ADAPTIVE OPTIMIZATION ALGORITHMS ON MANIFOLDS

Anonymous authors
Paper under double-blind review

ABSTRACT

In machine learning training, minimizing accumulated objective functions is a common task, while most optimization algorithms are designed for variables in Euclidean spaces. This paper generalizes the classical Adam and AMSGrad algorithms to manifold settings, which is often necessary for problems with geometric structures, introducing the RAdam and RAMSGrad algorithms. Under the geodesical convexity assumption, we establish the global convergence of RAMSGrad by deriving a regret bound. RAdam and RAMSGrad are implemented efficiently in the C++ package ROPTLIB. In the numerical experiments, RAdam and RAMSGrad are compared to the state-of-the-art methods including the Riemannian stochastic gradient and existing Riemannian adaptive optimization algorithms using the Karcher mean computation, the Poincaré embedding taxonomies, and the joint diagonalization. The proposed methods perform at least competitively or more effectively than the compared algorithms.

1 INTRODUCTION

Machine learning is a significant subfield of artificial intelligence, granting computers the ability to autonomously learn from large-scale data and make intelligent decisions. In recent years, with the rapid development of artificial intelligence and data science, machine learning has emerged as a widely recognized research hotspot in both academia and industry. It plays a central role in various domains, such as natural language processing (Nickel & Kiela, 2017; 2018; Balazevic et al., 2019; Mikolov et al., 2013), computer vision (Schroff et al., 2015; Tron & Daniilidis, 2017), and metric learning (Ying et al., 2017; Roy et al., 2018; Zadeh et al., 2016).

One of the core tasks in machine learning is optimization, particularly the problem of empirical risk minimization, which involves minimizing an accumulated form of objective functions:

$$\min_{w} \left\{ F(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right\}. \tag{1}$$

However, most optimization methods focus mainly on Euclidean spaces, which might make them less effective when the problem has an underlying manifold structure. Riemannian optimization concerns an optimization problem defined on a Riemannian manifold. Intuitively, a manifold can be viewed as a curved smooth space that may represent the structure of some data more accurately. One can also view a Riemannian manifold as a generalization of a Euclidean space. When the variables in an optimization problem must satisfy certain rules, such as symmetric positive definite (SPD) matrices, unit vectors, or low-rank matrices/tensors, Riemannian optimization smoothly handles these rules by exploring their geometry structures.

In this paper, we focus on problems in the form of

$$\min_{w \in \mathcal{M}} \left\{ F(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w; z_i) \right\}, \tag{2}$$

where $\mathcal{M}$ is a Riemannian manifold, and f_i is usually related to the i-th sample z_i in the dataset $\mathcal{D} = \{z_1, \cdots, z_n\}$. Riemannian optimization has been widely-used in many problems in the form of equation 2 including the principal eigenvector computation (Garber & Hazan, 2015; Zhang et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2016), the Karcher mean problem for SPD matrices (Jeuris et al., 2012; Bini & Iannazzo, 2013), blind source (Absil & Gallivan, 2006; Amari et al., 2000), matrix completion (Balzano et al., 2010; Sato et al., 2019) and embedding taxonomies (Nickel & Kiela, 2017).

For the optimization problem equation 1 with large datasets, the stochastic gradient descent (SGD) algorithm (Bottou, 1998; 2010) is often used. Its main idea is to use a small batch of data to estimate the gradient in each iteration, which effectively reduces the computational cost. Since then, a series of improved algorithms based on SGD have been proposed (Ruder, 2016), such as momentum (Qian, 1999), Nesterov Accelerated Gradient(NAG) (Nesterov, 1983), AdaGrad (Duchi et al., 2011), and RMSProp. Among them, the Adam algorithm (Kingma & Ba, 2015) is one of the most popular methods in practice. It combines the advantages of the momentum and the RMSProp algorithm. It calculates not only the first-order moment estimate of the gradients (i.e., momentum) but also the second-order moment estimate, allowing it to adaptively adjust the learning rate for each parameter. Later, Reddi et al. (2018) modified the second-moment estimation in the Adam algorithm, proposing the AMSGrad algorithm, which provides more stable convergence properties of the algorithm.

In the Euclidean setting, the algorithms mentioned above are commonly used and have shown effective performance. However, in the manifold setting equation 2, the existing works are still limited. This is because extending algorithms to Riemannian manifolds involves several challenges: there is no standard coordinate system on the manifold, and the manifold itself has a nonlinear geometric structure.

Bonnabel (2013) was one of the first to attempt in this area, he introduced the Riemannian SGD algorithm (RSGD), which is an extension of the SGD algorithm to Riemannian manifolds. Zhang & Sra (2016) proposed a new inequality for analyzing the behavior of optimization algorithms in Alexandrov spaces with bounded curvature, which can be applied to optimization problems on Riemannian manifolds. This inequality was also used in the convergence analysis of the algorithms therein. Furthermore, for general geodesically convex optimization problems, they provided a global complexity analysis for first-order algorithms for the first time, providing new perspectives and tools for understanding optimization algorithms in complex geometric settings. Subsequently, Zhang & Sra (2018) proposed a Riemannian Nesterov Accelerated Gradient (RAGD) algorithm and analyzed its convergence. RAGD uses an exponential mapping to update the parameters based on the NAG algorithm. Roy et al. (2018) utilized retraction and parallel transport to extend the momentum algorithm and the RMSProp algorithm, resulting in the creation of the cSGD-M algorithm and the cRMSProp algorithm. Bécigneul & Ganea (2019) extended Adam and AMSGrad algorithms on product manifolds, denoted by p-Adam and p-AMSGrad respectively, treating each manifold as a "coordinate" and utilizing parallel transport and exponential mapping to naturally extend Euclidean algorithms to product manifolds. Theoretically, they also provided proof of global convergence under geodesically convex conditions. Sakai & Iiduka (2025) proposed a general framework for Riemannian adaptive optimization with convergence guarantees and mini-batch analysis. Their proposed algorithms will lose adaptivity on non-product manifolds. Therefore, the extension of the Adam and AMSGrad algorithms to general Riemannian manifolds, which are not product manifolds, remains an area for further exploration. Kasai et al. (2019) proposed the Riemannian Adaptive Stochastic Gradient Algorithms(RASA). RASA cleverly avoids the need for vector transport by performing adaptive scaling on the row and column subspaces of the gradients. However, this subspace-based approach relies heavily on the specific algebraic structure of matrix manifolds and is not generalizable to arbitrary Riemannian manifolds. In contrast, our proposed methods utilize the intrinsic representation of tangent vectors coupled with vector transport by parallelization. This fundamentally geometric approach enables true element-wise adaptivity and is universally applicable to any general Riemannian embedded submanifolds of Euclidean space, without being restricted to specific matrix structures.

The contribution of this paper lies in the following aspects:

- We propose Riemannian Adam (RAdam) and Riemannian AMSGrad (RAMSGrad) by extending Adam and AMSGrad to Riemannian manifolds through vector transport by parallelization and exponential mapping, respectively.
- Under the conditions of geodesically convex problems, we prove the global convergence of the RAMSGrad algorithm.
- We provide a C++ implementation of RAdam and RAMSGrad, which relies on the Riemannian optimization package ROPTLIB (Huang et al., 2018).

- In the numerical experiments, the proposed RAdam and RAMSGrad are comprehensively compared to state-of-the-art algorithms using Karcher mean, joint diagonalization and Poincaré embedding problems. These baselines include the standard Riemannian stochastic gradient descent(RSGD), the Riemannian adaptive stochastic gradient algorithms(RASA), and existing adaptive methods for product manifolds(p-Adam and p-AMSGrad).

The paper is organized as follows. Section 2 introduces preliminaries on which this paper is based. In Section 3, we extend Adam and AMSGrad to Riemannian manifolds with the help of vector transport by parallelization and exponential mapping, proposing RAdam and RAMSGrad, and proving the convergence analysis of the RAMSGrad. In Section 4, we verify the effectiveness of the algorithms through three problems. Section 5 summarizes this paper and discusses the future research.

2 PRELIMINARIES

This section reviews the Riemannian optimization tools and manifold structures used in this paper, followed by Euclidean SGD, Adam, AMSGrad and related extensions. Unless otherwise specified, vector operations are componentwise. For $x, y \in \mathbb{R}^d$ and $c \in \mathbb{R}$, $x \pm c$, cx, x/c, $\sqrt{x}$, $x \odot y$, $x \oslash y$, and $\max(x, y)$ have their standard componentwise meanings. The superscript (i) denotes the i-th component of a vector.

2.1 RIEMANNIAN OPTIMIZATION

A d-dimensional smooth manifold $\mathcal{M}$ is locally homeomorphic to an open subset of $\mathbb{R}^d$, where $\mathbb{R}^d$ itself is a basic example with zero curvature. If $\mathcal{M}$ is a smooth subset of a linear space $\mathcal{E}$ whose topology and differential structure are inherited from $\mathcal{E}$, then $\mathcal{M}$ is called an embedded submanifold of $\mathcal{E}$. For $x \in \mathcal{M}$, the tangent space $\mathrm{T}_x\mathcal{M}$ is the linear space of tangent vectors to smooth curves on $\mathcal{M}$ passing through x, and the tangent bundle is denoted by $\mathrm{T}\mathcal{M} = \bigcup_{x\in\mathcal{M}} \mathrm{T}_x\mathcal{M}$.

