No-regret Online Learning over Riemannian Manifolds

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider online optimization over Riemannian manifolds, where a learner attempts to minimize a sequence of time-varying loss functions defined on Riemannian manifolds. Though many Euclidean online convex optimization algorithms have been proven useful in a wide range of areas, less attention has been paid to their Riemannian counterparts. In this paper, we study a Riemannian online gradient descent algorithm (R-OGD) on Hadamard manifolds for both geodesically convex and strongly geodesically convex loss functions, and a Riemannian bandit algorithm (R-BAN) on Hadamard homogeneous manifolds for geodesically convex functions. We establish upper bounds on the regret of the algorithms with respect to time horizon T, manifold curvature κ, and manifold dimension d. We also find a universal lower bound for the achievable regret by an online convex optimization algorithm on Hadamard manifolds. All the obtained regret bounds match the corresponding results in Euclidean spaces. Finally, numerical experiments of the Karcher mean problem validate our theoretical results.

1 Introduction

The *online optimization* has been widely studied in the past decades in online routing, spam filtering, and machine learning [5, 20, 8]. Without assuming a prior knowledge of loss functions, an online convex optimization algorithm predicts solutions before the loss function is revealed.

In this paper, we consider the following Riemannian online convex optimization (R-OCO) problem,

$$\min_{x_t \in \mathcal{K} \subset \mathcal{M}} f_t(x_t), t = 1, 2, \ldots, T, \tag{1}$$

where $\mathcal{M}$ is a complete Riemannian manifold equipped with a Riemannian metric g and $\mathcal{K}$ is a geodesically convex (g-convex) subset of the Riemannian manifold $\mathcal{M}$. We assume that $\{f_t\}_{t=1,2,\ldots,T}$ is a sequence of unknown loss functions and every f_t is a geodesically convex (g-convex) function with sufficient smoothness. The R-OCO problem (1) extends the online convex optimization in Euclidean spaces with potential applications in machine learning, online principal component analysis (PCA), and dictionary learning [26, 17, 39, 22].

The R-OCO problem (1) can be understood as a learning process of T rounds. At each round $t = 1, 2, 3, \ldots, T$, the online learner chooses a strategy x_t from the g-convex subset $\mathcal{K}$. Later or simultaneously, the adversary (or nature) produces a g-convex loss function $f_t : \mathcal{K} \to \mathbb{R}$ of which the learner has no prior knowledge. Finally, a learner receives the feedback and suffers the loss $f_t(x_t)$. Generally, there are two types of information feedback in consideration. One is the full information feedback, where the entire function f_t is revealed to the learner; the other is the bandit feedback,

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Table 1: Comparison of regret between our work and corresponding results in Euclidean spaces. T: the time horizon; n: dimension of a manifold; ζ and Λ: constants relied on the sectional curvature bound κ, the dimension n and the domain $\mathcal{K}$.

	Full information, g-convex	Full information, strongly g-convex	Bandit feedback	Universal lower bound
Our Work	$\mathcal{O}\big(\zeta^{\frac{1}{2}}\sqrt{T}\big)$	$\mathcal{O}\big(\zeta \log T\big)$	$\mathcal{O}\big(n^{\frac{1}{2}}\zeta^{\frac{1}{4}}\Lambda T^{\frac{3}{4}}\big)$	$\mathcal{O}\big(\sqrt{T}\big)$
Euclidean	$\mathcal{O}\big(\sqrt{T}\big)$ [40]	$\mathcal{O}\big(\log T\big)$ [21]	$\mathcal{O}\big(n^{\frac{1}{2}}T^{\frac{3}{4}}\big)$[18]	$\mathcal{O}\big(\sqrt{T}\big)$ [1]

where only the value $f_t(x_t)$ is revealed. The goal of the R-OCO is to minimize the *regret*, defined as

$$R(T) = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} f_t(x),$$

which measures the difference between the cost by $\{x_t\}_{t=1,\ldots,T}$ and the best-fixed point chosen in hindsight. An algorithm is called no-regret [34], if the regret of the algorithm goes sublinearly with the time horizon T.

For carrying out optimization over a manifold, some classical methods treat the manifold as a subset of an ambient Euclidean space and employ Euclidean constrained optimization techniques. For instance, [29] presented an algorithm for the online PCA problem, where the variables were updated in an embedding Euclidean space and then projected onto a manifold. However, in practical applications, the dimension of an embedding Euclidean space can be too high (e.g., the Grassmann manifold [13]), and computing the projection can be much expensive (e.g., the manifold of symmetric positive definite (SPD) matrices [38]). An alternative approach termed *Riemannian optimization* makes use of intrinsic geometry of manifolds so that it can optimize directly on the manifold as an unconstrained problem, and thus avoiding high dimension embedding and high computing cost for the projection. Furthermore, this viewpoint has shown benefits from the g-convexity, by which a nonconvex optimization problem can be converted into a g-convex one [7, 11]. Consequently, it is attractive to take a Riemannian approach in our problem (1).

Although there have been many existing algorithms for offline manifold optimization problems [2, 37, 6], limited research is obtained about the Riemannian online optimization problem. The authors in [36] proposed an online EM algorithm for hidden Markov chains on Hadamard homogeneous spaces. In addition, inspired by the zeroth-order oracle in [27], [28] studied the zeroth-order online optimization problem on Hadamard manifolds and derived bounds on the tracking error and regret asymptotically.

