Balanced LoRA: Removing Parameter Invariance to Accelerate Convergence

Anonymous Authors[1]

Abstract

Low-Rank Adaptation (LoRA) is the most widely adopted method for fine-tuning large language models. Notably, LoRA is inherently overparameterized: multiple pairs of low-rank factors can yield the same adapted weight matrix. We show—both theoretically and empirically—that these pairs exhibit significantly different condition numbers. As a result, converging to different loss minimizers directly impacts the convergence rate of LoRA. Building on this observation, we introduce Balanced Low-Rank Adaptation (BaLoRA), a variant of LoRA that projects iterates onto a balanced manifold. This manifold improves the conditioning of the loss landscape while preserving the adapted matrix. The projection step is computationally lightweight and integrates seamlessly into existing fine-tuning pipelines. Empirically, BaLoRA converges faster than standard LoRA and achieves superior performance across a range of fine-tuning tasks.

1. Introduction

Pretrained foundation models are now ubiquitous in natural language processing (Brown et al., 2020; Qin et al., 2023; Taori et al., 2023), computer vision (Awais et al., 2025), and multimodal learning (Li et al., 2022; Liu et al., 2023a), thanks to their ability to generalize from large-scale, diverse training data. Their massive number of parameters allows them to capture a wide range of patterns, making them ideal bases for building specialized fine-tuned models on task-specific data. However, as model sizes expand, full fine-tuning (updating all parameters) becomes impractical due to its prohibitive computational and memory costs.

To address this issue, parameter-efficient fine-tuning (PEFT) methods have become increasingly popular (Houlsby et al., 2019). They leverage the observation that overparameterized models often exhibit low intrinsic dimensionality (Li et al., 2018; Aghajanyan et al., 2021). These methods adapt large pretrained models by updating only a small subset of parameters, reducing computational costs while preserving performance. Among them, Low-Rank Adaptation (LoRA) (Hu et al., 2022) stands out as one of the most effective approaches. Rather than updating dense weight matrices, LoRA uses trainable low-rank matrices added to the frozen pretrained weights. Specifically, a pretrained weight matrix $W \in \mathbb{R}^{a\times b}$ is updated as $W + AB$, where $A \in \mathbb{R}^{a\times r}$, $B \in \mathbb{R}^{r\times b}$, and $r \ll \min(a, b)$ is the LoRA rank. By freezing W, this approach reduces the number of trainable parameters from $a \times b$ (full fine-tuning) to $r \times (a + b)$, achieving substantial memory savings while preserving adaptability.

Given its empirical success across diverse applications, LoRA has recently sparked theoretical interest, though still in its early stages. Recent studies have explored its expressivity in feedforward neural networks and transformers (Zeng & Lee, 2024), analyzed its fine-tuning dynamics in the Neural Tangent Kernel (NTK) regime (Malladi et al., 2023; Jang et al., 2024), examined the distinct roles of the A and B matrices (Zhu et al., 2024; Hayou et al., 2024b), and investigated the effects of various initialization strategies (Hayou et al., 2024a; Li et al., 2025).

In this paper, we analyze the asymptotic behavior of training dynamics of LoRA. Specifically, we bound the asymptotic convergence rate of LoRA when fine-tuning one layer of a deep, possibly non-linear, neural network, by establishing tight bounds on the loss condition number at convergence. Our study hinges on a key property of LoRA: its overparameterization. More precisely, for any invertible matrix $R \in \mathbb{R}^{r\times r}$, the low-rank factors $(AR, R^{-1}B)$ yield the same adapter AB. In particular, if (A, B) is a minimizer of the loss, then every $(AR, R^{-1}B)$ is also a minimizer, which yields a continuous manifold of minimizers. With that in mind, our novel theoretical analysis highlights that the conditioning of the loss can vary along this manifold: some minimizers are flatter than others, which, from an optimization perspective, makes them better candidates to converge to. Indeed, the loss around a flatter minimizer is

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

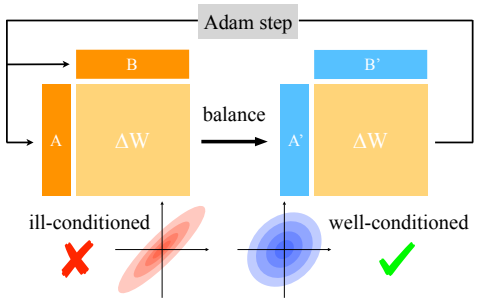

Figure 1. **BaLoRA in a nutshell.** BaLoRA projects the low-rank adapters (A, B) on the balanced manifold after each optimizer step. This projection improves the conditioning of the loss while preserving the product $\Delta W = AB = A'B'$.

better conditioned, so that the asymptotic convergence rate to the minimizer is faster.

Our analysis identifies that *balanced minimizers*—minimizers (A, B) satisfying $A^\top A = BB^\top$—achieve optimal conditioning of the loss. Although this balance condition has been studied in other contexts, such as linear networks (Nguegnang et al., 2024), matrix factorization (Ye & Du, 2021; Ghosh et al., 2025), and conservation laws in neural networks (Marcotte et al., 2023), it has not been explored for LoRA. Leveraging these insights, we introduce BaLoRA, an extension of LoRA that enforces balance during training to improve conditioning and accelerate convergence. BaLoRA is computationally lightweight, theoretically grounded, and compatible with standard optimization pipelines. Our contributions are the following:

- We introduce **BaLoRA**, a novel PEFT method that enforces balanced low-rank adapters throughout optimization with negligible computational overhead. (Section 3)
- We theoretically analyze the conditioning of LoRA's limiting points when fine-tuning one layer of a deep, possibly non-linear network, proving that balanced minimizers exhibit optimal conditioning. Consequently, BaLoRA converges to a better-conditioned minimizer than LoRA, improving its asymptotic convergence rate. (Section 2)
- When optimizing with gradient descent, we demonstrate that BaLoRA iterations can be reformulated as an intrinsic optimization scheme on the product AB, which provides an elegant and more interpretable geometric perspective on this algorithm. (Section 3)
- In our experiments on a range of large language models and datasets, BaLoRA consistently outperforms LoRA and matches or surpasses several state-of-the-art LoRA variants from the literature with negligible computational overhead. (Section 4)

1.1. Related Works

Parameter-efficient fine-tuning. To enable efficient adaptation without full retraining, residual adapters were first introduced for computer vision tasks (Rebuffi et al., 2017) and later extended to NLP through adapter-based transfer learning (Houlsby et al., 2019). Other prominent PEFT strategies include pruning-based approaches, such as Diff-Pruning (Guo et al., 2020), and low-rank adapters (Hu et al., 2022). LoRA and its variants have been widely adopted, ranging from bridging language models with non-language tasks via LIFT (Dinh et al., 2022) to fine-tuning image generation models (Fan et al., 2023). Theoretical analyses of LoRA in the Neural Tangent Kernel (NTK) regime have been conducted (Malladi et al., 2023; Jang et al., 2024), while its expressive power has been examined (Zeng & Lee, 2024).

LoRA optimization. LoRA is typically optimized using Adam (Kingma & Ba, 2017) or AdamW (Loshchilov & Hutter, 2017). Recent works have sought to adapt optimization strategies to low-rank structures. Riemannian approaches (Bogachev et al., 2025; Mo et al., 2025) tackle overparameterization through manifold-based optimization, but often require specialized solvers. Alternative algorithms for matrix factorization have been explored to strengthen convergence guarantees: Zhang & Fan (2024) analyze projected gradient descent and demonstrate convergence rates independent of condition numbers under specific assumptions; Ward & Kolda (2023) study alternating gradient descent, deriving bounds tied to spectral gaps; Zhang & Pilanci (2024) enhance gradient updates with Riemannian preconditioners; and Olikier et al. (2025) introduce Gauss–Southwell descent methods, emphasizing step-size and balancing interactions. Although matrix factorization shares similarities with LoRA, these studies do not address the fine-tuning of pretrained machine learning models.

Initialization and convergence dynamics. Standard LoRA initialization sets one low-rank matrix to zero and the other to Gaussian noise, ensuring the model initially retains the behavior of the pretrained model while enabling low-rank adaptations during training. The initial update A_0B_0 is scaled by a factor α/r, where α is a hyperparameter (Hu et al., 2022). Rank-stabilized scaling can be applied to mitigate gradient collapse at higher ranks (Kalajdzievski, 2023). The convergence dynamics of LoRA are closely tied to results on deep linear networks and matrix factorization. For small step sizes, gradient descent converges under balancing conditions (Nguegnang et al., 2024), which become exact conservation laws of the gradient flow in the vanishing step size limit, thereby explaining implicit biases (Marcotte et al., 2023). Random initialization can guarantee global convergence in asymmetric low-rank matrix factorization (Ye & Du, 2021). Large step sizes, however, may push training

toward the edge of stability (Cohen et al., 2021), a phenomenon extensively studied in linear networks (Ghosh et al., 2025; Chen & Bruna, 2023). For LoRA specifically, (Hayou et al., 2024b) propose assigning different learning rates to the low-rank factors to improve efficiency, while (Xu et al., 2025) analyze its dynamics through a gradient flow perspective, revealing an initial alignment phase followed by local convergence for small initialization scales.