We introduce the intrinsic representation of tangent vectors proposed by Huang et al. (2017), which will be used in the generalization of our algorithms later. Suppose a d-dimensional manifold is an embedded submanifold of a w-dimensional Euclidean space. A tangent vector η_x can be represented as a w-dimensional vector, called the extrinsic representation, denoted by η_x^{w} with a superscript w, or as a d-dimensional vector, called the intrinsic representation, denoted by η_x^{d} with a superscript d. Specifically, given $\eta_x \in \mathrm{T}_x\,\mathcal{M}$ and a basis B_x of $\mathrm{T}_x\,\mathcal{M}$, find $c \in \mathbb{R}^d$ such that $\eta_x = B_x c$, then c is called the intrinsic representation of the tangent vector η_x under the basis B_x. Using the intrinsic representation for computation brings several advantages. One is that it employs lower-dimensional vectors, reducing time and space complexity. The other is that the vector transport by parallelization by intrinsic representation is a simple identity mapping, which will be discussed later.

A Riemannian metric on $\mathcal{M}$ is an inner product $g_x : \mathrm{T}_x\mathcal{M} \times \mathrm{T}_x\mathcal{M} \to \mathbb{R}$, which induces the norm $\|\eta_x\|_x := \sqrt{g_x(\eta_x, \eta_x)}$, where $\eta_x \in \mathrm{T}_x\mathcal{M}$. A manifold equipped with such a metric is called a Riemannian manifold and is denoted by $(\mathcal{M}, g)$. The metric also induces the distance $d(x, y) = \inf_c \int_0^1 \|\dot{c}(t)\|_{c(t)}\,\mathrm{d}t$, where the infimum is taken over all smooth curves $c : [0, 1] \to \mathcal{M}$ satisfying $c(0) = x$ and $c(1) = y$. A geodesic is a smooth curve that locally minimizes this length.

Let $\mathrm{T}\mathcal{M} \oplus \mathrm{T}\mathcal{M} = \{(\eta_x, \xi_x) : x \in \mathcal{M}, \eta_x \in \mathrm{T}_x\mathcal{M}, \xi_x \in \mathrm{T}_x\mathcal{M}\}$ denote the Whitney sum of two tangent vectors. A vector transport $\mathcal{T} : \mathrm{T}\,\mathcal{M} \oplus \mathrm{T}\,\mathcal{M} \to \mathrm{T}\,\mathcal{M} : (\eta_x, \xi_x) \mapsto \mathcal{T}_{\eta_x}\xi_x$ with associated retraction R is a smooth mapping such that, $\forall x \in \mathcal{M}, \eta_x, \xi_x \in \mathrm{T}_x\,\mathcal{M}$, the following properties are satisfied: (i)$\mathcal{T}_{\eta_x}\xi_x \in \mathrm{T}_{R_x(\eta_x)}\,\mathcal{M}$; (ii)$\mathcal{T}_{0_x}\xi_x = \xi_x$; (iii)$\mathcal{T}_{\eta_x}$ is a linear mapping. Commonly encountered vector transports include vector transport by differentiated retraction, parallel transport, and vector transport by projection. However, these vector transports may have shortcomings such as no explicit expressions or expensive computational costs. This paper adopts vector transport by parallelization, a concept proposed by Huang et al. (2015). Its definition is as follows:

$$\mathcal{T}_{\eta_x}\xi_x = B_y B_x^{\flat}\xi_x, \tag{3}$$

where $y = R_x(\eta_x)$, $B : \mathcal{U} \to (\mathrm{T}\mathcal{M})^d : z \mapsto B_z$ is a smooth mapping, d is the dimension of $\mathcal{M}$, $\mathcal{U}$ is the neighborhood of x in $\mathcal{M}$, B_z is a set of bases of $\mathrm{T}_z\mathcal{M}$, the mapping $B_x^{\flat} : \mathrm{T}_x\mathcal{M} \to \mathbb{R}^d : \xi_x \mapsto v$ ensures that v satisfies $\xi_x = B_x v$. There also exists an intrinsic representation for vector transport

by parallelization. Assuming the intrinsic representation of $\xi_x \in \mathrm{T}_x\mathcal{M}$ is $v_x = B_x^\flat \xi_x$, the intrinsic representation of (3) can be written as

$$\mathcal{T}_{\eta_x}^{\mathrm{d}} v_x = B_y^\flat \mathcal{T}_{\eta_x} \xi_x = B_y^\flat B_y B_x^\flat B_x v_x = v_x. \tag{4}$$

It is not difficult to see that the intrinsic representation of vector transport by parallelization is essentially an identity mapping, making the computation process very simple. In particular, when $B_x = \bigcup_i \{b_i\}$ is an orthonormal basis, in the intrinsic representation, the Riemannian metric degenerates to the Euclidean metric, that is,

$$\tilde{g}_x(v_x, u_x) := g_x(\eta_x, \xi_x) = g_x(B_x v_x, B_x u_x) = v_x^T u_x, \tag{5}$$

where $\tilde{g}_x$ is the metric in intrinsic representation, $\eta_x = B_x v_x \in \mathrm{T}_x\mathcal{M}$, and $\xi_x = B_x u_x \in \mathrm{T}_x\mathcal{M}$.

Next, we directly present a generalization of convexity to Riemannian manifolds, containing the concepts of geodesically completeness, geodesically convex sets and geodesically convex functions on Riemannian manifolds, which will be used in the assumptions in Section 3.

A Riemannian manifold is geodesically complete, if every geodesic of the Riemannian manifold can be extended infinitely. A subset S of a Riemannian manifold $\mathcal{M}$ is called geodesically convex if, for every $x, y \in S$, there exists a geodesic segment $c : [0, 1] \to \mathcal{M}$ such that $c(0) = x, c(1) = y$ and $c(t)$ is in S for all $t \in [0, 1]$. For S which is a geodesically convex set, we say $f : S \to \mathbb{R}$ is geodesically convex if for all $x, y \in S$ and all geodesics c connecting x to y in S the function $f \circ c : [0, 1] \to \mathbb{R}$ is convex, that is, $f(c(t)) \leq (1 - t)f(x) + tf(y), \forall t \in [0, 1]$.

2.2 Riemannian Manifolds

The geometry structures and Riemannian optimization tools of the SPD manifold, Stiefel manifold and Poincaré ball, are reviewed in Appendix 6.1 as examples since they are used later in the numerical experiments.

2.3 SGD, Adam, AMSGrad, and Their Existing Extensions

In the Euclidean space $\mathbb{R}^n$, SGD (Bottou, 1998) is updated using the following form

$$x_{t+1} = x_t - \alpha \nabla F_t(x_t), \tag{6}$$

where $\nabla F_t(x_t)$ is the stochastic gradient, usually the gradient of the objective function on a batch of data at the t-th iteration, and $\alpha > 0$ is the stepsize (also called the learning rate in machine learning). When the optimization space becomes a Riemannian manifold, the update formula of RSGD (Bonnabel, 2013) is as follows:

$$x_{t+1} = R_{x_t}(-\alpha \operatorname{grad} F_t(x_t)), \tag{7}$$

where $\operatorname{grad} F_t(x_t)$ is the Riemannian stochastic gradient. The main differences between equation 6 and equation 7 are that RSGD uses the Riemannian gradient instead of the Euclidean gradient and that RSGD replaces the addition in Euclidean spaces with a retraction R_x, ensuring that the updated point x_{t+1} remains on the manifold.

The Adam algorithm proposed by Kingma & Ba (2015) is given by

$$\begin{aligned} &m_0 = 0, v_0 = 0, \epsilon > 0; \\ &\begin{cases} g_t = \nabla F_t(x_t) \\ m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t \odot g_t \\ \hat{m}_t = m_t / (1 - \beta_1^t) \\ \hat{v}_t = v_t / (1 - \beta_2^t) \\ x_{t+1} = x_t - \alpha \hat{m}_t \oslash (\sqrt{\hat{v}_t} + \varepsilon) \end{cases}, \end{aligned} \tag{8}$$

where m_t and v_t are the first and second moment estimates of the gradient, $\hat{m}_t$ and $\hat{v}_t$ are the initial bias correction terms for m_t and v_t, and β_1 and β_2 control the exponential decay rates for the gradients and second moments of gradients, respectively. Compared to SGD, the search direction of

Adam relies not only on the gradient of the current step but also incorporates gradient information from previous steps. Moreover, Adam is considered as an adaptive algorithm because it adjusts the stepsize for each coordinate component of the parameter based on the second moment estimates. This adaptive behavior allows Adam to handle sparse gradients effectively, often resulting in better performance than non-adaptive optimization methods like SGD (Kingma & Ba, 2015).

Reddi et al. (2018) pointed out an error in the convergence proof of Adam algorithm given in (Kingma & Ba, 2015) and corrected it, leading to the proposal of the AMSGrad algorithm. The modification part is as follows:

$$x_{t+1} = x_t - \alpha m_t \oslash (\sqrt{\hat{v}_t} + \varepsilon), \tag{9}$$

where $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ denotes the entry-wise maximum. This modification ensures that $\hat{v}_t$ is non-decreasing over time, which provides AMSGrad with convergence guarantee. A more recent work by Zhang et al. (2022) showed that Adam guaranteed convergence without any modifications.

It is not straightforward to generalize Adam and AMSGrad to Riemannian manifolds because there is no inherent coordinate system on the manifold itself in general. So Bécigneul & Ganea (2019) extended these two algorithms to product manifolds by viewing each component manifold as a "coordinate". They set different learning rates for different component manifolds based on the norm information of the gradient of each component manifold. However, when the product manifold degenerates into a single manifold, the same learning rate is used for the single "coordinate", thus losing its adaptability.

3 Generalization of Adam and AMSGrad on Riemannian Manifolds

In this section, we propose RAdam and RAMSGrad algorithms, which generalize Adam and AMSGrad to the more general case, i.e. non-product manifolds, and prove the global convergence of RAMSGrad for geodesically convex objective functions.

