Local Linear Convergence of Infeasible Optimization With Orthogonal Constraints

Youbang Sun[ID], Shixiang Chen[ID], Alfredo Garcia[ID], *Senior Member, IEEE*, and Shahin Shahrampour[ID]

Abstract—Many classical and modern machine learning algorithms require solving optimization tasks under orthogonality constraints. Solving these tasks with feasible methods requires a gradient descent update followed by a retraction operation on the Stiefel manifold, which can be computationally expensive. Recently, an infeasible retraction-free approach, termed the landing algorithm, was proposed as an efficient alternative. Motivated by the common occurrence of orthogonality constraints in tasks such as principle component analysis and training of deep neural networks, this letter studies the landing algorithm and establishes a novel linear convergence rate for smooth non-convex functions using only a local Riemannian PŁ condition. Numerical experiments demonstrate that the landing algorithm performs on par with the state-of-the-art retraction-based methods with substantially reduced computational overhead.

Index Terms—Riemannian optimization, orthogonality constraints, infeasible methods, linear convergence.

I. INTRODUCTION

ORTHOGONALITY constraints naturally appear in many machine learning problems, ranging from the classical principal component analysis (PCA) [17] and canonical correlation analysis (CCA) [18] to decentralized spectral analysis [19], low-rank matrix approximation [20] and dictionary learning [23]. More recently, due to the distinctive properties of orthogonal matrices, orthogonality constraints and regularization methods have been used for training deep neural networks [6], [30], providing improvements in model

robustness and stability [29], as well as for adaptive fine-tuning in large language models [36]. Optimization with orthogonality constraints is typically formulated as,

$$\min_{x \in \mathbb{R}^{d \times r}} f(x),$$
$$\text{s.t.} \quad x \in \text{St}(d, r) \triangleq \left\{ x \in \mathbb{R}^{d \times r} | x^\top x = I_r \right\}, \quad (1)$$

where $\text{St}(d, r)$ is referred to as the *Stiefel* manifold.

The Stiefel manifold can be seen as an equality constraint that is a non-convex set. As a result, traditional first-order optimization algorithms (e.g., gradient descent) fail to solve Problem (1) directly. The topic of functional constrained optimization [10] has recently emerged; although these works address tasks with non-convex constraints, they mostly require hierarchical algorithms for convergence. Another common approach is to consider a Riemannian optimization algorithm, where in each iteration, the algorithm takes a Riemannian gradient step instead of a Euclidean one. Then, the algorithm performs a retraction operation to ensure feasibility and stay on the manifold [4]. It has been shown by the previous literature that these retraction-based algorithms can achieve similar iteration complexity rates [11], [34] compared to their Euclidean counterparts.

While achieving favorable iteration complexity and satisfying the feasibility constraint in (1), these methods are computationally expensive due to the retraction operation in each iteration. A typical retraction operation on the Stiefel manifold usually requires calculating a matrix exponential, inversion, or square root. This means that a typical retraction requires $\mathcal{O}(dr^2)$ or more algebraic operations. When r is large, the calculation time for retractions could dominate the runtime of Riemannian algorithms. In addition, matrix inversion or exponentiation have generally not been well-suited to execute on GPUs until recently [5], making the application of (1) in modern machine learning computationally prohibitive.

As a mitigation, recent works have proposed *infeasible* approaches to solve manifold optimization problems. These methods do not require explicit retraction operations (or other projections) and are often referred to as *retraction-free* methods. While retraction-free optimization iterates are not strictly feasible, they gradually converge to the feasibility region.

In this letter, we focus on a recently proposed retraction-free (or infeasible) method for solving Problem 1, called the *landing algorithm* [1], [2]. Although [2] provided a global finite-time convergence guarantee for this algorithm,

the numerical results suggested that the actual convergence speed is much faster. Our work seeks to study this empirical observation and provide a theoretical justification for the faster convergence while exploring the possible assumptions required on the objective function in order to achieve this convergence guarantee. To this end,

1) In Section II, we introduce a background for the optimization on the Stiefel manifold and provide the retraction-free landing algorithm.
2) In Section III, we flesh out the set of assumptions on the objective function, notably the local Riemannian Polyak-Łojasiewicz (PŁ) condition on the Stiefel manifold. We then introduce the merit function for our analysis and provide a *local linear* convergence guarantee for the landing algorithm.
3) In Section IV, we present numerical experiments and demonstrate the advantage of the landing algorithm in terms of convergence speed and efficiency in a high-dimensional PCA task as well as training convolutional neural networks (CNNs).

A. Related Literature

1) Retraction-Based Riemannian Optimization: The classical optimization algorithms under Riemannian geometry rely on the diffeomorphic nature between a Riemannian manifold and the Euclidean space. A classical approach is the line-search method; this includes Armijo line search and accelerated line search [3]. However, these approaches are rarely used in modern machine learning with large datasets, since multiple evaluations along a geodesic are often undesirable. As a mitigation, many methods such as Riemannian gradient descent [34] use retractions to ensure the feasibility of the iterates. With both retractions and parallel transports, a wide range of optimization algorithms have been adapted to the manifold context. These methods include Riemannian accelerated gradient [35], Riemannian conjugate gradient [25], and adaptive Riemannian gradient methods [9].