Structural constraints. Low-rank models are inherently overparameterized, but structural constraints can mitigate their inefficiencies. Orthogonality has been explored for optimization on the Stiefel manifold (Park et al., 2025) and in QR-based initialization (OLoRA (Büyükakyüz, 2024)). Other approaches exploit richer decompositions: DoRA (Liu et al., 2024) decomposes weights into magnitude and direction; butterfly-based orthogonal fine-tuning (BOFT (Liu et al., 2023b)) and Householder reflection adaptation (HRA (Yuan et al., 2024)) leverage structured orthogonal parameterizations; SVFT (Lingam et al., 2024) utilizes singular vectors of pretrained weights; VeRA (Kopiczko et al., 2023) reduces parameters by sharing low-rank random matrices with compact scaling vectors; GOAT (Fan et al., 2025) employs SVD-structured priors with mixture-of-experts alignment to refine initialization and scaling; and LoRA Done RITE (Yen et al., 2024) enforces invariance of the optimization process under scaling and rotation transformations of adapters.

2. Balanced Minimizers Are Best Conditioned

In this section, we analyze theoretically the asymptotic convergence rate of LoRA while fine-tuning one weight matrix of a general deep neural network (e.g., a Transformer (Vaswani et al., 2023)).

For simplicity, we assume that LoRA is trained with gradient descent, as Adam (Kingma & Ba, 2017) lacks convergence guarantees even for simple convex quadratic objectives. Denote $f(A, B) \in \mathbb{R}$ the loss to optimize. We assume throughout this section that f takes the form of a regression loss $f(A, B) \coloneqq \frac{1}{2}\|h(AB) - Z\|_F^2$, where Z is the target matrix, $\|\cdot\|_F$ is the Frobenius norm and $h(AB) \in \mathbb{R}^{d\times n}$ is the output of a generic, possibly non-linear neural network.

LoRA iterations with step size γ and initialization (A_0, B_0) read

$$\begin{cases} A_{t+1} = A_t - \gamma \nabla_{A_t} f(A_t, B_t), \\ B_{t+1} = B_t - \gamma \nabla_{B_t} f(A_t, B_t). \end{cases} \tag{1}$$

When f is the quadratic loss over a linear neural network, iterations (1) are known to converge to a minimizer (A, B) of the loss (Nguegnang et al., 2024), with an unknown convergence rate. To gain insights on the convergence rate of f to its optimum, we focus on the condition number $\kappa \coloneqq \kappa(f)(A, B)$ of f at a minimizer (A, B). Letting H be the Hessian of f at (A, B), it is defined as $\kappa \coloneqq \lambda_{\max}(H)/\lambda_{\min\neq 0}(H)$, where $\lambda_{\min\neq 0}(H)$ is the smallest non-zero eigenvalue of H. The following classical result shows that a smaller condition number implies faster asymptotic convergence (*e.g.*, Bach (2024)).

Lemma 2.1. *Assume the iterations (1) converge to a minimizer (A, B) of f. Denoting H the Hessian of f at (A, B), let $L \coloneqq \lambda_{\max}(H)$ and $\mu \coloneqq \lambda_{\min\neq 0}(H)$. Then, $\limsup_{t\to+\infty} \frac{f(A_{t+1},B_{t+1})-f(A,B)}{f(A_t,B_t)-f(A,B)} \leq \max((1-\gamma\mu)^2, (1-\gamma L)^2)$. Taking $\gamma = 2/(L+\mu)$ to minimize the right-hand side, and denoting $\kappa \coloneqq L/\mu$, $\limsup_{t\to+\infty} \frac{f(A_{t+1},B_{t+1})-f(A,B)}{f(A_t,B_t)-f(A,B)} \leq \left(\frac{\kappa-1}{\kappa+1}\right)^2$. Hence, the smaller $\kappa \geq 1$, the faster the convergence to (A, B) asymptotically.*

It is therefore crucial to understand the condition number of the different minimizers of f. Losses of the form $f(A, B) = \frac{1}{2}\|h(AB) - Z\|_F^2$ have an r^2-dimensional manifold $\mathcal{M}$ of minimizers as, for any minimizer (A, B), the couple $(AR, R^{-1}B)$ for R an invertible $r \times r$ matrix leads to the same adapter AB, so is also a minimizer of f. Among such minimizers, we single out the submanifold $\mathcal{B}_{\min}$ of *balanced minimizers*, defined as all couples $(A, B) \in \mathcal{M}$ satisfying the *balancing condition* $A^\top A = BB^\top$. We prove in the next subsections that balanced minimizers are optimally conditioned, and discuss the balancing condition in Section 3.

2.1. The One-Layer Linear Case

We start with a tractable setting that still captures the complexity of the problem. Consider a pre-trained one-layer linear network $W \in \mathbb{R}^{a\times b}$ and a *target* $W^\star \in \mathbb{R}^{a\times b}$, representing the ideal fine-tuned model (Zeng & Lee, 2024). Let $Z \coloneqq W^\star - W$ be the gap between the pretrained and target models. LoRA then consists in minimizing the loss

$$f\colon (A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b} \mapsto \frac{1}{2}\|Z - AB\|_F^2 . \tag{2}$$

This setup generalizes matrix factorization (Ye & Du, 2021; Ghosh et al., 2025), where $\operatorname{rk} Z = r$, by allowing Z to have rank $\operatorname{rk} Z > r$, which, to the best of our knowledge, has not been studied before. As detailed in the following paragraphs, having $\operatorname{rk} Z > r$ makes the mathematical analysis significantly more involved, as the Hessian of f—which has to be computed and diagonalized to investigate the conditioning of f—becomes the sum of two terms whose codiagonalization is non-trivial, while there is only one term in the matrix factorization case.

The matrix factorization case. Let us first assume that $\operatorname{rk} Z = r$. Then, the problem reduces to matrix factorization, and the minimal value of the loss f is zero. We explicitly

compute the Hessian of f at any minimizer (A, B), and determine its full spectrum and corresponding condition number.

Proposition 2.2. *Let $(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b}$ be a global minimizer of the loss f (2). Assume $\operatorname{rk} Z = r$. The Hessian of f at (A, B) reads*

$$H = \begin{pmatrix} (BB^\top) \otimes I_a & B \otimes A \\ B^\top \otimes A^\top & I_b \otimes (A^\top A) \end{pmatrix},$$

and its eigenvalues are: 0 (with multiplicity r^2), $\sigma_i(A)^2 + \sigma_j(B)^2$ for $1 \le i, j \le r$ (each with multiplicity 1), $\sigma_i(A)^2$ (each with multiplicity $b - r$), and $\sigma_j(B)^2$ (each with multiplicity $a - r$). Therefore, $\kappa(f)(A, B) = (\sigma_1(A)^2 + \sigma_1(B)^2)/\min(\sigma_r(A)^2, \sigma_r(B)^2)$.

Proposition 2.2, proved in Section A.1, establishes a direct link between the condition number of a minimizer (A, B) and the singular values of A and B, which allows us to identify optimally conditioned minimizers.

Proposition 2.3. *Assume $\operatorname{rk} Z = r$. Then, all balanced minimizers (i.e., such that $A^\top A = BB^\top$) have the minimal condition number $\kappa_{\min} = 2\sigma_1(Z)/\sigma_r(Z)$ among all minimizers.*

Combining Proposition 2.3 with Proposition 2.1 yields a closed-form connection between the best asymptotic convergence rate of the dynamics (1) and the spectrum of the target matrix Z. In particular, it shows that a target with a more spread-out spectrum corresponds to a more challenging matrix factorization problem. Furthermore, our analysis shows that balanced minimizers achieve optimal (i.e., minimal) conditioning, making them ideal limiting points for fast asymptotic convergence. These explicit quantitative connections between spectral structure, conditioning, and convergence rates represent novel insights into the matrix factorization problem.

The general case. We now investigate how our insights for matrix factorization extend to the general case $\operatorname{rk}, Z \ge r$. This scenario, where the adapters have a lower rank than the target, reflects the typical case of low-rank adaptation. Here, the residual $AB - Z \neq 0$ introduces additional off-diagonal terms in the Hessian.

Proposition 2.4. *Let $(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b}$ be a global minimizer of the loss f (2). Assume $\operatorname{rk} Z \ge r$. The Hessian of f at (A, B) reads*

$$H = \begin{pmatrix} (BB^\top) \otimes I_a & B \otimes A \\ B^\top \otimes A^\top & I_b \otimes (A^\top A) \end{pmatrix} + \begin{pmatrix} 0 & (I_r \otimes (AB - Z))K_{r,b} \\ ((AB - Z)^\top \otimes I_r)K_{a,r} & 0 \end{pmatrix}$$

where $K_{k,\ell}$ is the $k\ell \times k\ell$ matrix such that $\operatorname{vec}(X^\top) = K_{k,\ell}\operatorname{vec}(X)$ for any $X \in \mathbb{R}^{k\times\ell}$, with vec the vectorization operator.