3.1 Algorithm Design

From the Euclidean Adam in equation 8, we can see that the algorithm requires a weighted sum of past gradients and the current gradient. Assume that the iteration proceeds to the t-th step and use equation 3 to transport the past gradient information $m_{t-1} \in \mathbb{R}^d$ in its intrinsic representation from $\mathrm{T}_{x_{t-1}}\mathcal{M}$ to $\mathrm{T}_{x_t}\mathcal{M}$, that is,

$$\mathcal{T}^{\mathrm{d}}_{\eta^{\mathrm{w}}_{t-1}} m_{t-1} = m_{t-1}, \tag{10}$$

where $\eta^{\mathrm{w}}_{t-1} \in \mathrm{T}_{x_{t-1}}\mathcal{M}$ satisfying $x_t = \mathrm{Exp}_{x_{t-1}}(\eta^{\mathrm{w}}_{t-1})$. The Riemannian gradient $g^{\mathrm{w}}_t = \operatorname{grad} F_t(x_t)$ is converted into the intrinsic representation g^{d}_t, so that linear combination of m_{t-1} and g^{d}_t can be performed in the space of the intrinsic representation of $\mathrm{T}_{x_t}\mathcal{M}$, i.e., $\mathbb{R}^d$. Subsequent operations for updating direction and stepsize are performed in the same linear space. When updating x_t, it is important to first convert the direction η^{d}_t to its extrinsic representation and then perform a retraction. Specifically, we present the RAdam and RAMSGrad in Algorithm 1 and Algorithm 2 respectively.

It is assumed throughout this paper that $\forall t$, B_{x_t} is an orthonormal basis, which ensures the property of equation 5. Note that the basis B_x is viewed as a coordinate system in $\mathrm{T}_x\mathcal{M}$. Therefore, the coordinate system on $\mathrm{T}_x\mathcal{M}$ varies as x.

Similar to the Euclidean version, RAMSGrad adds the max operation on top of the RAdam. Note that Steps 7 to 11 in Algorithm 2 are not generalized from its Euclidean version. These steps are used to control the maximum difference among the components in the vector $\sqrt{\hat{v}_t}$, which are used in the global convergence analysis in Theorem 3.5. The prescribed sequence $\{\phi(t)\}$ is decreasing and we choose $\phi(t) = C/t^a$ as an example for the analysis, where C is a positive constant and $a \in (0, 1)$.

Algorithm 1 RAdam

Require: Initial point $x_1 \in \mathcal{M}$; loss function $\{F_t\}$; exponential decay rate $\{\beta_{1t}\}, \beta_2 \in [0,1)$; stepsize $\{\alpha_t\}$; $\varepsilon > 0$; number of iterations T;

$m_0 = 0$, $v_0 = 0$;
for $t = 1$ to T **do**
  $g_t^{\mathrm{w}} = \mathrm{grad} F_t(x_t),\ g_t^{\mathrm{d}} = B_{x_t}^{\flat} g_t^{\mathrm{w}}$;
  $m_t = \beta_{1t} m_{t-1} + (1-\beta_{1t}) g_t^{\mathrm{d}}$;
  $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^{\mathrm{d}} \odot g_t^{\mathrm{d}}$;
  $\hat{m}_t = m_t / (1-\beta_{1t}^t)$;
  $\hat{v}_t = v_t / (1-\beta_2^t)$;
  $\eta_t^{\mathrm{d}} = -\alpha_t \hat{m}_t \oslash (\sqrt{\hat{v}_t} + \varepsilon),\ \eta_t^{\mathrm{w}} = B_{x_t} \eta_t^{\mathrm{d}}$;
  $x_{t+1} = \mathrm{Exp}_{x_t}(\eta_t^{\mathrm{w}})$;
end for
return x_t;

Algorithm 2 RAMSGrad

Require: initial point $x_1 \in \mathcal{M}$, loss function $\{F_t\}$, exponential decay rate $\{\beta_{1t}\},\ \beta_2 \in [0,1)$, stepsize $\{\alpha_t\}$, prescribed sequence $\{\phi(t)\}$; $\varepsilon > 0$; number of iterations T;

$m_0 = 0, v_0 = 0, \hat{v}_0 = 0$;
for $t = 1$ to T **do**
  $g_t^{\mathrm{w}} = \mathrm{grad} F_t(x_t),\ g_t^{\mathrm{d}} = B_{x_t}^{\flat} g_t^{\mathrm{w}}$;
  $m_t = \beta_{1t} m_{t-1} + (1-\beta_{1t}) g_t^{\mathrm{d}}$;
  $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^{\mathrm{d}} \odot g_t^{\mathrm{d}}$;
  $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$;
  $\sqrt{\hat{v}}_{t,max} = \max_i \{\sqrt{\hat{v}_t^{(i)}}\}$;
  $\sqrt{\hat{v}}_{t,min} = \min_i \{\sqrt{\hat{v}_t^{(i)}}\}$;
  if $\sqrt{\hat{v}}_{t,max} - \sqrt{\hat{v}}_{t,min} > \phi(t)$ **then**
    $\sqrt{\hat{v}_t} = \frac{\sqrt{\hat{v}_t} - \sqrt{\hat{v}}_{t,min}}{\sqrt{\hat{v}}_{t,max} - \sqrt{\hat{v}}_{t,min}} \cdot \phi(t) + \sqrt{\hat{v}}_{t,max} - \phi(t)$;
  end if
  $\eta_t^{\mathrm{d}} = -\alpha_t m_t \oslash (\sqrt{\hat{v}_t} + \varepsilon),\ \eta_t^{\mathrm{w}} = B_{x_t} \eta_t^{\mathrm{d}}$;
  $x_{t+1} = \mathrm{Exp}_{x_t}(\eta_t^{\mathrm{w}})$;
end for
return x_t;

3.2 Convergence Analysis

The global convergence of the RAMSGrad algorithm for geodesically convex functions is established in this section. The analysis relies on the following assumptions.

Assumption 3.1. *The Riemannian manifold $(\mathcal{M}, g)$ is geodesically complete with sectional curvature lower bounded.*

Assumption 3.2. *$\forall t \in \{1, 2, \ldots, T\}$, the function $F_t : \mathcal{M} \to \mathbb{R}$ is a differentiable geodesically convex function.*

Assumption 3.3. *All iterates stay in a bounded set, i.e., $d(x_i, x_j) \le D_\infty$.*

Assumption 3.4. *The gradient of the objective function is limited by G_∞, i.e. $\forall x \in \mathcal{M}$, $t \in [1, T]$, $\|\mathrm{grad} F_t(x)\|_x \le G_\infty$.*

Assumption 3.1 generally holds on some manifolds like Hadamard manifolds (Absil et al., 2008; Boumal, 2023; Zhang & Sra, 2016). Assumptions 3.2, 3.3 and 3.4 are standard in the convergence analysis of Riemannian adaptive methods (Reddi et al., 2018; Bécigneul & Ganea, 2019). Notably, regarding Assumption 3.3, bounding the maximum distance between iterates by a diameter D_∞ is geometrically indispensable. In Riemannian settings, the regret bound explicitly depends on D_∞ to control the curvature-dependent geometric penalty factor $\zeta(\kappa, D_\infty)$ arising from the Alexandrov space cosine inequality (see Lemma 6.1).

We need to prove the convergence of the RAMSGrad by limiting the regret value. At the end of the T-th iteration, the regret value is defined as follows:

$$Reg_T = \sum_{t=1}^{T} F_t(x_t) - \min_{x \in \mathcal{M}} \sum_{j=1}^{T} F_j(x).$$

Regret value is a core concept in online learning, which accumulates the deviation between the model's decisions and the optimal decisions in the ideal situation. An algorithm with low regret indicates that its performance is close to the theoretical optimal strategy. Theorem 3.5 gives a regret bound that is shown to be $o(T)$ when β_{1t} decreases exponentially, as shown in Corollary 3.6.

Theorem 3.5 (Convergence of RAMSGrad)**.** *Assume that $\{x_t\}$ and $\{\hat{v}_t\}$ are sequences obtained from Algorithm 2. For all $t \in [1, T]$, $\beta_{1t} \leq \beta_1 < 1$, $\sqrt{\hat{v}}_{t,max} - \sqrt{\hat{v}}_{t,min} \leq \phi(t) := C/t^a$, $a \in (0, 1)$, $\alpha_t = \alpha/t^b$, $b \in (0, 1)$, and $\gamma := \beta_1/\sqrt{\beta_2} < 1$, we can obtain the following regret bound*

$$\begin{aligned} Reg_T \leq & \frac{D_\infty^2}{2(1-\beta_1)\alpha} \left(C \frac{T^{1-(a-b)}}{1-(a-b)} + T^b \max_i \left\{ \sqrt{\hat{v}_T^{(i)}} \right\} \right) \\ & + \frac{\zeta(\kappa, D_\infty) + 1}{2} \frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)^2 \sqrt{1-\beta_2}(1-b)(1-\gamma)} + \frac{D_\infty^2}{2(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \sqrt{\hat{v}_t^{(i)}}}{\alpha_t}. \end{aligned} \tag{11}$$

Proof. See Appendix 6.2.3. □

Comparing Theorem 3.5 with Theorem 4 in (Reddi et al., 2018), we observe that the first term on the right side of the inequality is different in form due to the introduction of the bound related to $\hat{v}_t$, but it is still $o(T)(T \to \infty)$, when we select appropriate $a, b \in (0, 1)$ and $a - b \in (0, 1)$. The second term is $o(T)$ when $b \in (0, 1)$ and if the Riemannian manifold degenerates to Euclidean space, $\zeta(\kappa, D_\infty) = 1$. The last term is identical to (Reddi et al., 2018)[Theorem 4] except for a constant factor.