Contribution This paper aims to design no-regret algorithms for the R-OCO problem in both full information feedback and bandit feedback. The contribution of this paper is summarized as follows:

- We propose a Riemannian online gradient descent algorithm (R-OGD) for the R-OCO problem in the full information feedback, and then establish the upper regret bounds of the R-OGD algorithm for g-convex and strongly g-convex functions.
- We introduce a Riemannian bandit algorithm (R-BAN) for the R-OCO problem in the bandit feedback and then establish an upper regret bound for g-convex functions. Moreover, we develop a key technique to analyze the derivative of a local integration on homogeneous manifolds, which can be applied to estimate gradients in Riemannian optimization and beyond.
- We focus on the worst-case regret and present a universal lower regret bound of R-OCO algorithms with g-convex losses on Hadamard manifolds, which matches the upper bound achieved by the R-OGD algorithm for g-convex functions.

The established lower and upper bounds on the achievable bounds of R-OCO match their counterparts for Euclidean online convex optimization e.g., [40, 21, 18, 1]. We briefly list our results in Table 1.

Related Work The Euclidean online convex optimization was introduced in [40]. Inspired by the gradient descent method, [40] proposed the online gradient descent algorithm (OGD) of which the upper regret bound was proven to be $\mathcal{O}\big(\sqrt{T}\big)$. Then [21] proceeded with the study of the OGD

algorithm and established an upper regret bound $\mathcal{O}\big(\log T\big)$ for strongly convex functions. In addition, [1] gave a universal lower bound of $\mathcal{O}\big(\sqrt{T}\big)$ for online algorithms, which indicated that the bounds in [40] and [21] are essentially optimal. In the bandit setting, [18] provided a detailed exposition of a one-point bandit algorithm. By modifying the gradient in the OGD algorithm to a randomized estimator, the upper regret bound attained $\mathcal{O}\big(T^{\frac{3}{4}}\big)$ and $\mathcal{O}\big(T^{\frac{2}{3}}\big)$ for convex loss functions and strongly convex loss functions respectively. Extending the one-point bandit algorithm, [4] developed a multi-point bandit algorithm and presented upper regret bounds $\mathcal{O}\big(\sqrt{T}\big)$ and $\mathcal{O}\big(\log T\big)$ for convex and strongly convex loss functions. The Riemannian online algorithms proposed in the current work in full information feedback and one-point bandit feedback settings are extensions of the algorithms in ([40, 21, 18, 1]) to Riemannian manifolds.

Riemannian optimization has drawn much research attention in the past decades. Many basic algorithms in Euclidean spaces such as the gradient descent method, Newton's method, and trust-region methods have been adapted into a Riemannian setting [3, 6, 32, 33, 30, 23]. Some work of Riemannian stochastic optimization is intended to deal with time-varying optimization problems [12, 37, 38, 36]. Among them, [37] provided the first global complexity analysis for the R-SGD algorithm on geodesically convex problems over Hadamard manifolds and [36] proposed an online EM algorithm on Hadamard homogeneous spaces. However, These studies [12, 37, 38, 36] assume prior knowledge of loss functions, which is different from the framework of our R-OCO problem. The work about the R-OCO problem is fairly limited. Recently [28] has proposed a zeroth-order online algorithm on Hadamard manifolds and established the tracking error and regret bound in the asymptotic sense. In contrast, the regret bounds established for our online gradient-based/bandit Riemannian optimization algorithms are for any time, matching those for Euclidean online optimization.

2 Preliminaries

Riemannian manifolds A *Riemannian manifold* is a smooth manifold $\mathcal{M}$ equipped with a metric tensor g (or called Riemannian metric), which defines an inner product $\langle \cdot, \cdot \rangle_x$ in every tangent space $T_x\mathcal{M}$ of $x \in M$. The Riemannian metric g brings a distance structure on $\mathcal{M}$. A curve is a *geodesic* if it locally minimizes the distance, which is an analog of a straight line in Euclidean spaces. On Riemannian manifolds, a geodesic is uniquely determined by its start point and initial tangent vector. In this way, the *exponential map* $\exp_x : T_x\mathcal{M} \to \mathcal{M}$ is defined by mapping a vector $X \in T_p\mathcal{M}$ to $\gamma(1) \in \mathcal{M}$ for the geodesic γ such that $\gamma(0) = x$ and $\dot{\gamma}(0) = X$.

The curvature tensor reflects the geometry of manifolds. We focus on *sectional curvature* which is the Gauss curvature of a two-dimensional submanifold. Following [37], we consider the *Hadamard manifold*, which is a simply connected and complete manifold with nonpositive curvature. The Cartan-Hadamard theorem [10] shows that the exponential map has an inverse $\exp_x^{-1}(\cdot)$ globally on Hadamard manifolds and thus the distance $d(x, y)$ can be expressed as $\| \exp_x^{-1}(y)\|_x$.

Isometries of Riemannian manifolds have been widely studied in differential geometry [10, 9]. An isometry $\phi : \mathcal{M} \to \mathcal{M}$ is a diffeomorphism preserving distance, i.e., $d(x, y) = d(\phi(x), \phi(y))$ for all $x, y \in \mathcal{M}$. It is remarked that all isometries of a Riemannian manifold form a Lie group G. A Riemannian manifold is a *homogeneous manifold* if the group of isometries G acts on $\mathcal{M}$ transitively, i.e., for any points $x, y \in \mathcal{M}$ there exists an isometry such that $\phi(x) = y$. Some properties of homogeneous spaces are used in our work, for example, the Killing field that represents the infinitesimal symmetry of isometries. We left those properties in the appendix.