One can verify that H is symmetric, since $(I_r \otimes (AB - Z))K_{r,b}(x \otimes y) = (I_r \otimes (AB - Z))(y \otimes x) = y \otimes (AB - Z)x = K_{a,r}^\top((AB - Z)x \otimes y)$, as $K_{a,r}^\top = K_{r,a}$. The second term in H, that has positive and negative eigenvalues, makes the characterization of the conditioning of H more challenging, especially to lower bound $\lambda_{\min\neq 0}$. Below, we compute the sharpness of the Hessian at a minimizer and provide two bounds on its smallest eigenvalue. The proof is detailed in Section A.2.

Proposition 2.5. *Let $(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b}$ be a global minimizer of the loss f (2). The largest eigenvalue of the Hessian of f at (A, B) is $\lambda_{\max}(H) = \sigma_1(A)^2 + \sigma_1(B)^2$. Moreover, the smallest non-zero eigenvalue of H satisfies,*

$$\lambda_{\min\neq 0}(H) \ge \min(\sigma_r(A)^2, \sigma_r(B)^2) - \sigma_{r+1}(Z), \quad (3)$$

$$\lambda_{\min\neq 0}(H) \le \min(\sigma_r(A)^2, \sigma_r(B)^2)\,. \quad (4)$$

Finally, if (A, B) is balanced, the lower bound (3) is maximized*, equal to $\sigma_r(Z) - \sigma_{r+1}(Z)$, and becomes an* equality*: $\lambda_{\min\neq 0}(H) = \sigma_r(Z) - \sigma_{r+1}(Z)$.*

Compared to matrix factorization, here balanced minimizers are still optimally conditioned, but the key quantity governing the intrinsic hardness of LoRA optimization shifts from $\sigma_r(Z)$ to the r-spectral gap $\sigma_r(Z) - \sigma_{r+1}(Z)$. This gap quantifies how well the rank-r approximation separates from the discarded directions. The smaller the gap (and the larger $\sigma_1(Z)$), the slower the asymptotic convergence of the iterations (1) in the best case.

2.2. The Deep Non-Linear Case

The theoretical analysis in the previous section focuses on a simplified toy model (2). While a comprehensive understanding of the Hessian conditioning for deeper architectures remains challenging, we can still gain insights into why balancing constraints improve conditioning—specifically in the case of fine-tuning a single-layer adapter in the interpolating regime (where the minimum loss reaches zero).

Consider the regression loss $f(A, B) := \frac{1}{2}\|h(AB) - Z\|^2$, where Z is the target matrix, $\|\cdot\|_F$ is the Frobenius norm and $h(AB) \in \mathbb{R}^{d\times n}$ is the output of a generic, possibly deep and non-linear neural network, with n fixed inputs—h is seen as a function of the LoRA adapter AB. For instance, for a 2-layer MLP with weights (V, W) for which we fine-tune only the hidden-layer matrix W, one has $h(AB) := V\,\mathrm{ReLU}((W + AB)X)$, where $X \in \mathbb{R}^{b\times n}$ is the data matrix.

Assuming $nd \ge ab$ (a condition satisfied when sufficient data is available), the Jacobian $\partial h(AB)$ is a rectangular and injective matrix. We define its conditioning as $\kappa(\partial h(AB)) := \kappa(\partial h(AB)^\top\, \partial h(AB))^{1/2}$. The following Proposition, proved in Appendix A.3, provides a bound on the conditioning of the loss at a minimizer in the inter-

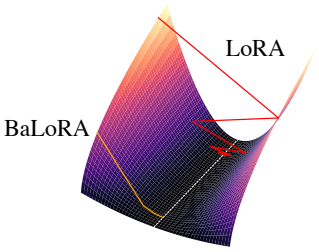

Figure 2. **The intuition behind BaLoRA.** By constraining the adapters to be balanced along the fine-tuning iterations, BaLoRA converges to balanced, and therefore, optimally conditioned, minimizer, reaching faster asymptotic convergence rates.

polation regime. It generalizes Proposition 2.2, which is recovered as a special case when $h = \mathrm{Id}$.

Proposition 2.6. *Let (A, B) be a minimizer of the loss $f(A, B) := \frac{1}{2}\|h(AB) - Z\|^2$, such that $h(AB) = Z$. One has the following upper-bound on the conditioning of f at (A, B):*

$$\kappa(f)(A, B) \leq \kappa(\partial h(AB))^2 \, \frac{\sigma_1(A)^2 + \sigma_1(B)^2}{\min(\sigma_r(A)^2, \sigma_r(B)^2)}.$$

Moreover, taking (A, B) to be balanced minimizes the upper-bound.

Proposition 2.6 shows that balancing the singular values of (A, B) minimizes the upper-bound on the conditioning at an interpolating point, which suggests an improved conditioning of the loss. Extending this analysis to the non-interpolating regime is an avenue for future work, and would bridge the gap with Proposition 2.5, which does not require interpolation but assumes $h = \mathrm{Id}$.

The results in this section suggest that steering the dynamics of (1) toward balanced adapters—whether through explicit or implicit regularization—can accelerate training in practice. Building on this insight, we propose a fine-tuning strategy that guides LoRA along balanced adapters, leading to faster convergence and improved stability in practice.

3. BaLoRA: Balanced Low-Rank Adaptation

Our results in Section 2 imply that balanced minimizers $(A, B) \in \mathcal{B}$ attain the optimal condition number, where $\mathcal{B} = \{(A, B) \mid A^\top A = BB^\top\}$ is the *balanced manifold* (Du et al., 2018). Building on this new insight, we thus propose to constrain LoRA iterations to stay on $\mathcal{B}$ by projecting the iterates after each optimizer (e.g., gradient or AdamW) step, to promote convergence to a better-conditioned minimizer with faster asymptotic rates (Figure 2).

As exposed in the following subsection, we introduce a submanifold $\mathcal{H} \subset \mathcal{B}$ of *hyperbalanced* matrices, which provides a more structured and efficient parameterization. We call Balanced Low-Rank Adaptation (BaLoRA) the novel fine-tuning method obtained by projecting to $\mathcal{H}$. Note that in the remainder, we use the term "manifold" with a slight abuse of language, since the sets we consider are manifolds with boundary.

3.1. Hyperbalanced Manifold and Balancing Map

Let $\mathbb{D}_+^r$ denote the set of $r \times r$ non-negative diagonal matrices with non-increasing diagonal values. We consider the submanifold (with boundary) $\mathcal{H} \subset \mathcal{B}$, which we call the *hyperbalanced manifold*

$$\mathcal{H} = \{(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b} \mid \exists\, S \in \mathbb{D}_+^r \text{ s.t. } A^\top A = BB^\top = S\}.$$

As detailed in Theorem B.1, one has the equivalent description of $\mathcal{H}$,

$$\mathcal{H} = \{\big(US^{1/2},\, S^{1/2}V\big) \mid U^\top U = VV^\top = I_r,\; S \in \mathbb{D}_+^r\}. \tag{5}$$

This reformulation has two main consequences.

Consequence 1: optimizing on $\mathcal{H}$ is equivalent to optimizing over low-rank matrices. Denoting $\mathcal{N}_r$ the set of rank-r matrices, (5) shows that $(A, B) \in \mathcal{H} \mapsto X := AB \in \mathcal{N}_r$ is surjective. For a function $g(X)$, define $f(A, B) := g(AB)$. Then, $\min_{X\in\mathcal{N}_r} g(X)$ and $\min_{(A,B)\in\mathcal{H}} f(A, B)$ are equivalent problems. Working with variables $(A, B) \in \mathcal{H}$ therefore provides a computationally convenient framework for low-rank optimization, while also taking advantage of the improved conditioning discussed in the previous section.

Consequence 2: definition of the balancing map P "projecting" onto $\mathcal{H}$. Given (A, B) with $X = AB$, take any reduced SVD with decreasing singular values $X = U\,S\,V^\top$. The *balancing map* is defined as

$$P(A, B) := \big(US^{1/2},\; S^{1/2}V^\top\big). \tag{6}$$

The reformulation in (5) shows that P "projects" onto $\mathcal{H}$. Although this is not an orthogonal projector, Theorem B.3 in the appendix shows that it exhibits a "projection-like" behavior, namely, it defines a smooth retraction (Absil & Malick, 2012) onto $\mathcal{H}$. Consequently, results from Riemannian optimization (Boumal, 2023) can be applied to analyze the convergence of the BaLoRA-GD method, *i.e.*, gradient descent combined with P to keep the iterates on $\mathcal{H}$, as detailed in the next section. Another important property of the balancing map P is that it preserves the product, unlike the orthogonal projector: denoting $(\tilde{A}, \tilde{B}) := P(A, B)$, then $\tilde{A}\tilde{B} = AB$. This preservation guarantees that the loss remains *unchanged*, which is essential for the intrinsic reformulation described in Section 3.2. The procedure to efficiently compute P is given in Algorithm 1, and has computational complexity $\mathcal{O}((a + b)r^2)$ for $a, b \gg r$, which adds negligible overhead to the cost of the optimizer step (see Section 4).

Algorithm 1 Balanced projection

Require: $(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times b}$
1: Compute polar decompositions $A = R_A S_A$ and $B = S_B R_B$
2: Compute $S = S_A S_B \in \mathbb{R}^{r\times r}$
3: Compute SVD decomposition $S = U\Sigma V^\top$
4: **Return** $A^{\text{proj}}, B^{\text{proj}} = R_A(U\Sigma^{1/2}), (\Sigma^{1/2}V^\top)R_B$

3.2. BaLoRA and BaLoRA-GD

Definition. The *BaLoRA* method consists in applying $P(A, B) = A^{\text{proj}}, B^{\text{proj}}$ at the end of each step of an optimization scheme, as defined in Algorithm 1. When combined with Adam, we simply refer to the resulting algorithm as *BaLoRA*. When applied to the iterates of gradient descent, we call it *BaLoRA-GD*.