2) Retraction-Free Approaches: In order to ensure the feasibility of the manifold constraint, the previously mentioned algorithms still require costly operations, such as retractions and parallel transports. Recently, several approaches have been proposed in an attempt to reformulate the constrained problem in (1) into equivalent unconstrained problems, which can be solved efficiently. Reference [26] introduced ODCGM, using a non-smooth penalty to ensure that stationarity is strictly recovered in the alternative merit function. Another approach stems from Fletcher's penalty function [14] and implements a smooth merit function. Reference [15] proposed a proximal linearized augmented Lagrangian algorithm (PLAM) and solved the optimization problem with an augmented Lagrangian method. Reference [33] proposed PenC, an exact penalty function model, and implemented a projected inexact gradient algorithm. PenC was later extended to a constraint dissolving function (CDF) approach in [32], with examples covering a wide range of manifolds. In this letter, our focus is on proving fast local convergence for the landing algorithm [1], [2] as it provides a simple approach to retraction-free optimization on the Stiefel manifold, and the empirical evidence in [2] suggested fast convergence speed around stationary points.

II. PROBLEM FORMULATION

A. Notations

We start this section with definitions and notations used in this letter. We denote the transpose of vector x (and matrix A) as $x^\top$ (and $A^\top$), respectively. The Frobenius norm of matrix A is denoted as $\|A\|_F$. The skew and symmetric parts of a square matrix A are denoted as $\text{skew}(A) = \frac{1}{2}(A - A^\top)$ and $\text{sym}(A) = \frac{1}{2}(A + A^\top)$, respectively. The matrix inner product of matrices A and B is defined as $\langle A, B \rangle \triangleq Tr(AB^\top)$. We denote the identity matrix of rank r as I_r. Given a set $\mathcal{S}$, we represent $\mathcal{D}(\mathcal{S}, \delta) \triangleq \{x \mid \text{dist}(\mathcal{S}, x) \leq \delta\}$, where $\text{dist}(\mathcal{S}, x) = \min_{y \in \mathcal{S}} \|x - y\|_F$.

B. Riemannian Optimization

In the Euclidean space, the function gradient $\nabla f(x)$ is calculated as the differential of f evaluated at x. The classical gradient descent (GD) algorithm seeks to minimize the objective function by taking a step towards the negative Euclidean gradient, and the iterate is defined as $x_{k+1} = x_k - \alpha \nabla f(x_k)$, with α denoting the step size. On the other hand, the Riemannian gradient $\text{grad} f(x)$ is defined on the tangent space of a point x on the manifold with respect to a given metric. For the Stiefel manifold specifically, the Riemannian gradient with respect to the canonical metric [13] is given by the following closed-form expression:

$$\text{grad} f(x) = \text{skew}\left(\nabla f(x)x^\top\right)x. \tag{2}$$

The Riemannian GD uses $\text{grad} f(x)$ to perform a retraction operation on the iterate, which allows the iterate to move while staying on $\text{St}(d, r)$; the Riemannian GD algorithm updates the iterates as

$$x_{k+1} = \mathcal{R}_{x_k}(-\alpha \text{grad} f(x_k)),$$

where $\mathcal{R}_x : T_x\text{St}(d, r) \to \text{St}(d, r)$ denotes the retraction operation, which maps vectors in the tangent space at point x to points on the Stiefel manifold. Though there are multiple choices of retraction operations on the Stiefel manifold, including exponential retraction, Cayley retraction, and projection retraction, all of them require high computational costs. To address this issue, many recent application-focused works such as deep learning-related studies [8] use a penalty function instead and aim to minimize $f(x) + \lambda \mathcal{N}(x)$, where $\mathcal{N}(x)$ penalizes the constraint violation. Although these methods are conceptually simple and have relatively low computational complexity, they only find an approximately accurate solution, and the end product is often infeasible.

C. Retraction-Free Algorithm

Contrary to the existing retraction-based and penalty-based algorithms, this letter focuses on the landing algorithm proposed by [1] as an infeasible, retraction-free numerical solver for Problem (1). Unlike retraction-based algorithms, the iterates of the landing algorithm do not strictly stay on the Stiefel manifold, avoiding high computational costs. The landing algorithm also differs from conventional penalty methods as it converges to a critical point on the manifold.

The landing algorithm updates are based on the landing field, defined as

$$\Lambda(x) \triangleq \text{grad} f(x) + \lambda \nabla p(x), \tag{3}$$

Algorithm 1 Retraction-Free Landing Algorithm

1: **Input:** initial point $x_0 \in \text{St}(d, r)^\epsilon$, $\alpha > 0, \lambda > 0, \epsilon \in (0, \frac{3}{4})$.