Function Classes We now define a few classes of functions that are useful in Riemannian optimization. A set $\mathcal{K}$ is called *geodesically convex* (g-convex) if, for any points $x, y \in \mathcal{K}$, there admits a geodesic $\gamma \subset \mathcal{K}$ connecting x and y. A function $f : \mathcal{K} \to \mathbb{R}$ is called *geodesically convex* (or g-convex) if for any geodesic $\gamma : [0, 1] \to \mathcal{M}$, there holds

$$f(\gamma(t)) \leq (1 - t)f(\gamma(0)) + tf(\gamma(1)).$$

The g-convexity has some equivalent conditions. When f is differentiable, which means that the *gradient* $\nabla f(x) \in T_x\mathcal{M}$ exists for every $x \in \mathcal{M}$, the g-convexity is equivalent to the following condition

$$f(y) \geq f(x) + \langle \nabla f(x), \exp_x^{-1}(y)\rangle, \forall x, y \in \mathcal{M}.$$

Furthermore, if f is twice differentiable, the g-convexity is equivalent to

$$Hess(f)(x)(X, X) = \nabla_X \nabla_X f(x) - \nabla_{\nabla_X X} f(x) \geq 0, \forall x \in \mathcal{M}.$$

A function $f : \mathcal{M} \to \mathbb{R}$ is *geodesically μ-strongly convex* (or μ-strongly g-convex) if there exists a constant $\mu > 0$ such that for any $x, y \in \mathcal{M}$ there holds

$$f(y) \geq f(x) + \langle \nabla f(x), \exp_x^{-1}(y) \rangle + \frac{\mu}{2} d^2(x, y).$$

We term a function to be *geodesically L-Lipschitz* (or g-L-Lipschitz) if there exists a constant $L > 0$ such that for any $x, y \in \mathcal{M}$, there holds

$$|f(y) - f(x)| \leq L \cdot d(x, y),$$

which is equivalent to

$$\|\nabla f(x)\| \leq L, \forall x \in \mathcal{M},$$

if f is differentiable.

3 Riemannian Online Convex Optimization with Full Information Feedback

This section is devoted to the study of the R-OCO problem in the full information feedback. In this section, we first propose our R-OGD algorithm and then analyze the upper regret bounds of R-OGD for both g-convex and strongly g-convex functions. In addition, a universal lower regret bound is presented to illustrate that the regret bound of the R-OGD algorithm in g-convex cases is tight up to a constant.

3.1 Riemannian Online Gradient Algorithm

In the full information setting, we consider the following assumptions, which were used in the existing literature of Euclidean online convex optimization and Riemannian optimization (e.g., [40, 6, 23]).

Assmuption 1. *There exists a $x^\star \in \mathcal{M}$ such that $x^\star = \arg\min \sum_{t=1}^{T} f_t(x)$.*

Assmuption 2. *$(\mathcal{M}, g)$ is a Hadamard manifold with the sectional curvature lower bounded by a constant $-\kappa$ ($\kappa \geq 0$).*

Remark 1. The Hadamard manifold plays an important role in Riemannian geometry [19]. Some basic spaces, such as the Euclidean space $\mathbb{R}^n$, the Hyperbolic space H^n, and the manifold of SPD matrices, are all Hadamard manifolds [6, 23].

Assmuption 3. *The g-convex set $\mathcal{K}$ is a bounded set with the diameter D, i.e.,*

$$d(x, y) \leq D, \forall x, y \in \mathcal{K}.$$

Assmuption 4. *For all $t = 1, \ldots, T$, f_t are differentiable and g-L-Lipschitz.*

We now propose our Riemannian online gradient descent algorithm (R-OGD) in Algorithm 1, where the exponential map replaced the vector addition in the Euclidean online gradient descent [40].

Algorithm 1: Riemannian Online Gradient Descent Algorithm (R-OGD)

Input: Manifold $\mathcal{M}$, time T, step size $\{\eta_t\}$
Output: $\{x_t\}_{t=1,\ldots,T}$
for $t = 1$ *to* T *do* **do**
  Play x_t and observe the function f_t;
  Update

$$x_{t+1} = \mathcal{P}_{\mathcal{K}}(\exp_{x_t}(-\nabla f_t(x_t))),$$

  where $\exp_{x_t}(\cdot)$ is the exponential map at x_t and $\mathcal{P}_{\mathcal{K}}$ is the Riemannian projection mapping of x onto $\mathcal{K}$, i.e., $\mathcal{P}_{\mathcal{K}}(x) := \arg\min_{y \in \mathcal{K}} d(x, y)$;
  Return x_{t+1}, and suffer from the loss $f_t(x_t)$;
end

3.2 Regret Upper Bounds

In Theorems 1 and 2 we present upper bounds of the regret along the R-OGD algorithm for g-convex loss functions and strongly g-convex functions, respectively. We introduce $\zeta(\kappa, d) = \frac{\sqrt{\kappa} \cdot d}{\tanh\left(\sqrt{\kappa} \cdot d\right)}$. By direct observation, ζ is a increasing function of the variables κ and d when $\kappa d \geq 0$.

