RETRACTION-FREE OPTIMIZATION OVER THE STIEFEL MANIFOLD WITH APPLICATION TO THE LORA FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Optimization over the Stiefel manifold has played a significant role in various machine learning tasks. Many existing algorithms either use the retraction operator to keep each iterate staying on the manifold, or solve an unconstrained quadratic penalized problem. The retraction operator in the former corresponds to orthonormalization of matrices and can be computationally costly for large-scale matrices. The latter approach usually equips with an unknown large penalty parameter. To address the above issues, we propose a retraction-free and penalty parameter-free algorithm, which lands on the manifold. Moreover, our convergence theory allows for the use of a constant step size, improving upon the result in (Ablin & Peyré, 2022), which only guarantees convergence to a neighborhood. A key component of the analysis is the convex-like property of the quadratic penalty of the Stiefel manifold, which enables us to explicitly characterize the constant penalty parameter. As an application, we introduce a new algorithm, Manifold-LoRA, which employs the landing technique and a carefully designed step size strategy to accelerate low-rank adaptation (LoRA) in fine-tuning large language models. Numerical experiments on the benchmark datasets demonstrate the efficiency of our proposed method.

1 INTRODUCTION

Optimization over the Stiefel manifold has attracted considerable attention in the context of machine learning, e.g., RNN (Arjovsky et al., 2016), batch normalization (Cho & Lee, 2017), distributionally robust optimization (Chen et al., 2017), and vision transformer (Kong et al., 2023). The mathematical formulation of this class of problems is:

$$\min_{X\in\mathbb{R}^{d\times r}} \; f(X) \quad \text{subject to} \quad X \in \mathrm{St}(d,r) := \{X \in \mathbb{R}^{d\times r} : X^\top X = I\}, \tag{1}$$

where $r \leq d$ and $f : \mathbb{R}^{d\times r} \to \mathbb{R}$ is a continuously differentiable function. The most popular methods for solving (1) are retraction-based algorithms, which have been extensively studied in the context of manifold optimization (Absil et al., 2008; Wen & Yin, 2013; Hu et al., 2020; Boumal, 2023). Recently, to alleviate the possible computational burden of the retraction operator, some retraction-free methods have been developed in (Gao et al., 2018; 2022; Xiao et al., 2024; Ablin & Peyré, 2022). The ideas in these papers are based on a combination of the manifold geometry and a penalty function for the manifold constraint, which involves an unknown but sufficiently large penalty parameter. For large-scale machine learning applications, retraction-free algorithms are preferred. However, designing retraction-free algorithms with a known penalty parameter for solving (1) remains a challenge.

Another motivation for studying retraction-free methods arises from its application in the fine-tuning of large language models (LLMs). Recently, LLMs have revolutionized the field of natural language processing (NLP), achieving unprecedented performance across various applications (Radford et al., 2019; Qin et al., 2023). To tailor pretrained LLMs for specific downstream tasks, the most common approach is full fine-tuning, which requires prohibitively large computational resources due to the need to adapt all model weights, hindering the deployment of large models. As a result, parameter-efficient fine-tuning (PEFT) has gained widespread attention for requiring few trainable parameters while delivering comparable or even superior results to full fine-tuning. This paradigm involves inserting learnable modules or designating only a small portion of weights as trainable, keeping the main model frozen (Houlsby et al., 2019; Li & Liang, 2021; Zaken et al., 2021). Among fine-tuning methods, low-rank adaptation (LoRA) (Hu et al., 2021) has become the de facto standard among parameter-efficient fine-tuning techniques. It assumes that the change in weights lies in a *low intrinsic dimension*, thereby modelling the update $\Delta W \in \mathbb{R}^{d\times m}$ by two low-rank (not greater than a small integer r) matrices $A \in \mathbb{R}^{r\times m}$ and $B \in \mathbb{R}^{d\times r}$, i.e., $\Delta W = BA$. Since $r \ll d$, the requirements on both storage and computation are significantly reduced. Due to its decompositional nature, there is redundancy in the representation of ΔW. Traditional optimization methods for LoRA are unable to exploit this redundancy, which consequently undermines model performance. Instead, we reformulate LoRA fine-tuning as an optimization problem over the product of Stiefel manifolds and Euclidean spaces. Therefore, we propose an algorithmic framework called Manifold-LoRA to accelerate the fine-tuning process and enhance model performance. Moreover, by exploiting projected gradients and incorporating a parameter-free penalty, the overhead that our method incurs is relatively negligible. Our contributions are as follows:

- We first prove the existence of explicit choice for the penalty parameter by establishing a strong convexity-like condition of the nonconvex penalty problem associated with the Stiefel manifold constraint. Our convergence theory also allows for the use of a constant step size, which improves the result of convergence to neighborhood (Ablin & Peyré, 2022) and simplifies the hyperparameter tuning process. Furthermore, for the given penalty parameter, under mild conditions, we prove that the iterates of our proposed retraction-free gradient descent method eventually land on the Stiefel manifold and achieve the optimality of (1).
- Building upon the established landing theory of retraction-free and penalty parameter-free method and the AdamW framework, we propose a new method, Manifold-LoRA, which employs a carefully designed step size strategy to accelerate the training process of fine-tuning. Compared with the conventional AdamW method, we use the penalized gradient instead of the usual gradient, and the computational overhead is negligible.
- Numerical experiments are conducted on a wide range of NLP tasks, demonstrating the efficiency of our algorithm. Specifically, compared to the vanilla LoRA, our Manifold-LoRA with half the trainable parameters not only delivers fast convergence but also yields improved generalization. In particular, our method converges twice as fast as baseline methods on several typical datasets, including the SQuAD 2.0 dataset and the CoLA dataset.

1.1 Related Work

Optimization over the Stiefel manifold. Optimization over the Stiefel manifold has attracted lots of attention due to its broad applications. Through the use of retraction, known as the generalization of the exponential map, the Riemannian gradient descent is proposed (Absil et al., 2008; Boumal, 2023; Hu et al., 2020), where all iterates lie on the manifold. When such retraction is computationally costly, the authors (Gao et al., 2018) develop a retraction-free algorithm based on the augmented Lagrangian method. More recently, by defining the constraint dissolving operator and adding a sufficiently large penalty term, the authors (Xiao et al., 2024) convert the manifold constrained problem (1) into an unconstrained problem and then apply unconstrained optimization algorithms. Inspired by the convergence of Oja's flow, a retraction-free method is developed in (Ablin & Peyré, 2022) for the squared Stiefel manifold (i.e., $d = r$), where the landing flow consists of the projected gradient and the gradient of the penalty function. All of these methods rely on an unknown penalty parameter to ensure the convergence. This motivates us to design penalty parameter-free algorithms, which could significantly reduce the need for tuning parameters in practical implementations.

LoRA. There are numerous variants of LoRA aiming to improve performance or reduce memory usage. AdaLoRA (Zhang et al., 2023), a well-known successor, introduces the idea of adaptively adjusting the rank of different layers by incorporating an additional vector $\boldsymbol{g}$ to serve as the diagonal of a singular value matrix. This approach leverages a revised sensitivity-based importance measure to decide whether to disable entries in vector $\boldsymbol{g}$ and in matrices A and B. A similar work, SoRA (Ding et al., 2023), adopts the same model architecture as AdaLoRA, but proposes a different way to update vector $\boldsymbol{g}$ after training. This update rule is the proximal gradient of $\mathcal{L}_1$ loss, acting as a post-pruning method. Additionally, based on the idea that networks with random initialization contain subnetworks that are optimal(Frankle & Carbin, 2018), VeRA is proposed in (Kopiczko et al., 2023) to reduce memory overhead. Although LoRA has gained significant popularity and various variants have been developed, the potential for efficient training through leveraging the manifold geometry to reduce redundancy has not been well-explored.