Corollary 3.6. *Suppose $\beta_{1t} = \beta_1 \lambda^{t-1}$, $0 < \lambda < 1$, we can derive the following inequality:*

$$\begin{aligned} Reg_T \leq & \frac{D_\infty^2}{2(1-\beta_1)\alpha} \left(C \frac{T^{1-(a-b)}}{1-(a-b)} + T^b \max_i \left\{ \sqrt{\hat{v}_T^{(i)}} \right\} \right) \\ & + \frac{\zeta(\kappa, D_\infty) + 1}{2} \frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)^2 \sqrt{1-\beta_2}(1-b)(1-\gamma)} + \frac{d \beta_1 D_\infty^2 G_\infty}{2\alpha(1-\beta_1)(1-\lambda)^2}. \end{aligned} \tag{12}$$

Proof. See Appendix 6.2.4. □

Comparing Corollary 3.6 with the existing Euclidean result (Reddi et al., 2018)[Corollary 1], we observe that the first term and second term on the right side of the inequality are same as the case that Theorem 3.5 is being compared with the result in (Reddi et al., 2018)[Theorem 4]. The last term is identical to (Reddi et al., 2018)[Corollary 1] except for a constant factor. These prove that the RAMSGrad algorithm converges in the sense that $Reg_T = o(T)$.

4 Numerical Experiments

In this section, we conduct experiments on three problems: Poincaré embedding (Nickel & Kiela, 2017), Karcher mean (Jeuris et al., 2012) and joint diagonalization (Theis et al., 2009) based on the Riemann optimization package ROPTLIB (Huang et al., 2018). Poincaré embedding problems have been used to test the performance of Riemannian adaptive optimization algorithms (Nickel & Kiela, 2017); Karcher mean is a geodesically strongly convex function with a unique minimizer; the joint diagonalization problems are known not to satisfy the assumptions of the analysis, but the proposed algorithms work well empirically. Due to the space limitation, the numerical results of Karcher mean and joint diagonalization are given in the Appendix 6.3.1. To ensure the reproducibility of experiments, the detailed test environment and core parameter settings of all experiments in this paper are summarized in Appendix 6.3.

4.1 POINCARÉ EMBEDDING

Testing Problem and Dataset

For large datasets with hierarchical structure, Nickel & Kiela (2017) chose to use the Poincaré ball for embedding. Compared to Euclidean space, the Poincaré ball greatly reduces the dimensionality of the embeddings. Given a set of pairs of objects with relationships $\mathcal{D} = \{(u, v)\}$, the goal is to learn embeddings $x = \{x_i\}_{i=1}^n$ for all objects in $\mathcal{D}$ such that related objects are close to each other in the embedding space. This corresponds to solving the following optimization problem:

$$\min_{x \in \prod_{i=1}^n \mathcal{B}^d} F(x) = \frac{1}{|\mathcal{D}|} \sum_{(u,v)\in\mathcal{D}} -\log \frac{e^{-d(x_u, x_v)}}{\sum_{v' \in \mathcal{N}(u)} e^{-d(x_u, x_{v'})}},$$

where $\prod_{i=1}^n \mathcal{B}^d$ represents the product manifold of n Poincaré balls, $|\mathcal{D}|$ represents the size of the dataset, and $\mathcal{N}(u) = \{v' | (u, v') \notin \mathcal{D}\} \cup \{v\}$ is the negative sample set for object u (including v). The distance function $d(u, v)$ is given by (17).

WordNet is a large English lexical database, and we use a subset of its noun hierarchy's transitive closure, with mammals as the root node. The mammals subset consists of about 1180 nouns and approximately 6540 hypernym-hyponym relationships, meaning the number of embedded objects n is 1180, and the number of samples $|\mathcal{D}|$ is 6540.

Results and Analysis

We compare the proposed algorithm with RSGD and the AMSGrad and Adam on product manifolds from (Bécigneul & Ganea, 2019), denoted as p-RAMSGrad and p-RAdam. Each algorithm uses a fixed stepsize, and we tried different values from the set $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3\}$. In the experiments, we used addition retraction and exponential mapping. Figures 1, Table 1, and Table 2 show the results of the former, while Figure 2, Table 3, and Table 4 present the results of the latter.

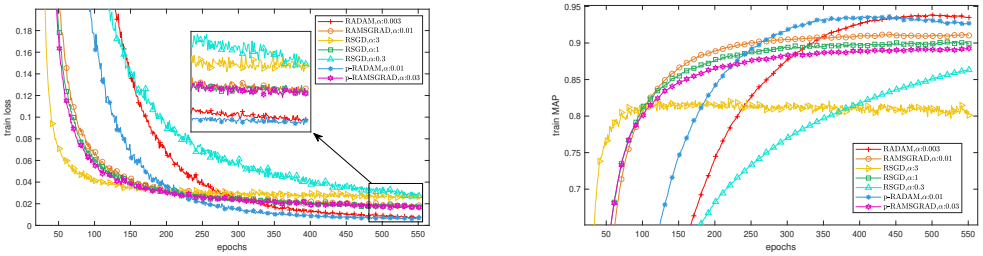


Figure 1: For reconstruction, the result of addition retraction is used.

Table 1: For reconstruction, the addition retraction is used, and the results of each algorithm are recorded at the stopping point.

Algorithms	RAMSGrad	RAdam	p-RAMSGrad	p-RAdam	RSGD	RSGD	RSGD
α	0.01	0.003	0.03	0.01	0.3	1	3
train loss	1.86×10^{-2}	7.51×10^{-3}	1.67×10^{-2}	7.09×10^{-3}	2.73×10^{-2}	1.70×10^{-2}	2.66×10^{-2}
MAP	0.910	0.935	0.893	0.927	0.863	0.899	0.801
training time(s)	665.07	454.31	790.55	786.32	459.46	457.73	459.39

Table 2: For link prediction, the addition retraction is used, and the MAP of each algorithm on the test set.

Algorithms	RAMSGrad	RAdam	p-RAMSGrad	p-RAdam	RSGD
α	0.003	0.001	0.01	0.003	0.3
MAP	0.481	0.494	0.461	0.490	0.449

The results show that in terms of reconstruction, whether it is addition retraction or exponential mapping, the RAdam exhibited the best overall performance with relatively low train loss, the highest

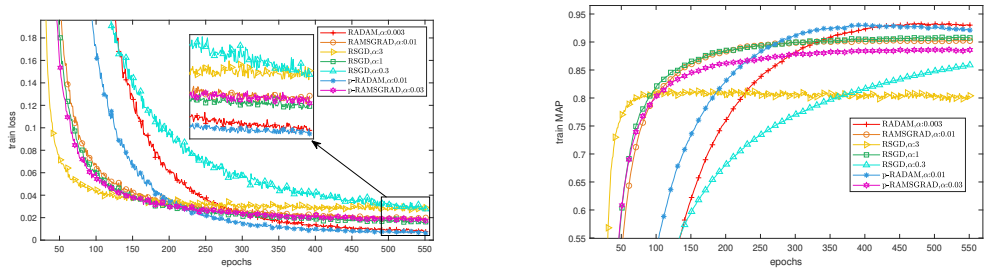


Figure 2: For reconstruction, the result of the exponential mapping is used.

Table 3: For reconstruction, the exponential mapping is used, and the results of each algorithm are recorded at the stopping point.

Algorithms	RAMSGrad	RAdam	p-RAMSGrad	p-RAdam	RSGD	RSGD	RSGD
α	0.01	0.003	0.03	0.01	0.3	1	3
train loss	2.00×10^{-2}	7.55×10^{-3}	1.73×10^{-2}	6.28×10^{-3}	2.76×10^{-2}	1.62×10^{-2}	2.90×10^{-2}
MAP	0.902	0.930	0.886	0.921	0.859	0.908	0.804
training time(s)	674.87	461.15	795.71	790.95	463.51	463.16	460.83

Table 4: For link prediction, the exponential mapping is used, and the MAP of each algorithm on the test set.

Algorithms	RAMSGrad	RAdam	p-RAMSGrad	p-RAdam	RSGD
α	0.003	0.001	0.01	0.003	0.3
MAP	0.508	0.509	0.467	0.504	0.482

MAP, and the shorter training time. The MAP of the RAMSGrad was generally higher than that of the p-RAMSGrad, and its training time was about 120 seconds shorter than that of the p-AMSGrad, mainly due to the more complex parallel transport operations used in p-AMSGrad.

For the link prediction, the RAdam and RAMSGrad outperformed the p-RAdam and p-RAMSGrad, demonstrating stronger generalization capabilities. Additionally, the application of exponential mapping in this task helped improve the MAP of all algorithms on the test set. Although the training time required for exponential mapping was longer than that for addition retraction, this increase was not significant in the experiments due to the sparsity of parameter updates in this problem.

While the primary theoretical contribution of our proposed RAdam and RAMSGrad lies in enabling element-wise adaptivity on non-product manifolds, we use the Poincaré embedding task—a product manifold problem—to establish a direct and fair baseline comparison with p-RAdam and p-RAMSGrad (Bécigneul & Ganea, 2019). This serves to demonstrate that our intrinsic coordinate approach remains highly competitive and robust even on the specific geometrical structures where product-manifold adaptive methods were originally designed to operate.