2: **for** $k = 0, 1, \ldots$ **do**

3:    Calculate the landing field $\Lambda(x)$ and update x as:

$$\Lambda(x_k) = \text{grad} f(x_k) + \lambda x_k \left(x_k^\top x_k - I_r \right),$$

$$x_{k+1} = x_k - \alpha \Lambda(x_k).$$

4: **end for**

---

where the penalty function $p(x) \triangleq \frac{1}{4} \| x^\top x - I_r \|_F^2$ and $\lambda > 0$ is a constant. The exact update for the landing algorithm is provided in Algorithm 1, where α denotes the step size. Though the iterate x_k may not belong to $\text{St}(d, r)$, $\text{grad} f(x_k)$ is still well-defined as in (2) and is referred to as the *relative gradient*.

Observe that when x is on the manifold, the penalty function $p(x)$ and its gradient $\nabla p(x)$ are equal to zero; hence, $\Lambda(x) = \text{grad} f(x)$ for $x \in \text{St}(d, r)$. In addition, the two terms of the landing field are orthogonal to each other, since $\langle \text{grad} f(x), \nabla p(x) \rangle = \langle \text{skew}(\nabla f(x) x^\top), x(x^\top x - I_r) x^\top \rangle = 0$ [2]. Therefore, $\Lambda(x) = 0$ if and only if $\nabla p(x) = 0$ and $\text{grad} f(x) = 0$, which shows that $\Lambda(x) = 0$ when x is a first-order stationary point of $f(x)$ on the manifold.

While the landing algorithm allows the iterates to move outside of the manifold, one must ensure that they stay close enough to the manifold within a *safety region*.

Definition 1 (Safety Region [2]): With $\epsilon \in (0, 3/4)$, we define the safety region as

$$\text{St}(d, r)^\epsilon \triangleq \left\{ x \in \mathbb{R}^{d \times r} \mid \left\| x^\top x - I_r \right\|_F \le \epsilon \right\}. \tag{4}$$

It has been shown that for the landing field $\Lambda(x)$ defined in (3), a safety step size exists such that the algorithm never leaves the safety region.

Proposition 1 (Safety Step Size [2]): Let $x_k \in \text{St}(d, r)^\epsilon$. Assuming that $\| \text{grad} f(x_k) \|_F \le G$, the next iterate satisfies $x_{k+1} \in \text{St}(d, r)^\epsilon$ if the step size α satisfies

$$\alpha \le \alpha_{safe} \triangleq \min \left\{ \frac{\lambda \epsilon (1 - \epsilon)}{G^2 + \lambda^2 (1 + \epsilon) \epsilon^2}, \sqrt{\frac{\epsilon}{2G^2}}, \frac{1}{2\lambda} \right\}.$$

III. THEORETICAL ANALYSIS

In this section, we first state our theoretical assumptions and then introduce the merit function for our analysis. Next, we provide our main theoretical result, which is the local linear convergence of the landing algorithm.

A. Technical Assumptions

Let us start by introducing the following assumptions on the objective function in (1). These assumptions are considered to be standard in the literature [16], [34] and are necessary for our improved convergence guarantee.

Assumption 1 (Lipschitz Smoothness): We assume that the function $f : \mathbb{R}^{d \times r} \to \mathbb{R}$ is twice continuously differentiable and L-smooth, i.e., for all $x, y \in \mathbb{R}^{d \times r}$, we have

$$f(y) \le f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \| y - x \|_F^2. \tag{5}$$

Different from the retraction-based algorithms, since the iterates can go outside the manifold in the infeasible algorithms, we assume the smoothness of the function in the entire Euclidean space instead of the Stiefel manifold. We also note that the smoothness is defined with respect to the Euclidean gradient instead of the Riemannian gradient.

Next, we discuss the local gradient growth of the nonconvex function $f(x)$ in the sense of Polyak-Łojasiewicz, formally defined below.

Assumption 2 (Local Riemannian PŁ Condition): The objective function $f(x) : \mathbb{R}^{d \times r} \to \mathbb{R}$ satisfies the local Riemannian Polyak-Łojasiewicz (PŁ) condition on the Stiefel manifold with a factor $\mu > 0$ if

$$\left| f(x) - f_{\mathcal{S}}^* \right| \le \frac{1}{2\mu} \| \text{grad} f(x) \|_F^2, \tag{6}$$

for any point $x \in \text{St}(d, r) \cap \mathcal{D}(\mathcal{S}, 2\delta)$, where $\mathcal{S}$ denotes the set of all local minima with a given value $f_{\mathcal{S}}^*$.

Different from Assumption 1, the assumption on the PŁ condition is strictly on the manifold, and the condition deals with the Riemannian gradient $\text{grad} f(x)$ instead of the Euclidean gradient $\nabla f(x)$.

The local Riemannian PŁ condition is less restrictive compared to other commonly studied scenarios. It is a relaxation of the global Riemannian PŁ condition, which could be restrictive in this case since the Stiefel manifold has a positive curvature. This assumption is also easier to satisfy compared to the geodesic strong convexity. Many traditional tasks involving the Stiefel manifold, such as the PCA problem and the generalized quadratic problem, satisfy the local PŁ condition but do not meet the other two conditions mentioned above. These are sometimes referred to as quadratic problems and have been shown to satisfy Assumption 2, demonstrated in [22].

