approximation as well as the resulting approximation error.

B Proofs

B.1 Proof of Theorem 2

By Equation (12) we have that $\hat{\boldsymbol{A}}_{\mathrm{SR}} = \boldsymbol{A}^* + \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top}$. In the following the expectation is always taken over $\boldsymbol{U}^* = (\boldsymbol{U}_1^*, \dots, \boldsymbol{U}_T^*)$, where $\boldsymbol{U}_t^* \in \mathbb{R}^{d\times k}$ satisfies $\mathrm{vec}(\boldsymbol{U}_t^*) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{dk})$. Then,

$$
\begin{aligned}
\mathbb{E}_{\boldsymbol{U}^*}\left[\sum_{t=1}^{T} \mathcal{L}_t(\hat{\boldsymbol{A}}_{\mathrm{SR}})\right] &= \sum_{t=1}^{T} \mathbb{E}\left[\left\|\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{A}^* - \frac{1}{T}\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right\|_F^2\right] \\
&= \sum_{t=1}^{T} \mathbb{E}\left[\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \frac{1}{T}\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right\|_F^2\right] \\
&= \sum_{t=1}^{T} \mathbb{E}\left[\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I} - \frac{1}{T}\sum_{s=1}^{T}\left(\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right)\right\|_F^2\right] \\
&= \sum_{t=1}^{T} \mathbb{E}\left[\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right\|_F^2 + \left\|\frac{1}{T}\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right\|_F^2 - \right. \\
&\qquad\qquad \left. \frac{2}{T}\,\mathrm{tr}\left\{\left(\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right)\left(\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right)\right\}\right] \\
&= T\mathbb{E}\left[\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right\|_F^2\right] + \frac{1}{T}\mathbb{E}\left[\left\|\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right\|_F^2\right] \\
&\qquad\qquad - \frac{2}{T}\mathbb{E}\left[\mathrm{tr}\left\{\left(\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right)\left(\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right)\right\}\right]
\end{aligned}
$$

Using the fact that each $\boldsymbol{U}_t^*\boldsymbol{U}_t^*$ is an independent sample of a $d \times d$ Wishart distribution with scatter matrix $\boldsymbol{I}$ and k degrees of freedom, each term is computed as

$$
\begin{aligned}
\mathbb{E}\left[\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right\|_F^2\right] &= kd(d+1) \\
\mathbb{E}\left[\left\|\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right\|_F^2\right] &= Tkd(d+1) \\
\mathbb{E}\left[\mathrm{tr}\left\{\left(\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - k\boldsymbol{I}\right)\left(\sum_{s=1}^{T}\boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top} - k\boldsymbol{I}\right)\right\}\right] &= kd(d+1)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\mathbb{E}_{\boldsymbol{U}^*}\left[\sum_{t=1}^{T} \mathcal{L}_t(\hat{\boldsymbol{A}}_{\mathrm{SR}})\right] &= Tkd(d+1) + kd(d+1) - \frac{2}{T}kd(d+1) \\
&= (T + 1 - \frac{2}{T})kd(d+1) = \Omega\left(Tkd^2\right)
\end{aligned}
$$

B.2 Proof of Propositions 3,4

Recall $\hat{\boldsymbol{A}}_{\text{SR}} = \boldsymbol{A}^* + \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{U}_t^* \boldsymbol{U}_t^{*\top}$. Then for any $\boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d\times r}$,

$$\begin{aligned}\mathcal{L}_{T+1}^*(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{A}}_{\text{SR}}) &= \frac{1}{2}\left\|\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \hat{\boldsymbol{A}}_{\text{SR}} - \boldsymbol{Q}\boldsymbol{V}^\top\right\|_F^2 \\ &= \frac{1}{2}\left\|\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \sum_{t=1}^{T}\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{Q}\boldsymbol{V}^\top\right\|_F^2\end{aligned}$$

By the assumption in Remark 1 we have that $\operatorname{rank}\left(\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \sum_{t=1}^{T}\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}\right) = k(T+1)$. Then Proposition 3 follows from that fact that $\operatorname{rank}(\boldsymbol{Q}\boldsymbol{V}^\top) \leq r$.

Further, as $T \to \infty$, the strong law of large numbers implies that $\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} \to \mathbb{E}\left[\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}\right] = k\boldsymbol{I}$. Thus for large T,

$$\left\|\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{Q}\boldsymbol{V}^\top\right\|_F^2 \approx \left\|\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - k\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{V}^\top\right\|_F^2 \tag{15}$$

Using classic low-rank matrix factorization results, the $\boldsymbol{Q}^*\boldsymbol{V}^{*\top}$ that minimizes $\left\|\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - k\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{V}^\top\right\|_F^2$ will exactly capture the r eigen-directions of $\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - k\boldsymbol{I}$ with largest magnitude. But, $\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - k\boldsymbol{I}$ has $d-k$ eigenvalues of magnitude k, so $\boldsymbol{Q}^*\boldsymbol{V}^{*\top}$ can only capture r of them. Thus, $\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - k\boldsymbol{I} - \boldsymbol{Q}^*\boldsymbol{V}^{*\top} \geq (d-k-r)k^2$. Since $\boldsymbol{Q}^*\boldsymbol{V}^{*\top}$ minimized this quantity, we have that

$$\mathcal{L}_{T+1}^*(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{A}}_{\text{SR}}) \geq (d-k-r)k^2 \quad \forall \boldsymbol{Q}, \boldsymbol{V} \in \mathbb{R}^{d\times r}$$

Thus, $\mathcal{L}_{T+1}(\boldsymbol{Q}, \boldsymbol{V}\,; \hat{\boldsymbol{A}}_{\text{SR}})$ scales as $(d-k-r)k^2 \approx (d-r)k^2$ since $k \ll d$.

B.3 Proof of theorem 5

Since $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ and $\mathcal{L}^* \geq 0$ we must have that $\nabla_{\boldsymbol{A}}\mathcal{L}^* = 0$.

Thus, $\hat{\boldsymbol{A}} = \boldsymbol{A}^* - \frac{1}{T}\sum_{j=1}^{T}\left(\hat{\boldsymbol{U}}_j\hat{\boldsymbol{U}}_j^\top - \boldsymbol{U}_j^*\boldsymbol{U}_j^{*\top}\right)$. Plugging this into $\mathcal{L}^*$ gives

$$\begin{aligned}0 = \mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) &= \frac{1}{2}\sum_{t=1}^{T}\left\|\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \left(\boldsymbol{A}^* - \frac{1}{T}\sum_{s=1}^{T}\left(\hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^\top - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right)\right) - \boldsymbol{U}_t\boldsymbol{U}_t^\top\right\|_F^2 \\ &= \frac{1}{2}\sum_{t=1}^{T}\left\|\boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{U}_t\boldsymbol{U}_t^\top - \frac{1}{T}\sum_{s=1}^{T}\left(\hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^\top - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right)\right\|_F^2.\end{aligned}$$

Thus each term of the summation is zero, so for all $t, s \in [T]$,

$$\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^T - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*T} = \hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^T - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*T}.$$

Combining these results gives that

$$\begin{aligned}\hat{\boldsymbol{A}} &= \boldsymbol{A}^* - \frac{1}{T}\sum_{s=1}^{T}\left(\hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^\top - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right) \\ &= \boldsymbol{A}^* - \left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right)\end{aligned}$$

Let $\boldsymbol{C} = -\left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right)$. Then $\hat{\boldsymbol{A}} = \boldsymbol{A}^* + \boldsymbol{C}$ and $rank(\boldsymbol{C}) \leq rank(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top) + rank(\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}) \leq 2k$.