5 Conclusions

This paper successfully extends Adam and AMSGrad to Riemannian manifolds using vector transport by parallelization, introducing the RAdam and RAMSGrad algorithms. Under the assumption of geodesic convexity, we provide an upper bound on the regret of the RAMSGrad, thus proving its global convergence. In practice, we implemented the proposed algorithms using the Riemannian optimization package ROPTLIB, further expanding the solver library of ROPTLIB. Various problems were used to verify the effectiveness of our algorithms, and the experiments show that the RAdam and RAMSGrad algorithm demonstrates performance comparable to or even better than baseline algorithms. Additionally, our algorithms can be adapted to extend first-order optimization algorithms like Momentum and RMSProp to Riemannian manifolds through vector transport by parallelization, which could be explored in future work.

AI USE STATEMENT

We did not use Large Language Models or other AI tools in paper writing.

REPRODUCIBILITY STATEMENT

For algorithms we proposed, source code will be submitted as supplementary material; for theoretical results, clear explanations of a complete proof of the claims are included in the appendix; for any datasets used in the experiments, a complete description of the data processing steps is provided in the main text or the appendix.

REFERENCES

P-A Absil and Kyle A Gallivan. Joint diagonalization on the oblique manifold for independent component analysis. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, pp. V–V. IEEE, 2006.

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

Shun-ichi Amari, Tian-Ping Chen, and Andrzej Cichocki. Nonholonomic orthogonal learning algorithms for blind source separation. *Neural Computation*, 12(6):1463–1484, 2000.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational Poincaré graph embeddings. *Advances in Neural Information Processing Systems*, 32, 2019.

Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 704–711, 2010. doi: 10.1109/ALLERTON.2010.5706976.

Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1eiqi09K7.

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.

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

Léon Bottou. Online learning and stochastic approximations. *Online Learning in Neural Networks*, 17(9):142, 1998.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pp. 177–186. Springer, 2010.

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

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

P Thomas Fletcher and Sarang Joshi. Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Processing*, 87(2):250–262, 2007.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *Advances in neural information processing systems*, 31, 2018.

Daniel Garber and Elad Hazan. Fast and simple PCA via convex optimization. *ArXiv*, abs/1509.05647, 09 2015. URL https://api.semanticscholar.org/CorpusID:3166860.

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

Wen Huang, P-A Absil, and Kyle A Gallivan. Intrinsic representation of tangent vectors and vector transports on matrix manifolds. *Numerische Mathematik*, 136:523–543, 2017.

Wen Huang, P-A Absil, Kyle A Gallivan, and Paul Hand. Roptlib: an object-oriented C++ library for optimization on riemannian manifolds. *ACM Transactions on Mathematical Software (TOMS)*, 44(4):1–21, 2018.

Ben Jeuris, Raf Vandebril, and Bart Vandereycken. A survey and comparison of contemporary algorithms for computing the matrix geometric mean. *Electronic Transactions on Numerical Analysis*, 39:379–402, 2012.

Hiroyuki Kasai, Pratik Jawanpuria, and Bamdev Mishra. Riemannian adaptive stochastic gradient algorithms on matrix manifolds. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3262–3271. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kasai19a.html.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proceedings of 3rd International Conference on Learning Representations*, 2015.

Aaron Q Li, Amr Ahmed, Sujith Ravi, and Alexander J Smola. Reducing the sampling complexity of topic models. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 891–900, 2014.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 2013.

Maher Moakher. On the averaging of symmetric positive-definite tensors. *Journal of Elasticity*, 82: 273–296, 2006.

Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. *Doklady Akademii Nauk SSSR*, 269(3):543–547, 1983.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in Neural Information Processing Systems*, 30, 2017.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pp. 3779–3788. PMLR, 2018.

Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12 (1):145–151, 1999.

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

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryQu7f-RZ.

Soumava Kumar Roy, Zakaria Mhammedi, and Mehrtash Harandi. Geometry aware constrained optimization techniques for deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4460–4469, 2018.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *ArXiv*, abs/1609.04747, 2016. URL https://api.semanticscholar.org/CorpusID:17485266.

Hiroyuki Sakai and Hideaki Iiduka. A general framework of riemannian adaptive optimization methods with a convergence analysis, 2025. URL https://arxiv.org/abs/2409.00859.

Hiroyuki Sato, Hiroyuki Kasai, and Bamdev Mishra. Riemannian stochastic variance reduced gradient algorithm with retraction and vector transport. *SIAM Journal on Optimization*, 29(2):1444–1472, 2019. doi: 10.1137/17M1116787.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823, 2015.

Fabian J Theis, Thomas P Cason, and P A Absil. Soft dimension reduction for ICA by joint diagonalization on the Stiefel manifold. In *Independent Component Analysis and Signal Separation: 8th International Conference, ICA 2009, Paraty, Brazil, March 15-18, 2009. Proceedings 8*, pp. 354–361. Springer, 2009.

Roberto Tron and Kostas Daniilidis. The space of essential matrices as a Riemannian quotient manifold. *SIAM Journal on Imaging Sciences*, 10(3):1416–1445, 2017.

A.A. Ungar. *A Gyrovector Space Approach to Hyperbolic Geometry*. Synthesis lectures on mathematics and statistics. Morgan & Claypool Publishers, 2009. ISBN 9781598298222. URL https://books.google.com/books?id=dlFTgrhqm3IC.

Shihui Ying, Zhijie Wen, Jun Shi, Yaxin Peng, Jigen Peng, and Hong Qiao. Manifold preserving: An intrinsic approach for semisupervised distance metric learning. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):2731–2742, 2017.

Xinru Yuan, Wen Huang, P-A Absil, and Kyle A Gallivan. Computing the matrix geometric mean: Riemannian versus Euclidean conditioning, implementation techniques, and a Riemannian BFGS method. *Numerical Linear Algebra with Applications*, 27(5):e2321, 2020.

Pourya Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric mean metric learning. In *International Conference on Machine Learning*, pp. 2464–2471. PMLR, 2016.

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

Hongyi Zhang and Suvrit Sra. Towards Riemannian accelerated gradient methods. In *31th Annual Conference on Learning Theory (COLT'18)*. PMLR, 2018.

Hongyi Zhang, Sashank J Reddi, and Suvrit Sra. Riemannian SVRG: Fast stochastic optimization on Riemannian manifolds. *Advances in Neural Information Processing Systems*, 29, 2016.

Yushun Zhang, Congliang Chen, Naichen Shi, Ruoyu Sun, and Zhi-Quan Luo. Adam can converge without any modification on update rules. In *Advances in Neural Information Processing Systems*, volume 35, pp. 28386–28399, 2022.

6 Appendix

6.1 Riemannian Manifolds

6.1.1 SPD manifold

The SPD manifold $\mathcal{S}_{++}^n$ is the set of all $n \times n$ real symmetric positive definite matrices, that is,

$$\mathcal{S}_{++}^n = \{X \in \mathbb{R}^{n\times n} : X = X^T, X \succ 0\}.$$

It can be regarded as an embedded submanifold of $\mathbb{R}^{n\times n}$, with the embedding space dimension n^2, while the dimension of $\mathcal{S}_{++}^n$ is $\frac{n(n+1)}{2}$. It is often equipped with the following affine invariant Riemannian metric:

$$g_X(\xi_X, \eta_X) = \text{trace}(\xi_X X^{-1} \eta_X X^{-1}), \tag{13}$$

where $\eta_X, \xi_X \in \mathrm{T}_X\mathcal{S}_{++}^n = \{X \in \mathbb{R}^{n\times n} : X = X^T\}$. According to the properties of affine invariant metrics and symmetric matrices, one can find an orthonormal basis as follows:

$$B_X = \{Le_ie_i^TL^T, i = 1, \cdots, n\} \cup \left\{\frac{1}{\sqrt{2}}L(e_ie_j^T + e_je_i^T)L^T, i < j, i = 1, \cdots, n, j = 1, \cdots, n\right\},$$

where $X = LL^T$ is the Cholesky decomposition, and $\{e_1, \cdots, e_n\}$ is the canonical basis of $\mathbb{R}^n$. Through this basis, we can achieve the intrinsic representation of tangent vectors. For more details, please refer to reference (Yuan et al., 2020). In the experiments of this paper, we use the second order approximation of the exponential mapping proposed by Jeuris et al. (2012), given by

$$R_X(\eta_X) = X + \eta_X + \frac{1}{2}\eta_X X^{-1}\eta_X. \tag{14}$$

This retraction is cheaper in computational cost compared to the exponential mapping, and it can also avoid numerical overflow.

6.1.2 STIEFEL MANIFOLD

The Stiefel manifold $\mathrm{St}(n,p)$ with $p \leq n$ is the collection of all n-row by p-column orthogonal matrices, that is,

$$\mathrm{St}(n,p) = \left\{X \in \mathbb{R}^{n\times p} : X^T X = I_p\right\},$$

where I_p is a p-order identity matrix. It can be seen as an embedded submanifold of $\mathbb{R}^{n\times p}$, where the dimension of the embedding space is np, and the dimension of $\mathrm{St}(n,p)$ is $np - \frac{p(p+1)}{2}$. Its tangent space is

$$\mathrm{T}_X\mathrm{St}(n,p) = \{X\Omega + X_\perp K : \Omega = -\Omega^T, K \in \mathbb{R}^{(n-p)\times p}\},$$

where $X_\perp$ is a $n \times (n-p)$ matrix such that $[X\ X_\perp] \in \mathbb{R}^{n\times n}$ is orthonormal. By observing the structure of this tangent space, one can find the following basis

$$\begin{aligned} B_X = &\left\{\frac{1}{\sqrt{2}}X(e_i e_j^T - e_j e_i^T), i = 1, \cdots, p, j = i+1, \cdots, p\right\} \\ &\cup \{X_\perp \tilde{e}_i e_j^T, i = 1, \cdots, n-p, j = 1, \cdots, p\}, \end{aligned} \tag{15}$$

where $\{e_1, \cdots, e_p\}$ is the canonical basis of $\mathbb{R}^p$, and $\{\tilde{e}_1, \cdots, \tilde{e}_{n-p}\}$ is the canonical basis of $\mathbb{R}^{n-p}$. Note that equation 15 is orthonormal with respect to the metric $g_X(\eta_X, \xi_X) = \mathrm{trace}(\eta_X^T \xi_X)$. Please refer to reference (Huang et al., 2017) for more details on the intrinsic and extrinsic representations of tangent vectors using this basis. The qf-retraction (Absil et al., 2008) defined by

$$R_X(\eta_X) = \mathrm{qf}(X + \eta_X),$$

is used, where $\mathrm{qf}(A)$ denotes the Q factor after QR decomposition, and the diagonal elements of the R factor are non-negative.