B. Merit Function

To better understand the dynamics of the landing algorithm, we consider the following merit function [2], which is defined with respect to the objective in (1) as

$$\mathcal{L}(x) \triangleq f(x) + h(x) + \gamma p(x),$$

$$\text{and} \quad h(x) \triangleq -\frac{1}{2} \langle \text{sym}\left(x^\top \nabla f(x) \right), x^\top x - I_r \rangle. \tag{7}$$

From the definition of $\mathcal{L}$, one can prove that for any point $x \in \text{St}(d, r)$, $\nabla \mathcal{L}(x) = \nabla f(x) - x \, \text{sym}(x^\top \nabla f(x))$ (see eq. (44) in [2]). Similar to [2, Proposition 6], for a given $\epsilon \in (0, 3/4)$, γ is required to satisfy the following condition:

$$\gamma \ge \frac{2}{3 - 4\epsilon} \left(L(1 - \epsilon) + 3s + \hat{L}^2 \frac{(1 + \epsilon)^2}{\lambda (1 - \epsilon)} \right),$$

where $s = \sup_{x \in \text{St}(d, r)^\epsilon} \left\| \text{sym}\left(x^\top \nabla f(x) \right) \right\|_F$, and

$$\hat{L} = \max \left(L, \max_{x \in \text{St}(d, r)^\epsilon} \| \nabla f(x) \|_F \right). \tag{8}$$

The following proposition on the properties of $\mathcal{L}(x)$ is also given in [2] and is useful for our technical analysis.

Proposition 2: The merit function $\mathcal{L}(x)$ satisfies the following properties.

1) $\mathcal{L}(x)$ is $L_{\mathcal{L}}$-smooth on $x \in \text{St}(d, r)^\epsilon$, with $L_{\mathcal{L}} \le L_{f+h} + (2 + 3\epsilon)\gamma$, where L_{f+h} is the smoothness of $f + h$.

2) For $\rho = \min\{\frac{1}{2}, \frac{\gamma}{4\lambda(1+\epsilon)}\}$ and $x \in \mathrm{St}(d, r)^\epsilon$, we have

$$\langle \Lambda(x), \nabla \mathcal{L}(x) \rangle \geq \rho \|\Lambda(x)\|_{\mathrm{F}}^2.$$

The first property in Proposition 2 shows that $\mathcal{L}$ is indeed a smooth function. The second property establishes a relationship between the landing field and the gradient of the merit function. Since $\langle \Lambda(x), \nabla \mathcal{L}(x) \rangle \geq 0$, the landing field always points towards an ascent direction for the merit function. Therefore, a landing step can be seen as a descent step on the merit function. Moreover, since $\langle \mathrm{grad} f(x), \nabla p(x) \rangle = 0$, the second property shows that within the neighborhood $\mathrm{St}(d, r)^\epsilon$, if $\nabla \mathcal{L}(x) = 0$, we have $\nabla p(x) = 0$ and $\mathrm{grad} f(x) = 0$.

In addition, due to the Lipschitz smoothness of f, it is easy to show that $\Lambda(x)$ is first-order Lipschitz continuous; we denote its Lipschitz constant as L_Λ. L_Λ and $L_\mathcal{L}$ can be set to be on the same order as L given ideal λ and γ. For the ease of analysis below, we denote $L' \triangleq \max\{\hat{L}, L_\Lambda, L_\mathcal{L}\}$.

C. Local Linear Convergence of Landing Algorithm

In this section, we study the convergence of the landing algorithm under the local Riemannian PŁ condition. Since this letter focuses on the local convergence guarantees of the landing algorithm, and that Assumption 2 is only applicable to $x \in \mathrm{St}(d, r) \cap \mathcal{D}(\mathcal{S}, 2\delta)$, we only characterize the convergence rate assuming that the iterates have reached a neighborhood of a local minimum $x^* \in \mathcal{S}$. This assumption is not surprising; even some classical methods (e.g., Newton's method) achieve faster local rates when initialization is close enough to a local minimizer. Nevertheless, one can guarantee that iterates of Algorithm 1 can reach a neighborhood of a *stationary* point given the general convergence theorem in [27] as well as the global finite-time convergence of the landing algorithm in [2].

Without loss of generality, we assume $f_\mathcal{S}^* = 0$ and omit $f_\mathcal{S}^*$ when applicable. We first present the following lemma to show that for any x in the neighborhood of a local minimizer and in the safety region, the merit function is locally dominated by the landing field.

Lemma 1 (Pseudo Gradient Domination [28]): Let $x \in \mathrm{St}(d, r)^\epsilon \cap \mathcal{D}(\mathcal{S}, \delta)$, and without the loss of generality assume $f_\mathcal{S}^* = 0$. Then, under Assumptions 1 and 2, we have,

$$\mathcal{L}(x) \leq \frac{1}{\mu'} \|\Lambda(x)\|_{\mathrm{F}}^2, \tag{9}$$

where $\frac{1}{\mu'} = \max\{\frac{1}{\mu}, \frac{2(3+2\epsilon)^2 \hat{L}^2 + \mu L'}{2\mu\lambda^2(1-\epsilon)^2}\}$.