Theorem 1 (Convex Cases)**.** *Suppose Assumptions 1-4 hold. Assume that f_t is g-convex for any $t = 1, \ldots, T$. Then the R-OGD algorithm with step sizes $\{\eta_t = \frac{D}{L\sqrt{\zeta(\kappa,D)t}}\}$ guarantees the following for all $T \geq 1$:*

$$R(T) \leq \frac{3}{2} DL\sqrt{\zeta(\kappa, D) \cdot T}$$

Theorem 2 (Strongly-convex Cases)**.** *Suppose Assumptions 1-4 hold. Assume that f_t is μ-strongly g-convex for any $t = 1, \ldots, T$. Then the R-OGD algorithm with step sizes $\{\eta_t = \frac{1}{\mu t}\}$ guarantees the following for all $T \geq 1$:*

$$R(T) \leq \frac{L^2 \zeta(\kappa, D)}{2\mu}(1 + \log T).$$

The proofs of Theorems 1 and 2 are in the appendix. A major challenge in proving Theorems 1 and 2 is that there is no vector space structure on Riemannian manifolds. Thanks to the trigonometric distance bound proposed in [37], we manage to obtain the regret $\mathcal{O}\big(\sqrt{T}\big)$ and $\mathcal{O}\big(\log T\big)$ for g-convex loss functions and strongly g-convex loss functions. By slightly moving κ to zero, the results recover the regret bounds for Euclidean gradient descent in [40] and [21].

Theorems 1 and 2 also reveals the influence of curvature on the regret bounds. Since $\zeta(\kappa, d)$ is an increasing function of κ, the upper regret bounds in the R-OGD algorithm are higher than those in Euclidean spaces and the increase of κ raises the upper regret bound. Therefore, a proper Riemannian metric should be chosen in the optimization to avert the high sectional curvature bound.

3.3 Regret Lower Bound

This section is intended to answer the question of whether there exists an algorithm that attains a tighter regret bound than $\mathcal{O}\big(\sqrt{T}\big)$ for g-convex functions. Theorem 3 answers the question negatively.

Theorem 3. *Suppose Assumptions 1-4 hold. Then for any Hadamard manifold $\mathcal{M}$, the Riemannian online convex optimization incurs the regret $\mathcal{O}\big(DL\sqrt{T}\big)$ for any possible strategy in the worst case.*

The proof of Theorem 3 is in the appendix. The result illustrates that, same as that in Euclidean spaces, the regret of a Riemannian online algorithm can not be less than $\mathcal{O}\big(\sqrt{T}\big)$ in the worst case. Moreover, Theorem 3 shows that the regret of the R-OGD algorithm in Theorem 1 is tight up to a constant.

4 Riemannian Online Convex Optimization with Bandit Feedback

In this section, we consider the Riemannian online convex optimization with the one-point bandit feedback. We first present the Riemannian bandit algorithm (R-BAN) on Hadamard homogeneous manifolds and then analyze the (expected) upper regret bound for our algorithm. In the following of this section, for any point x, we denote $B_\delta(x)$ as the ball centered at x with radius δ and $S_\delta(x)$ as the sphere centered at x with radius δ.

4.1 Riemannian Bandit Algorithm

In the bandit setting, Assumptions 2-4 need to be slightly modified to adapt to the bandit case.

Assmuption 5. *$\mathcal{M}$ is an n-dimensional homogeneous Hadamard manifold with the sectional curvature lower bounded by a constant $-\kappa$ ($\kappa > 0$).*

Remark 2. The homogeneous Hadamard manifold is widely studied in differential geometry [9, 10]. The property of homogeneity has received much attention in machine learning [35, 36, 15]. It has been seen that many manifolds often considered in Riemannian optimization, such as the Euclidean space $\mathbb{R}^n$, the Hyperbolic space H^n, and the manifold of SPD matrices, are Hadamard homogeneous manifolds.

Note that on Hadamard homogeneous manifolds, the volume and surface area of a ball is only related to the radius but not to the center of the ball, thus we denote V_δ as the volume of $B_\delta(x)$ and S_δ as the surface area ($(n-1)$-dim volume) of $S_\delta(x)$ for all x in the manifold $\mathcal{M}$.

Assmuption 6. *Suppose there exists an interior point $p \in \mathcal{K}$ such that the set $\mathcal{K}$ contains a ball with radius r centered at p, and is also contained in a ball with radius D, i.e.,*

$$B_r(p) \subset \mathcal{K} \subset B_D(p).$$

Assmuption 7. *For any $t = 1, \ldots, T$, f_t is differentiable, g-L-Lipschitz and bounded by C.*

Inspired by the Euclidean bandit algorithm, we replace the gradient $\nabla f_t(x_t)$ with a randomized estimator g_t and propose our Riemannian bandit optimization algorithm (R-BAN) in Algorithm 2 over Hadamard homogeneous manifolds.

Algorithm 2: Riemannian Bandit Algorithm (R-BAN)

Input: Manifold $\mathcal{M}$, time T, step size ν, parameter δ, α.
Output: Sequence $\{x_t\}_{t=1,\ldots,T}$
for $t = 1$ *to* T *do* **do**
  Pick x_t uniformly from $S_\delta(y_t)$;
  Play x_t and observe $f_t(x_t)$;
  Construct the gradient estimator

$$g_t = f_t(x_t)\frac{\exp_{y_t}^{-1}(x_t)}{\|\exp_{y_t}^{-1}(x_t)\|};$$

  Update with the rule

$$y_{t+1} = P_{(1-\alpha)K}(\exp_{y_t}(-\nu g_t)),$$

  where we denote $P_{(1-\alpha)K}$ as the projection mapping onto the shrinking set $(1-\alpha)\mathcal{K} = \{\exp_p((1-\alpha)u) | u = \exp_p^{-1}(x), x \in \mathcal{K}\}$.
  Return x_t and suffer from the loss $f_t(x_t)$;
end