While BaLoRA (with Adam) is a heuristic method whose theoretical analysis is beyond the scope of this work, we show that the gradient descent variant, BaLoRA-GD, exhibits a striking intrinsic behavior. Recall that BaLoRA-GD iterates for $k \geq 0$ and some stepsize $\tau_k > 0$, starting from any initialization (A_0, B_0), read $(A_{k+1}, B_{k+1}) = P\big(A_k - \tau_k \nabla_A f(A_k, B_k), B_k - \tau_k \nabla_B f(A_k, B_k)\big)$.

Intrinsic BaLoRA-GD. Consider a loss function $f(A, B) = g(X)$, where $X = AB$ and $g\colon \mathbb{R}^{a\times b} \to \mathbb{R}$ is smooth. This general setting encompasses all LoRA losses. As shown in Proposition 3.1 below, proved in Appendix A.4, the BaLoRA-GD iteration can be written entirely as an intrinsic gradient descent on the manifold of rank-r matrices $\mathcal{N}_r$ endowed with a Riemannian metric. For $X \in \mathcal{N}_r$, the inverse of this metric is given by the symmetric positive (semi)definite linear operator $H_X[W] := (XX^\top)^{1/2}W + W(X^\top X)^{1/2}$. When restricted to symmetric positive definite (SPD) matrices, this operator reduces to $H_X[W] = XW + WX$ and coincides with the inverse of the Bures metric. This metric is well known in optimal transport on Gaussian measures and provides a natural tool for optimization over the cone of SPD matrices (Bhatia et al., 2019). By abuse of terminology, we will refer to H_X as the inverse Bures metric on the larger manifold $\mathcal{N}_r$.

Proposition 3.1 (Intrinsic update on $X = AB$)**.** *Let $f(A, B) = g(AB)$. Denote by (A_k, B_k) the BaLoRA-GD iterates and set $X_k := A_k B_k$. Then for $k \geq 1$,*

$$X_{k+1} = R\big(X_k, -\tau_k \Delta_k\big), \tag{7}$$

$$\text{where } R(X, \delta) = X + \delta - H_X^{-1}[\delta]\, X^\top H_X^{-1}[\delta], \tag{8}$$

and $\Delta_k := H_{X_k}[\nabla g(X_k)]$ is the Riemannian gradient associated with the Bures metric, R is a retraction on $\mathcal{N}_r$.

Note that although $H_X^{-1}[\delta]$ is not uniquely defined when X is rank-deficient, the quantity $R(X, \delta)$ is uniquely defined. Moreover, (7) may fail for $k = 0$ if (A_0, B_0) does not belong to the balancing set $\mathcal{H}$.

Equation (7) is the canonical way to express a Riemannian gradient descent on a manifold using a retraction (Boumal, 2023). When $\tau_k \to 0$, this iteration converges to the gradient flow $\dot{X} = -H_X[\nabla g(X)]$. BaLoRA-GD can therefore be interpreted as an efficient implementation of gradient descent with respect to the Bures metric, leveraging computations on the factored variables (A, B) instead of working directly with X.

BaLoRA initialization. Throughout the paper, we initialize BaLoRA in the same way as standard LoRA, with $A = 0$ and B following the Kaiming initialization. However, the BaLoRA method is compatible with a variety of different initializations, such as the OLoRA (Büyükakyüz, 2024) or the LoRA-GA (Wang et al., 2024) initialization. Investigating whether some of these initializations are particularly suited to BaLoRA is an important question, although outside of the scope of this paper.

4. Experiments

We present an empirical evaluation of BaLoRA, demonstrating its benefits in terms of performance and convergence speed, with negligible computational overhead. Our experiments cover fine-tuning tasks on both synthetic and real-world data, with different pre-trained architectures. We also provide ablation studies that highlight the robustness of BaLoRA with respect to hyperparameter choice.

4.1. Synthetic Experiments

We first compare the dynamics of BaLoRA and standard LoRA on a toy framework which aligns closely with our theoretical analysis. Specifically, we consider the optimization problems, $\min_{(A,B)} \|W^\star - (W + AB)\|_F^2$ and $\min_{(A,B)} \|W^\star - (W_1 + A_1B_1)(W_2 + A_2B_2)\|_F^2$ for $(A, B) \in \mathbb{R}^{a\times r} \times \mathbb{R}^{r\times a}$. The first problem is encompassed by the setting studied in Section 2, while the second extends this setting to fine-tuning simultaneously both layers of a 2-layer linear network, which introduces more complex interactions between layers. We apply LoRA and BaLoRA, optimized with Adam, to these problems across eight different initialization seeds. In Figure 3, we report the loss over iterations for a fixed learning rate. We see that for both configurations (one or two layers), BaLoRA starts slower than LoRA, then enters a fast convergence regime where it significantly outperforms LoRA. This confirms our insights from Section 2 and extends their scope to fine-tuning two layers simultaneously.

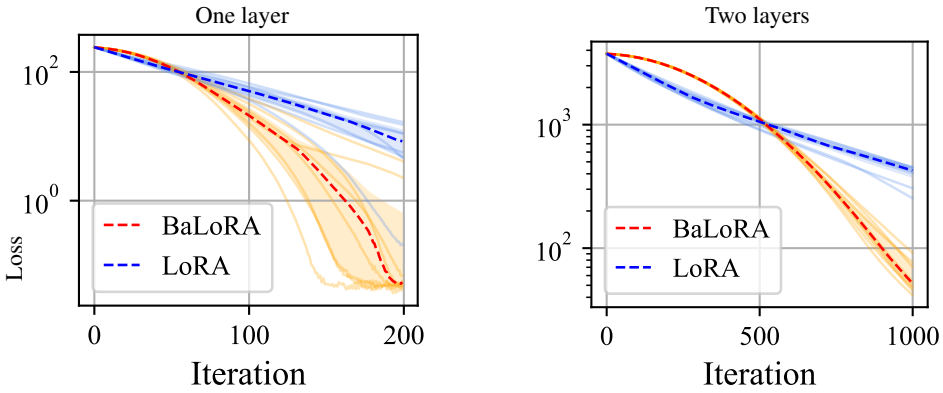

Figure 3. **Synthetic experiments.** Evolution of the loss of LoRA vs. BaLoRA. The dotted lines are the median of 8 curves with different seeds for the initialization, for a fixed target. Both methods use the standard LoRA init, with $A_0 = 0$, B_0 random Gaussian, a scaling $\alpha/r = 1$, and a LoRA rank of 4. The left (resp. right) plot corresponds to a square one-layer linear network of size 20 (resp. a two-layer linear network of size 20, whose layers are both fine-tuned). After a slower start, BaLoRA converges faster in both situations.

4.2. Experiments with Large Language Models

We scale up the experiments to large language models and real-world data. We fine-tune the pretrained models GPT-2 (Radford et al., 2019), Llama-3.2-3B (Meta AI, 2024) and Qwen-2.5-3B (Qwen et al., 2025), evaluating their abilities in language modeling with the dataset Wikitext-2-raw-v1 (Merity et al., 2016), in dialogue with the dataset WizardLM (Xu et al., 2023), their mathematical reasoning with the datasets MetaMathQA (Yu et al., 2023) and GSM8K (Cobbe et al., 2021), their coding abilities with the dataset CodeFeedback (Zheng et al., 2025), and their general natural language understanding with OpenHermes (Teknium, 2023). Since our focus is on optimization speed, we compare methods by reporting the loss on held-out test sets.

General setup. In all the experiments, we simultaneously fine-tune all MLP layers with the optimizer AdamW (Loshchilov & Hutter, 2017), while keeping the attention layers frozen. The learning rate remains constant throughout the training. We choose a LoRA rank equal to 8 for all methods. For each method, we run a sweep of learning rates and scalings—we call scaling the scalar α that multiplies the LoRA adapter: $W + \alpha AB$—and report the test loss for the best choice of such hyperparameters, i.e., the choice that gives the best test loss at the end of the fine-tuning.

Baselines. We compare BaLoRA with several related LoRA-variants:

1. *Standard LoRA* (Hu et al., 2022): each pretrained weight matrix is fine-tuned by adding a trainable product of low-rank adapters AB. B is initialized with Kaiming initialization and A is initially set to 0.
2. *LoRA-GA* (Wang et al., 2024): the adapters A, B are initially balanced and such that AB is the best rank-r approximation of the full gradient of the loss (as a function of the weights, without LoRA adapters). Then, A, B are optimized without constraints, as in LoRA. In particular, (A, B) does not stay balanced.
3. *OLoRA* (Büyükakyüz, 2024): A, B are initialized as the QR decomposition of the pretrained weight matrix, truncated at rank r. In particular, A, B are not balanced and are then optimized as in LoRA.
4. *DoRA* (Liu et al., 2024): the structure of the low-rank adaptation is changed by decoupling the magnitude of AB and its direction. Both components are learnable.