1.2 Notation

For a matrix $X \in \mathbb{R}^{d\times r}$, we use $\|X\|$ to denote its Frobenius norm. For a squared matrix $A \in \mathbb{R}^{r\times r}$, we define $\mathrm{sym}(A) = \frac{A+A^\top}{2}$ and use $\mathrm{diag}(A) \in \mathbb{R}^r$ to denote its diagonal part. For two matrices $X, Y \in \mathbb{R}^{d\times r}$, we use $\langle X, Y\rangle := \sum_{i=1}^d \sum_{j=1}^r X_{ij}Y_{ij}$ to denote their Euclidean inner product. For a differential function $f : \mathbb{R}^{d\times r} \to \mathbb{R}$, we use $\nabla f(X)$ to denote its Euclidean gradient at X. We define $U_{\mathrm{St}(d,r)}(\frac{1}{8}) = \{X \in \mathbb{R}^{d\times r} \mid \mathrm{dist}(X, \mathrm{St}(d,r)) < \frac{1}{8}\}$ and $\bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8}) = \{X \in \mathbb{R}^{d\times r} \mid \mathrm{dist}(X, \mathrm{St}(d,r)) \le \frac{1}{8}\}$ with $\mathrm{dist}(X, \mathrm{St}(d,r)) := \min_{Y\in \mathrm{St}(d,r)} \|Y - X\|$.

2 Preliminaries

2.1 Retraction-based manifold optimization

Manifold optimization has attracted much attention in the past few decades, as evident in works such as Absil et al. (2008); Hu et al. (2020); Boumal (2023). For the Stiefel manifold $\mathrm{St}(d,r)$, its tangent space is denoted by $T_X\mathcal{M}$. The tangent space $T_X\mathcal{M}$ of $\mathcal{M}$ at X is defined as the set of all tangent vectors. For a differentiable f, the Riemannian gradient $\widetilde{\mathrm{grad}} f(X) \in T_X\mathcal{M}$ is the unique tangent vector satisfying

$$\left\langle \widetilde{\mathrm{grad}} f(X), \xi \right\rangle_X = \mathrm{d}f(X)[\xi], \forall \xi \in T_X\mathcal{M},$$

where $\langle\cdot,\cdot\rangle_X$ is the Riemannian metric and $\mathrm{d}f$ denotes the differential of function f. If $\mathcal{M}$ is a submanifold embedded in $\mathbb{R}^{d\times r}$, the function f can be extended to $\mathbb{R}^{d\times r}$, and setting the Riemannian metric as the Euclidean metric, then the Riemannian gradient of f at X can be computed as

$$\widetilde{\mathrm{grad}} f(X) = \mathcal{P}_{T_X\mathcal{M}}(\nabla f(X)),$$

where $\mathcal{P}_{T_X\mathcal{M}}$ represents the orthogonal projection onto $T_X\mathcal{M}$. The normal space $N_X\mathcal{M}$ is defined as the orthogonal complement of $T_X\mathcal{M}$ in $\mathbb{R}^{d\times r}$. In the design of Riemannian algorithms, an essential concept is the so-called retraction operator. A retraction operator $\mathcal{R}$ at X, denoted as $\mathcal{R}_X$, is a mapping from $T_X\mathcal{M}$ to $\mathcal{M}$ that satisfies the following two conditions:

- $\mathcal{R}_X(0_X) = X$, where 0_X is the zero element of $T_X\mathcal{M}$.
- $\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{R}_X(t\xi_X)\mid_{t=0} = \xi_X$ for any $\xi_X \in T_X\mathcal{M}$.

It is well-known that the retraction operator is a generalization of the exponential map (Absil et al., 2008). The iterative scheme of a Riemannian gradient descent method is usually given by

$$X_{k+1} = \mathcal{R}_{X_k}(t_k \widetilde{\mathrm{grad}} f(X_k)),$$

where $t_k > 0$ is a step size. For the Stiefel manifold $\mathrm{St}(d,r)$, the Riemannian gradient is $\widetilde{\mathrm{grad}} f(X) = \nabla f(X) - X\mathrm{sym}(X^\top \nabla f(X))$, and there are several choices for the retraction $\mathcal{R}$, such as the exponential map, the Cayley transform, the QR decomposition, and the polar decomposition, see (Hu et al., 2020) for details. Among them, the Cayley transformation proposed by (Wen & Yin, 2013) is popularly used. It can be expressed as

$$\mathcal{R}_X^{\mathrm{Cayley}}(-\eta) = X - U\left(I_{2r} + \frac{1}{2}V^\top U\right)^{-1} V^\top X,$$

where $U = [(I - \frac{1}{2}XX^\top)\eta, X], V = [X, -(I - \frac{1}{2}XX^\top)\eta] \in \mathbb{R}^{d\times(2r)}$. This needs to inverse an $(2r)$-by-$(2r)$ matrix and the total computational flops from (Jiang & Dai, 2015) is $4dr^2 + \frac{40}{3}r^3$, which could be fast calculated for small r.

2.2 PROXIMAL SMOOTHNESS

The notion of proximal smoothness, as introduced by (Clarke et al., 1995), refers to the characteristic of a closed set whereby the nearest-point projection becomes a singleton when the point is close enough to the set. This property facilitates algorithmic and theoretical advancements by endowing nonconvex sets with convex-like structural attributes. Specifically, for any positive real number γ, we define the γ-tube around $\mathcal{M}$ as $U_{\mathcal{M}}(\gamma) := \{X : \mathrm{dist}(X, \mathcal{M}) < \gamma\}$. We say a closed set $\mathcal{M}$ is γ-proximally smooth if the projection operator $\mathcal{P}_{\mathcal{M}}(X) := \mathrm{argmin}_{Y \in \mathcal{M}} \|Y - X\|^2$ is a singleton whenever $X \in U_{\mathcal{M}}(\gamma)$.

Obviously, any closed and convex set is proximally smooth for arbitrary $\gamma \in (0, \infty)$. According to (Clarke et al., 1995, Corollary 4.6), a closed set $\mathcal{M}$ is convex if and only if it is proximally smooth with a radius of γ for every $\gamma > 0$. It is worth noting that the Stiefel manifold is 1-proximally smooth. By following the proof in (Clarke et al., 1995, Theorem 4.8),

$$\left\|\mathcal{P}_{\mathrm{St}(d,r)}(X) - \mathcal{P}_{\mathrm{St}(d,r)}(X)\right\| \leq 2\|X - Y\|, \;\; \forall X, Y \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{2}). \tag{2}$$

Note that for any closed convex set $\mathcal{M} \subset \mathbb{R}^{d\times r}$, the projection operator $\mathcal{P}_{\mathcal{M}}$ is 1-Lipschitz continuous over $\mathbb{R}^{d\times r}$. The singleton property and the Lipschitz continuity (2) from the proximal smoothness make $\mathrm{St}(d, r)$ locally behave like a convex set.

3 RETRACTION-FREE AND PENALTY PARAMETER-FREE OPTIMIZATION OVER THE STIEFEL MANIFOLD

In this section, we focus on the design of retraction-free and penalty parameter-free algorithms for solving problem (1). We will first present the retraction-free algorithm and then show how the penalty parameter can be explicitly determined by characterizing the landscape of the penalty function.

3.1 RETRACTION-FREE ALGORITHMS

Inspired by the retraction-free algorithms (Gao et al., 2018; Xiao et al., 2024; Ablin & Peyré, 2022), we consider the following retraction-free gradient descent method for problem (1):

$$X_{k+1} = X_k - \alpha \mathrm{grad} f(X_k) - \mu X_k(X_k^\top X_k - I), \tag{3}$$

where $\alpha, \mu > 0$ are step sizes and the projected gradient $\mathrm{grad} f(X_k) := \nabla f(X_k) - X_k \mathrm{sym}(X_k^\top \nabla f(X_k))$. Note that the tangent space of $\mathrm{St}(d, r)$ is $T_{X_k}\mathrm{St}(d, r) := \{\xi \in \mathbb{R}^{d\times r} : X_k^\top \xi + \xi^\top X_k = 0\}$. Then, for $X_k \in \mathrm{St}(d, r)$, $\mathrm{grad} f(X_k)$ is the projection of the Euclidean gradient $\nabla f(X_k)$ to the tangent space, i.e., $\mathrm{grad} f(X_k) = \widetilde{\mathrm{grad}} f(X_k)$. Note that the term $X_k(X_k^\top X_k - I)$ is exactly the gradient of the following quadratic penalty function

$$\varphi(X) := \frac{1}{4}\|X^\top X - I\|^2.$$

As will be shown in our theorem, the negative gradient $-\nabla\varphi(X_k)$ pulls the iterate X_{k+1} back to the manifold, while the use of the projected gradient $\mathrm{grad} f(X_k)$ is crucial for ensuring its asymptotic orthogonality with $\nabla\varphi(X_k)$, resulting in landing on the manifold and convergence to a stationary point. This differs with the usual penalty method, which optimizes $f(X) + \mu\varphi(X)$ using the update $X_{k+1} = X_k - \alpha\nabla f(X_k) - \mu X_k(X_k^\top X_k - I)$, and requires $\mu \to \infty$ to guarantee the feasibility.