Note the the effective remaining test-task dimension is

$$\begin{aligned}\operatorname{rank}\left(\boldsymbol{A}^* + \boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \boldsymbol{A}^* - \boldsymbol{C}\right) &= \operatorname{rank}\left(\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top} - \boldsymbol{C}\right) \\ &\leq \operatorname{rank}\left(\boldsymbol{U}_{T+1}^*\boldsymbol{U}_{T+1}^{*\top}\right) + \operatorname{rank}(\boldsymbol{C}) \\ &\leq 3k\end{aligned}$$

B.4 Proof of Theorem 8

Proof. Since $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, we have that for all $t, s \in [T]$,

$$\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^T - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*T} = \hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^T - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*T} \tag{16}$$

Applying this to the first three tasks and rearranging gives that

$$\begin{aligned}\boldsymbol{U}_1^*\boldsymbol{U}_1^{*T} &= \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^T + \boldsymbol{U}_2^*\boldsymbol{U}_2^{*T} - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^T \qquad (17) \\ &= \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^T + \boldsymbol{U}_3^*\boldsymbol{U}_3^{*T} - \hat{\boldsymbol{U}}_3\hat{\boldsymbol{U}}_3^T. \qquad (18)\end{aligned}$$

We first show that $\operatorname{im}(\hat{\boldsymbol{U}}_1) = \operatorname{im}(\boldsymbol{U}_1^*)$.
Since $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*T} \succcurlyeq 0$, we must have that $\operatorname{im}(\hat{\boldsymbol{U}}_2) \subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_2^*)$ and $\operatorname{im}(\hat{\boldsymbol{U}}_3) \subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_3^*)$, as otherwise there would exist a vector on $\ker\left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^T + \boldsymbol{U}_2^*\boldsymbol{U}_2^{*T}\right) \cap \ker(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^T)^\perp$ whose existence contradicts the positive semi-definiteness of $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*T}$.
Thus,

$$\begin{aligned}\operatorname{im}(\boldsymbol{U}_1^*) &\subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_2^*) \qquad (19) \\ \operatorname{im}(\boldsymbol{U}_1^*) &\subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_3^*) \qquad (20)\end{aligned}$$

Using that fact that for subspaces $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$, $\boldsymbol{X} \subseteq \boldsymbol{Y} \implies \boldsymbol{X} + \boldsymbol{Z} \subseteq \boldsymbol{Y} + \boldsymbol{Z}$, we can add $\operatorname{im}(\boldsymbol{U}_2^*)$ and $\operatorname{im}(\boldsymbol{U}_3^*)$ to both sides of 19 and 20 respectively. This gives:

$$\begin{aligned}\operatorname{im}(\boldsymbol{U}_1^*) \oplus \operatorname{im}(\boldsymbol{U}_2^*) &\subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_2^*) \qquad (21) \\ \operatorname{im}(\boldsymbol{U}_1^*) \oplus \operatorname{im}(\boldsymbol{U}_3^*) &\subseteq \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\boldsymbol{U}_3^*). \qquad (22)\end{aligned}$$

For $t \in \{2,3\}$, we clearly have that $\dim\left(\mathrm{im}(\hat{\boldsymbol{U}}_1) + \mathrm{im}(\boldsymbol{U}_t^*)\right) \leq \dim \mathrm{im}(\hat{\boldsymbol{U}}_1) + \dim \mathrm{im}(\boldsymbol{U}_t^*) \leq 2k$, and $\dim\left(\mathrm{im}(\boldsymbol{U}_1^*) + \mathrm{im}(\boldsymbol{U}_t^*)\right) = 2k$. Thus,

$$(\mathrm{im}(\boldsymbol{U}_1^*) \oplus \mathrm{im}(\boldsymbol{U}_2^*)) = \left(\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)\right) \tag{23}$$

$$(\mathrm{im}(\boldsymbol{U}_1^*) \oplus \mathrm{im}(\boldsymbol{U}_3^*)) = \left(\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_3^*)\right) \tag{24}$$

Lemma 12. $\left([\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]\right) = \mathrm{im}(\hat{\boldsymbol{U}}_1)$

Proof. Clearly, $\mathrm{im}(\hat{\boldsymbol{U}}_1) \subseteq \left([\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]\right)$. To show the converse, consider $\boldsymbol{x} \in \left([\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]\right)$.

By assumption there exists some $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d} \in \mathbb{R}^k$ such that

$$\boldsymbol{x} = \hat{\boldsymbol{U}}_1 \boldsymbol{a} + \boldsymbol{U}_2^* \boldsymbol{b} \tag{25}$$

$$= \hat{\boldsymbol{U}}_1 \boldsymbol{c} + \boldsymbol{U}_3^* \boldsymbol{d} \tag{26}$$

Thus,

$$\hat{\boldsymbol{U}}_1(\boldsymbol{a} - \boldsymbol{c}) + \boldsymbol{U}_2^* \boldsymbol{b} - \boldsymbol{U}_3^* \boldsymbol{d} = \boldsymbol{0}. \tag{27}$$

By Equation 23, we can write

$$\begin{aligned} \mathrm{im}(\boldsymbol{U}_2^*) &= ([\mathrm{im}(\boldsymbol{U}_1^*) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\boldsymbol{U}_2^*) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]) \\ &= \left([\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\boldsymbol{U}_2^*) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]\right) \end{aligned}$$

Thus, $\mathrm{im}(\hat{\boldsymbol{U}}_1) \cap [\mathrm{im}(\boldsymbol{U}_2^*) \oplus \mathrm{im}(\boldsymbol{U}_3^*)] \subseteq \mathrm{im}(\hat{\boldsymbol{U}}_1) \cap \mathrm{im}(\boldsymbol{U}_2^*) = \{\boldsymbol{0}\}$, so

$$\mathrm{im}(\hat{\boldsymbol{U}}_1) \cap [\mathrm{im}(\boldsymbol{U}_2^*) \oplus \mathrm{im}(\boldsymbol{U}_3^*)] = \{\boldsymbol{0}\} \tag{28}$$

Applying Equation (28) to Equation (27) implies that $\boldsymbol{a} = \boldsymbol{c}$ and $\boldsymbol{b} = \boldsymbol{d} = \boldsymbol{0}$. Thus $\boldsymbol{x} = \hat{\boldsymbol{U}}_1 \boldsymbol{a} \in \mathrm{im}(\hat{\boldsymbol{U}}_1)$, so $\left([\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_2^*)] \cap [\mathrm{im}(\hat{\boldsymbol{U}}_1) \oplus \mathrm{im}(\boldsymbol{U}_3^*)]\right) \subseteq \mathrm{im}(\hat{\boldsymbol{U}}_1)$. □

Then Equations (19) and (20) combined with Lemma (12) implies that $\mathrm{im}(\boldsymbol{U}_1^*) \subseteq \mathrm{im}(\hat{\boldsymbol{U}}_1)$ but $\dim(\mathrm{im}(\boldsymbol{U}_1^*)) = \dim(\mathrm{im}(\hat{\boldsymbol{U}}_1)) = k$, so $\mathrm{im}(\boldsymbol{U}_1^*) = \mathrm{im}(\hat{\boldsymbol{U}}_1)$.

Since the initial assumptions about $\hat{\boldsymbol{U}}_1$ and $\boldsymbol{U}_1^*$ analogously hold for the corresponding matrices for tasks 2 and 3, by the exact same argument we can show that

$$\mathrm{im}(\boldsymbol{U}_t^*) = \mathrm{im}(\hat{\boldsymbol{U}}_t) \quad \forall t \in [T]. \tag{29}$$

Then by equation (16), $\mathrm{im}(\boldsymbol{U}_1^*) \supseteq \mathrm{im}\left(\hat{\boldsymbol{U}}_1 \hat{\boldsymbol{U}}_1^T - \boldsymbol{U}_1^* \boldsymbol{U}_1^{*T}\right) = \mathrm{im}\left(\hat{\boldsymbol{U}}_2 \hat{\boldsymbol{U}}_2^T - \boldsymbol{U}_2^* \boldsymbol{U}_2^{*T}\right) \subseteq \mathrm{im}(\boldsymbol{U}_2^*)$. Thus,

$$\begin{aligned} \mathrm{im}\left(\hat{\boldsymbol{U}}_1 \hat{\boldsymbol{U}}_1^T - \boldsymbol{U}_1^* \boldsymbol{U}_1^{*T}\right) &\subseteq \mathrm{im}(\boldsymbol{U}_1^*) \cap \mathrm{im}(\boldsymbol{U}_2^*) \\ &= \{\boldsymbol{0}\}. \end{aligned}$$

Thus $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^T = \boldsymbol{U}_1^*\boldsymbol{U}_1^{*T}$. Then by Equation (16), $\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^T = \boldsymbol{U}_t^*\boldsymbol{U}_t^{*T}$ for all $t \in [T]$. Lastly, since $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, we have that $\nabla_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, so

$$\hat{\boldsymbol{A}} = \boldsymbol{A}^* + \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{U}_t\boldsymbol{U}_t^\top = \boldsymbol{A}^*$$

□

B.5 Proof of Theorem 11

Clearly if $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$, then $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ is an SOSP. The reverse direction is the challenging part of the proof. We equivalently prove that if $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ is a critical point and $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) \neq 0$, then $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ has a negative eigenvalue.