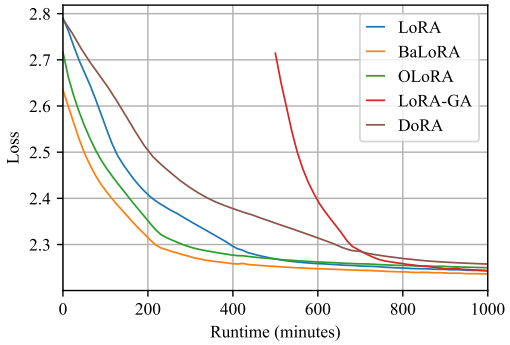

Figure 4. Test loss when fine-tuning Llama-3.2-3B on Wikitext as a function of the training time. The initialization time is taken into account; the full gradient estimation in the LoRA-GA init takes ≈ 500 minutes, which makes it slower than the other methods.

Wikitext. We fine-tune GPT-2 (Radford et al., 2019), Llama-3.2-3B (Meta AI, 2024) and Qwen-2.5-3B (Qwen et al., 2025) on the train split of Wikitext-2-raw-v1 (Merity et al., 2016) (36.7k samples), with a context length of 1024 tokens. Performance is evaluated with the cross-entropy loss on the test split (4.36k samples). Test losses are reported in Tables 1 and 2, and Figure 4 reports the evolution of the test loss as a function of the runtime, to take into account the computational overhead of each LoRA variant during training. Finally, we compare the sensitivity of each method to the learning rate and the initialization scaling in Figures 5 and 6. BaLoRA ranks itself in the top-3, and stands out

Table 1. Fine-tuning Llama-3.2-3B on several datasets (1 epoch): best test loss in bold.

Method	Code	WizardLM	OpenHermes	Wikitext
LoRA	0.919	0.597	0.660	2.278
BaLoRA	**0.918**	**0.596**	**0.659**	**2.274**

Table 2. Fine-tuning GPT-2 on Wikitext-2-raw-v1: best test loss in bold.

Method	Epoch 1	Epoch 2
LoRA	3.261	3.251
BaLoRA	**3.258**	**3.247**
DoRA	3.261	3.252

as the best method when taking into account computational overheads (Figure 4).

GSM8K and MetaMathQA. We fine-tune Llama-3.2-3B on GSM8K (7.47k train samples) and Qwen-2.5-3B on a 30k subset of MetaMathQA. The results are in Table 3 for Qwen, and in Figure 7 and Table 4 for Llama. In both cases, BaLoRA performs on par with the best methods, especially when constraining the runtime.

CodeFeedback, WizardLM and OpenHermes. We compare LoRA and BaLoRA when fine-tuning Llama-3.2-3B on the datasets CodeFeedback, WizardLM and OpenHermes. Results are reported in Table 1 and Figure 8. BaLoRA consistently outperforms LoRA on this range of datasets.

Discussion. In our experiments, *none of the above baselines* consistently outperforms the others: the performance appears to depend very much on the model and dataset. In Table 1, we observe that BaLoRA consistently matches or outperforms standard LoRA, and more generally, BaLoRA ranks among the top-performing variants evaluated in our study, especially when taking computational overhead into account (see, for instance, Figure 4).

5. Conclusion

This paper presents a theoretical analysis of the convergence dynamics of LoRA, revealing how its inherent overparameterization induced a variety of condition numbers at different minimizers of the loss. We identify balanced minimizers that achieve optimal conditioning as a critical factor for efficient optimization. Leveraging this insight, we introduce BaLoRA, a novel extension of LoRA that explicitly enforces balance by projecting adapters onto the hyperbalanced manifold after each optimization step. This projection preserves the adapted weight matrix while systematically improving its conditioning, resulting in faster convergence and greater robustness to hyperparameter choices, all with negligible computational overhead. Our empirical evaluations

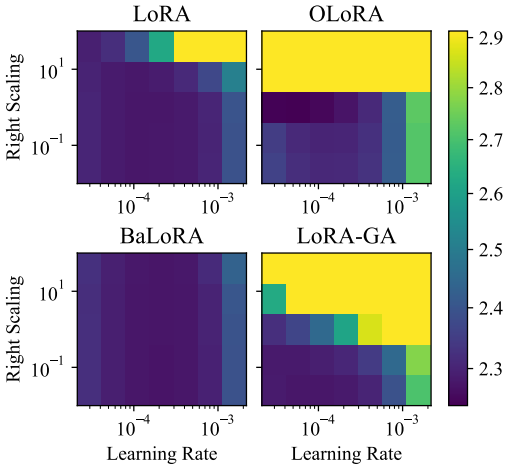

Figure 5. Hyperparameter sensitivity analysis (learning rates, initialization scalings) when fine-tuning Llama-3.2-3B on Wikitext-2-raw-v1. We observe that BaLoRA is significantly more stable to high scalings than all methods, and more stable to high learning rates than OLoRA and LoRA-GA.

Table 3. Fine-tuning Qwen-2.5-3B on MetaMathQA: best test loss in bold, second best underlined. Most methods achieve a similar final loss.

Method	Epoch 1
LoRA	<u>0.1379</u>
BaLoRA	0.1383
DoRA	**0.1379**
OLoRA	0.1384
LoRA-GA	0.1456

on large language models (GPT-2, Llama-3.2, Qwen-2.5) demonstrate that BaLoRA consistently outperforms standard LoRA across multiple datasets. Moreover, it matches or surpasses several state-of-the-art LoRA variants from the literature, both in terms of final accuracy and stability across a range of learning rates and initialization scales.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Absil, P.-A. and Malick, J. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1): 135–158, 2012. doi: 10.1137/110834512.

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Zong, C., Xia, F., Li, W., and

Navigli, R. (eds.), *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)*, pp. 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Awais, M., Naseer, M., Khan, S., Anwer, R. M., Cholakkal, H., Shah, M., Yang, M.-H., and Khan, F. S. Foundation models defining a new era in vision: a survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Bach, F. *Learning theory from first principles*. MIT press, 2024.

Bhatia, R., Jain, T., and Lim, Y. On the bures–wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2019. doi: 10.1016/j.exmath.2018.01.002.

Bogachev, V., Aletov, V., Molozhavenko, A., Bobkov, D., Soboleva, V., Alanov, A., and Rakhuba, M. Riemannlora: A unified riemannian framework for ambiguity-free lora optimization. *arXiv preprint arXiv:2507.12142*, 2025.

Boumal, N. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press, Cambridge, UK, 2023. ISBN 9781009166171. doi: 10.1017/9781009166164.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Büyükakyüz, K. Olora: Orthonormal low-rank adaptation of large language models. *arXiv preprint arXiv:2406.01775*, 2024.

Chen, L. and Bruna, J. Beyond the edge of stability via two-step gradient updates. In *International Conference on Machine Learning*, pp. 4330–4391. PMLR, 2023.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Cohen, J., Kaur, S., Li, Y., Kolter, J. Z., and Talwalkar, A. Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*, 2021.

Dinh, T., Zeng, Y., Zhang, R., Lin, Z., Gira, M., Rajput, S., Sohn, J.-y., Papailiopoulos, D., and Lee, K. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784, 2022.

Du, S. S., Hu, W., and Lee, J. D. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018. URL https://arxiv.org/abs/1806.00900.

Fan, C., Lu, Z., Liu, S., Gu, C., Qu, X., Wei, W., and Cheng, Y. Make lora great again: Boosting lora with adaptive singular values and mixture-of-experts optimization alignment. *arXiv preprint arXiv:2502.16894*, 2025.

Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., and Lee, K. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Ghosh, A., Kwon, S. M., Wang, R., Ravishankar, S., and Qu, Q. Learning dynamics of deep linear networks beyond the edge of stability. *arXiv preprint arXiv:2502.20531*, 2025.

Guo, D., Rush, A. M., and Kim, Y. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020.

Hayou, S., Ghosh, N., and Yu, B. The impact of initialization on lora finetuning dynamics. *Advances in Neural Information Processing Systems*, 37:117015–117040, 2024a.

Hayou, S., Ghosh, N., and Yu, B. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024b.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Jang, U., Lee, J. D., and Ryu, E. K. LoRA training in the NTK regime has no spurious local minima. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=s1sdx6vNsU.

Kalajdzievski, D. A rank stabilization scaling factor for fine-tuning with lora. *arXiv preprint arXiv:2312.03732*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Kopiczko, D. J., Blankevoort, T., and Asano, Y. M. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryup8-WCW.

Li, J., Li, D., Xiong, C., and Hoi, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Li, S., Luo, X., Tang, X., Wang, H., Chen, H., Luo, W., Li, Y., He, X., and Li, R. Beyond zero initialization: Investigating the impact of non-zero initialization on lora fine-tuning dynamics. *arXiv preprint arXiv:2505.23194*, 2025.