Compared with the popularly used Cayley transformation-based retraction-type algorithms, the computational cost therein is $4dr^2 + \frac{40}{3}r^3$, which is more than twice the cost of our method at $2dr^2$ for any r. Moreover, retractions on the Stiefel manifold involve complex orthogonalization procedures, such as matrix inversion in the Cayley transformation, which are difficult to scale and parallelize. In contrast, the landing update (3) can be executed using scalable BLAS3 operations.

3.2 EXPLICIT CHOICE FOR THE PENALTY PARAMETER

It is known that a large penalty parameter yields better feasibility (Nocedal & Wright, 1999, Chapter 17). To make the iterative scheme (3) be penalty parameter-free, we need a careful investigation on the landscape of the following optimization problem:

$$\min_{X\in\mathbb{R}^{d\times r}} \quad \varphi(X). \tag{4}$$

It can be easily verified that problem (4) is nonconvex and its optimal solution set is $\mathrm{St}(d,r)$. The key of obtaining an explicit formula of μ is to establish certain strong convexity-type inequality and show the gradient descent method with step size μ has linear convergence.

For any $X \in \mathbb{R}^{d\times r}$, let us denote $\bar{X} := \mathcal{P}_{\mathrm{St}(d,r)}(X)$. Let $X = USV^\top$ be the singular value decomposition with orthogonal matrices $U \in \mathbb{R}^{d\times r}, V \in \mathbb{R}^{r\times r}$ and diagonal matrix $S \in \mathbb{R}^{r\times r}$, then $\bar{X} = UV^\top$. Building on these notations, we demonstrate that problem (4) satisfies the restricted secant inequality (RSI) (Zhang & Yin, 2013), which serves as an alternative to the strong convexity in the linear convergence analysis of gradient-type methods.

Lemma 1. *For any $X \in \mathbb{R}^{d\times r}$ with $\|X - \bar{X}\| \le \frac{1}{8}$, we have*

$$\left\langle \nabla\varphi(X), X - \bar{X}\right\rangle \ge \|X - \bar{X}\|^2. \tag{5}$$

With the above RSI, we have the linear convergence of the gradient descent update for (4), i.e.,

$$X_{k+1} = X_k - \mu\nabla\varphi(X_k). \tag{6}$$

Lemma 2. *Let the sequence $\{X_k\}$ be generated by* (6) *with $\mu = \frac{1}{3}$. Suppose that $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\|X_{k+1} - \bar{X}_{k+1}\|^2 \le \frac{2}{3}\|X_k - \bar{X}_k\|^2. \tag{7}$$

The proofs of Lemmas 1 and 2 can be found in Appendix A.

3.3 Landing on the Stiefel manifold

Building on the established linear convergence of gradient descent for problem (4), we are now able to show that the iterates generated by (3) will land on the Stiefel manifold eventually, and the limiting point is a stationary point of (1), i.e., $\mathrm{grad} f(X_\infty) = 0$.

Let us start with the Lipschitz continuity of $\mathrm{grad} f(X)$. For any $X \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})$, we define $\mathcal{P}_{T_X\mathrm{St}(d,r)}(U) = U - X\mathrm{sym}(X^\top U)$ for $U \in \mathbb{R}^{d\times r}$. We first have the following quadratic upper bound on f from its twice differentiability and the compactness of $\mathrm{St}(d,r)$.

Lemma 3. *There exists a constant $L > 0$ such that for any $X, Y \in \mathrm{St}(d,r)$, the following quadratic upper bound holds:*

$$f(Y) \le f(X) + \langle \mathrm{grad} f(X), Y - X\rangle + \frac{L}{2}\|Y - X\|^2. \tag{8}$$

In addition, there exists a constant $\hat{L} > 0$ such that for any $X \in \mathrm{St}(d,r), Y \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})$,

$$\|\mathrm{grad} f(X) - \mathrm{grad} f(Y)\| \le \hat{L}\|X - Y\|. \tag{9}$$

By the linear convergence result in Lemma 2, we have the following bound on the feasibility error.

Lemma 4. *Let $\{X_k\}$ be the sequence generated by* (3) *with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\|X_{k+1} - \bar{X}_{k+1}\| \le \sqrt{\frac{2}{3}}\|X_k - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\|. \tag{10}$$

The following one-step descent lemma on f is crucial in establishing the convergence.

Lemma 5. *Let $\{X_k\}$ be the sequence generated by* (3) *with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. We have*

$$\begin{aligned} f(\bar{X}_{k+1}) - f(\bar{X}_k) \le &-(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\|\mathrm{grad} f(X_k)\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\ &+ \frac{1}{2}\left(4\hat{D}_f + 8\hat{L}^2 + 8L + 3\right)\|X_k - \bar{X}_k\|^2. \end{aligned} \tag{11}$$

From the above lemma, the one-step descrease on f is related to both the gradient norm of f and the feasibility error. Regarding convergence, we need both $\mathrm{grad} f(X_k)$ and $\|X_k^\top X_k - I\|$ converge to 0. The following theorem shows that the retraction-free and penalty parameter-free update (3) converges.

Theorem 1. *Let $\{X_k\}$ be the sequence generated by* (3) *with $\mu = \frac{1}{3}$ and $\|X_0 - \bar{X}_0\| \le \frac{1}{8}$. If the step size $\alpha < \frac{1}{2c_1}$ for some c_1 large enough, then we have*

$$\min_{k=0,\ldots,K} \|\mathrm{grad} f(X_k)\|^2 \le \frac{1}{K}, \quad \min_{k=0,\ldots,K} \|X_k^\top X_k - I\|^2 \le \frac{1}{K}. \tag{12}$$

The proofs of the above lemmas and theorem are presented in Appendix A.

Remark 1. *In comparison to the landing algorithm (Ablin & Peyré, 2022), which only addresses the squared Stiefel manifold and requires tuning both parameters α and μ, our method handles general Stiefel manifolds and only requires searching for the parameter α, as indicated by Theorem 1.*

Remark 2. *Theorem 1 establishes the exact convergence of our proposed retraction-free method* (3) *with a constant step size. In contrast, the landing algorithm in (Ablin & Peyré, 2022) converges only to a neighborhood whose size depends on the step size, as discussed in the paragraph following Proposition 10 of their paper. Moreover, our iteration complexity of $\mathcal{O}(1/K)$ is on par with retraction-based algorithms (Boumal et al., 2019).*

4 Accelerate LoRA fine-tuning with landing

In this section, we will first clarify where the Stiefel manifold constraint comes from in the LoRA fine-tuning. Then, we will apply the above developed retraction-free and penalty parameter-free method to enhance LoRA fine-tuning.

4.1 Manifold optimization formulation of LoRA fine-tuning

In neural networks, the dense layers perform matrix multiplication, and the weight matrices in these layers usually have a full rank. However, when adapting to a specific task, pre-trained language models have been shown to have a low intrinsic dimension, allowing them to learn efficiently even with a random projection to a smaller subspace. One possible drawback in the current LoRA fine-tuning framework is that the low-rank decomposition ΔW into product BA is not unique. Specifically, for any invertible matrix C, it holds that $BA = (BC)(C^{-1}A)$. Note that BC shares the same column space with B. This suggests us optimizing the subspace generated by B instead of B itself. Numerous studies in the field of low-rank optimization, e.g., (Boumal & Absil, 2011; Dai et al., 2011; 2012), investigate the manifold geometry of the low-rank decomposition and develop efficient algorithms. However, such geometry has not been explored in the LoRA fine-tuning.

To address such redundancy (i.e., the non-uniqueness of BA representations), we regard B as the basis through the manifold constraint and A as the coordinate of ΔW under B. Hence, the optimization problem can be formulated as

$$\min_{A,B} \quad \mathcal{L}(BA), \quad \text{subject to} \quad B \in \mathrm{St}(d,r) \text{ or } B \in \mathrm{Ob}(d,r), \tag{13}$$

where $\mathrm{Ob}(d,r) := \{B \in \mathbb{R}^{d\times r} : \mathrm{diag}(B^\top B) = \mathbf{1}\}$ and $\mathcal{L}$ represents the loss function. Compared to the Stiefel manifold $\mathrm{St}(d,r)$, the oblique manifold $\mathrm{Ob}(d,r)$ necessitates that the matrix B has unit norms in its columns, without imposing requirements for orthogonality between the columns. Problem (13) is an optimization problem over the product of manifolds and Euclidean spaces.