4.2 Challenge Arising from Geometry in Riemannian Manifolds

Since Algorithm 2 is an extension of the Euclidean bandit algorithm in [18], it is worth reviewing the analysis in [18]. In the Euclidean setting, we uniformly choose x_t on the $S_\delta(y_t)$ and update y_t by the rule

$$\begin{cases} \tilde{g}_t = f(x_t)\frac{x_t - y_t}{\|x_t - y_t\|}, \\ y_{t+1} = \mathcal{P}_{(1-\alpha)\mathcal{K}}(y_t - \nu g_t). \end{cases} \tag{2}$$

The basic idea for the analysis is to introduce the *smoothed loss function* [18]

$$\hat{f}_t^E(x) = \mathbb{E}_{u \in B_\delta(x)}[f(u)] = \frac{1}{V_\delta}\int_{B_\delta(x)} f_t(u)du,$$

where $\hat{f}_t^E$ is a convex approximation of f_t when δ is small. It is shown that $\frac{n}{\delta}\tilde{g}_t$ is an unbiased estimator of the gradient $\nabla \hat{f}_t^E(y_t)$, hence the bandit algorithm is actually an expected gradient descent method with the loss function $\hat{f}_t^E$. In this way, a regret bound of the bandit algorithm is established by [18].

Back to the Riemannian case, we attempt to generalize the analysis of [18] in parallel by defining the "Riemannian version" of the smoothed loss function,

$$\hat{f}_t(x) = \mathbb{E}_{u \in B_\delta(x)}[f(u)] = \frac{1}{V_\delta}\int_{B_\delta(x)} f_t(u)\omega,$$

where ω is the volume element with the respect to the metric g. Analyzing this smoothed loss function in the Riemannian space, however is fundamentally challenging due to the following two reasons.

(1) The gradient is hard to compute. Computing the gradient of $\hat{f}_t$ is quite different from that in Euclidean spaces, due to the absence of the commutativity of the derivative operator ∇ and the integration operator $\int_{B_\delta(y_t)}$. In Euclidean spaces, the derivative operator ∇ commutes with the integration operator $\int_{B_\delta(y_t)}$. Accordingly, for the Euclidean smoothed loss function $\hat{f}_t^E$, there holds

$$\nabla \hat{f}_t^E(y_t) = \frac{1}{V_\delta} \nabla \int_{B_\delta(y_t)} f_t(u) du = \frac{1}{V_\delta} \int_{B_\delta(y_t)} \nabla f_t(u) du, \tag{3}$$

which derives the fact $\frac{n}{\delta} E[\tilde{g}_t] = \nabla \hat{f}_t^E(y_t)$. However, on Riemannian manifolds the derivative operator ∇ does not commute with the integration operator $\int_{B_\delta(y_t)}$. Consequently, the equation (3) fails to hold for functions on Riemannian manifolds.

(2)The convexity may not hold. Another essential challenge for regret analysis is the convexity of $\hat{f}_t$. In Euclidean spaces, one can easily conclude the convexity of $\hat{f}_t^E$. However, the convexity may not hold for Riemannian manifolds. Through calculation, the hessian of $\hat{f}_t$ on Riemannian manifolds is

$$Hess(\hat{f}_t)(X, X) = \frac{1}{V_\delta} \int_{B_\delta(x)} (Hess(f_t)(\eta, \eta)(u) + \langle \nabla_\eta \eta, \nabla f_t(u) \rangle) \omega,$$

where η is a Killing field such that $\eta(x) = X$. Since the qudratic form $Hess(f_t)(\eta, \eta)(x) + \langle \nabla_\eta \eta, \nabla f_t(x) \rangle$ can be negative at some $\eta \in T_p\mathcal{M}$, the g-convexity of $\hat{f}_t$ is violated for some small δ.

4.3 Gradient Bound and Approximate g-Convexity

To address the first difficulty, we propose a key technique to analyze the derivative of local integration by introducing the Killing vector field. With the help of this technique, we manage to compute the gradient of $\hat{f}_t$ in Lemma 1.

Lemma 1. *Let $\mathcal{M}$ be a homogeneous Hadamard manifold. Suppose that f is a C^1 function on $\mathcal{M}$, Then the smoothed function $\hat{f}(x) = \frac{1}{V_\delta} \int_{B_\delta(x)} f\omega$ satisfies,*

1. $\nabla \hat{f}(x) = \frac{1}{V_\delta} \int_{S_\delta x} f(u) \frac{\exp_x^{-1}(u)}{\| \exp_x^{-1}(u) \|} = \frac{S_\delta}{V_\delta} \mathbb{E}_{u \in S_\delta(x)} \left[f(u) \frac{\exp_x^{-1}(u)}{\| \exp_x^{-1}(u) \|} \right], \forall x \in \mathcal{M}$;
2. *if $|f(x)| < C$, then there exist a number $\epsilon(n)$ such that*

$$\|\nabla \hat{f}(x)\| \leq \frac{S_\delta}{V_\delta} C \leq \frac{n}{\delta} C + \frac{(n^2 - n)}{3n + 6} \kappa \delta C$$

holds for all $0 < \delta < \epsilon(n)$.