Assume for the sake of contradiction that $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ is a critical point and $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) \neq 0$. Then,

$$\nabla_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = T(\hat{\boldsymbol{A}} - \boldsymbol{A}^*) + \sum_{t=1}^{T}\left(\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}\right) = \boldsymbol{0} \tag{30}$$

$$\nabla_{\boldsymbol{U}_t}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 2\left(\hat{\boldsymbol{A}} - \boldsymbol{A}^* + \hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}\right)\hat{\boldsymbol{U}}_t = \boldsymbol{0} \tag{31}$$

Thus,

$$\hat{\boldsymbol{A}} = \boldsymbol{A}^* - \frac{1}{T}\sum_{t=1}^{T}\left(\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top}\right). \tag{32}$$

Define $\boldsymbol{B}_t(\hat{\boldsymbol{U}}) = \hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \frac{1}{T}\sum_{s=1}^{T}\left(\hat{\boldsymbol{U}}_s\hat{\boldsymbol{U}}_s^\top - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top}\right)$. Despite being a slight abuse of notation, we refer to $\boldsymbol{B}_t(\hat{\boldsymbol{U}})$ as just $\boldsymbol{B}_t$ for the remainder of the proof.

Then (31) equivalently states:

$$\boldsymbol{B}_t\hat{\boldsymbol{U}}_t = 0. \tag{33}$$

Note that by construction, $\sum_{t=1}^{T}\boldsymbol{B}_t = 0$.

Considering $\mathcal{L}$ as a function of the flattened vector $[\mathrm{vec}(\boldsymbol{A}); \mathrm{vec}(\boldsymbol{U}_1); \mathrm{vec}(\boldsymbol{U}_2)]$, and let $\boldsymbol{U}_1 = [\boldsymbol{x}_1\ \ldots\ \boldsymbol{x}_k]$, $\boldsymbol{U}_2 = [\boldsymbol{y}_1\ \ldots\ \boldsymbol{y}_k]$, we compute the Hessian

$$\nabla^2\mathcal{L} = \begin{bmatrix} \nabla^2_{\boldsymbol{A}}\mathcal{L} & \nabla_{\boldsymbol{U}_1}\nabla_{\boldsymbol{A}}\mathcal{L} & \nabla_{\boldsymbol{U}_2}\nabla_{\boldsymbol{A}}\mathcal{L} \\ (\nabla_{\boldsymbol{U}_1}\nabla_{\boldsymbol{A}}\mathcal{L})^\top & \nabla^2_{\boldsymbol{U}_1}\mathcal{L} & \boldsymbol{0} \\ (\nabla_{\boldsymbol{U}_2}\nabla_{\boldsymbol{A}}\mathcal{L})^\top & \boldsymbol{0} & \nabla^2_{\boldsymbol{U}_2}\mathcal{L} \end{bmatrix} \tag{34}$$

where

$$\begin{aligned}
\nabla^2_{\boldsymbol{A}}\mathcal{L}^* &= 2\boldsymbol{I}_{d^2} \\
\nabla_{\boldsymbol{U}_1}\nabla_{\boldsymbol{A}}\mathcal{L}^* &= \left[(\boldsymbol{x}_1 \oplus \boldsymbol{x}_1)\ \dots\ (\boldsymbol{x}_k \oplus \boldsymbol{x}_k)\right] \in \mathbb{R}^{d^2 \times dk} \\
\nabla_{\boldsymbol{U}_2}\nabla_{\boldsymbol{A}}\mathcal{L}^* &= \left[(\boldsymbol{y}_1 \oplus \boldsymbol{y}_1)\ \dots\ (\boldsymbol{y}_k \oplus \boldsymbol{y}_k)\right] \in \mathbb{R}^{d^2 \times dk} \\
\nabla^2_{\boldsymbol{U}_1}\mathcal{L}^* &= 2(\boldsymbol{A} + \boldsymbol{U}_1\boldsymbol{U}_1^\top - \boldsymbol{A}^* - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}) \otimes \boldsymbol{I}_k \\
&\quad + 2\begin{bmatrix} \boldsymbol{x}_1\boldsymbol{x}_1^\top + \|\boldsymbol{x}_1\|_2^2\,\boldsymbol{I} & \boldsymbol{x}_1^\top\boldsymbol{x}_2\boldsymbol{I} + \boldsymbol{x}_2\boldsymbol{x}_1^\top & \dots & \boldsymbol{x}_1^\top\boldsymbol{x}_k\boldsymbol{I} + \boldsymbol{x}_k\boldsymbol{x}_1^\top \\ \boldsymbol{x}_2^\top\boldsymbol{x}_1\boldsymbol{I} + \boldsymbol{x}_1\boldsymbol{x}_2^\top & \boldsymbol{x}_2\boldsymbol{x}_2^\top + \|\boldsymbol{x}_2\|_2^2\,\boldsymbol{I} & \dots & \boldsymbol{x}_2^\top\boldsymbol{x}_k\boldsymbol{I} + \boldsymbol{x}_k\boldsymbol{x}_2^\top \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_k^\top\boldsymbol{x}_1\boldsymbol{I} + \boldsymbol{x}_1\boldsymbol{x}_k^\top & \dots & \dots & \boldsymbol{x}_k\boldsymbol{x}_k^\top + \|\boldsymbol{x}_k\|_2^2\,\boldsymbol{I} \end{bmatrix} \\
\nabla^2_{\boldsymbol{U}_2}\mathcal{L}^* &= 2(\boldsymbol{A} + \boldsymbol{U}_2\boldsymbol{U}_2^\top - \boldsymbol{A}^* - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}) \otimes \boldsymbol{I}_k \\
&\quad + 2\begin{bmatrix} \boldsymbol{y}_1\boldsymbol{y}_1^\top + \|\boldsymbol{y}_1\|_2^2\,\boldsymbol{I} & \boldsymbol{y}_1^\top\boldsymbol{y}_2\boldsymbol{I} + \boldsymbol{y}_2\boldsymbol{y}_1^\top & \dots & \boldsymbol{y}_1^\top\boldsymbol{y}_k\boldsymbol{I} + \boldsymbol{y}_k\boldsymbol{y}_1^\top \\ \boldsymbol{y}_2^\top\boldsymbol{y}_1\boldsymbol{I} + \boldsymbol{y}_1\boldsymbol{y}_2^\top & \boldsymbol{y}_2\boldsymbol{y}_2^\top + \|\boldsymbol{y}_2\|_2^2\,\boldsymbol{I} & \dots & \boldsymbol{y}_2^\top\boldsymbol{y}_k\boldsymbol{I} + \boldsymbol{y}_k\boldsymbol{y}_2^\top \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{y}_k^\top\boldsymbol{y}_1\boldsymbol{I} + \boldsymbol{y}_1\boldsymbol{y}_k^\top & \dots & \dots & \boldsymbol{y}_k\boldsymbol{y}_k^\top + \|\boldsymbol{y}_k\|_2^2\,\boldsymbol{I} \end{bmatrix}
\end{aligned}$$

Note that $\oplus$ denotes the Kronecker sum defined as $\boldsymbol{X} \oplus \boldsymbol{Y} = \boldsymbol{I} \otimes \boldsymbol{X} + \boldsymbol{Y} \otimes \boldsymbol{I}$ where $\otimes$ is the Kronecker product.