4.2 Manifold-LoRA

The retraction-free method is well-suited to address (13), simultaneously minimizing the loss function $\mathcal{L}(BA)$ and constraint violation. To control the constraint violation, we use the quadratic penalties $R_s(B) := \|B^\top B - I\|^2$ and $R_o(B) := \|\mathrm{diag}(B^\top B) - 1\|^2$ for the Stiefel manifold and oblique manifold, respectively. As shown in the landing theory in Section 3, we shall use the projected

Algorithm 1: Manifold-LoRA

Input: Initial point $A_0, B_0, \mu \in \mathbb{R}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $upper\_bound \geq lower\_bound > 0$, $\epsilon = 10^{-8}$, $\lambda > 0$, and $k = 0$.

```
while Stopping conditions not met do
    for C ∈ {A, B} do
        if C = B then
            Set g(C_k) according to (14) or (15) using the stochastic estimate of ∇_B L(B_k A_k)
            // Projected gradient for matrix B
        else
            Set g(C_k) to be the stochastic estimate of ∇_A L(B_k A_k)
        end
        m(C_k) ← β_1 m(C_k) + (1 − β_1) g(C_k)
        v(C_k) ← β_2 v(C_k) + (1 − β_2) g_t^2(C_k)
        m̂(C_k) ← m(C_k) / (1 − β_1^t)
        v̂(C_k) ← v(C_k) / (1 − β_2^t)
        η(C_k) ← clip(norm_{C_k}, upper_bound, lower_bound)
                                        // Scheduling step size of matrix A and B
        C_k ← C_{k−1} − η_t(C_k) ( m̂_t(C_k) / ( sqrt(v̂_t(C_k)) + ε ) ) − λ C_{k−1}
        if C = B then
            C_k ← C_k − μ∇R_s(C_k)( or ∇R_o(C_k))     // Apply penalty gradient for
             matrix B
        end
    end
    k ← k + 1
end
```

gradient of the loss part instead of the Euclidean gradient. For the Stiefel manifold and the oblique manifold, the respective projected gradients are

$$\mathrm{grad}_B \mathcal{L}(BA) = \nabla_B \mathcal{L}(BA) - B\mathrm{sym}(B^\top \nabla_B \mathcal{L}(BA)) \tag{14}$$

and

$$\mathrm{grad}_B \mathcal{L}(BA) = \nabla_B \mathcal{L}(BA) - B\mathrm{diag}(\mathrm{diag}(B^\top \nabla_B \mathcal{L}(BA))). \tag{15}$$

Thus, the gradients of our retraction-free method for A and B are $\nabla_A \mathcal{L}(BA)$ and $\mathrm{grad}_B \mathcal{L}(BA) + \mu \nabla R_s(B)(\text{ or } \nabla R_o(B))$.

Note that B and A represent the basis and the coordinate of ΔW, respectively. This results in different magnitudes and different Lipschitz constants of their gradient function. In fact, let $X = BA$. It follows

$$\nabla_A \mathcal{L}(BA) = B^\top \nabla_X \mathcal{L}(X), \quad \nabla_B \mathcal{L}(BA) = \nabla_X \mathcal{L}(X) A^\top.$$

Then,

$$\begin{aligned} \|\nabla_A \mathcal{L}(BA_1) - \nabla \mathcal{L}(BA_2)\| &\leq \|B\|_2 L_g \|A_1 - A_2\|, \\ \|\nabla_B \mathcal{L}(B_1 A) - \nabla \mathcal{L}(B_2 A)\| &\leq \|A\|_2 L_g \|B_1 - B_2\|, \end{aligned}$$

where L_g is the Lipschitz constant of $\nabla_X \mathcal{L}(X)$ and $\|\cdot\|_2$ represent the matrix ℓ_2 norm (i.e., the largest singular value). Note that the step size generally should be propositional to the reciprocal of Lipschitz constant for the gradient type algorithms (Nocedal & Wright, 1999; Bottou et al., 2018). Hence, we schedule the learning rates for the two matrices based on their respective ℓ_2 norms. Having prepared the above, we incorporate the AdamW optimizer (Loshchilov & Hutter, 2018) with our manifold-accelerated technique to enhance the LoRA fine-tuning, as presented in Algorithm 1.

5 Experiments

In this section, we delve into the experimental results and their detailed analysis. This discussion is structured around two principal areas: (1) the performance gain compared to other mainstream fine-tuning methods and accelerated convergence achieved through our manifold-constrained optimization approach; (2) the convergence of matrix B onto the manifold, illustrated by the heat map of $B^\top B$.

5.1 Natural language understanding

We evaluate our backbone model DeBERTaV3-base (He et al., 2021) on GLUE (Wang et al., 2018) benchmark containing nine subdatasets, including MNLI (Williams et al., 2017), SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), QQP (Wang et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Bentivogli et al., 2009), MRPC (Dolan & Brockett, 2005), and STS-B (Wang et al., 2018).

Manifold-LoRA exhibits superior performance on GLUE benchmark compared to other memory-equivalent methods. Experimental results of the GLUE benchmark are recorded in Table 1. It can be seen that our method is superior to other baselines on most tasks. Notably, for RTE and STS-B datasets, both sphere-constrained (i.e., oblique manifold-constrained) and Stiefel-constrained have an obvious performance gain even with only half the trainable parameters compared to the LoRA baseline, i.e., Sphere$_{r=8}$ and Stiefel$_{r=8}$ beat LoRA$_{r=16}$. Note that Manifold-LoRA and the baselines have the same memory requirement under same rank r.

Manifold-LoRA achieves faster convergence across multiple datasets. In addition, with the help of manifold geometry, the fine-tuning process can be significantly accelerated compared to the vanilla AdamW optimizer, achieving a lower training loss, as shown in Figure 1. Particularly, on the CoLA dataset presented in Figure 1a, our approach achieves the same training loss as the standard Adam optimizer but requires nearly half the number of epochs.

5.2 Question Answering

We conduct an evaluation on two question answering datasets: SQuAD v1.1 (Rajpurkar et al., 2016) and SQuADv2.0 (Rajpurkar et al., 2018). Manifold-LoRA is used to fine-tune DeBERTaV3-base for these tasks, which are treated as sequence labeling problems predicting the probability of each token as the start or end of an answer span. The main experimental results are presented in Table 2.

Manifold-LoRA surpasses full fine-tuning on question answering task . Notably, our proposed algorithm outperforms fine-tuning methods, which requires three times larger memory consumption compared to Manifold-LoRA. Moreover, as demonstrated in Table 2, Manifold-LoRA outperforms all other baselines on both Stiefel and Sphere settings, regardless of whether $r = 8$ or $r = 16$.

Our method converges twice as fast as baseline methods on SQuAD datasets. Additionally, we plot the training loss against epochs in Figure 2. We can suggest that the proposed Manifold-LoRA method achieves a 2x speed-up in training epochs compared to AdamW, while simultaneously improving model performance. We also illustrate the heat map of $B^\top B$ in Figure 3, which indicates that the matrix B lands on the manifold eventually. This supports our assertion that landing on manifold enhances the performance of LoRA.

5.3 Natural Language Generation

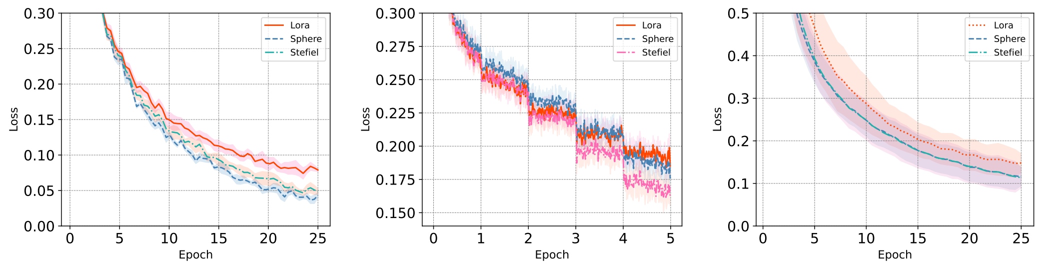
The E2E NLG Challenge(Novikova et al., 2017), as introduced by Novikova, provides a dataset for training end-to-end, data-driven natural language generation systems, widely used in data-to-text evaluations. The E2E dataset comprises approximately 42,000 training examples, 4,600 validation examples, and 4,600 test examples, all from the restaurant domain. We test our method on the E2E dataset using GPT-2 Medium and Large models, following the experimental setup outlined by LoRA. For LoRA, we set the hyperparameters to match those specified in the original paper. The results from the E2E dataset are recorded in Table 3, where we focus on comparing LoRA and Manifold-LoRA. The results clearly indicate that our proposed algorithm outperforms the established baselines.