We note that μ' depends on a number of factors. In practice, if we set $\lambda = \mathcal{O}(L')$ and also L' and $\hat{L}$ are of the same order, the two terms in μ' are of the same order, and we have $\mu' \approx \mu$.

Next, we use the following lemma to demonstrate the quadratic growth of the merit function $\mathcal{L}(x)$ around the set of local minima $\mathcal{S}$.

Lemma 2 (Quadratic Growth): Let Assumptions 1 and 2 hold. For $x \in \mathrm{St}(d, r)^\epsilon \cap \mathcal{D}(\mathcal{S}, \delta)$, the merit function $\mathcal{L}(x)$ satisfies

$$\mathcal{L}(x) \geq \frac{\mu'\rho^2}{4} \mathrm{dist}(\mathcal{S}, x)^2. \tag{10}$$

Proof: Since $x \in \mathrm{St}(d, r)^\epsilon$, given Proposition 2, it is clear that $\|\nabla \mathcal{L}(x)\|_{\mathrm{F}} \geq \rho \|\Lambda(x)\|_{\mathrm{F}}$. Combined with Lemma 1, we have,

$$\mathcal{L}(x) \leq \frac{1}{\mu'} \|\Lambda(x)\|_{\mathrm{F}}^2 \leq \frac{1}{\mu'\rho^2} \|\nabla \mathcal{L}(x)\|_{\mathrm{F}}^2.$$

Therefore, we know that the local Euclidean PŁ condition for $\mathcal{L}(x)$ holds on $\forall x \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$. Given [24, Proposition 2.2], the local Euclidean PŁ condition on $\mathcal{L}(x)$ implies the quadratic growth relationship (10). ∎

With the introduction of the previous two lemmas, we can now present our theorem. We show that when x is in the neighborhood of a local minimizer, Algorithm 1 exhibits a local linear convergence under Assumption 2.

Theorem 1: Let function f satisfy Assumptions 1 and 2, and the initial point $x_0 \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$ is such that $\mathcal{L}(x_0) \leq \frac{\mu'\rho^2\delta^2}{16}$. Choose the step size $0 < \alpha \leq \min\{\frac{\rho}{L'}, \alpha_{safe}\}$ and let iterates $\{x_k\}$ follow Algorithm 1. Then, the merit function defined in (7) converges linearly $\forall k \geq 0$, such that

$$\mathcal{L}(x_k) \leq \left(1 - \frac{\alpha\rho\mu'}{2}\right)^k \mathcal{L}(x_0). \tag{11}$$

Proof: We prove the theorem by induction. Suppose that $x_k \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$ and that $\mathcal{L}(x_k) \leq \frac{\mu'\rho^2\delta^2}{16}$ for some k. Since the merit function is L'-Lipschitz smooth, using the update of the landing algorithm, we get

$$\begin{aligned}
\mathcal{L}(x_{k+1}) &\leq \mathcal{L}(x_k) + \langle \nabla \mathcal{L}(x_k), x_{k+1} - x_k \rangle + \frac{L'}{2} \|x_{k+1} - x_k\|_{\mathrm{F}}^2 \\
&= \mathcal{L}(x_k) - \alpha \langle \nabla \mathcal{L}(x_k), \Lambda(x_k) \rangle + \frac{\alpha^2 L'}{2} \|\Lambda(x_k)\|_{\mathrm{F}}^2 \\
&\leq \mathcal{L}(x_k) - \alpha\rho \|\Lambda(x_k)\|_{\mathrm{F}}^2 + \frac{\alpha^2 L'}{2} \|\Lambda(x_k)\|_{\mathrm{F}}^2 \\
&\leq \left(1 - \mu'\left(\alpha\rho - \frac{\alpha^2 L'}{2}\right)\right) \mathcal{L}(x_k),
\end{aligned}$$

where the second to last line is from Proposition 2, and the last inequality is from Lemma 1. Now, if we choose a safe step size $0 < \alpha \leq \min\{\frac{\rho}{L'}, \alpha_{safe}\}$, we get $\alpha\rho - \frac{\alpha^2 L'}{2} \geq \frac{\alpha\rho}{2}$. This implies the monotonicity of $\mathcal{L}$, and

$$\mathcal{L}(x_{k+1}) \leq \left(1 - \frac{\alpha\rho\mu'}{2}\right) \mathcal{L}(x_k). \tag{12}$$

Given that $x_k \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$, from Lemma 2 we get $\mathrm{dist}(\mathcal{S}, x_k) \leq \sqrt{\frac{4\mathcal{L}(x_k)}{\mu'\rho^2}} \leq \frac{\delta}{2}$. Therefore, for the next iteration $k + 1$, we have

$$\begin{aligned}
\mathrm{dist}(\mathcal{S}, x_{k+1}) &\leq \mathrm{dist}(\mathcal{S}, x_k) + \|x_k - x_{k+1}\|_{\mathrm{F}} \\
&= \mathrm{dist}(\mathcal{S}, x_k) + \alpha \|\Lambda(x_k)\|_{\mathrm{F}} \\
&\leq \mathrm{dist}(\mathcal{S}, x_k) + \alpha L' \mathrm{dist}(\mathcal{S}, x_k) \leq \delta.
\end{aligned}$$

The last line stems from the Lipschitz continuity of $\Lambda(x)$ and the fact that $\alpha \leq \frac{\rho}{L'} \leq \frac{1}{2L'}$. This relationship combined with the safety guarantee ensures that for the next iteration, $x_{k+1} \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$ and that $\mathcal{L}(x_{k+1}) \leq \frac{\mu'\rho^2\delta^2}{16}$ due to (12).