Remark 3. The proof of Lemma 1 can be found in the appendix. The first part of the lemma proceeds the result in [18], while the second part gives us an easy-to-compute bound of the gradient. From the proof, we develop a key technique that transform derivation of integration on $B_\delta(x)$ to integration of derivative of corresponding Killing vector field on $B_\delta(x)$, i.e.,

$$\nabla_X \Big(\int_{B_\delta(x)} f(u)\omega \Big) = \int_{B_\delta(x)} \nabla_\eta f(u) \omega, \tag{4}$$

where η is a Killing field such that $\eta(x) = X$. This technique does not rely on the curvature and other specific manifold structures. Hence, it can be a basic tool for optimization on homogeneous spaces and maybe in broader areas.

To overcome the second difficulty, we notice that though the function $\hat{f}_t$ may not be g-convex, it is very close to be g-convex. We conclude it in Lemma 2.

Lemma 2. *Suppose that $\mathcal{M}$ is a Hadamard homogeneous manifold and $\mathcal{K}$ is a convex and bounded set of $\mathcal{M}$. Then there exists a constant $\rho \geq 0$ only related to the set $\mathcal{K}$ such that for any g-convex and g-L-Lipschitz function f, the smoothed function $\hat{f}$ satisfies*

$$\hat{f}(y) - \hat{f}(x) - \langle \nabla \hat{f}(x), \exp_x^{-1}(y) \rangle \geq -2\rho\delta L, \forall x, y \in \mathcal{K}.$$

The proof of Lemma 2 is in the appendix. It is worth mentioning the constant ρ describes how close a smoothed function is from being g-convex. Notice that ρ does not relate to the function $\hat{f}$, which indicates that the gap from g-convexity is uniform for all smoothed functions. Moreover, if the manifold $\mathcal{M}$ reduces to a Euclidean space, then we can find $\rho = 0$ and $\hat{f}$ is convex, which coincides with the result in Euclidean spaces. Additionally, though the value of ρ is generally difficult to calculate, our algorithm analysis and parameter selection do not depend on the specific value of ρ (see Theorem 4).

4.4 Regret Bound

After surmounting the difficulties above, we now carry out the analysis of the expected regret bounds of Algorithm 2.

Theorem 4. *Suppose Assumptions 1 and 5-7 holds. Assume that f_t is g-convex for any $t = 1, \ldots, T$. Denote $B = \frac{n^2-n}{3n+6}\kappa$, $\Delta = BCD\sqrt{\zeta(\kappa, D)} + 3L + 2C/r$ and $\Lambda = \sqrt{\Delta} + \frac{2\rho L}{\sqrt{\Delta}}$. For any $T > \frac{(nCD\sqrt{\zeta(\kappa,D)})^2}{\Delta^2\epsilon(n)^4}$, take $\alpha = \frac{\delta}{r}$, $\nu = \frac{D}{C\sqrt{\zeta(\kappa,D)T}}$,$\delta = T^{-\frac{1}{4}}\sqrt{\frac{CDn\sqrt{\zeta(\kappa,D)}}{\Delta}}$. Then the expected regret of Algorithm 2 is upper bounded by*

$$\mathbb{E}[R(T)] \leq 2T^{\frac{3}{4}}\sqrt{nCD\sqrt{\zeta(\kappa, D)}}\Lambda.$$

The proof of Theorem 4 can be seen in the appendix. Theorem 4 shows that the regret of the Riemannian bandit algorithm achieves $\mathcal{O}\big(T^{\frac{3}{4}}\big)$ for g-convex loss functions on homogeneous Hadamard manifolds, which is same as the regret bounds in Euclidean spaces [18].

Our work is different from the result in [28] in two aspects. First, in [28] the tracking error and regrets bounds are given asymptotically, while in Theorem 4, the regret bound is expressed explicitly with the time horizon T. Second, to estimate the gradient, [28] used a biased estimator of the gradient, while in our bandit algorithm the gradient estimator g_t of the smoothed function is unbiased.

5 Numerical Experiment

To validate our results, we apply our R-OGD and R-BAN algorithms to the Karcher mean problem. We also compare our result with the Riemannian zeroth online (R-OZO) algorithm[28]. The Karcher mean problem, which is also known as finding the Riemannian centroid of a set of points on a manifold, has applications in various domains, such as diffusion tensor magnetic resonance imaging (DT-MRI) [16, 31], radar signal processing [25], and face recognition [24]. In this section, we study an online form of the Karcher mean problem, which is to average a set of N time-varying points $\{A_{t,1}, A_{t,2}, A_{t,3}, \ldots, A_{t,N}\}$ on a manifold. In this situation the loss function is defined as

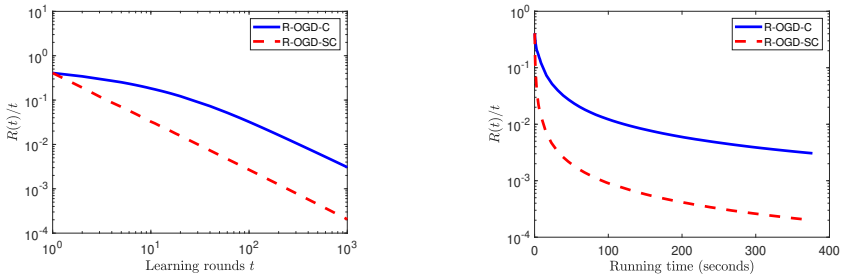
$$f_t(X_t) = \frac{1}{N}\sum_{i=1}^{N} d^2(X_t, A_{t,i}), t = 1, 2, \ldots, T$$

where $d(X, Y)$ is the Riemannian distance of the manifold. It is remarked that f_t is g-convex and 2-strongly g-convex so that we can apply our algorithms. In particular, we test the R-OGD and the R-BAN algorithm respectively on the manifold of SPD matrices and the hyperbolic space. All experiments are performed in Matlab R2018a on a 64 bit Windows platform with a 3.6 GHz CPU (Intel Core i7-6850K) and the code used for the numerical experiments is provided in the supplementary materials.