6 Conclusion

Optimization over the Stiefel manifold has been widely used in machine learning tasks. In this work, we develop a retraction-free and penalty parameter-free gradient method, and prove that the generated iterates eventually land on the manifold and achieve the optimality simultaneously. Moreover, our convergence theory enables the use of a constant step size, improving on previous results that only ensured convergence to a neighborhood. We then apply this landing theory to avoid the possible redundancy of LoRA fine-tuning in LLMs. Specifically, we reformulate the LoRA fine-tuning as an optimization problem over the Stiefel manifold, and propose a new algorithm, Manifold-LoRA,

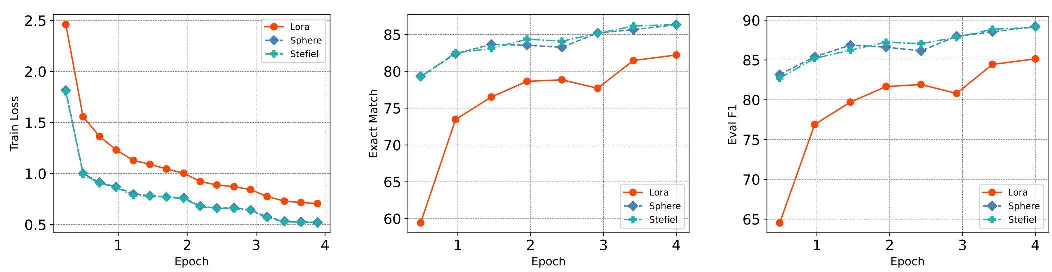
Table 1: We present results using DeBERTaV3-base on the GLUE benchmark. For MNLI, we report the accuracy (combining matched and mismatched sets), with the left panel representing matched subset and the right panel representing mismatched subset. For CoLA, we report Matthew's correlation, and for STS-B, we report Pearson correlation. For all other tasks, we report accuracy. All metrics are same as the original LoRA paper (Hu et al., 2021). Higher values are better for all metrics. The best results are highlighted in **bold**.

Method	# Params	MNLI Acc	SST-2 Acc	CoLA Mcc	QQP Acc / F1	QNLI Acc	RTE Acc	MRPC Acc	STS-B Corr	All Ave.
Full FT	184.42M	90.45/**90.60**	95.48	68.17	**91.99/89.12**	93.60	79.28	88.93	90.92	87.85
Adapter	0.61M	90.13/90.16	94.86	69.37	91.38/88.46	93.54	81.87	89.12	91.52	88.06
BitFit	0.06M	87.08/86.39	94.88	69.11	87.96/84.35	92.19	76.52	87.06	90.96	85.65
LoRA$_{r=8}$	0.30M	90.20/90.08	94.93	68.14	90.78/87.68	93.85	80.15	90.40	90.29	87.60
LoRA$_{r=16}$	0.59M	90.44/90.12	95.41	68.19	90.92/87.77	94.00	80.58	90.20	90.34	87.74
Sphere$_{r=8}$	0.30M	90.37/90.09	95.48	69.55	91.25/88.34	94.02	82.44	91.55	91.26	88.44
Sphere$_{r=16}$	0.59M	**90.52**/90.19	95.64	**70.14**	91.46/88.65	**94.29**	82.16	**91.67**	**91.59**	**88.63**
Stiefel$_{r=8}$	0.30M	90.25/89.99	95.46	69.85	91.44/88.60	94.09	**83.16**	91.18	91.22	88.52
Stiefel$_{r=16}$	0.59M	90.26/90.28	**95.76**	68.92	91.71/89.00	94.10	82.16	91.10	91.51	88.48



(a) Loss curves on CoLA dataset. (b) Loss curves on QQP dataset. (c) Loss curves on STSB dataset.

Figure 1: The figures illustrate that both sphere constrained and Stiefel constrained manifold-LoRA achieve a faster convergence rate and attain a lower training loss within same optimization steps compared to LoRA method on three distinct datasets CoLA, QQP, STS-B.



(a) SQuADv2.0 Train Loss (b) SQuADv2.0 Eval Exact Match (c) SQuADv2.0 Eval F1

Figure 2: The figures compare the training loss, evaluation exact match, and evaluation F1 metrics against epochs for the SQuADv2.0 dataset. It can be clearly seen that our proposed Manifold-LoRA method almost achieves a 2x speed-up in training epochs compared to the vanilla LoRA.

which incorporates a careful analysis of step sizes to enable fast training using the landing properties. Extensive experimental results demonstrate that our approach not only accelerates the training process but also yields significant performance improvements.

Our study suggests several potential directions for future research. Although the established landing theory focuses on the Stiefel manifold, extending this theory to general manifolds, is one potential direction. Additionally, evaluating the performance of Manifold-LoRA on LLMs with billions of parameters would be valuable. Due to the heterogeneity of different layers, incorporating adaptive ranks for ΔW across different layers is another possible direction. This may be achievable by adding sparsity regularization to the coordinate matrix A.

Table 2: Results with DeBERTaV3-base on SQuAD v1.1 and SQuADv2.0. We report EM/F1. The best results in each setting are shown in **bold**.

Methods	Params	SQuADv1.1	SQuADv2.0
Full FT	184M	86.30 / 92.85	84.30 / 87.58
Adapter$_{r=16}$	0.61M	87.46 / 93.41	85.30 / 88.23
Adapter$_{r=32}$	1.22M	87.53 / 93.51	85.42 / 88.36
Bitfit	0.07M	80.26 / 88.79	74.21 / 87.19
LoRA$_{r=8}$	1.33M	87.90 / 93.88	85.56 / 88.52
LoRA$_{r=16}$	2.65M	87.94 / 93.75	85.90 / 88.81
Sphere$_{r=8}$	1.33M	88.51 / **94.25**	86.33 / 89.20
Sphere$_{r=16}$	2.65M	88.32 / 94.03	86.15 / 89.03
Stiefel$_{r=8}$	1.33M	**88.68** / 94.23	86.35 / 89.09
Stiefel$_{r=16}$	2.65M	88.25 / 94.04	**86.41 / 89.22**

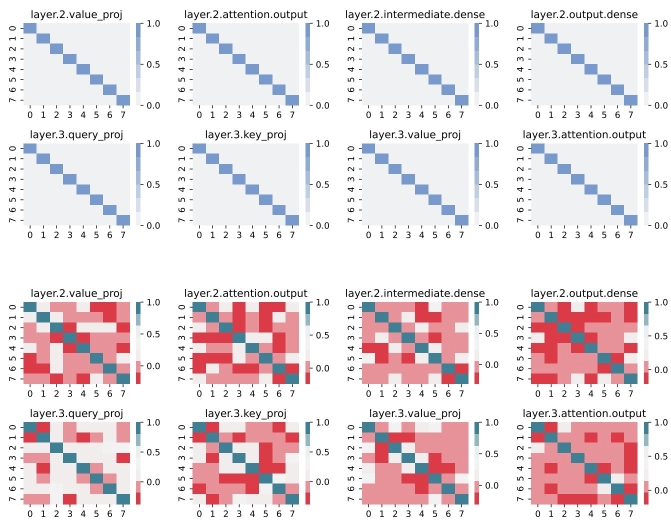


Figure 3: The heat map of $B^\top B$ with the Stiefel manifold (the first and second rows) and the oblique manifold (the third and fourth rows) at the end of training on SQuADv2.0 dataset.

Table 3: GPT-2 medium (M) and large (L) models were evaluated on the E2E NLG Challenge. * denotes results from previously published works.