Therefore, given the initial condition on x_0 and by recursion, $x_k \in \mathcal{D}(\mathcal{S}, \delta) \cap \mathrm{St}(d, r)^\epsilon$ and $\mathcal{L}(x_k) \leq \frac{\mu'\rho^2\delta^2}{16}$ is guaranteed to be satisfied by all k. Hence, we get

$$\mathcal{L}(x_k) \leq \left(1 - \frac{\alpha\rho\mu'}{2}\right)^k \mathcal{L}(x_0).$$

∎

Theorem 1 guarantees the exponential convergence of the merit function $\mathcal{L}(x)$ while ensuring the iterates never leave the safe neighborhood region. Using Lemma 2 and the Lipschitz
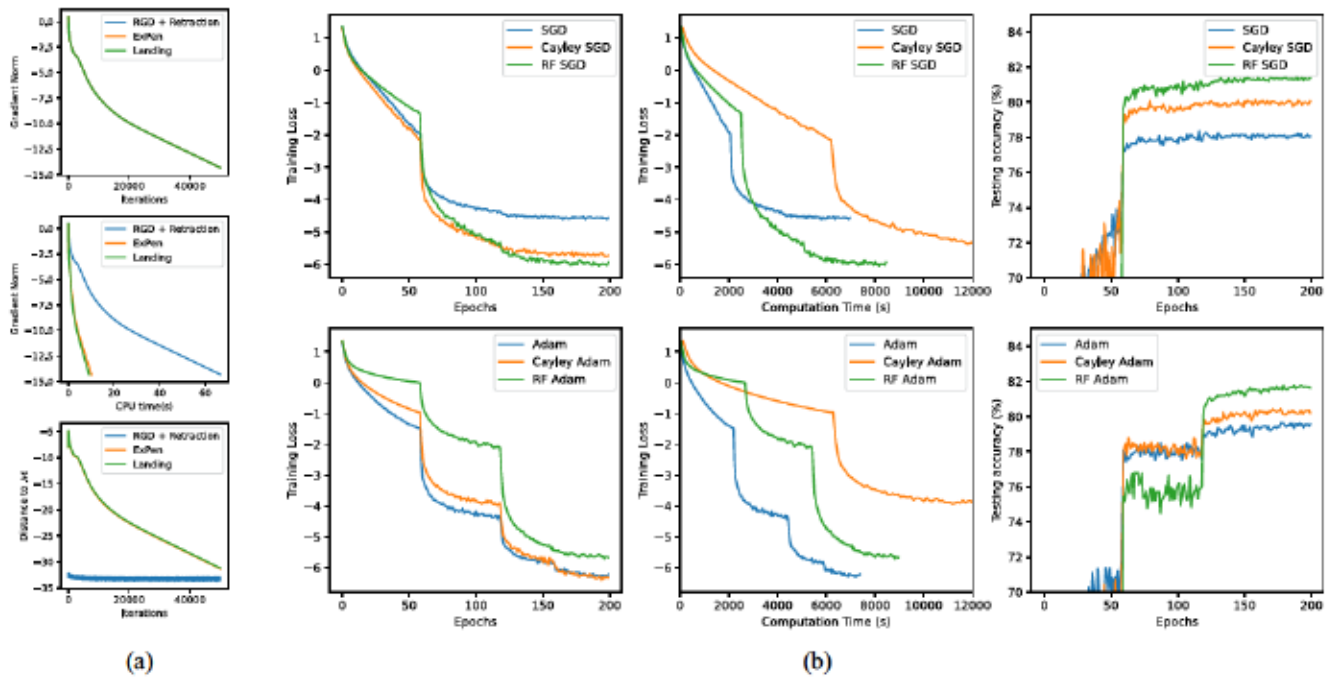
Fig. 1. (a) Performance of the landing algorithm on PCA. (b) Comparison of retraction-free methods vs. SGD and Adam-type optimizers and their Cayley counterparts for ResNet on CIFAR-100 dataset.

continuity of $\Lambda(x)$, linear convergence rates for $\mathrm{dist}(\mathcal{S}, x_k)$ and $\|\Lambda(x_k)\|_{\mathrm{F}}$ can be easily derived.

IV. NUMERICAL RESULTS

For the experiments, we first evaluate the landing algorithm on a PCA task with synthetic data (Section IV-A) and compare it with existing methods. Then, we evaluate it on training orthogonal-constrained CNNs (Section IV-B).