Lingam, V. C., Neerkaje, A., Vavre, A., Shetty, A., Gudur, G. K., Ghosh, J., Choi, E., Dimakis, A., Bojchevski, A., and Sanghavi, S. Svft: Parameter-efficient fine-tuning with singular vectors. *Advances in Neural Information Processing Systems*, 37:41425–41446, 2024.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023a.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

Liu, W., Qiu, Z., Feng, Y., Xiu, Y., Xue, Y., Yu, L., Feng, H., Liu, Z., Heo, J., Peng, S., et al. Parameter-efficient orthogonal finetuning via butterfly factorization. *arXiv preprint arXiv:2311.06243*, 2023b.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Malladi, S., Wettig, A., Yu, D., Chen, D., and Arora, S. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, pp. 23610–23641. PMLR, 2023.

Marcotte, S., Gribonval, R., and Peyré, G. Abide by the law and follow the flow: Conservation laws for gradient flows. *Advances in neural information processing systems*, 36: 63210–63221, 2023.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models, 2016.

Meta AI. Llama 3.2 model card. https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_2/, 2024. Version 3.2, 3B parameter model.

Mo, Z., Huang, L.-K., and Pan, S. J. Parameter and memory efficient pretraining via low-rank riemannian optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

Nguegnang, G. M., Rauhut, H., and Terstiege, U. Convergence of gradient descent for learning linear neural networks. *Advances in Continuous and Discrete Models*, 2024(1):23, 2024.

Olikier, G., Uschmajew, A., and Vandereycken, B. Gauss–southwell type descent methods for low-rank matrix optimization. *Journal of Optimization Theory and Applications*, 206(1):6, 2025.

Park, J., Kang, M., Lee, S., Lee, H., Kim, S., and Lee, J. Riemannian optimization for lora on the stiefel manifold. *arXiv preprint arXiv:2508.17901*, 2025.

Qin, C., Zhang, A., Zhang, Z., Chen, J., Yasunaga, M., and Yang, D. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*, 2023.

Qwen, :, Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rebuffi, S.-A., Bilen, H., and Vedaldi, A. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30, 2017.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Stanford alpaca: An instruction-following llama model, 2023.

Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023. URL https://huggingface.co/datasets/teknium/OpenHermes-2.5.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Wang, S., Yu, L., and Li, J. Lora-ga: Low-rank adaptation with gradient approximation, 2024. URL https://arxiv.org/abs/2407.05000.

Ward, R. and Kolda, T. Convergence of alternating gradient descent for matrix factorization. *Advances in Neural Information Processing Systems*, 36:22369–22382, 2023.

Xu, C., Sun, Q., Zheng, K., Geng, X., Zhao, P., Feng, J., Tao, C., and Jiang, D. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

Xu, Z., Min, H., Luo, J., MacDonald, L. E., Tarmoun, S., Mallada, E., and Vidal, R. Understanding the learning dynamics of loRA: A gradient flow perspective on low-rank adaptation in matrix factorization. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=hphdX8WlcT.

Ye, T. and Du, S. S. Global convergence of gradient descent for asymmetric low-rank matrix factorization. *Advances in Neural Information Processing Systems*, 34: 1429–1439, 2021.

Yen, J.-N., Si, S., Meng, Z., Yu, F., Duvvuri, S. S., Dhillon, I. S., Hsieh, C.-J., and Kumar, S. Lora done rite: Robust invariant transformation equilibration for lora optimization. *arXiv preprint arXiv:2410.20625*, 2024.

Yu, L., Jiang, W., Shi, H., Yu, J., Liu, Z., Zhang, Y., Kwok, J. T., Li, Z., Weller, A., and Liu, W. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Yuan, S., Liu, H., and Xu, H. Bridging the gap between low-rank and orthogonal adaptation via householder reflection adaptation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=LzLeAscHnj.

Zeng, Y. and Lee, K. The expressive power of low-rank adaptation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=likXVjmh3E.

Zhang, F. and Pilanci, M. Riemannian preconditioned loRA for fine-tuning foundation models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=IwqE4QqBew.

Zhang, T. and Fan, X. Projected gradient descent algorithm for low-rank matrix estimation. *arXiv preprint arXiv:2403.02704*, 2024.

Zheng, T., Zhang, G., Shen, T., Liu, X., Lin, B. Y., Fu, J., Chen, W., and Yue, X. Opencodeinterpreter: Integrating code generation with execution and refinement, 2025. URL https://arxiv.org/abs/2402.14658.

Zhu, J., Greenewald, K., Nadjahi, K., de Ocáriz Borde, H. S., Gabrielsson, R. B., Choshen, L., Ghassemi, M., Yurochkin, M., and Solomon, J. Asymmetry in low-rank adapters of foundation models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=PHrrbfrMEl.

A. Postponed Proofs

A.1. Proof of Proposition 2.2

Computing H is straightforward. To diagonalize this matrix, notice that $H = MM^\top$ with

$$M \coloneqq \begin{pmatrix} B \otimes I_a \\ I_b \otimes A^\top \end{pmatrix} \in \mathbb{R}^{r(a+b)\times ab}.$$

- **Kernel of H.** The kernel of H is equal to the kernel of $M^\top$, which can be written as $\{\operatorname{vect}(AR, -RB) : R \in \mathbb{R}^{r\times r}\}$. Indeed, unvectorizing the equation $M^\top \operatorname{vect}(D, E) = 0$ gives $DB + AE = 0$, and such (D, E) can be rewritten as $(AR, -RB)$ for $R = A^+D = -EB^+$. Therefore, $\ker H$ is of dimension r^2.

- **Non-zero spectrum of H.** To find the non-zero eigenvalues of H and associated eigenvectors, we use the following observation.

  Lemma A.1. *Let M be a matrix and x be an eigenvector of $M^\top M$ associated with a non-zero eigenvalue λ. Then Mx is an eigenvector of $MM^\top$, associated with the eigenvalue λ.*

  We have
  $$M^\top M = (B^\top B) \otimes I_a + I_b \otimes (AA^\top).$$
  Let (λ, x) be an eigenpair of $AA^\top$ and (μ, y) an eigenpair of $B^\top B$. Then $(\lambda + \mu, y \otimes x)$ is an eigenpair of $M^\top M$. Therefore, denoting $\lambda_1, \dots, \lambda_r$ and $\mu_1, \dots, \mu_r$ the non-zero eigenvalues of $AA^\top$ and $B^\top B$ respectively, with associated unit eigenvectors $x_1, \dots, x_r$ and $y_1, \dots, y_r$, and denoting $x_{r+1}, \dots, x_a$ and $y_{r+1}, \dots, y_b$ unit bases of $\ker AA^\top$ and $\ker B^\top B$, the eigenpairs of $M^\top M$ associated with non-zero eigenvalues are
  $$(\lambda_i + \mu_j, y_j \otimes x_i)_{1\le i,j\le r} \cup (\lambda_i, y_{r+j} \otimes x_i)_{\substack{1\le j\le b-r \\ 1\le i\le r}} \cup (\mu_j, x_j \otimes x_{r+i})_{\substack{1\le i\le a-r \\ 1\le j\le r}}.$$
  Using Lemma A.1, the eigenpairs of $MM^\top$ with non-zero eigenvalues are thus
  $$(\lambda_i + \mu_j, M(y_j \otimes x_i))_{1\le i,j\le r} \cup (\lambda_i, M(y_{r+j} \otimes x_i))_{\substack{1\le j\le b-r \\ 1\le i\le r}} \cup (\mu_j, M(y_j \otimes x_{r+i}))_{\substack{1\le i\le a-r \\ 1\le j\le r}}.$$

A.2. Proof of Proposition 2.5

Sharpness of the Hessian. Let us first compute the largest eigenvalue of H. Denote $H_1 \coloneqq \begin{pmatrix} (BB^\top)\otimes I_a & B\otimes A \\ B^\top \otimes A^\top & I_b \otimes (A^\top A) \end{pmatrix}$ and $H_2 \coloneqq \begin{pmatrix} 0_{ar\times ar} & (I_r \otimes (AB - Z))K_{r,b} \\ ((AB-Z)^\top \otimes I_r)K_{a,r} & 0_{br\times br.} \end{pmatrix}$, so that $H = H_1 + H_2$.

Lemma A.2. *The largest eigenvalue of H_2 is $\sigma_{r+1}(Z)$. The smallest eigenvalue of H_2 is $-\sigma_{r+1}(Z)$.*

Proof. Denote $G \coloneqq (I_r \otimes (AB - Z))K_{r,b} = (((AB - Z)^\top \otimes I_r)K_{a,r})^\top \in \mathbb{R}^{ra\times rb}$. Let $(u_1, \dots, u_{ra})$ and $(v_1, \dots, v_{rb})$ be respectively the left and right eigenvectors of G. Denote $\tau_1 \ge \cdots \ge \tau_\varrho$ the singular values of G, with $\varrho \coloneqq \operatorname{rk} G = r(\min(a, b) - r)$.

- The Kernel of H_2 is the span of $\{\begin{pmatrix} u_j \\ 0 \end{pmatrix} : j = \varrho + 1, \dots, ar\} \cup \{\begin{pmatrix} 0 \\ v_k \end{pmatrix} : k = \varrho + 1, \dots, br\}$.

- For $i = 1, \dots, \varrho$, let $x_i^+ \coloneqq \begin{pmatrix} u_i \\ v_i \end{pmatrix}$ and $x_i^- \coloneqq \begin{pmatrix} u_i \\ -v_i \end{pmatrix}$. Then x_i^+ (resp. x_i^-) is an eigenvector of H_2 associated with the eigenvalue σ_i (resp. $-\sigma_i$). The σ_i can be easily computed, they are of the form $-\sigma_k(Z)$ for $k = r + 1, \dots, \min(a, b)$, which proves the result.