6.1.3 POINCARÉ BALL

The Poincaré ball is defined by

$$\mathcal{B}^n = \{x \in \mathbb{R}^n : \|x\| < 1\},$$

where $\|\cdot\|$ denotes the Euclidean norm, i.e., $\|v\| = \sqrt{v^T v}$. Clearly, the Poincaré ball is an open subset of $\mathbb{R}^n$, making it an embedded submanifold of $\mathbb{R}^n$. The dimensions of both the embedding space and $\mathcal{B}^n$ are n. For any point $x \in \mathcal{B}^n$, the tangent space is $\mathbb{R}^n$. In the Poincaré ball, the Riemannian metric is

$$g_x(\eta_x, \xi_x) = \left(\frac{2}{1 - \|x\|^2}\right)^2 \eta_x^T \xi_x,\ \forall \eta_x, \xi_x \in \mathrm{T}_x\mathcal{B}^n. \tag{16}$$

An orthonormal basis on $\mathrm{T}_x\mathcal{B}^n$ with respect to Metric equation 16 is defined by

$$B_x = \left\{\frac{1 - \|x\|^2}{2}e_i, i = 1, \cdots, n\right\},$$

where $\{e_1, \cdots, e_n\}$ is the canonical basis of $\mathbb{R}^n$. Therefore, the conversion between the extrinsic and intrinsic representations of tangent vectors is given by

$$\eta_x = B_x v_x = \frac{1 - \|x\|^2}{2}v_x, \text{ and } v_x = B_x^\flat \eta_x = \frac{2}{1 - \|x\|^2}\eta_x,$$

where $\eta_x \in \mathrm{T}_x\mathcal{B}^n,\ v_x \in \mathbb{R}^n$. The distance between any two points x and y in $\mathcal{B}^n$ is

$$d(x,y) = \operatorname{arcosh}\left(1 + 2\frac{\|x-y\|^2}{(1-\|x\|^2)(1-\|y\|^2)}\right). \tag{17}$$

Let $\lambda_x = \frac{2}{1-\|x\|^2}$. Two retractions are commonly used: one is the exponential mapping

$$\exp_x(v) = x \oplus \left(\tanh\left(\frac{\lambda_x\|v\|}{2}\right)\frac{v}{\|v\|}\right), \tag{18}$$

where $\oplus$ is the generalized Möbius addition (see (Ganea et al., 2018)) which is a binary operation in $\mathcal{B}^n$ given by

$$x \oplus y := \frac{(1+2(x\cdot y)+||y||_2^2)x + (1-||x||_2^2)y}{1+2(x\cdot y)+||x||_2^2||y||_2^2}. \tag{19}$$

The other is the retraction by addition, i.e., $R_x(\eta_x) = x + \eta_x$. Since the image of the retractions may not still be in $\mathcal{B}^n$, an addition projection is further used

$$\mathrm{proj}(x) = \begin{cases} x/\|x\| - \varepsilon & \text{if } x \geq 1; \\ x & \text{otherwise,} \end{cases}$$

where ε is set to 10^{-5}. Note that such additional projection has been used before in (Bécigneul & Ganea, 2019). The parallel transport along a geodesic is given by $P_{x\to y}(v) = \frac{\lambda_x}{\lambda_y}\cdot \mathrm{gyr}[y,-x]v$, where gyr is the map given by the equations $\mathrm{gyr}[u,v]w = w + 2\frac{Au+Bv}{D}$, $A = -u\cdot w\,\|v\|^2 + v\cdot w + 2(u\cdot v)(v\cdot w)$, $B = -v\cdot w\,\|u\|^2 - u\cdot w$, and $D = 1 + 2u\cdot v + \|u\|^2\|v\|^2$, see details in (Ungar, 2009).

6.2 Convergence Analysis

6.2.1 Lemma 6.1

Lemma 6.1 (Cosine inequality in Alexandrov spaces)**.** *If a, b, c are the sides of a geodesic triangle in an Alexandrov space with curvature lower bounded by κ, and A is the angle between sides b and c, then*

$$a^2 \leq \zeta(\kappa,c)b^2 + c^2 - 2bc\cos(A),$$

where $\zeta(\kappa,c) = \frac{\sqrt{|\kappa|}c}{\tanh(\sqrt{|\kappa|}c)}$.

Proof. See Lemma 6 in Reference (Zhang & Sra, 2016). □

6.2.2 Lemma 6.2

Lemma 6.2. *Assume that $\{m_t\}$ and $\{\hat{v}_t\}$ are sequences obtained from Algorithm 2. Let $\hat{V}_t = \mathrm{diag}(\hat{v}_t^{(1)},\cdots,\hat{v}_t^{(d)})$. For all $t \in [1,T]$, $\beta_{1t} \leq \beta_1 \leq 1$, $\alpha_t = \alpha/t^b, b \in (0,1)$ and $\gamma := \beta_1/\sqrt{\beta_2} < 1$, the following inequality can be obtained:*

$$\sum_{t=1}^{T}\alpha_t\|\hat{V}_t^{-\frac{1}{4}}m_t\|^2 \leq \frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)\sqrt{1-\beta_2}(1-b)(1-\gamma)}. \tag{20}$$

Proof. First, we claim that $g_{1:T}^{d(i)}$ represents the vector formed by the i-th component of g^d from Step 1 to Step T, i.e., $g_{1:T}^{d(i)} = (g_1^{d(i)}, g_2^{d(i)},\cdots,g_T^{d(i)})^T$. Separate the last term on the left side of (20) and

find an upper bound for it:

$$
\begin{aligned}
&\alpha_T \|\hat{V}_T^{-\frac{1}{4}} m_T\|^2 = \alpha_T \sum_{i=1}^{d} \frac{(m_T^{(i)})^2}{\sqrt{\hat{v}_T^{(i)}}} \leq \alpha_T \sum_{i=1}^{d} \frac{(m_T^{(i)})^2}{\sqrt{v_T^{(i)}}} \\
=&\frac{\alpha}{T^b} \sum_{i=1}^{d} \frac{\left(\sum_{j=1}^{T}(1-\beta_{1j})(\prod_{k=1}^{T-j}\beta_{1,T-k+1}) \cdot g_j^{d(i)}\right)^2}{\sqrt{(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}(g_j^{d(i)})^2}} \\
\leq&\frac{\alpha}{T^b} \sum_{i=1}^{d} \frac{\left(\sum_{j=1}^{T}(1-\beta_{1j})\prod_{k=1}^{T-j}\beta_{1,T-k+1}\right) \cdot \left(\sum_{j=1}^{T}(1-\beta_{1j})\prod_{k=1}^{T-j}\beta_{1,T-k+1}(g_j^{d(i)})^2\right)}{\sqrt{(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}(g_j^{d(i)})^2}} \\
\leq&\frac{\alpha}{T^b} \sum_{i=1}^{d} \frac{\left(\sum_{j=1}^{T}\beta_1^{T-j}\right)\left(\sum_{j=1}^{T}\beta_1^{T-j}(g_j^{d(i)})^2\right)}{\sqrt{(1-\beta_2)\sum_{j=1}^{T}\beta_2^{T-j}(g_j^{d(i)})^2}} \\
\leq&\frac{\alpha}{T^b(1-\beta_1)\sqrt{1-\beta_2}} \sum_{i=1}^{d}\sum_{j=1}^{T} \frac{\beta_1^{T-j}(g_j^{d(i)})^2}{\sqrt{\beta_2^{T-j}(g_j^{d(i)})^2}} \\
=&\frac{\alpha}{T^b(1-\beta_1)\sqrt{1-\beta_2}} \sum_{i=1}^{d}\sum_{j=1}^{T} \gamma^{T-j}\left|g_j^{d(i)}\right|,
\end{aligned}
$$

where the first inequality follows from $v_T^{(i)} \leq \hat{v}_T^{(i)}$, the second equality follows from the recursive definitions of m_t and v_t, the second inequality follows from the Cauchy-Schwarz inequality, the third inequality follows from the decay of β_{1j} with respect to j, and the fourth inequality follows from $\sum_{j=1}^{T}\beta_1^{T-j} = \frac{1-\beta_1^T}{1-\beta_1} \leq \frac{1}{1-\beta_1}$. Similarly, we can control the first $T-1$ terms, so we have

$$
\begin{aligned}
\sum_{t=1}^{T}\alpha_t\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 \leq&\frac{\alpha}{(1-\beta_1)\sqrt{1-\beta_2}} \sum_{t=1}^{T}\frac{1}{t^b}\sum_{i=1}^{d}\sum_{j=1}^{t}\gamma^{t-j}\left|g_j^{d(i)}\right| \\
=&\frac{\alpha}{(1-\beta_1)\sqrt{1-\beta_2}} \sum_{i=1}^{d}\sum_{t=1}^{T}\left|g_t^{d(i)}\right|\sum_{j=t}^{T}\frac{\gamma^{j-t}}{j^b} \\
\leq&\frac{\alpha}{(1-\beta_1)\sqrt{1-\beta_2}} \sum_{i=1}^{d}\sum_{t=1}^{T}\left|g_t^{d(i)}\right|\frac{1}{t^b}\sum_{j=t}^{T}\gamma^{j-t} \\
\leq&\frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)\sqrt{1-\beta_2}(1-b)(1-\gamma)}
\end{aligned}
$$

where the first equation follows from rearranging the summation, the third inequality follows from $\sum_{t=1}^{T}|g_t^{d(i)}|\frac{1}{t^b} \leq G_\infty \sum_{t=1}^{T}\frac{1}{t^b} \leq \frac{1}{1-b}G_\infty T^{1-b}$ which is based on the Assumption 3.4 and $\sum_{j=t}^{T}\gamma^{j-t} \leq \frac{1}{1-\gamma}$. It can be seen that $\sum_{t=1}^{T}\alpha_t\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2$ can be controlled by $O(T^{1-b})$. □