Model	Parameters	BLEU	NIST	MET	ROUGE-L	CIDEr
GPT-2 M (FT)*	354.92M	68.2	8.62	46.2	71.0	2.47
GPT-2 M (Adapter$^{\mathrm{L}}$)*	11.09M	68.9	8.71	46.1	71.3	2.47
GPT-2 M (Adapter$^{\mathrm{H}}$)*	11.09M	$67.3_{\pm.6}$	$8.50_{\pm.07}$	$46.0_{\pm.2}$	$70.7_{\pm.2}$	$2.44_{\pm.01}$
GPT-2 M (FT$^{\mathrm{Top2}}$)*	25.19M	68.1	8.59	46.0	70.8	2.41
GPT-2 M (PreLayer)*	0.35M	69.7	8.81	46.1	71.4	2.49
GPT-2 M (LoRA)	0.35M	68.9	8.69	46.5	71.5	2.51
GPT-2 M(Stiefel)	0.35M	70.1	8.82	**46.8**	**71.7**	**2.53**
GPT-2 M(Sphere)	0.35M	**70.3**	**8.83**	46.7	**71.7**	2.52
GPT-2 L (FT)*	774.03M	68.5	8.78	46.0	69.9	2.45
GPT-2 L (Adapter$^{\mathrm{L}}$)*	23.00M	$68.9_{\pm.3}$	$8.70_{\pm.04}$	$46.1_{\pm.1}$	$71.3_{\pm.2}$	$2.45_{\pm.02}$
GPT-2 L (PreLayer)*	0.77M	70.3	8.85	46.2	71.7	2.47
GPT-2 L (LoRA)	0.77M	70.1	8.82	46.7	72.0	2.53
GPT-2 L(Stiefel)	0.77M	70.4	8.86	**46.8**	72.1	2.53
GPT-2 L(Sphere)	0.77M	**70.9**	**8.92**	**46.8**	**72.5**	**2.55**

REFERENCES

Pierre Ablin and Gabriel Peyré. Fast and accurate optimization on the orthogonal manifold without retraction. In *International Conference on Artificial Intelligence and Statistics*, pp. 5636–5657. PMLR, 2022.

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *International conference on machine learning*, pp. 1120–1128. PMLR, 2016.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The Fifth PASCAL Recognizing Textual Entailment Challenge. *TAC*, 7(8):1, 2009.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

Nicolas Boumal and Pierre-antoine Absil. RTRMC: A Riemannian trust-region method for low-rank matrix completion. *Advances in neural information processing systems*, 24, 2011.

Nicolas Boumal, Pierre-Antoine Absil, and Coralia Cartis. Global rates of convergence for nonconvex optimization on manifolds. *IMA Journal of Numerical Analysis*, 39(1):1–33, 2019.

Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. Robust optimization for non-convex objectives. *Advances in Neural Information Processing Systems*, 30, 2017.

Shixiang Chen, Alfredo Garcia, Mingyi Hong, and Shahin Shahrampour. Decentralized Riemannian gradient descent on the Stiefel manifold. In *International Conference on Machine Learning*, pp. 1594–1605. PMLR, 2021.

Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. *Advances in Neural Information Processing Systems*, 30, 2017.

Francis H Clarke, Ronald J Stern, and Peter R Wolenski. Proximal smoothness and the lower-C2 property. *Journal of Convex Analysis*, 2(1-2):117–144, 1995.

Wei Dai, Olgica Milenkovic, and Ely Kerman. Subspace evolution and transfer (SET) for low-rank matrix completion. *IEEE Transactions on Signal Processing*, 59(7):3120–3132, 2011.

Wei Dai, Ely Kerman, and Olgica Milenkovic. A geometric approach to low-rank matrix completion. *IEEE Transactions on Information Theory*, 58(1):237–247, 2012.

Kangkang Deng and Jiang Hu. Decentralized projected Riemannian gradient method for smooth optimization on compact submanifolds. *arXiv preprint arXiv:2304.08241*, 2023.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. *arXiv preprint arXiv:2311.11696*, 2023.

Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Bin Gao, Xin Liu, Xiaojun Chen, and Ya-xiang Yuan. A new first-order algorithmic framework for optimization problems with orthogonality constraints. *SIAM Journal on Optimization*, 28(1): 302–332, 2018.

Bin Gao, Guanghui Hu, Yang Kuang, and Xin Liu. An orthogonalization-free parallelizable framework for all-electron calculations in density functional theory. *SIAM Journal on Scientific Computing*, 44(3):B723–B745, 2022.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jiang Hu, Xin Liu, Zai-Wen Wen, and Ya-Xiang Yuan. A brief introduction to manifold optimization. *Journal of the Operations Research Society of China*, 8:199–248, 2020.

Shengding Hu, Ning Ding, Weilin Zhao, Xingtai Lv, Zhen Zhang, Zhiyuan Liu, and Maosong Sun. Opendelta: A Plug-and-play Library for Parameter-efficient Adaptation of Pre-trained Models. *arXiv preprint arXiv:2307.03084*, 2023.

Bo Jiang and Yu-Hong Dai. A framework of constraint preserving update schemes for optimization on Stiefel manifold. *Mathematical Programming*, 153(2):535–575, 2015.

Lingkai Kong, Yuqing Wang, and Molei Tao. Momentum Stiefel Optimizer, with Applications to Suitably-Orthogonal Attention, and Optimal Transport. In *International Conference on Learning Representations*, 2023.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The E2E dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. *arXiv preprint arXiv:1806.03822*, 2018.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1):397–434, 2013.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

Nachuan Xiao, Xin Liu, and Kim-Chuan Toh. Dissolving constraints for Riemannian optimization. *Mathematics of Operations Research*, 49(1):366–397, 2024.

Jinming Xu, Shanying Zhu, Yeng Chai Soh, and Lihua Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pp. 2055–2060. IEEE, 2015.

Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

Hui Zhang and Wotao Yin. Gradient methods for convex minimization: better rates under weaker conditions. *arXiv preprint arXiv:1303.4645*, 2013.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023.

A PROOFS

Proof of Lemma 1

Proof. Denote the SVD of X by $X = USV^\top$. Then, it holds that $\mathrm{dist}(X, \mathrm{St}(d,r)) = \|X - \bar{X}\| = \|s - 1\|_2$, where $s = \mathrm{diag}(S)$. Based on the assumption that $\|X - \bar{X}\| \le \frac{1}{8}$, we have $\frac{7}{8} \le s_i \le \frac{9}{8}$ for any i. Therefore, it follows that

$$\begin{aligned}
\langle \nabla\varphi(X), X - \bar{X} \rangle &= \langle USV^\top (VS^2V^\top - I), USV^\top - UV^\top \rangle \\
&= \langle U(S^3 - S)V^\top, U(S - I)V^\top \rangle \\
&= \mathrm{tr}((S^3 - S)(S - I)) \\
&\ge \min_i s_i(s_i + 1)\|s - 1\|_2^2 \\
&\ge \frac{3}{2}\|s - 1\|_2^2 \\
&= \frac{3}{2}\|X - \bar{X}\|^2,
\end{aligned}$$

where the last inequality comes from $\min_i s_i(s_i + 1) \ge \frac{105}{64} \ge \frac{3}{2}$. This completes the proof. $\square$

Proof of Lemma 2

Proof. It follows from $\frac{7}{8} \le s_i \le \frac{9}{8}$ that

$$\|\nabla\varphi(X_k)\|^2 = \mathrm{tr}((S^3 - S)^2) \le 6\|X_k - \bar{X}_k\|^2. \tag{16}$$

Hence, we have

$$\begin{aligned}
\|X_{k+1} - \bar{X}_{k+1}\|^2 &\le \|X_{k+1} - \bar{X}_k\|^2 \\
&= \|X_k - \frac{1}{3}\nabla\varphi(X_k) - \bar{X}_k\|^2 \\
&= \|X_k - \bar{X}_k\|^2 - \frac{2}{3}\langle X_k - \bar{X}_k, \nabla\varphi(X_k)\rangle + \frac{1}{9}\|\nabla\varphi(X_k)\|^2 \\
&\le (1 - 1 + \frac{2}{3})\|X_k - \bar{X}_k\|^2 \\
&= \frac{2}{3}\|X_k - \bar{X}_k\|^2,
\end{aligned}$$

where the first inequality is from $\bar{X}_{k+1} = \mathrm{argmin}_{X \in \mathrm{St}(d,r)} \|X - X_{k+1}\|^2$ and the second inequality is due to Lemma 1 and (16). $\square$

Proof of Lemma 3

Proof. Due to the twice differentiability of f and the compactness of $\mathrm{St}(d,r)$, the inequality (8) directly follows from (Chen et al., 2021, Lemma 2.4) and (Deng & Hu, 2023, Lemma 4.2), where $L := L_f + D_f$ with L_f being the Lipschitz constant of $\nabla f(X)$ over $\mathrm{St}(d,r)$ and $D_f := \max_{X \in \mathrm{St}(d,r)} \|\nabla f(X)\|$.