□

The eigenvectors of H_1 form a basis of the space $\mathbb{R}^{(a+b)r}$. We will prove that for any eigenvector u of this basis, it holds $|Hu| \le (\sigma_1(A)^2 + \sigma_1(B)^2)|u|$.

- If u is in the kernel of H_1 and has unit norm, then $|Hu| = |H_2 u| \leq \|H_2\|_2 = \sigma_{r+1}(Z) \leq \sigma_1(Z) \leq \sigma_1(A)\sigma_1(B) \leq \sigma_1(A)^2 + \sigma_1(B)^2$.
- If u is of the form $M(y_j \otimes x_i)$ with the notations of the proof of Proposition 2.2, then

$$H_2 u = \begin{pmatrix} (I_r \otimes (AB - Z))(A^\top x_i \otimes y_j) \\ ((AB - Z)^\top \otimes I_r)(x_i \otimes B y_j) \end{pmatrix} = 0, \tag{9}$$

  so $|Hu| = |H_1 u| \leq (\sigma_1(A)^2 + \sigma_1(B)^2)|u|$.
- If u is of the form $M(y_{r+j} \otimes x_i)$ for $1 \leq i \leq r$ and $1 \leq j \leq b-r$, it is easy to check that $H_1 u$ and $H_2 u$ are orthogonal. Then:

$$\begin{aligned} |Hu| &= \sqrt{|H_1 u|^2 + |H_2 u|^2} \\ &\leq \sqrt{\sigma_i(A)^4 |u|^2 + \sigma_{r+1}(Z)^2 |u|^2} \\ &= \sqrt{\sigma_i(A)^4 + \sigma_{r+1}(Z)^2}|u|. \end{aligned}$$

  We therefore need to prove that $\sqrt{\sigma_i(A)^4 + \sigma_{r+1}(Z)^2} \leq \sigma_1(A)^2 + \sigma_1(B)^2$, or equivalently that $\sigma_1(A)^2 + \sigma_1(B)^2 - \sqrt{\sigma_1(A)^4 + \sigma_{r+1}(Z)^2} \geq 0$. We have $\sigma_1(B)^2 \geq \sigma_1(Z)^2/\sigma_1(A)^2$. Then

$$\begin{aligned} &\sigma_1(A)^2 + \sigma_1(B)^2 - \sqrt{\sigma_1(A)^4 + \sigma_{r+1}(Z)^2} \\ &\quad\geq \sigma_1(A)^2 + \sigma_1(Z)^2/\sigma_1(A)^2 - \sqrt{\sigma_1(A)^4 + \sigma_{r+1}(Z)^2} \\ &\quad\geq \sigma_1(A)^2 + \sigma_{r+1}(Z)^2/\sigma_1(A)^2 - \sqrt{\sigma_1(A)^4 + \sigma_{r+1}(Z)^2} \\ &\quad= (\sigma_1(A)^4 + \sigma_{r+1}(Z)^2)\left(\frac{1}{\sigma_1(A)^2} - \frac{1}{\sqrt{\sigma_1(A)^4 + \sigma_{r+1}(Z)^2}}\right) \\ &\quad\geq 0, \end{aligned}$$

  which proves the result.

Smallest non-zero eigenvalue of the Hessian. We have $\lambda_{\min\neq 0}(H) = \lambda_{r(a+b)-r^2}(H)$, as the Kernel of H has dimension r^2. Equation 9 shows that all the eigenvalues of H_1 are also eigenvalues of H_2. Therefore, $\lambda_{r(a+b)-r^2}(H) \leq \min(\sigma_r(A)^2, \sigma_r(B)^2)$. For the lower bound, we apply the Weyl inequality:

$$\begin{aligned} \lambda_{r(a+b)-r^2}(H) &= \lambda_{r(a+b)-r^2}(H_1 + H_2) \\ &\geq \lambda_{r(a+b)-r^2}(H_1) + \lambda_{r(a+b)}(H_2) \\ &= \min(\sigma_r(A)^2, \sigma_r(B)^2) - \sigma_{r+1}(Z), \end{aligned}$$

according to Lemma A.2.

When the minimizer is balanced, it holds $\sigma_r(A)^2 = \sigma_r(B)^2 = \sigma_r(Z)$. This is the maximal value for the lower bound. Indeed, let (A, B) be any minimizer of the loss f, i.e., $AB = LR_r(Z)$. Denote $U\Sigma V^\top$ the thin SVD of $LR_r(Z)$, i.e., with $\Sigma \succ 0$ of size $r \times r$. We can write $A = U\Sigma^{1/2}P$, $B = P^{-1}\Sigma^{1/2}V^\top$ for some invertible matrix $P \in GL_r(\mathbb{R})$. Then $\sigma_r(Z) = \sigma_r(U\Sigma V^\top) = \sigma_r(\Sigma^{1/2}PP^{-1}\Sigma^{1/2}) \geq \sigma_r(\Sigma^{1/2}P)\sigma_r(P^{-1}\Sigma^{1/2})$ by the Weyl inequality. Hence, $\sigma_r(Z) \geq \sigma_r(A)\sigma_r(B)$. We have proven that balanced minimizers maximize the lower bound $\min(\sigma_r(A)^2, \sigma_r(B)^2) - \sigma_{r+1}(Z)$. Now, it is easy to check that when (A, B) is balanced, the vector $\begin{pmatrix} o_r \otimes u_{r+1} \\ v_{r+1} \otimes o_r \end{pmatrix}$ with

- o_r an eigenvector of $A^\top A = BB^\top$ associated with eigenvalue $\sigma_r(Z)$,
- u_{r+1} the column $r+1$ of U,
- v_{r+1} the column $r+1$ of V,

is an eigenvector of H with the eigenvalue $\sigma_r(Z) - \sigma_{r+1}(Z)$, which proves that $\lambda_{\min\neq 0}(H) = \sigma_r(Z) - \sigma_{r+1}(Z)$ when (A, B) is balanced.

A.3. Proof of Proposition 2.6

We denote $\theta := (A, B)$ and $\phi := \varphi(\theta) := AB$. At an interpolation point, $h(\phi) = Z$, so the Gauss–Newton identity gives $\partial^2 f(\theta) = \partial\varphi(\theta)^\top \, \partial h(\phi)^\top \, \partial h(\phi) \, \partial\varphi(\theta)$. Since $dn \geq ab$, standard singular value inequalities yield $\kappa(f)(\theta) \leq \kappa(\partial h(\phi))^2 \, \kappa(\partial\varphi(\theta))^2$, where $\kappa(\partial\varphi(\theta))$ is defined similarly as $\kappa(\partial h(AB))$ (since ϕ also increases dimensions), but with singular values defined on the orthogonal complement of $\ker(\partial\varphi(\theta))$ (equivalently, by ignoring the zero singular values due to gauge invariance). Proposition 2.2 provides an upper bound on the conditioning of these nonzero singular values, namely $\kappa(\partial\varphi(\theta))^2 \leq \frac{\sigma_1(A)^2+\sigma_1(B)^2}{\min(\sigma_r(A)^2,\sigma_r(B)^2)}$, which gives the claimed inequality.

A.4. Proof of Proposition 3.1

Since $f(A, B) = g(AB)$, the chain rule yields, writing $G_k = \nabla g(X_k)$, $\nabla_A f(A_k, B_k) = G_k B_k^\top$, $\nabla_B f(A_k, B_k) = A_k^\top G_k$. Hence, the pre-projection product is

$$\begin{aligned}\widetilde{X}_k = \widetilde{A}_k\widetilde{B}_k &= (A_k - \tau_k G_k B_k^\top)\,(B_k - \tau_k A_k^\top G_k) \\ &= X_k - \tau_k\big(A_k A_k^\top G_k + G_k B_k^\top B_k\big) + \tau_k^2\, G_k\, X_k^\top G_k.\end{aligned}$$

By construction of the projection P, the product is preserved by P, hence $X_{k+1} = A_{k+1}B_{k+1} = \widetilde{X}_k$. Since $(A_k, B_k) \in \mathcal{H}$, we take an SVD $X_k = U_k S_k V_k^\top$ and balanced factors $A_k = U_k S_k^{1/2}$ and $B_k = S_k^{1/2} V_k^\top$. Then, $A_k A_k^\top = U_k S_k U_k^\top = (X_k X_k^\top)^{1/2}$, $B_k^\top B_k = V_k S_k V_k^\top = (X_k^\top X_k)^{1/2}$. Substituting into $\widetilde{X}_k$ gives the claimed formula $X_{k+1} = X_k - \tau_k H_{X_k}[G_k] + \tau_k^2\, G_k\, X_k^\top G_k$,

B. Structure of the hyperbalanced manifold $\mathcal{H}$

The following proposition further details the structure of the set $\mathcal{H}$.