6.2.3 Proof of Theorem 3.5

Proof. From F is a geodesically convex function, we obtain

$$
F_t(x) - F_t(y) \leq \langle -\text{grad}F_t(x),\ \text{Exp}_x^{-1}(y)\rangle_x,\ \forall x, y \in \mathcal{M}. \tag{21}
$$

Let $y_{t+1} = -B_{x_t}(\alpha_t m_t/\sqrt{\hat{v}_t})$, $y_* = \text{Exp}_{x_t}^{-1}(x_*)$, where Exp^{-1} is the inverse mapping of the retraction. Obviously, 0_{x_t}, y_{t+1} and y_* are all in the same linear space $\text{T}_{x_t}\mathcal{M}$, so we get the

following equation

$$\begin{aligned}
&\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat (y_{t+1} - y_*)\|^2 \\
=&\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat (-\alpha_t B_{x_t} \hat{V}_t^{-\frac{1}{2}} m_t - y_*)\|^2 \\
=&\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat y_*\|^2 + \alpha_t^2 \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 - 2\langle \hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat y_*, -\alpha_t \hat{V}_t^{-\frac{1}{4}} m_t\rangle \\
=&\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat \mathrm{Exp}_{x_t}^{-1}(x_*)\|^2 + \alpha_t^2 \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 - 2\langle B_{x_t}^\flat \mathrm{Exp}_{x_t}^{-1}(x_*), -\alpha_t m_t\rangle \\
=&\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat \mathrm{Exp}_{x_t}^{-1}(x_*)\|^2 + \alpha_t^2 \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 - 2\langle \mathrm{Exp}_{x_t}^{-1}(x_*), -\alpha_t B_{x_t} m_t\rangle_{x_t},
\end{aligned}$$

where $\hat{V}_t = \mathrm{diag}(\hat{v}_t^{(1)}, \cdots, \hat{v}_t^{(d)})$, and the last equation is obtained from equation 5. Substituting $m_t = \beta_{1t} m_{t-1} + (1-\beta_{1t}) B_{x_t}^\flat g_t^w$ into the last term and rearranging it gives

$$\begin{aligned}
\langle -g_t^w, \mathrm{Exp}_{x_t}^{-1}(x_*)\rangle_{x_t} = &\frac{1}{2\alpha_t(1-\beta_{1t})} \left(\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat \mathrm{Exp}_{x_t}^{-1}(x_*)\|^2 - \|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat (y_{t+1} - y_*)\|^2 \right) \\
&+ \frac{1}{2(1-\beta_{1t})} \alpha_t \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 + \frac{\beta_{1t}}{1-\beta_{1t}} \langle B_{x_t} m_{t-1}, \mathrm{Exp}_{x_t}^{-1}(x_*)\rangle_{x_t}.
\end{aligned} \tag{22}$$

Note that $\|B_{x_t}^\flat (y_{t+1} - y_*)\| \neq \|B_{x_{t+1}}^\flat \mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|$, so we cannot directly substitute one for the other. We need to find the relationship between these two terms. Consider the triangle in the tangent space $\mathrm{T}_{x_t}\mathcal{M}$ with vertices at $x_t,\ y_{t+1}$ and y_*. Using the law of cosines in Euclidean space, we can obtain the following equation

$$d(y_{t+1}, y_*)^2 = d(x_t, y_{t+1})^2 + d(x_t, y_*)^2 - 2d(x_t, y_{t+1}) d(x_t, y_*) \cos(\angle y_{t+1} x_t y_*). \tag{23}$$

Similarly, consider the geodesic triangle on the manifold $\mathcal{M}$ with vertices at $x_t,\ x_{t+1}$ and x_*. By using Assumption 3.1 and Lemma 6.1, we have

$$d(x_{t+1}, x_*)^2 \leq \zeta(\kappa, d(x_t, x_*)) d(x_{t+1}, x_t)^2 + d(x_t, x_*)^2 - 2d(x_t, x_{t+1}) d(x_t, x_*) \cos(\angle x_{t+1} x_t x_*), \tag{24}$$

where $\zeta(\kappa, c) := \frac{c\sqrt{|\kappa|}}{\tanh(c\sqrt{|\kappa|})}$. Combining (23) and (24) we get

$$d(x_{t+1}, x_*)^2 \leq d(y_{t+1}, y_*)^2 + (\zeta(\kappa, d(x_t, x_*)) - 1) d(x_t, y_{t+1})^2. \tag{25}$$

Here we use $d(x_t, y_{t+1}) = d(x_t, x_{t+1})$, $d(x_t, y_*) = d(x_t, x_*)$ and $\cos(\angle y_{t+1} x_t y_*) = \cos(\angle x_{t+1} x_t x_*)$. In fact, (25) can also be rewritten as

$$\|B_{x_{t+1}}^\flat \mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|^2 \leq \|B_{x_t}^\flat (y_{t+1} - y_*)\|^2 + \left(\zeta\left(\kappa, D_\infty\right) - 1\right) \alpha_t^2 \|\hat{V}_t^{-\frac{1}{2}} m_t\|^2. \tag{26}$$

The last term on the right side of the above inequality is because the function $\zeta(\kappa, c)$ is monotonically increasing with respect to the c, and according to Assumption 3.3, we know that $d(x_t, x_*) \leq D_\infty$. Then, from $y_{t+1} = x_t - B_{x_t}(\alpha_t m_t / \sqrt{\hat{v}_t})$, we can deduce

$$d(x_t, y_{t+1})^2 = \|y_{t+1} - x_t\|_{x_t}^2 = \|B_{x_t}(\alpha_t m_t / \sqrt{\hat{v}_t})\|_{x_t}^2 = \|\alpha_t m_t / \sqrt{\hat{v}_t}\|^2 = \alpha_t^2 \|\hat{V}_t^{-\frac{1}{2}} m_t\|^2.$$

Using the compatibility of norms for the formula (26) we have

$$\begin{aligned}
&\|B_{x_{t+1}}^\flat \mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|^2 \\
\leq &\|\hat{V}_t^{-\frac{1}{4}}\|^2 \|\hat{V}_t^{\frac{1}{4}} B_{x_t}^\flat (y_{t+1} - y_*)\|^2 + (\zeta(\kappa, D_\infty) - 1) \alpha_t^2 \|\hat{V}_t^{-\frac{1}{4}}\|^2 \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2.
\end{aligned} \tag{27}$$

Based on (22) and (27), we can get the following results:

$$
\begin{aligned}
&\langle -g_t^w, \mathrm{Exp}_{x_t}^{-1}(x_*)\rangle_{x_t} \\
&\le \frac{1}{2\alpha_t(1-\beta_{1t})}\left(\|\hat{V}_t^{\frac{1}{4}} B_{x_t}^{\flat}\mathrm{Exp}_{x_t}^{-1}(x_*)\|^2 - \frac{\|B_{x_{t+1}}^{\flat}\mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|^2}{\|\hat{V}_t^{-\frac{1}{4}}\|^2} + (\zeta(\kappa, D_\infty)-1)\alpha_t^2\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2\right) \\
&\quad + \frac{1}{2(1-\beta_{1t})}\alpha_t\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 + \frac{\beta_{1t}}{1-\beta_{1t}}\langle B_{x_t} m_{t-1}, \mathrm{Exp}_{x_t}^{-1}(x_*)\rangle_{x_t} \\
&\le \frac{1}{2\alpha_t(1-\beta_{1t})}\left(\|\hat{V}_t^{\frac{1}{4}}\|^2\cdot\|B_{x_t}^{\flat}\mathrm{Exp}_{x_t}^{-1}(x_*)\|^2 - \frac{\|B_{x_{t+1}}^{\flat}\mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|^2}{\|\hat{V}_t^{-\frac{1}{4}}\|^2}\right) + \frac{\zeta(\kappa, D_\infty)\alpha_t}{2(1-\beta_{1t})}\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 \\
&\quad + \frac{\beta_{1t}}{1-\beta_{1t}}\langle m_{t-1}, B_{x_t}^{\flat}\mathrm{Exp}_{x_t}^{-1}(x_*)\rangle \\
&\le \frac{1}{2\alpha_t(1-\beta_1)}\left(\max_i\left\{\sqrt{\hat{v}_t^{(i)}}\right\}\|\mathrm{Exp}_{x_t}^{-1}(x_*)\|_{x_t}^2 - \min_i\left\{\sqrt{\hat{v}_t^{(i)}}\right\}\|\mathrm{Exp}_{x_{t+1}}^{-1}(x_*)\|_{x_{t+1}}^2\right) \qquad (28) \\
&\quad + \frac{\zeta(\kappa, D_\infty)\alpha_t}{2(1-\beta_1)}\|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 + \frac{\alpha_{t-1}}{2(1-\beta_1)}\|\hat{V}_{t-1}^{-\frac{1}{4}} m_{t-1}\|^2 \qquad (29) \\
&\quad + \frac{\beta_{1t}}{2\alpha_t(1-\beta_1)}\|\hat{V}_t^{\frac{1}{4}}\|^2\cdot\|B_{x_t}^{\flat}\mathrm{Exp}_{x_t}^{-1}(x_*)\|^2. \qquad (30)
\end{aligned}
$$