For the second argument, we have

$$\begin{aligned}
&\|\mathrm{grad} f(X) - \mathrm{grad} f(Y)\| \\
\le&\|\mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(X)) - \mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(Y))\| + \|\mathcal{P}_{T_X\mathrm{St}(d,r)}(\nabla f(Y)) - \mathrm{grad} f(Y)\| \\
\le&L_f\|X - Y\| + \frac{1}{2}\|X(X^\top\nabla f(Y) + \nabla f(Y)^\top X) - Y(Y^\top\nabla f(Y) + \nabla f(Y)^\top Y)\| \\
\le&L_f\|X - Y\| + \frac{1}{2}\|X((X - Y)^\top\nabla f(Y) + \nabla f(Y)^\top(X - Y))\| \\
&+ \frac{1}{2}\|(X - Y)(Y^\top\nabla f(Y) + \nabla f(Y)^\top Y)\| \\
\le&L_f\|X - Y\| + \frac{1}{2}(2\hat{D}_f + 3\hat{D}_f)\|X - Y\| \\
=&(L_f + \frac{5}{2}\hat{D}_f)\|X - Y\|,
\end{aligned}$$

where $\hat{D}_f := \max_{X \in \bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})} \|\nabla f(X)\|$, the second inequality is due to the contractive property of $\mathcal{P}_{T_X \mathrm{St}(d,r)}$, and the last inequality is from the fact that $\|Y\|_2 \leq \frac{3}{2}$. By setting $\hat{L} = L_f + \frac{5}{2}\hat{D}_f$, we complete the proof. $\square$

Proof of Lemma 4

Proof. It follows that

$$\begin{aligned}
\|X_{k+1} - \bar{X}_{k+1}\| &\leq \|X_{k+1} - \bar{X}_k\| \\
&\leq \|X_k - \mu\varphi(X_k) - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\| \\
&\leq \sqrt{\frac{2}{3}}\|X_k - \bar{X}_k\| + \alpha\|\mathrm{grad} f(X_k)\|.
\end{aligned}$$

We complete the proof. $\square$

Proof of Lemma 5

Proof. First, let us prove the following equality

$$\langle \mathrm{grad} f(X), \nabla\phi(X)\rangle = \left\langle \nabla f(X), \mathcal{P}_{T_X \mathrm{St}(d,r)}(\nabla\phi(X))\right\rangle.$$

In fact, using the definition of $\langle A, B\rangle = \mathrm{tr}(A^\top B)$, we have

$$\begin{aligned}
\langle \mathrm{grad} f(X), \nabla\phi(X)\rangle &= \left\langle \nabla f(X) - X\mathrm{sym}(X^\top \nabla f(X)), \nabla\phi(X)\right\rangle \\
&= \langle \nabla f(X), \nabla\phi(X)\rangle - \left\langle X\mathrm{sym}(X^\top \nabla f(X)), \nabla\phi(X)\right\rangle \\
&= \langle \nabla f(X), \nabla\phi(X)\rangle - \left\langle \mathrm{sym}(X^\top \nabla f(X)), X^\top\nabla\phi(X)\right\rangle \\
&= \langle \nabla f(X), \nabla\phi(X)\rangle - \left\langle X^\top \nabla f(X), \mathrm{sym}(X^\top\nabla\phi(X))\right\rangle \\
&= \langle \nabla f(X), \nabla\phi(X)\rangle - \left\langle \nabla f(X), X\mathrm{sym}(X^\top\nabla\phi(X))\right\rangle \\
&= \left\langle \nabla f(X), \mathcal{P}_{T_X \mathrm{St}(d,r)}(\nabla\phi(X))\right\rangle.
\end{aligned}$$

Then, it follows from (8) that

$$\begin{aligned}
&f(\bar{X}_{k+1}) - f(\bar{X}_k) \leq \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - \bar{X}_k\right\rangle + \frac{L}{2}\|\bar{X}_{k+1} - \bar{X}_k\|^2 \\
&\leq \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - X_{k+1} + X_k - \bar{X}_k\right\rangle + \left\langle \mathrm{grad} f(\bar{X}_k), X_{k+1} - X_k\right\rangle \\
&\quad + 2L\|X_{k+1} - X_k\|^2 \\
&\leq \left\langle \mathrm{grad} f(\bar{X}_k), \bar{X}_{k+1} - X_{k+1}\right\rangle + \left\langle \mathrm{grad} f(\bar{X}_k), X_{k+1} - X_k\right\rangle \\
&\quad + 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
&= \left\langle \mathrm{grad} f(\bar{X}_k) - \mathrm{grad} f(\bar{X}_{k+1}), \bar{X}_{k+1} - X_{k+1}\right\rangle + \langle \mathrm{grad} f(X_k), X_{k+1} - X_k\rangle \\
&\quad + \left\langle \mathrm{grad} f(\bar{X}_k) - \mathrm{grad} f(X_k), X_{k+1} - X_k\right\rangle \\
&\quad + 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
&\leq 2\hat{L}^2\|X_{k+1} - X_k\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 - \alpha\|\mathrm{grad} f(X_k)\|^2 \qquad (17) \\
&\quad - \mu\langle \mathrm{grad} f(X_k), \nabla\varphi(X_k)\rangle + \frac{1}{2}(\hat{L}^2\|X_k - \bar{X}_k\|^2 + \|X_{k+1} - X_k\|^2) \\
&\quad + 4L(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
&\leq -\alpha\|\mathrm{grad} f(X_k)\|^2 - \mu\left\langle \nabla f(X_k), \mathcal{P}_{T_{X_k}\mathrm{St}(d,r)}(\nabla\varphi(X_k))\right\rangle + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\
&\quad + \frac{1}{2}\|X_k - \bar{X}_k\|^2 + (4\hat{L}^2 + 4L + 1)(\alpha^2\|\mathrm{grad} f(X_k)\|^2 + \mu^2\|\nabla\varphi(X_k)\|^2) \\
&\leq -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\|\mathrm{grad} f(X_k)\|^2 + \frac{1}{2}\|X_{k+1} - \bar{X}_{k+1}\|^2 \\
&\quad + (6\mu\hat{D}_f + \frac{1}{2} + 6(4\hat{L}^2 + 4L + 1)\mu^2)\|X_k - \bar{X}_k\|^2,
\end{aligned}$$

where the second inequality is from the 2-Lipschitz continuity of $\mathcal{P}_{\mathrm{St}(d,r)}$ over $\bar{U}_{\mathrm{St}(d,r)}(\frac{1}{8})$, the third inequality is due to the facts that $X_k - \bar{X}_k \in N_{\bar{X}_k}\mathrm{St}(d,r)$ and $\langle A, B\rangle \le \frac{1}{2}(\|A\|^2 + \|B\|^2)$ for any $A, B \in \mathbb{R}^{d\times r}$, and the last inequality comes from

$$\|\mathcal{P}_{T_{X_k}\mathrm{St}(d,r)}(\nabla\varphi(X_k))\| = \|X_k(X_k^\top X_k - I)^2\| \le 6\|X_k - \bar{X}_k\|^2.$$

Plugging $\mu = \frac{1}{3}$ into (17) gives (11). $\square$

Proof of Theorem.