Lemma 13. *$\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ if and only if $\boldsymbol{B}_t = \boldsymbol{0}$ for each $t \in [T]$.*

Proof. Since $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$ is a critical point, then plugging Equation (32) into the definition of $\mathcal{L}$ gives that

$$\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = \frac{1}{2}\sum_{t=1}^{T} \|\boldsymbol{B}_t\|_F^2 .$$

Thus $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ if and only if $\boldsymbol{B}_t = \boldsymbol{0} \quad \forall t$. □

Lemma 14. *If $\nabla^2_{\boldsymbol{U}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) \succcurlyeq \boldsymbol{0}$, then the eigenvectors corresponding to the non-zero eigenvalues of $\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top$ are the leading non-negative eigenvectors of $\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \hat{\boldsymbol{A}}$ for all $t \in [T]$.*

Proof. Consider the function $\bar{f}_t(\boldsymbol{U}_t; \hat{\boldsymbol{A}}) = \frac{1}{2}\left\|\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \hat{\boldsymbol{A}} - \boldsymbol{U}_t\boldsymbol{U}_t^\top\right\|_F^2$. $\bar{f}_t$ is simply the *tth* summand in $\mathcal{L}^*$ where $\boldsymbol{A} = \hat{\boldsymbol{A}}$ is fixed and we only consider the variable $\boldsymbol{U}_t$. Minimizing $\bar{f}_t$ is identical to the problem of symmetric matrix factorization.

Using well-known properties of symmetric matrix factorization, since $\nabla \bar{f}_t(\hat{\boldsymbol{U}}_t) = \boldsymbol{0}$, we must have that $\hat{\boldsymbol{U}}_t = \boldsymbol{V}_t\boldsymbol{\Gamma}$ where the columns of $\boldsymbol{V}_t$ are the properly scaled eigenvectors of $\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \hat{\boldsymbol{A}}$ with non-negative eigenvalues where each column has norm equal to the square root of its corresponding eigenvalue, and $\boldsymbol{\Gamma} \in O_k$ is some orthogonal matrix. Further, if the eigenvectors corresponding to the non-zero eigenvalues of $\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top$ are not the leading non-negative eigenvectors, then $\nabla^2 \bar{f}_t(\hat{\boldsymbol{U}}) \not\succcurlyeq \boldsymbol{0}$ by [ZQW20]. Since $\nabla^2 \bar{f}_t(\hat{\boldsymbol{U}}_t)$ is a diagonal block of $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})$, $\nabla^2 \bar{f}_i(\hat{\boldsymbol{U}}_t) \not\succcurlyeq \boldsymbol{0}$ would imply $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) \not\succcurlyeq \boldsymbol{0}$. □

Remark 3. *Without loss of generality, we can assume that the eigenvectors corresponding to the non-zero eigenvalues of $\hat{\boldsymbol{U}}_t\hat{\boldsymbol{U}}_t^\top$ are the leading non-negative eigenvectors of $\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \hat{\boldsymbol{A}}$ for all i.*

Lemma 15. $\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right)\boldsymbol{x} = \left(\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right)\boldsymbol{x}$ *for all* $\boldsymbol{x} \in \operatorname{im}(\hat{\boldsymbol{U}}_1) + \operatorname{im}(\hat{\boldsymbol{U}}_2)$.

Proof. Recall $\boldsymbol{B}_1 = \frac{1}{2}\left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top + \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}\right)$. Then applying first-order stationarity and the fact that $\boldsymbol{B}_2 = -\boldsymbol{B}_1$, we have

$$\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right)\hat{\boldsymbol{U}}_1 = \left(\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right)\hat{\boldsymbol{U}}_1$$
$$\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right)\hat{\boldsymbol{U}}_2 = \left(\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right)\hat{\boldsymbol{U}}_2.$$

□

Corollary 16. $\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top$ *and* $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$ *share an eigenbasis.*

Proof. Using the lemma, any non-zero eigenvector-eigenvalue pair of $\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top$ is also an eigenvector-eigenvalue pair of $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$. Denote the space defined by the span of these eigenvectors as $\boldsymbol{S}$. Then all other eigenvectors of $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$ are orthogonal to $\boldsymbol{S}$, so they are also 0-eigenvectors of $\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top$. Thus the two matrices share an eigenbasis. □

Corollary 17. $\dim\left(\operatorname{im}\hat{\boldsymbol{U}}_1 + \operatorname{im}\hat{\boldsymbol{U}}_2\right) \le 2k-1$, *i.e., the set of columns of* $\hat{\boldsymbol{U}}_1$ *and* $\hat{\boldsymbol{U}}_2$ *are not linearly independent.*

Proof. Assume for contradiction that the vectors in the set $\boldsymbol{S} = \{\hat{\boldsymbol{U}}_1\boldsymbol{e}_i \mid i = 1,\dots,k\} \cup \{\hat{\boldsymbol{U}}_2\boldsymbol{e}_i \mid i = 1,\dots,k\}$ are linearly independent, where $\boldsymbol{e}_i$ is the ith standard basis vector in $\mathbb{R}^k$.

Then note that $\left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top\right)\boldsymbol{x} \neq \boldsymbol{0}$ and $\left(\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}\right)\boldsymbol{x} \neq \boldsymbol{0}$ for all $\boldsymbol{x} \in \boldsymbol{S}$. By Lemma (15), $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top$ and $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ agree for each vector on the $2k$-dimensional space $\operatorname{span}(\boldsymbol{S})$. But, both $\operatorname{rank}(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top), \operatorname{rank}(\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}) \le 2k$ by construction. Then by dimension counting, $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top$ and $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ must send $\operatorname{span}\{\boldsymbol{S}\}^\perp$ to $\boldsymbol{0}$. Thus, $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top$ and $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ agree on the entire basis formed by concatenating basis vectors of $\operatorname{span}\{\boldsymbol{S}\}^\perp$ with those of $\operatorname{span}(\boldsymbol{S})$. This implies that $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top = \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ and thus $\boldsymbol{B}_1 = \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} + \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} = \boldsymbol{0}$. Then $\boldsymbol{B}_2 = -\boldsymbol{B}_1 = \boldsymbol{0}$ so by Lemma 13, $\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 0$ which is a contradiction. □

Lemma 18. $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$ *has exactly* k *positive and* k *negative eigenvalues.*

Proof. First, note that $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ has exactly k positive eigenvalues and $k-d$ eigenvalues of $\boldsymbol{0}$. Then $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - (\boldsymbol{U}_1^*\boldsymbol{e}_1)(\boldsymbol{U}_1^*\boldsymbol{e}_1)^\top$ has rank $k+1$ because of the linear independence of the columns of the combined set of columns $\boldsymbol{U}_1^*$ and $\boldsymbol{U}_2^*$. Further, since we subtract $(\boldsymbol{U}_1^*\boldsymbol{e}_1)(\boldsymbol{U}_1^*\boldsymbol{e}_1)^\top$, we must be accumulating an additional negative eigenvalue relative to $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$. Continuing this process shows that subtracting $(\boldsymbol{U}_1^*\boldsymbol{e}_{j+1})(\boldsymbol{U}_1^*\boldsymbol{e}_{j+1})^\top$ from $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \sum_{t=1}^{j}(\boldsymbol{U}_1^*\boldsymbol{e}_i)(\boldsymbol{U}_1^*\boldsymbol{e}_i)^\top$ contributes exactly one more negative eigenvalue, since $\boldsymbol{U}_1^*\boldsymbol{e}_{j+1}$ can never be written as a linear combination of $\{\boldsymbol{U}_1^*\boldsymbol{e}_1, \dots \boldsymbol{U}_1^*\boldsymbol{e}_k, \boldsymbol{U}_2^*\boldsymbol{e}_1, \dots \boldsymbol{U}_2^*\boldsymbol{e}_j\}$ for $0 < j < k$. The result then follows from induction. □

Lemma 19. $\operatorname{rank}(\hat{\boldsymbol{U}}_1) = \operatorname{rank}(\hat{\boldsymbol{U}}_2) = k$.

Proof. Assume for contradiction that $\operatorname{rank}(\hat{\boldsymbol{U}}_1) = m < k$ without loss of generality. Since by Remark (3) we assume the columns of $\hat{\boldsymbol{U}}_1$ are the leading k non-negative eigenvectors of $\boldsymbol{A}^* + \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \hat{\boldsymbol{A}} = \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \boldsymbol{B}_1$, this must imply that $\boldsymbol{A}^* + \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \hat{\boldsymbol{A}} - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top = -\boldsymbol{B}_1 \preccurlyeq \boldsymbol{0}$.