Karcher Mean on the SPD Manifold On the manifold of SPD matrices $\{X \in \mathbb{R}^{n\times n} | X^T = X, X \succ 0\}$, we run the R-OGD algorithm with two kinds of step size: the step size for convex cases $\eta_t = D/(L\sqrt{\zeta(\kappa, D)}t)$ and the step size for strongly g-convex cases $\eta_t = 1/(2t)$. In these two cases, we set $[n, N, T] = [100, 10, 1000]$. Matrices $A_{i,t}$ are randomly generated by the method in Manopt toolbox [14][1], and their norms are normalized to 1. We plot the average regret $\frac{R(t)}{t}$ versus the learning round t and the running time in Figure 1. As seen, the regrets of the R-OGD algorithm go

[1] https://www.manopt.org/about.html, GNU General Public License (GPL) version 3

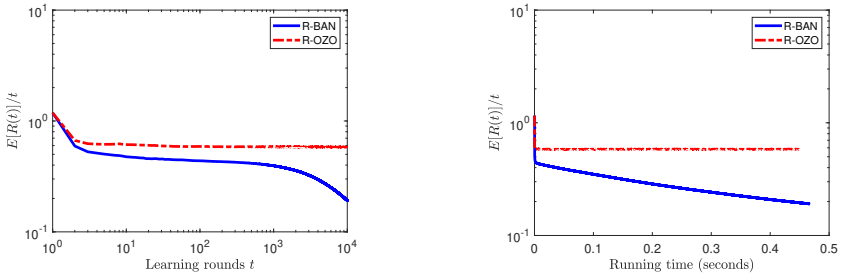
sublinearly with t and the average regret of the R-OGD-C converges faster than that of R-OGD-SC in terms of the iteration as well as the running time, which are consistent with the theoretical results in Theorems 1 and 2.



(a) Average regret vs learning rounds (b) Average regret vs running time

Figure 1: The R-OGD algorithm in the SPD matrix Karcher mean problems. The blue line (R-OGD-C) stands for the R-OGD algorithm with step size for g-convex cases and the red line (R-OGD-C) stands for the R-OGD algorithm with the step size for strongly g-convex cases.

Karcher Mean on the Hyperbolic Space We test the performance of the R-BAN algorithm on the hyperbolic space $H^n = \{x \in \mathbb{R}^{n+1} | -x_{n+1}^2 + \sum_{i=1}^n x_i^2 = -1\}$. Consider the case $[n, N, T] = [10, 10, 10000]$ and set our parameter $\delta = 0.1873$ and $\nu = 0.005$. The data $A_{i,t}$ are also produced by Manopt toolbox [14] with the norm normalized to 1. Figure 2 compares the performance between the R-BAN algorithm and the R-OZO algorithm for 100 random runs with different initial values. We observe that our R-BAN algorithm can achieve a sublinear expected regret and requires less information, since the R-OZO needs function values of two points in this cases.



(a) Expected average regret vs learning rounds (b) Expected average regret vs running time

Figure 2: Comparing the R-BAN algorithm and the zeroth-order online in the Hyperbolic Karcher mean problems. The blue line stands for our R-BAN, and the red line stands for the R-OZO algorithm.

6 Conclusion

We have considered the online optimization problem on Riemannian manifolds in the full information and bandit feedback setting. We developed the R-OGD algorithm on Hadamard manifolds and the R-BAN algorithm on Hadamard homogeneous manifolds. The upper regret bounds of the R-OGD and R-BAN algorithm, together with a universal lower regret bound were established with the influence of curvature clearly indicated. All of the regret bounds matched their Euclidean counterpart.

A limitation of our work is that we do not take retraction into consideration. A retraction map is a cheap approximation of the exponential map on manifolds and is a sensible choice in many real scenarios. In future work, we intend to design Riemannian online optimization methods with the retraction map, so that they can be more effective in large-scale optimization problems.

References

[1] Jacob Abernethy, Peter L Bartlett, Alexander Rakhlin, and Ambuj Tewari. Optimal strategies and minimax lower bounds for online convex games. 2008.

[2] P-A Absil, Christopher G Baker, and Kyle A Gallivan. Trust-region methods on riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

[3] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[4] Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Proceedings of the 23rd Annual Conference on Learning Theory (COLT)*, June 2010.

[5] Shmuel Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6:382–392, 1954.

[6] Kwangjun Ahn and Suvrit Sra. From nesterov's estimate sequence to riemannian acceleration. *arXiv preprint arXiv:2001.08876*, 2020.

[7] Zeyuan Allen-Zhu, Ankit Garg, Yuanzhi Li, Rafael Oliveira, and Avi Wigderson. Operator scaling via geodesically convex optimization, invariant theory and polynomial identity testing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 172–181, 2018.

[8] Sébastien Arnold, Pierre-Antoine Manzagol, Reza Babanezhad Harikandeh, Ioannis Mitliagkas, and Nicolas Le Roux. Reducing the variance in online optimization by transporting past gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[9] Valerii Berestovskii and Yurii Nikonorov. *Riemannian manifolds and homogeneous geodesics*. Springer, 2020.

[10] Marcel Berger. *Geometry i*. Springer Science & Business Media, 2009.