Proof. First, we show $X_k \in \bar{U}_{\mathrm{St}(n,d)}(\frac{1}{8})$ for any $k \ge 0$ if $\alpha \le \frac{1}{45\hat{D}_f}$. In fact, by proof of induction, we have from (10) that

$$\|X_{k+1} - \bar{X}_{k+1}\| \le \sqrt{\frac{2}{3}}\|X_k - \bar{X}_k\| + \frac{1}{45\hat{D}_f}\|\mathrm{grad} f(X_k)\| \le \frac{1}{8}.$$

Moreover, applying (Xu et al., 2015, Lemma 2) to (10) yields

$$\sum_{k=0}^{K}\|X_k - \bar{X}_k\|^2 \le 60\alpha^2\sum_{k=0}^{K}\|\mathrm{grad} f(\bar{X}_k)\|^2 + 4. \tag{18}$$

Then, summing (11) over $k = 0, \ldots, K$ gives

$$\begin{aligned}
&f(\bar{X}_{K+1}) - f(\bar{X}_0)\\
\le& -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2)\sum_{k=0}^{K}\|\mathrm{grad} f(X_k)\|^2\\
&+ \frac{1}{2}\left(4\hat{D}_f + 8\hat{L}^2 + 8L + 4\right)\sum_{k=0}^{K+1}\|X_k - \bar{X}_k\|^2\\
\le& -(\alpha - (4\hat{L}^2 + 4L + 1)\alpha^2 + 30(4\hat{D}_f + 8\hat{L}^2 + 8L + 4)\alpha^2)\sum_{k=0}^{K}\|\mathrm{grad} f(X_k)\|^2\\
&+ \frac{1}{2}\left(4\hat{D}_f + 8\hat{L}^2 + 8L + 4\right)(60\alpha^2\|\mathrm{grad} f(X_{K+1})\|^2 + 4).
\end{aligned} \tag{19}$$

Define $c_1 = 244\hat{L}^2 + 244L + 120\hat{D}_f + 121$ and $c_2 = (30\hat{D}_f^2 + 2)(4\hat{D}_f + 8\hat{L}^2 + 8L + 4)$. Then, we have

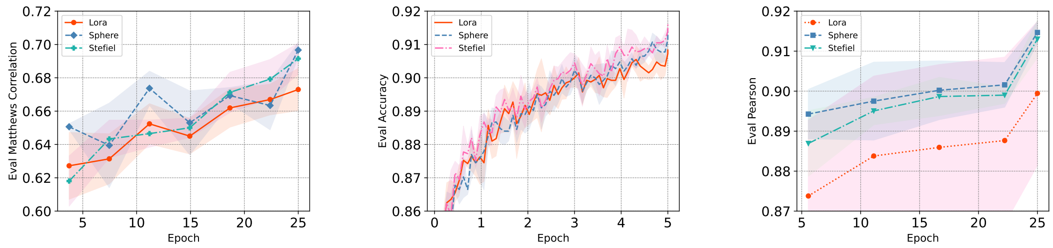
$$\alpha(1 - c_1\alpha)\sum_{k=0}^{K}\|\mathrm{grad} f(X_k)\|^2 \le f(\bar{X}_0) - f(\bar{X}_{K+1}) + c_2.$$

Therefore, for any $\alpha \le \frac{1}{2c_1}$ (which implies $\alpha \le \frac{1}{45\hat{D}_f}$), taking $K \to \infty$ gives $\sum_{k=0}^{\infty}\|\mathrm{grad} f(X_k)\|^2 < \infty$. Then by (12), $\sum_{k=0}^{\infty}\|X_k - \bar{X}_k\|^2 < \infty$. These lead to (12). $\square$

B Experimental Details

Baselines We compare our approach against several baseline methods, including full fine-tuning, Adapter (Houlsby et al., 2019), BitFit (Zaken et al., 2021) and LoRA (Hu et al., 2021). The variants of the Adapter method are excluded from the baselines, as their performance are relatively similar.

Implementation Details Our code is based on Pytorch (Paszke et al., 2019), Huggingface Transformers (Wolf et al., 2020) and an open-source plug-and-play library for parameter-efficient fine-tuning opendelta (Hu et al., 2023). The bottleneck dimension for the Adapter is set to 16 or 32, ensuring that the number of trainable parameters aligns closely with that of the LoRA method and the new layers are inserted into the attention layer and feed-forward layer. The update of LoRA is scaled by a hyper-parameter α. This value is typically left unmodified, as it is usually set as 16 or 32 and never tuned (Hu et al., 2021; Yang & Hu, 2020). The exponential moving average parameters β_1 and β_2 of AdamW (Loshchilov & Hutter, 2017) are set to their default values of 0.9 and 0.999, respectively. All the experiments are conducted on NVIDIA A800 GPUs.



(a) CoLA evaluation matthews correlation

(b) QQP evaluation accuracy

(c) STS-B evaluation pearson

Figure 4: Performance on the validation sets across three datasets. The COLA dataset is evaluated using the matthews correlation metric, QQP is measured by accuracy, and STS-B is evaluated by Pearson correlation, all plotted against the number of epochs.

B.1 Experimental results

We present the omitted experimental results in Section 5. We plot the evaluation loss during training to further demonstrate that Manifold-LoRA not only accelerates the optimization process but also achieves better performance metrics more quickly in comparison to the vanilla Adam optimizer. This highlights Manifold-LoRA's effectiveness in reaching superior results faster during evaluation.

Manifold-LoRA yields a faster convergence rate. As shown in Figure 4c, both Oblique and Stiefel constrained have a pronounced convergence speed improvement compared to the vanilla LoRA, simultaneously achieving better performance.

Manifold-LoRA typically maintains lower variance compared to other methods. The plotted results represent the average performance over five random seeds, with the shaded regions indicating the variance. As shown in Figure 4, the variance (shaded area) for Manifold-LoRA is smaller compared to LoRA, demonstrating its more stable performance.

B.2 Hyperparameters

In this section, we list the hyperparameters used in GLUE benchmark, question answering and E2E benchmark. To make a fair comparison, All hyperparameters such as Batch size, learning rate scheduler remain the same across experiments, except the additional parameters introduced by the Manifold-LoRA.

Table 4: Hyperparameter setup of Manifold-LoRA for E2E benchmark.

Method	Hyperparamter	GPT-2(M)	GPT-2(L)
	Warmup Steps	500	
	LR Schedule	Linear	
	Weight Decay	0.01	
	β_1	0.9	
	β_2	0.999	
	LoRA dropout	0	
	Batch Size	8	
	Learning Rate	2e-4	
	Epochs	5	
Sphere(r=4)	μ	1	0.9
	Lower	0.5	0.5
	Upper	2	2
Stiefel(r=4)	μ	1	1.1
	Lower	0.5	0.5
	Upper	4	2

Table 5: Hyperparameter setup of Manifold-LoRA for question answering tasks. For LoRA and our algorithms, new layers are inserted into $W_q, W_k, W_v, W_o, FC_1, FC_2$.

Method	Hyperparamter	SQuADv1.1	SQuADv2.0
	Warmup Ratio	0.06	
	LR Schedule	Linear	
	Weight Decay	0.1	
	β_1	0.9	
	β_2	0.999	
	Batch Size	64	
	Learning Rate	3e-3	
	Epochs	4	
Sphere(r=8)	μ	0.85	0.85
	Lower	0.25	0.25
	Upper	0.75	0.5
Sphere(r=16)	μ	0.9	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5
Stiefel(r=8)	μ	0.85	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5
Stiefel(r=16)	μ	0.9	0.85
	Lower	0.25	0.25
	Upper	0.5	0.5

Table 6: Hyperparameter configurations of Manifold-LoRA for GLUE benchmark

Method	Hyperparameter	MNLI	SST-2	CoLA	QQP	QNLI	RTE	MRPC	STS-B
	Warmup Ratio	0.06							
	LR Schedule	Linear							
	Max Sequence Length	256							
	Weight Decay	0.1							
	β_1	0.9							
	β_2	0.999							
	Batch Size	32							
	LoRA Layer	W_q, W_v							
	Epochs	7	24	25	5	5	50	30	25
	Learning rate	5e-4	8e-4	5e-4	5e-4	1.2e-3	1.2e-3	1e-3	2.2e-3
Sphere(r=16)	μ	1	0.9	0.8	0.9	0.95	1.2	0.85	0.9
	Lower	0.25	0.25	0.5	0.5	0.5	0.5	1	1
	Upper	2	2	2	4	2	2	4	4
Sphere(r=8)	μ	0.95	0.95	1	0.9	1	0.9	0.85	1
	Lower	2	0.5	1	0.5	0.5	0.25	2	1
	Upper	8	2	8	2	2	0.5	4	8
Stiefel(r=16)	μ	0.8	0.85	0.95	0.9	0.95	1.2	0.8	1
	Lower	2	0.5	2	0.5	0.5	0.5	1	1
	Upper	8	1	8	4	1	2	4	16
Stiefel(r=8)	μ	0.8	0.95	0.95	0.9	0.85	0.9	1	1
	Lower	2	0.5	2	0.5	0.5	0.25	1	1
	Upper	8	2	8	2	2	1	4	16