Proposition B.1 (Equivalent descriptions of $\mathcal{H}$)**.** *The set $\mathcal{H}$ is a smooth manifold in a neighborhood of full-rank points, with dimension equal to that of the rank-r manifold $\mathcal{N}_r := \{X \in \mathbb{R}^{a\times b} : \operatorname{rank}(X) \leq r\}$. The product mapping $(A, B) \mapsto AB$ is a surjective map from $\mathcal{H}$ onto $\mathcal{N}_r$. If one locally fixes a consistent sign convention for the singular vectors in an SVD decomposition $X = USV^\top$, then the mapping $X \mapsto (US^{1/2},\, S^{1/2}V^\top)$ defines a smooth local inverse at points X with non-repeated singular values. Equivalently, $\mathcal{H}$ admits the explicit description*

$$\mathcal{H} = \{\big(US^{1/2},\, S^{1/2}V^\top\big) \;:\; U^\top U = V^\top V = I_r,\;\; S \in \mathbb{D}_+^r\}. \tag{10}$$

Proof. We prove Equation (5). ($\subseteq$) Take $(A, B) \in \mathcal{H}$ with $A^\top A = BB^\top = S \in \mathbb{D}_+^r$. Set $U := AS^{-1/2}$ and $V := B^\top S^{-1/2}$. Then $U^\top U = I_r$, $V^\top V = I_r$, and $A = US^{1/2}$, $B = S^{1/2}V^\top$. ($\supseteq$) Conversely, if $A = US^{1/2}$ and $B = S^{1/2}V^\top$ with $U^\top U = V^\top V = I_r$ and $S \in \mathbb{D}_+^r$, then $A^\top A = S$ and $BB^\top = S$, so $(A, B) \in \mathcal{H}$. □

Remark B.2 (Smoothness of P). It is important to note that the definition of P in Equation (6) is not entirely unambiguous: singular vectors are determined only up to sign, and in the presence of repeated singular values they are even invariant under rotations within the degenerate subspace. When analyzing the convergence of optimization schemes over $X = AB$, this ambiguity is harmless, as discussed in Section 3.2. However, to guarantee that P is smooth, one must restrict attention to points X with distinct singular values and adopt a locally consistent sign convention for the singular vectors.

With Remark B.2 in mind, the next proposition shows that P locally enables the definition of a retraction map ((Absil & Malick, 2012)) that preserves the product AB.

Proposition B.3 (Properties of P)**.** *Let P be as in Equation (6). Then $P(A, B) \in \mathcal{H}$, and if $(A, B) \in \mathcal{H}$, then $P(A, B) = (A, B)$. Furthermore, P preserves the product $\Pi(A, B) := AB$, i.e., $\Pi\big(P(A, B)\big) = \Pi(A, B)$. The map P acts locally as a first–order retraction on $\mathcal{H}$: for any $Z := (A, B) \in \mathcal{H}$ such that AB has distinct singular values, and for any $\Delta \in T_Z\mathcal{H}$ in the tangent plane of $\mathcal{H}$ at Z, the map $(Z, \Delta) \mapsto P(Z+\Delta)$ defines a first-order retraction, namely $P(Z+\Delta) = Z+\Delta+o(\Delta)$.*

Proof. Fix $(A, B) \in \mathcal{H}$ and assume $Z := AB$ has distinct singular values. By standard perturbation theory for the SVD (with a consistent choice of signs), the reduced SVD $Z \mapsto (U, S, V)$ depends C^1-smoothly on Z in a neighborhood of Z, hence the map

$$P(A', B') = \big(U(A'B')\, S(A'B')^{1/2},\; S(A'B')^{1/2} V(A'B')^\top\big)$$

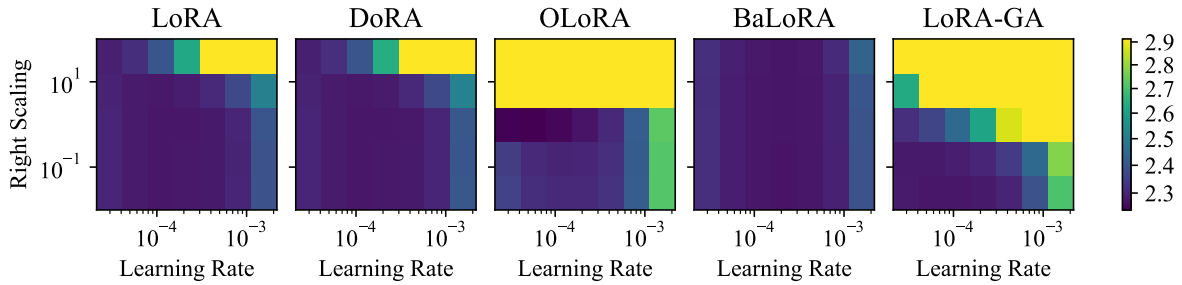

Figure 6. Hyperparameter sensitivity analysis of BaLoRA, LoRA and variants for a grid of learning rates and initialization scalings, when fine-tuning Llama-3.2-3B on Wikitext-2-raw-v1. We observe that BaLoRA is significantly more stable to high scalings than the other methods, and more stable to high learning rates than OLoRA and LoRA-GA.

Table 4. Results of fine-tuning Llama-3.2-3B on GSM8K. Best loss is in bold, second best loss underlined.

Method	Epoch 1	Epoch 2
LoRA	<u>0.498</u>	0.492
BaLoRA	0.506	0.493
DoRA	**0.497**	<u>0.492</u>
OLoRA	0.510	0.503
LoRA-GA	0.504	**0.491**

is C^1 in (A', B') near (A, B). Moreover, P fixes $\mathcal{H}$: if $(A', B') \in \mathcal{H}$ then $P(A', B') = (A', B')$.

Let $\Delta \in T_{(A,B)}\mathcal{H}$. By the definition of the tangent space of an embedded submanifold, there exists a C^1 curve $\gamma : (-\epsilon, \epsilon) \to \mathcal{H}$ with $\gamma(0) = (A, B)$ and $\dot{\gamma}(0) = \Delta$. Since P fixes $\mathcal{H}$ pointwise, $P(\gamma(t)) = \gamma(t)$ for all t small. Differentiating at $t = 0$ and using the chain rule yields

$$DP(A, B)[\Delta] = \frac{d}{dt}\Big|_{t=0} P(\gamma(t)) = \frac{d}{dt}\Big|_{t=0} \gamma(t) = \Delta.$$

Because P is C^1, its first-order expansion at (A, B) gives, for any $\varepsilon \to 0$,

$$P\big((A, B) + \varepsilon\Delta\big) = P(A, B) \; + \; \varepsilon\, DP(A, B)[\Delta] \; + \; o(\varepsilon) = (A, B) \; + \; \varepsilon\Delta \; + \; o(\varepsilon).$$

□

C. Additional Empirical Results

We provide in this section some additional figures.

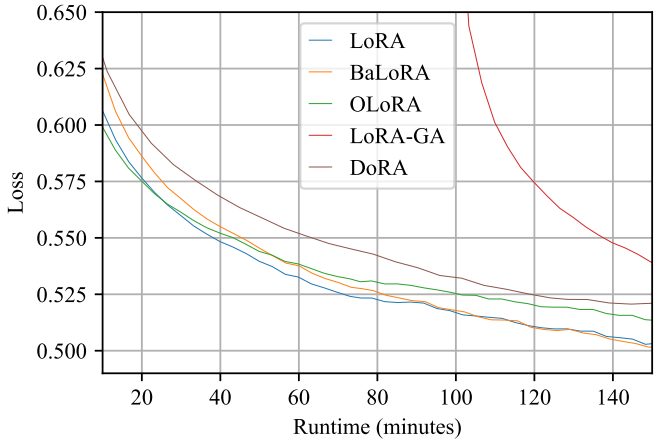

Figure 7. Test loss evolution over fine-tuning of Llama-3.2-3B on GSM8K as a function of the training time. The initialization time is taken into account, which explains why LoRA-GA is slower than the other methods.

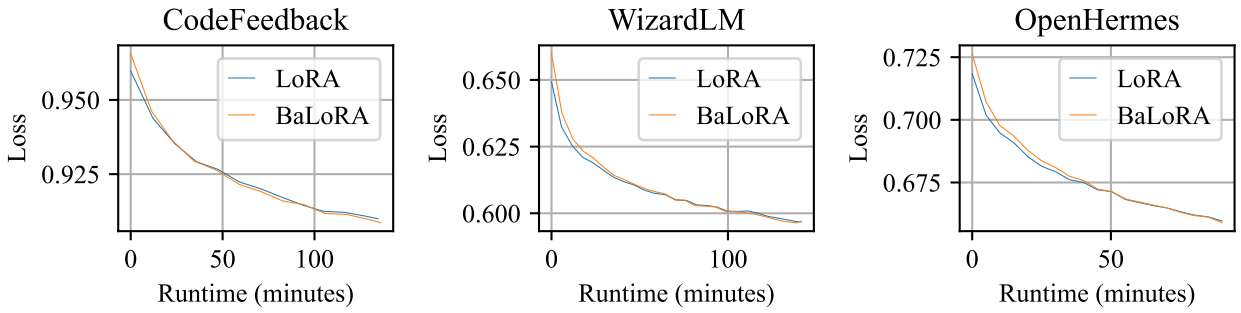

Figure 8. Test loss evolution over fine-tuning of Llama-3.2-3B on CodeFeedback, WizardLM and OpenHermes as a function of the training time. BaLoRA slightly outperforms LoRA in all plots.