Plugging in the definition of $\boldsymbol{B}_1$ gives that $\frac{1}{2}\left(\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top + \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}\right) \succcurlyeq \mathbf{0}$. Thus, $\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top \succcurlyeq \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} + \hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} \succcurlyeq \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$. This contradicts the fact from Lemma (18) that $\boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top} - \boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top}$ has k positive eigenvalues. □

With this lemma, we will prove the existence of a direction of $\nabla^2\mathcal{L}^*$ with negative curvature. Instead of directly working with this matrix, we instead use the Schur complement to work with a different form.

Theorem 20. *(Schur Complement) Since $\nabla^2_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) = 2\boldsymbol{I} \succ \mathbf{0}$, $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) \succcurlyeq \mathbf{0}$ if and only if $\nabla^2_{\boldsymbol{U}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) - \left(\nabla_{\boldsymbol{A}}\nabla_{\boldsymbol{U}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right)\left(\nabla^2_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right)^{-1}\left(\nabla_{\boldsymbol{U}}\nabla_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right) \succcurlyeq \mathbf{0}$.*

Define $\boldsymbol{M} = \nabla^2_{\boldsymbol{U}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}}) - \left(\nabla_{\boldsymbol{A}}\nabla_{\boldsymbol{U}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right)\left(\nabla^2_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right)^{-1}\left(\nabla_{\boldsymbol{U}}\nabla_{\boldsymbol{A}}\mathcal{L}^*(\hat{\boldsymbol{A}}, \hat{\boldsymbol{U}})\right)$.
For example, when $k = 2$ and letting $\boldsymbol{U}_1 = [\boldsymbol{x}_1\ \boldsymbol{x}_2]$, $\boldsymbol{U}_2 = [\boldsymbol{y}_1\ \boldsymbol{y}_2]$, we have

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{12}^\top & \boldsymbol{M}_{22} \end{bmatrix},$$

where

$$\boldsymbol{M}_{11} = \begin{bmatrix} 2\boldsymbol{B}_1 + \boldsymbol{x}_1\boldsymbol{x}_1^\top + \|\boldsymbol{x}_1\|_2^2 & \boldsymbol{x}_1^\top\boldsymbol{x}_2\boldsymbol{I} + \boldsymbol{x}_2\boldsymbol{x}_1^\top \\ \boldsymbol{x}_2^\top\boldsymbol{x}_1\boldsymbol{I} + \boldsymbol{x}_1\boldsymbol{x}_2^\top & 2\boldsymbol{B}_1 + \boldsymbol{x}_2\boldsymbol{x}_2^\top + \|\boldsymbol{x}_2\|_2^2 \end{bmatrix}$$

$$\boldsymbol{M}_{12} = \begin{bmatrix} -\boldsymbol{x}_1^\top\boldsymbol{y}_1\boldsymbol{I} - \boldsymbol{y}_1\boldsymbol{x}_1^\top & -\boldsymbol{x}_1^\top\boldsymbol{y}_2\boldsymbol{I} - \boldsymbol{y}_2\boldsymbol{x}_1^\top \\ -\boldsymbol{x}_2^\top\boldsymbol{y}_1\boldsymbol{I} - \boldsymbol{y}_1\boldsymbol{x}_2^\top & \boldsymbol{x}_2^\top\boldsymbol{y}_2\boldsymbol{I} - \boldsymbol{y}_2\boldsymbol{x}_2^\top \end{bmatrix}$$

$$\boldsymbol{M}_{22} = \begin{bmatrix} 2\boldsymbol{B}_2 + \boldsymbol{y}_1\boldsymbol{y}_1^\top + \|\boldsymbol{y}_1\|_2^2 & \boldsymbol{y}_1^\top\boldsymbol{y}_2\boldsymbol{I} + \boldsymbol{y}_2\boldsymbol{y}_1^\top \\ \boldsymbol{y}_2^\top\boldsymbol{y}_1\boldsymbol{I} + \boldsymbol{y}_1\boldsymbol{y}_2^\top & 2\boldsymbol{B}_2 + \boldsymbol{y}_2\boldsymbol{y}_2^\top + \|\boldsymbol{y}_2\|_2^2 \end{bmatrix}$$

For brevity, we do not include the full form of $\boldsymbol{M}$ for general k. However, we can make an easy simplification that will allow for a much cleaner expression.

Using Corollaries (16) and (17), there is an eigenvector $\boldsymbol{z}$ of $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$ with eigenvalue $\lambda \neq 0$ such that $\boldsymbol{z} \in \ker\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right)$. Assume without loss of generality that $\lambda > 0$, and consider $\boldsymbol{\alpha} \in \mathbb{R}^{2k}$. Define the function $g(\cdot\,; \boldsymbol{z}) : \mathbb{R}^{2k} \to \mathbb{R}$ parameterized by $\boldsymbol{z}$ such that $g(\boldsymbol{\alpha}; \boldsymbol{z}) = (\boldsymbol{\alpha} \otimes \boldsymbol{z})^\top \boldsymbol{M} (\boldsymbol{\alpha} \otimes \boldsymbol{z})$, where we partition $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1; \boldsymbol{\alpha}_2]$, $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathbb{R}^k$. Then after some algebra,

$$g(\boldsymbol{\alpha}; \boldsymbol{z}) = \left\|\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}_1 + \hat{\boldsymbol{U}}_2\boldsymbol{\alpha}_2\right\|_2^2 + \lambda\left(\|\boldsymbol{\alpha}_1\|_2^2 - \|\boldsymbol{\alpha}_2\|_2^2\right). \tag{35}$$

We prove the existence of $\boldsymbol{\alpha} \in \mathbb{R}^{2k}, \boldsymbol{x} \in \mathbb{R}^d$ such that $g(\boldsymbol{\alpha}; \boldsymbol{x}) < 0$ considering two different cases. Define $N^- : S_d \to \mathbb{Z}$ as the function that returns the number of strictly negative eigenvalues of its input.

Case 1: $N^-\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) < k$.

Using Corollary (17), we can pick $\boldsymbol{\alpha}$ such that $\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}_1 + \hat{\boldsymbol{U}}_2\boldsymbol{\alpha}_2 = 0$, $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \neq 0$.

Because $N^-\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) < k$, $N^-\left(\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}\right) = k$, and $\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top$ and $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*\top} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*\top}$ share an eigenbasis by Corollary 16, there exists $\boldsymbol{z}^- \in \mathbb{R}^d$ that is a λ^--eigenvector of $\boldsymbol{U}_2^*\boldsymbol{U}_2^{*T} - \boldsymbol{U}_1^*\boldsymbol{U}_1^{*T}$, $\lambda^- < 0$, where $\boldsymbol{z} \in \ker\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right)$

Then for the same choice of $\boldsymbol{\alpha}$,

$$\begin{aligned} \operatorname{sign}\left(g\left(\boldsymbol{\alpha};\boldsymbol{z}\right)\right) &= \operatorname{sign}\left(\|\boldsymbol{\alpha}_1\|_2^2 - \|\boldsymbol{\alpha}_2\|_2^2\right) \\ \operatorname{sign}\left(g\left(\boldsymbol{\alpha};\boldsymbol{z}^-\right)\right) &= \operatorname{sign}\left(\|\boldsymbol{\alpha}_2\|_2^2 - \|\boldsymbol{\alpha}_1\|_2^2\right). \end{aligned}$$

Then if $\|\boldsymbol{\alpha}_1\|_2 \neq \|\boldsymbol{\alpha}_2\|_2$, one of the above expressions is negative and thus $\boldsymbol{M}$ has a negative eigenvalue. This then implies $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}},\hat{\boldsymbol{U}}) \not\succcurlyeq 0$.