[11] Dario A Bini and Bruno Iannazzo. Computing the karcher mean of symmetric positive definite matrices. *Linear Algebra and its Applications*, 438(4):1700–1710, 2013.

[12] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

[13] Nicolas Boumal and P-A Absil. Low-rank matrix completion via preconditioned optimization on the grassmann manifold. *Linear Algebra and its Applications*, 475:200–239, 2015.

[14] Nicolas Boumal, Bamdev Mishra, P.-A. Absil, and Rodolphe Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(42):1455–1459, 2014.

[15] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021.

[16] Guang Cheng, Hesamoddin Salehian, and Baba C Vemuri. Efficient recursive algorithms for computing the mean diffusion tensor and applications to dti segmentation. In *European Conference on Computer Vision*, pages 390–401. Springer, 2012.

[17] Jiashi Feng, Huan Xu, and Shuicheng Yan. Online robust pca via stochastic optimization. In *Advances in neural information processing systems*, pages 404–412. Citeseer, 2013.

[18] Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. Online convex optimization in the bandit setting: Gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '05, page 385–394, USA, 2005. Society for Industrial and Applied Mathematics.

[19] Mohammad Ghomi and Joel Spruck. Total curvature and the isoperimetric inequality in cartan-hadamard manifolds, 2021.

[20] Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

[21] Elad Hazan, Adam Kalai, Satyen Kale, and Amit Agarwal. Logarithmic regret algorithms for online convex optimization. In *International Conference on Computational Learning Theory*, pages 499–513. Springer, 2006.

[22] 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(2):199–248, 2020.

[23] Wen Huang, P-A Absil, and Kyle A Gallivan. A riemannian symmetric rank-one trust-region method. *Mathematical Programming*, 150(2):179–216, 2015.

[24] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel methods on the riemannian manifold of symmetric positive definite matrices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[25] J Lapuyade-Lahorgue and F Barbaresco. Radar detection using siegel distance between autoregressive processes, application to hf and x-band radar. In *2008 IEEE Radar Conference*, pages 1–6. IEEE, 2008.

[26] Kuang-Chih Lee and David Kriegman. Online learning of probabilistic appearance manifolds for video-based recognition and tracking. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 852–859. IEEE, 2005.

[27] Jiaxiang Li, Krishnakumar Balasubramanian, and Shiqian Ma. Stochastic zeroth-order riemannian derivative estimation and optimization, 2021.

[28] Alejandro I Maass, Chris Manzie, Dragan Nesic, Jonathan H Manton, and Iman Shames. Online zeroth-order optimisation on riemannian manifolds. *arXiv preprint arXiv:2010.00211*, 2020.

[29] Jiazhong Nie, Wojciech Kotłowski, and Manfred K Warmuth. Online pca with optimal regret. *The Journal of Machine Learning Research*, 17(1):6022–6070, 2016.

[30] Chunhong Qi, Kyle A Gallivan, and P-A Absil. Riemannian bfgs algorithm with applications. In *Recent advances in optimization and its applications in engineering*, pages 183–192. Springer, 2010.

[31] Yogesh Rathi, Allen Tannenbaum, and Oleg Michailovich. Segmenting images on the tensor manifold. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[32] Wolfgang Ring and Benedikt Wirth. Optimization methods on riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 22(2):596–627, 2012.

[33] Berkant Savas and Lek-Heng Lim. Quasi-newton methods on grassmannians and multilinear approximations of tensors. *SIAM Journal on Scientific Computing*, 32(6):3352–3393, 2010.

[34] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[35] Hao Tang, Zhiao Huang, Jiayuan Gu, Bao-Liang Lu, and Hao Su. Towards scale-invariant graph-related problem solving by iterative homogeneous gnns. *Advances in Neural Information Processing Systems*, 33, 2020.

[36] Quinten Tupker, Salem Said, and Cyrus Mostajeran. Online learning of riemannian hidden markov models in homogeneous hadamard spaces. *arXiv preprint arXiv:2102.07771*, 2021.

[37] Hongyi Zhang and Suvrit Sra. First-order methods for geodesically convex optimization. In *Conference on Learning Theory*, pages 1617–1638, 2016.

[38] Jingzhao Zhang, Hongyi Zhang, and Suvrit Sra. R-spider: A fast riemannian stochastic optimization algorithm with curvature independent rate. *arXiv preprint arXiv:1811.04194*, 2018.

[39] Shengping Zhang, Shiva Kasiviswanathan, Pong Yuen, and Mehrtash Harandi. Online dictionary learning on symmetric positive definite manifolds with vision applications. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[40] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 928–935. AAAI Press, 2003.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [No] The content of our research is purely theoretical. From the current point of view, we have not found that it has applications that violate the social ethics.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Assumption 1-7.
   (b) Did you include complete proofs of all theoretical results? [Yes] We left them in the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code used for the numerical experiments is provided in the supplementary materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Although we run the R-BAN algorithm 100 times in our experiment, the standard error is too small (like 10^{-5}), thus we do not plot the error bars.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] All experiments are performed in Matlab R2018a on 64 bit Windows platform with 3.6 GHz CPU (Intel Core i7-6850K), see Section 5.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] For generating data from manifolds, we used the method that in Manopt toolbox and cited the original paper in Section 5.
   (b) Did you mention the license of the assets? [Yes] The lisence of Manopt is GNU General Public License (GPL) version 3.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] We did not include any new assets.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] Our experiment did not involve data managed by others.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We didn't use personally identifiable information in our experiments.

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [No] Our research did not include crowdsourcing or conducted research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]