A. High-Dimensional PCA Task

We start the numerical results with a traditional PCA task. The objective function for PCA can be written as:

$$\max_{x \in \mathbb{R}^{d \times r}} \langle A^\top Ax, xD \rangle \quad \text{s.t. } x \in \mathrm{St}(d, r),$$

where $A \in \mathbb{R}^{m \times d}$ is the data matrix and D is a diagonal matrix with $[D]_{11} > \cdots > [D]_{rr} > 0$. We want to find the top-r principal components of the data matrix. Let us set $d = 500$, $r = 20$, $m = 1000$ in our experiment.

We compare the performance of the landing algorithm with two other iterative gradient-based algorithms.[1] We first consider a traditional Riemannian GD algorithm using projection as its retraction step. Secondly, we consider a state-of-the-art retraction-free approach named ExPen [31]. All the algorithms are run with the same hyper-parameters and with the same initial points x_0. The results are provided in Fig. 1(a).

It is easy to verify that as long as the data matrix A is not ill-conditioned and $rank(A) \geq r$, Assumption 2 is satisfied [22]. We plot the dynamics of gradient norm $\|\mathrm{grad} f(x)\|$ and the distance of iterates to $\mathcal{M}$, i.e., $\|x^\top x - I_r\|_{\mathrm{F}}$, with respect to iterations as well as the CPU time. (i) All three algorithms

exhibit similar linear convergence in terms of iterations, but the two retraction-free algorithms converge much faster in CPU time. (ii) Though the two retraction-free algorithms fail to strictly satisfy feasibility constraints initially, they eventually converge to a feasible critical point on $\mathrm{St}(d, r)$. (iii) While the landing algorithm performs similarly to ExPen, it does not require calculating the gradient of an alternative point other than x; not only is it more efficient, but also it is more suitable for implementation in modern neural networks, which we present in the next experiment.

B. Training CNNs Without Retractions

We adopt the experimental setting in [21] to compare retraction-free algorithms with the state-of-the-art *Cayley SGD* and *Cayley Adam*, as well as vanilla SGD and Adam, which are suitable for unconstrained Euclidean problems. Although no strict function assumptions are satisfied by neural networks due to the nature of ReLU activations, some works [7] suggested that the loss of wide neural networks can be coupled with that of a quadratic model. We refer to the retraction-free methods as RF-SGD and RF-Adam, respectively. For the adaptation of RF-Adam, we refer to [21, Algorithm 2] for details but replace the *Cayley_loop* with a retraction-free landing update.

We evaluate all algorithms with Wide ResNet (WRN-28-10) on the CIFAR-100 dataset, where generalization is an important metric. All hyper-parameters are adapted from [21] with no modifications. We present the results in Fig. 1(b), suggesting that orthogonality constraints (imposed in Riemannian optimization) may help with better generalization, a phenomenon also observed in other works (e.g., [12]). We note that the mini-batch training in neural networks introduces stochastic noise in gradient evaluations, negatively impacting

[1]The code for the PCA experiment is available at *https://github.com/sundave1998/Landing-Linear-Convergence*.

convergence. In our experiments, all three methods are run with a learning rate decay scheduler in accordance with the previous literature, resulting in a multi-stage convergence with jumps every 50 epochs. In terms of generalization, both RF-SGD and RF-Adam perform better than their Cayley counterparts [21], which suggests that compared to inexact Cayley updates, RF-based algorithms are perhaps more accurate and/or have more desirable properties in training neural networks.

We also plot the convergence of loss vs. computation time. Retraction-free (RF) methods have much faster convergence compared to Cayley-retraction methods. We note that the update in [21] is already an approximate solution with emphasis on efficiency, yet the landing algorithm helps with faster convergence and moderately improves the generalization.

V. CONCLUSION

In this letter, we studied a faster convergence for the retraction-free landing algorithm, addressing optimization under orthogonality constraints. We used the local Riemannian PŁ condition on the objective function to derive a local linear convergence rate for the landing algorithm. We also provided numerical results as a verification of the convergence speed while highlighting the computational efficiency of the algorithm compared to existing benchmarks.

For future research, generalization of these techniques to general manifold constraints, such as the Grassmann and Hyperbolic manifolds, is an interesting direction. In addition, given the wide application of orthogonality constraints in modern neural network structures, developing adaptive and momentum-based retraction-free optimization algorithms is another interesting open question.