Otherwise $\|\boldsymbol{\alpha}_1\|_2 = \|\boldsymbol{\alpha}_2\|_2$. Then $g\left(\boldsymbol{\alpha};\boldsymbol{z}\right) = 0$, but $\nabla_{\boldsymbol{\alpha}_1} g(\boldsymbol{\alpha};\boldsymbol{z}) = \hat{\boldsymbol{U}}_1^\top\left(\hat{\boldsymbol{U}}_1\bar{\boldsymbol{\alpha}}_1 + \hat{\boldsymbol{U}}_2\boldsymbol{\alpha}_2\right) - 2\lambda\boldsymbol{\alpha}_2 = -2\lambda\boldsymbol{\alpha}_2 \neq 0$. Thus $g(\boldsymbol{\alpha};\boldsymbol{z}) = 0$ and $\nabla g(\boldsymbol{\alpha};\boldsymbol{z}) \neq 0$ so there exists $\bar{\boldsymbol{\alpha}}$ in an infinitesimal neighborhood around $\boldsymbol{\alpha}$ where $g(\bar{\boldsymbol{\alpha}}\,;\boldsymbol{z}) < 0$. Thus $\boldsymbol{M}$ has a negative eigenvalue so $\nabla^2\mathcal{L}^*(\hat{\boldsymbol{A}},\hat{\boldsymbol{U}}) \not\succcurlyeq 0$.

Case 2: $N^-\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) = k$.

Define $m = \dim\left(\operatorname{im}(\hat{\boldsymbol{U}}_1) \cap \operatorname{im}(\hat{\boldsymbol{U}}_2)\right)$. By Corollary 17, $m \geq 1$, so we can select orthogonal matrix $\boldsymbol{\Gamma} \in O_k$ such that $\hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\boldsymbol{e}_1 \in \left(\operatorname{im}(\hat{\boldsymbol{U}}_1) \cap \operatorname{im}(\hat{\boldsymbol{U}}_2)\right)$. Define $\boldsymbol{y} = \hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\boldsymbol{e}_1$.

Clearly for any $\boldsymbol{B} \in S_d$ and $\boldsymbol{R} \in S_d^+$, $N^-(\boldsymbol{B}) \geq N^-(\boldsymbol{B}+\boldsymbol{R})$. Then since $N^-\left(-\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) = k$ by Lemma (19), we have that

$$\begin{aligned} k = N^-(-\hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top) &\geq N^-(\boldsymbol{y}\boldsymbol{y}^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top) = N^-\left(\left(\hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\boldsymbol{e}_1\right)\left(\hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\boldsymbol{e}_1\right)^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) \\ &\geq N^-\left(\left(\hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\right)\left(\hat{\boldsymbol{U}}_2\boldsymbol{\Gamma}\right)^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) = N^-\left(\hat{\boldsymbol{U}}_2\hat{\boldsymbol{U}}_2^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) = k, \end{aligned}$$

Thus, $N^-(\boldsymbol{y}\boldsymbol{y}^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top) = k$. But, since $\boldsymbol{y} \in \operatorname{im}(\hat{\boldsymbol{U}}_1)$, $\operatorname{rank}\left(\boldsymbol{y}\boldsymbol{y}^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top\right) = k$. Thus,

$$\boldsymbol{y}\boldsymbol{y}^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top \preccurlyeq 0. \tag{36}$$

Take $\boldsymbol{\alpha}$ such that $\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}_1 = -\boldsymbol{y}$ and $\boldsymbol{\alpha}_2 = \boldsymbol{\Gamma}\boldsymbol{e}_1$. Then

$$\begin{aligned} \boldsymbol{y}_1\boldsymbol{y}_1^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top &= \left(\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}\right)\left(\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}\right)^\top - \hat{\boldsymbol{U}}_1\hat{\boldsymbol{U}}_1^\top & (37) \\ &= \hat{\boldsymbol{U}}_1\left(\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\top - \boldsymbol{I}\right)\hat{\boldsymbol{U}}_1^\top \preccurlyeq 0. & (38) \end{aligned}$$

Therefore $\|\boldsymbol{\alpha}_1\|_2 \leq 1$.

Then $g\left(\boldsymbol{\alpha};\boldsymbol{z}\right) = \left\|\hat{\boldsymbol{U}}_1\boldsymbol{\alpha}_1 + \hat{\boldsymbol{U}}_2\boldsymbol{\alpha}_2\right\|_2^2 + \lambda\left(\|\boldsymbol{\alpha}_1\|_2^2 - \|\boldsymbol{\alpha}_2\|_2^2\right) = \lambda\left(\|\boldsymbol{\alpha}_1\|_2^2 - 1\right) \leq 0.$

If $g\left(\boldsymbol{\alpha};\boldsymbol{z}\right) < 0$, then we are done. Otherwise, $g\left(\boldsymbol{\alpha};\boldsymbol{z}\right) = 0$. Then the same analysis from Case 1 will show that $\nabla g(\boldsymbol{\alpha};\boldsymbol{z}) \neq \boldsymbol{0}$, so there exists $\bar{\boldsymbol{\alpha}}$ in an infinitesimal neighborhood around $\boldsymbol{\alpha}$ where $g(\bar{\boldsymbol{\alpha}}\,;\boldsymbol{z})$ is strictly negative. This then implies our desired result.

B.6 Derivation of Equation (8)

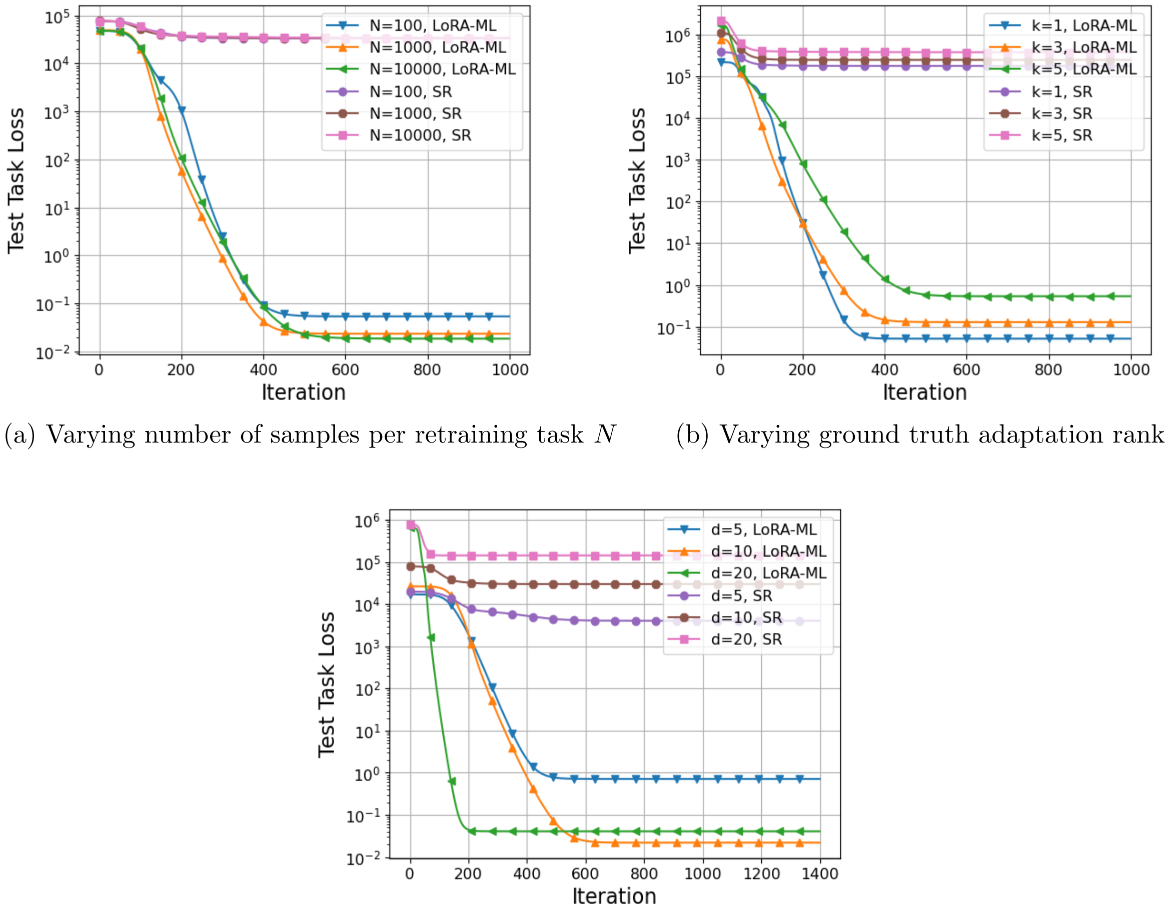
Recall our generative model where each input sample $\boldsymbol{x} \in \mathbb{R}^d$ satisfies $\mathbb{E}\left[\boldsymbol{x}\right] = \boldsymbol{0}$ and $\mathbb{E}\left[\boldsymbol{x}\boldsymbol{x}^\top\right] = \sigma_x^2 \boldsymbol{I}_d$, each noise sample is generated independently of $\boldsymbol{x}$ as $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_\epsilon^2 \boldsymbol{I}_d)$, and $\boldsymbol{y} = (\boldsymbol{A}^* + \boldsymbol{R}_t^*)\boldsymbol{x} + \boldsymbol{\epsilon}$. Then,

$$\begin{aligned}
2\mathbb{E}[\mathcal{L}_t^1(\boldsymbol{A})] &= \mathbb{E}\left[\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2\right] \\
&= \mathbb{E}\left[\|(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A})\boldsymbol{x} + \boldsymbol{\epsilon}\|_2^2\right] \\
&= \mathbb{E}\left[\|(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\boldsymbol{x}\|_2^2 + \|\boldsymbol{\epsilon}\|_2^2 + 2\boldsymbol{\epsilon}^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\boldsymbol{x}\right] \\
&= \mathbb{E}\left[\operatorname{tr}\left(\boldsymbol{x}^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\boldsymbol{x}\right)\right] + \sigma_\epsilon^2 + 2\mathbb{E}\left[\boldsymbol{\epsilon}\right]^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\mathbb{E}\left[\boldsymbol{x}\right] \\
&= \operatorname{tr}\left\{(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\mathbb{E}\left[\boldsymbol{x}\boldsymbol{x}^\top\right]\right\} + \sigma_\epsilon^2 \\
&= \sigma_x^2 \operatorname{tr}\left((\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)^\top(\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t)\right) + \sigma_\epsilon^2 \\
&= \sigma_x^2 \left\|\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t\right\|_F^2 + \sigma_\epsilon^2
\end{aligned}$$

Thus, $\mathbb{E}[\mathcal{L}_t^1(\boldsymbol{A}_t)] = \frac{1}{2}\left(\sigma_x^2 \left\|\boldsymbol{A}^* + \boldsymbol{R}_t^* - \boldsymbol{A}_t\right\|_F^2 + \sigma_\epsilon^2\right)$. Then $\mathbb{E}[\mathcal{L}_t^{n_t}(\boldsymbol{A}_t)] = \mathbb{E}[\mathcal{L}_t^1(\boldsymbol{A}_t)]$ by linearity of expectation, so

$$\frac{1}{\sigma_x^2}\left(\mathbb{E}\left[\mathcal{L}_t^{n_t}(\boldsymbol{A}_t)\right] - \frac{\sigma_\epsilon^2}{2}\right) = \frac{1}{2}\left\|\boldsymbol{A}^* + \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \boldsymbol{A}_t\right\|_F^2$$

C Additional Experiments

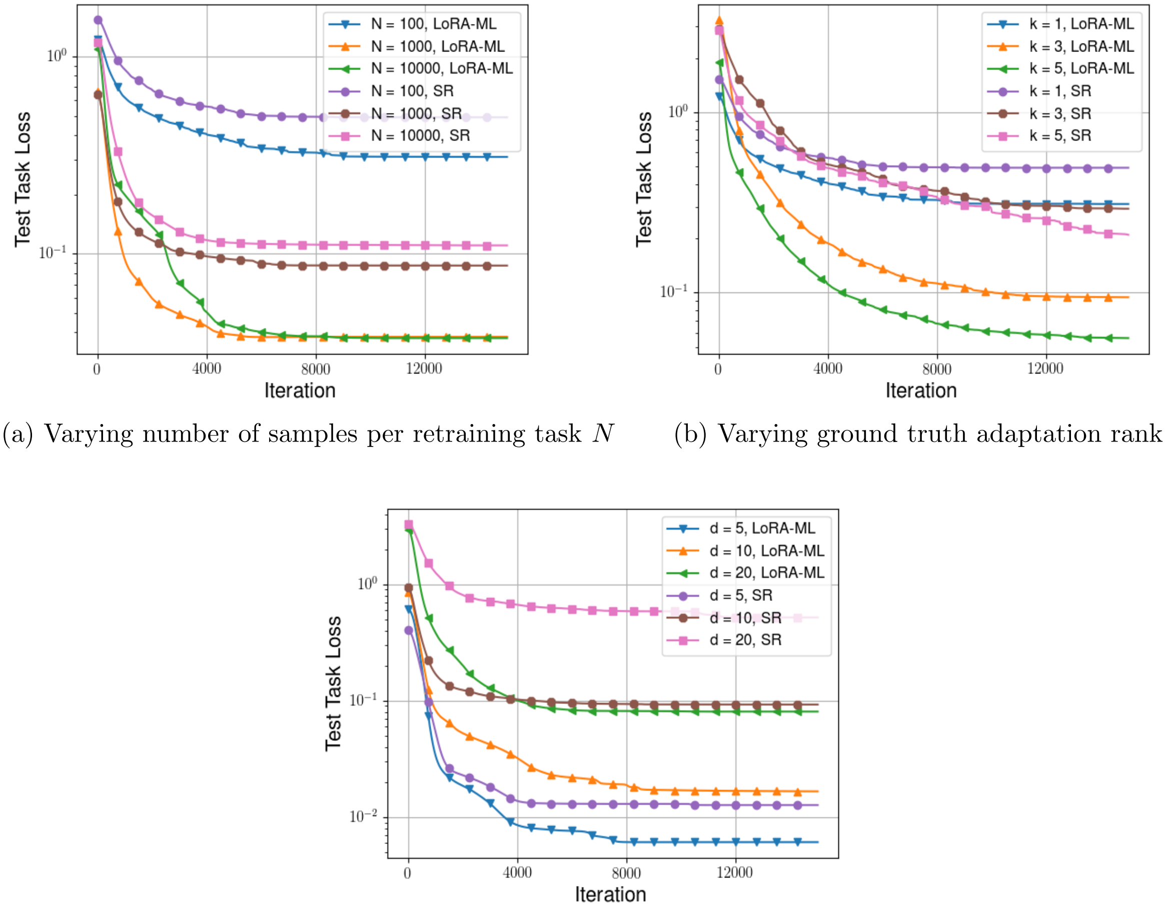


(a) Varying number of samples per retraining task N

(b) Varying ground truth adaptation rank k

(c) Varying ambient dimension d

Figure 3: Linear model fine-tuning performance, additional ablations



(a) Varying number of samples per retraining task N

(b) Varying ground truth adaptation rank k

(c) Varying ambient dimension d

Figure 4: Shallow network fine-tuning performance, additional ablations

C.1 Synthetic Experiment Data Parameters

Experiment	T	n	N	k	d	σ_x	σ_ϵ
Linear, varying T	{2,3,5}	100	5000	1	10	1	.1
Linear, varying n	3	{100,1000,10000}	5000	1	10	1	.1
Linear, varying N	3	100	{100,1000,10000}	1	10	1	.1
Linear, varying k	3	100	1000	{1,3,5}	20	1	.1
Linear, varying d	3	100	5000	1	{5,10,20}	1	.1
Shallow Network, varying T	{2,3,5}	100	1000	1	10	1	.1
Shallow Network, varying n	3	{100,1000,10000}	1000	1	10	1	.1
Shallow Network, varying N	3	100	{100,1000,10000}	1	10	1	.1
Shallow Network, varying k	3	100	1000	{1,3,5}	10	1	.1
Shallow Network, varying d	3	100	1000	1	{5,10,20}	1	.1