REFERENCES

[1] P. Ablin and G. Peyré, "Fast and accurate optimization on the orthogonal manifold without retraction," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2022, pp. 5636–5657.
[2] P. Ablin, S. Vary, B. Gao, and P.-A. Absil, "Infeasible deterministic, stochastic, and variance-reduction algorithms for optimization under orthogonality constraints," 2023, *arXiv:2303.16510*.
[3] P. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ, USA: Princeton Univ. Press, 2008.
[4] P. Absil and J. Malick, "Projection-like retractions on matrix manifolds," *SIAM J. Optim.*, vol. 22, no. 1, pp. 135–158, 2012.
[5] M. Adil, R. Madani, S. Tavakkol, and A. Davoudi, "A first-order numerical algorithm without matrix operations," 2022, *arXiv:2203.05027*.
[6] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1120–1128.
[7] Y. Bai and J. D. Lee, "Beyond linearization: On quadratic and higher-order approximation of wide neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–35.
[8] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep CNNs?" in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.
[9] G. Bécigneul and O. Ganea, "Riemannian adaptive optimization methods," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–16.
[10] D. Boob, Q. Deng, and G. Lan, "Stochastic first-order methods for convex and nonconvex functional constrained optimization," *Math. Program.*, vol. 197, no. 1, pp. 215–279, 2023.
[11] N. Boumal, P.-A. Absil, and C. Cartis, "Global rates of convergence for nonconvex optimization on manifolds," *IMA J. Numer. Anal.*, vol. 39, no. 1, pp. 1–33, 2019.
[12] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," 2015, *arXiv:1511.06068*.
[13] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 2, pp. 303–353, 1998.
[14] R. Fletcher and S. Leyffer, "Nonlinear programming without a penalty function," *Math. Program.*, vol. 91, pp. 239–269, Jan. 2002.
[15] B. Gao, X. Liu, and Y. Yuan, "Parallelizable algorithms for optimization problems with orthogonality constraints," *SIAM J. Sci. Comput.*, vol. 41, no. 3, pp. A1949–A1983, 2019.
[16] A. Han, B. Mishra, P. Jawanpuria, P. Kumar, and J. Gao, "Riemannian hamiltonian methods for min-max optimization on manifolds," *SIAM J. Optim.*, vol. 33, no. 3, pp. 1797–1827, 2023.
[17] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. Educ. Psychol.*, vol. 24, no. 6, p. 417, 1933.
[18] H. Hotelling, "Relations between two sets of variates," in *Breakthroughs in Statistics: Methodology and Distribution*. New York, NY, USA: Springer, 1992, pp. 162–190.
[19] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *J. Comput. Syst. Sci.*, vol. 74, no. 1, pp. 70–83, 2008.
[20] N. K. Kumar and J. Schneider, "Literature survey on low rank approximation of matrices," *Linear Multilinear Algebra*, vol. 65, no. 11, pp. 2212–2244, 2017.
[21] J. Li, F. Li, and S. Todorovic, "Efficient Riemannian optimization on the Stiefel manifold via the Cayley transform," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–16.
[22] H. Liu, A. M.-C. So, and W. Wu, "Quadratic optimization with orthogonality constraint: Explicit Łojasiewicz exponent and linear convergence of retraction-based line-search and stochastic variance-reduced gradient methods," *Math. Program.*, vol. 178, no. 1, pp. 215–262, Nov. 2019.
[23] H. Raja and W. U. Bajwa, "Cloud K-SVD: A collaborative dictionary learning algorithm for big, distributed data," *IEEE Trans. Signal Process.*, vol. 64, no. 1, pp. 173–188, Jan. 2016.
[24] Q. Rebjock and N. Boumal, "Fast convergence to non-isolated minima: Four equivalent conditions for C^2 functions," *Math. Program.*, 2024, to be published.
[25] H. Sato, "A Dai–Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions," *Comput. Optim. Appl.*, vol. 64, pp. 101–118, May 2016.
[26] S. Schechtman, D. Tiapkin, M. Muehlebach, and E. Moulines, "Orthogonal directions constrained gradient method: From non-linear equality constraints to stiefel manifold," in *Proc. 30th 6th Annu. Conf. Learn. Theory*, 2023, pp. 1228–1258.
[27] R. Schneider and A. Uschmajew, "Convergence results for projected line-search methods on varieties of low-rank matrices via Łojasiewicz inequality," *SIAM J. Optim.*, vol. 25, no. 1, pp. 622–646, 2015.
[28] Y. Sun, S. Chen, A. Garcia, and S. Shahrampour, "Global convergence of decentralized retraction-free optimization on the Stiefel manifold," 2024, *arXiv:2405.11590*.
[29] A. Trockman and J. Kolter, "Orthogonalizing convolutional layers with the Cayley transform," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–21.
[30] E. Vorontsov, C. Trabelsi, S. Kadoury, and C. Pal, "On orthogonality and learning recurrent networks with long term dependencies," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3570–3578.
[31] N. Xiao and X. Liu, "Solving optimization problems over the Stiefel manifold by smooth exact penalty function," 2021, *arXiv:2110.08986*.
[32] N. Xiao, X. Liu, and K. Toh, "Dissolving constraints for Riemannian optimization," *Math. Oper. Res.*, vol. 49, no. 1, pp. 366–397, 2024.
[33] N. Xiao, X. Liu, and Y. Yuan, "A class of smooth exact penalty function methods for optimization problems with orthogonality constraints," *Optim. Methods Softw.*, vol. 37, no. 4, pp. 1205–1241, 2022.
[34] H. Zhang and S. Sra, "First-order methods for geodesically convex optimization," in *Proc. Conf. Learn. Theory*, 2016, pp. 1617–1638.
[35] H. Zhang and S. Sra, "Towards Riemannian accelerated gradient methods," 2018, *arXiv:1806.02812*.
[36] Q. Zhang et al., "Adaptive budget allocation for parameter-efficient fine-tuning," in *Proc. 11th Int. Conf. Learn. Represent.*, 2023, pp. 1–17.