Table 2: Synthetic Data Parameters

Hyperparameter	**Standard Retraining**	**Meta-LoRA-8**	**Meta-LoRA-16**
Learning Rate	5e-5	3e-5	5e-5
Learning Rate Schedule	Linear	Linear	Linear
Batch Size	6	4	4
Epochs	30	30	30
Optimizer	AdamW	AdamW	AdamW
LoRA Rank	N/A	8	16
LoRA Dropout	N/A	0.1	.1
LoRA Alpha	N/A	16	16

Table 3: Retraining Hyperparameters

D LLM Training Hyperparameters

Hyperparameter	**Rank-k LoRA Fine-Tuning**
Learning Rate	3e-5
Learning Rate Schedule	Linear
Batch Size	6
Epochs	30
Optimizer	AdamW
LoRA Rank	k
LoRA Dropout	.1
LoRA Alpha	16

Table 4: Rank-k LoRA Fine-Tuning Hyperparameters, $k \in \{8, 16\}$

D.1 Note on Number of Trainable Parameters

For simplicity assume our model architecture consisted of m layers, where each layer was parameterized by a $d \times d$ matrix, and we use rank-k adaptations for each layer for our Meta-LoRA objective, where $k \ll d$. Then the standard retraining method uses md^2 trainable parameters, while minimizing the Meta-LoRA objective uses $m(d^2 + 2kdT)$ trainable parameters. Although Meta-LoRA uses some additional parameters, since k is small relative to d and we work in the setting where $k(T+1) < d$, asymptotically $m(d^2 + 2kdT) = O(md^2)$ so the increase in trainable parameters is minor. After running either of these retraining procedures, the fine-tuning stages are identical and require the same number of trainable parameters no matter which retraining procedure was run.

E Theory Notes

E.1 Non-Uniqueness of Global Min for $T = 2$

Consider $T = 2$, $k = 1$, $d = 2$, $\boldsymbol{A}^* = \boldsymbol{0}$, and $\boldsymbol{u}_t^* = \boldsymbol{e}_t$ for $t = 1, 2$, where $\boldsymbol{e}_t$ is the t_{th} standard basis vector. Clearly the ground truth perturbations $\boldsymbol{u}_i^*$ are orthonormal and thus linearly independent. The set of global minima of $\mathcal{L}^*$ are $(\boldsymbol{A}, \boldsymbol{U})$ such that $\boldsymbol{A} = \frac{1}{T}\sum_{t=1}^{T}\left(\boldsymbol{u}_t^*\boldsymbol{u}_t^{*\top} - \boldsymbol{u}_t\boldsymbol{u}_t^\top\right)$ and $\boldsymbol{u}_t\boldsymbol{u}_t^\top -$

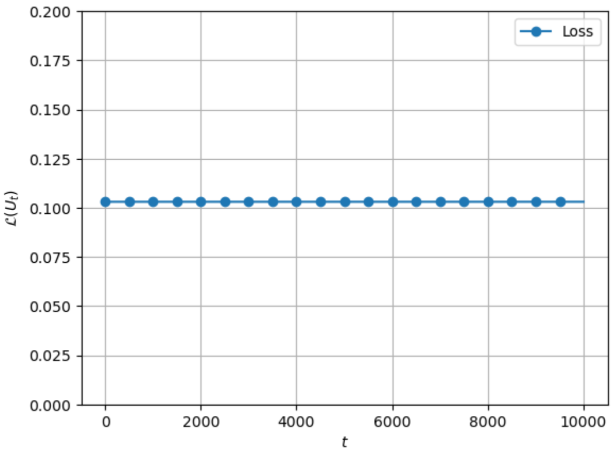


Figure 5: Loss does not decrease near these spurious local minima

$\boldsymbol{u}_t^* \boldsymbol{u}_t^{*\top} - \frac{1}{T}\sum_{s=1}^{T}\left(\boldsymbol{u}_s\boldsymbol{u}_s^\top - \boldsymbol{u}_s^*\boldsymbol{u}_s^{*\top}\right) = \boldsymbol{0}$. It is not hard to see that a global minimum follows from any set values of $\boldsymbol{u}_1, \boldsymbol{u}_2$ such that $\boldsymbol{u}_1\boldsymbol{u}_1^\top - \boldsymbol{u}_2\boldsymbol{u}_2^\top = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. When properly parameterized, this system of equations defines a hyperbola where each point corresponds to a global minimum of $\mathcal{L}^*$.

E.2 Spurious Local Minima

We observe that for $T \geq 3$, for certain tasks $\boldsymbol{U}^* = (\boldsymbol{U}_1^*, \boldsymbol{U}_2^*, \boldsymbol{U}_3^*)$, it is possible to find points $\boldsymbol{U}$ that are local minima, but not global minima. To find these points, we sample true tasks $\boldsymbol{U}^*$ from a normal distribution and use a numerical solver to find zeros of the gradient of the reduced loss

$$\hat{\mathcal{L}}(\boldsymbol{U}) = \sum_{t=1}^{T}\left\|\boldsymbol{U}_t\boldsymbol{U}_t^\top - \boldsymbol{U}_t^*\boldsymbol{U}_t^{*\top} - \frac{1}{T}\sum_{s=1}^{T}(\boldsymbol{U}_s\boldsymbol{U}_s^\top - \boldsymbol{U}_s^*\boldsymbol{U}_s^{*\top})\right\|_F^2 .$$

Through the Schur complement argument used to prove Theorem 11, we can see that $\hat{\mathcal{L}}$ has a spurious local minimum only if $\mathcal{L}$ has a spurious local minimum.

Typically, these zeros are close to the global minimum. Occasionally, it is possible to find a point $\hat{\boldsymbol{U}}$ with gradients close to 0 and with positive definite Hessians. We then confirm that these are close to the spurious local minimum through the following argument.

Consider the function

$$r(\boldsymbol{U}) = \text{vec}(\boldsymbol{U} - \hat{\boldsymbol{U}})^\top \text{vec}(\nabla\hat{\mathcal{L}}(\boldsymbol{U})).$$

Clearly, there is a minimum of $\hat{\mathcal{L}}$ in the δ-ball of $\hat{\boldsymbol{U}}$ if $r(\boldsymbol{U}) > 0$ for all $\boldsymbol{U}$ on the boundary of the δ-ball. As r is continuous, if for some small enough $\epsilon, \gamma > 0$ if $r(\boldsymbol{U}) > \gamma > 0$ for all $\boldsymbol{U}$ on the ϵ-net of the boundary of the δ-ball, then there exists a spurious local minimum in the δ-ball around $\hat{\boldsymbol{U}}$. Numerically, such points and ϵ, δ, and γ can be found which would imply that spurious local minima exist, barring any errors due to numerical computation. To confirm, we run gradient descent from this point and observe that the loss stays constant.

F Example Pseudocode for Minimizing (5)

Algorithm 1 Meta-Adapter Training

```
Input: Tasks $\mathcal{T}_t$, $t \in [T]$, learning rate $\eta$, number of epochs $N_e$, batches per epoch $N_b$
Initialize: Model parameters $\boldsymbol{W}_0, \boldsymbol{\theta}_0^{(t)}$ for all $t = 1, \ldots, T$
for epoch $e = 1$ to $N_e$ do
  for $b = 1, \ldots, N_b$ do
    for $t = 1, \ldots, T$ do
      Load next batch $\beta_{t,b}$ from $\mathcal{T}_i$
      Compute gradient $\boldsymbol{g}^{(t)} = \nabla_{\boldsymbol{W}, \boldsymbol{\theta}^{(t)}} \Big( \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \beta_{t,b}} \mathcal{L}( \big( \Phi_{\mathrm{FT}} \left( \boldsymbol{x}\, ; \boldsymbol{W}, \boldsymbol{\theta}^{(t)} \right), \boldsymbol{y} \big) \Big)$
      Update adapter parameters: $\boldsymbol{\theta}_{e+1}^{(t)} \leftarrow \boldsymbol{\theta}_e^{(t)} - \eta_e \boldsymbol{g}_{\boldsymbol{\theta}^{(t)}}$
    end for
    Update base parameters: $\boldsymbol{W}_{e+1} \leftarrow \boldsymbol{W}_e - \eta_e \sum_{t=1}^{T} \boldsymbol{g}_{\boldsymbol{W}}^{(t)}$
  end for
end for
```