For the last inequality, we used the Cauchy-Schwarz inequality and the Young inequality. Let (28), (29), (30) be $A_t,\ B_t,\ C_t$ respectively. According to Assumption 3.2, we know that the function F_t is a retraction convex function. Combining this with (21), we obtain

$$
\sum_{t=1}^{T} F_t(x_t) - F_t(x_*) \le \sum_{t=1}^{T}\langle -g_t^w,\ \mathrm{Exp}_{x_t}^{-1}(x_*)\rangle_{x_t} \le \sum_{t=1}^{T} A_t + B_t + C_t. \qquad (31)
$$

Next, we provide upper bounds for $A_t,\ B_t$ and C_t respectively. First, we consider A_t part:

$$
\begin{aligned}
\sum_{t=1}^{T} A_t &\le \frac{1}{2(1-\beta_1)}\frac{1}{\alpha_1}\max_i\left\{\sqrt{\hat{v}_1^{(i)}}\right\}\|\mathrm{Exp}_{x_1}^{-1}(x_*)\|_{x_1}^2 \\
&\quad + \frac{1}{2(1-\beta_1)}\sum_{t=2}^{T}\left(\frac{\max_i\left\{\sqrt{\hat{v}_t^{(i)}}\right\}}{\alpha_t} - \frac{\min_i\left\{\sqrt{\hat{v}_{t-1}^{(i)}}\right\}}{\alpha_{t-1}}\right)\|\mathrm{Exp}_{x_t}^{-1}(x_*)\|_{x_t}^2 \\
&\le \frac{D_\infty^2}{2(1-\beta_1)}\left(\sum_{t=1}^{T-1}\frac{\max_i\left\{\sqrt{\hat{v}_t^{(i)}}\right\} - \min_i\left\{\sqrt{\hat{v}_t^{(i)}}\right\}}{\alpha_t} + \frac{\max_i\left\{\sqrt{\hat{v}_T^{(i)}}\right\}}{\alpha_T}\right) \\
&\le \frac{D_\infty^2}{2(1-\beta_1)}\left(\sum_{t=1}^{T-1}\frac{Ct^b}{\alpha t^a} + \frac{T^b\max_i\left\{\sqrt{\hat{v}_T^{(i)}}\right\}}{\alpha}\right) \\
&\le \frac{D_\infty^2}{2(1-\beta_1)\alpha}\left(C\frac{T^{1-(a-b)}}{1-(a-b)} + T^b\max_i\left\{\sqrt{\hat{v}_T^{(i)}}\right\}\right). \qquad (32)
\end{aligned}
$$

The second inequality is because $\|\mathrm{Exp}_{x_t}^{-1}(x_*)\| \le D_\infty$. The third inequality follows from $\sqrt{\hat{v}}_{t,max} - \sqrt{\hat{v}}_{t,min} \le C/t^a,\ a \in (0,1)$. The last inequality is based on $\sum_{t=1}^{T-1} t^{b-a} \le \int_0^T t^{b-a}\mathrm{d}t = T^{1-(a-b)}/(1-(a-b))$. Then, handle the B_t term. Using Lemma 6.2, we can easily bound B_t for all $t \in [1, T]$:

$$\begin{aligned}\sum_{t=1}^{T} B_t &\leq \sum_{t=1}^{T} \frac{\zeta(\kappa, D_\infty)\alpha_t}{2(1-\beta_1)} \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 + \frac{\alpha_t}{2(1-\beta_1)} \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 \\ &= \frac{\zeta(\kappa, D_\infty)+1}{2(1-\beta_1)} \sum_{t=1}^{T} \alpha_t \|\hat{V}_t^{-\frac{1}{4}} m_t\|^2 \\ &\leq \frac{\zeta(\kappa, D_\infty)+1}{2(1-\beta_1)} \frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)\sqrt{1-\beta_2}(1-b)(1-\gamma)} \\ &= \frac{\zeta(\kappa, D_\infty)+1}{2} \frac{\alpha d G_\infty T^{1-b}}{(1-\beta_1)^2\sqrt{1-\beta_2}(1-b)(1-\gamma)} \end{aligned} \tag{33}$$

Finally, handle the C_t term:

$$\sum_{t=1}^{T} C_t \leq \frac{D_\infty^2}{2(1-\beta_1)} \sum_{t=1}^{T} \beta_{1t} \frac{\|\hat{V}_t^{\frac{1}{4}}\|^2}{\alpha_t} \leq \frac{D_\infty^2}{2(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t}\sqrt{\hat{v}_t^{(i)}}}{\alpha_t}. \tag{34}$$

Combining (31),(32),(33) and (34), we obtain the result of Theorem 3.5. □

6.2.4 Proof of Corollary 3.6

Proof. Combining Assumption 3.4 with $v_t^{(i)} = (1-\beta_2)\sum_{j=1}^{T} \beta_2^{T-j} (g_j^{d(i)})^2$, we obtain

$$v_t^{(i)} \leq (1-\beta_2) \sum_{j=1}^{T} \beta_2^{T-j} G_\infty^2 \leq (1-\beta_2) G_\infty^2 \frac{1}{(1-\beta_2)} = G_\infty^2. \tag{35}$$

Since $\hat{v}_t^{(i)} = \max(\hat{v}_{t-1}^{(i)}, v_t^{(i)})$, we have $\hat{v}_t^{(i)} \leq G_\infty^2$, $\forall t \in \mathbb{Z}_+$. Combining (35) with $\beta_{1t} = \beta_1 \lambda^{t-1}$ and $\alpha_t = \frac{\alpha}{t^b}$, we obtain the following result:

$$\frac{D_\infty^2}{2(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t}\sqrt{\hat{v}_t^{(i)}}}{\alpha_t} \leq \frac{d D_\infty^2 G_\infty}{2(1-\beta_1)} \sum_{t=1}^{T} \frac{\beta_{1t}}{\alpha_t} = \frac{d\beta_1 D_\infty^2 G_\infty}{2\alpha(1-\beta_1)} \sum_{t=1}^{T} \lambda^{t-1} t^b. \tag{36}$$

Let $S_T = \sum_{t=1}^{T} \lambda^{t-1} t^b$, then

$$S_T - \lambda S_T = 1 + \sum_{t=1}^{T-1} \lambda^t ((t+1)^b - t^b) - \lambda^T T^b \leq \sum_{t=0}^{T-1} \lambda^t = \frac{1-\lambda^T}{1-\lambda} \leq \frac{1}{1-\lambda}. \tag{37}$$

Furthermore, we can derive

$$S_T \leq \frac{1}{(1-\lambda)^2}. \tag{38}$$

Substituting (38) into (36) proves this corollary. □

6.3 Additional Numerical Experiments

6.3.1 Karcher mean and Joint Diagonalization

Testing Problem and Dataset of Karcher mean

The Karcher mean has a wide range of applications in various fields, including image processing (Rathi et al., 2007), medical imaging (Fletcher & Joshi, 2007), and solid mechanics (Moakher, 2006). The core of this problem is to find a point on a Riemannian manifold that minimizes the sum of squared Riemannian distances to all points in the sample. Specifically, given a set of $n \times n$ SPD matrices $A_1, A_2, \cdots, A_k$, we aim to solve the following optimization problem:

$$\min_{X \in \mathcal{S}_{++}^n} F(X) = \frac{1}{2k} \sum_{i=1}^{k} \delta^2(A_i, X), \tag{39}$$

where $\delta(X, Y) = \| \log(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}) \|_F$ is the geodesic distance under metric (13).

In this experiment, the dataset consists of 1000 randomly generated 10×10 SPD matrices. Specifically, we generate a 10×10 with random entries uniformly distributed between 0 and 1, and then compute $A_i = R_i^T R_i$ to obtain the SPD matrices. The initial point X_0 is a 10×10 identity matrix.

Testing Problem and Dataset of Joint Diagonalization
The joint diagonalization (JD) on the Stiefel manifold is a key preprocessing step in independent component analysis. Specifically, this process aims to optimize the following problem:

$$\min_{X \in \mathrm{St}(n,p)} F(X) = -\frac{1}{N} \sum_{i=1}^{N} \|\mathrm{diag}(X^T C_i X)\|_F^2, \tag{40}$$

where $\mathrm{diag}(A)$ denotes the diagonal elements of matrix A forming a diagonal matrix, $\|\mathrm{diag}(A)\|_F^2$ represents the sum of the squares of the diagonal elements of matrix A, and $C_{i\,i=1}^{N}$ are given symmetric matrices.

In this experiment, n is set to 40, p to 30, and N to 1000. The set $C_{i\,i=1}^{N}$ consists of synthetic datasets, specifically constructed as $C_i = \mathrm{diag}(1, 2, \cdots, n) + R_i + R_i^T$ (Huang et al., 2015), where R_i is a 40×40 matrix with entries drawn from a standard normal distribution. It is obvious that C_i is a symmetric matrix. The initial point X_0 is a randomly generated 40×30 column-orthogonal matrix.

Testing Environment
The experiments in this section were performed on an Apple MacBook Air with an Apple M3 chip, 24 GB unified memory, and 512 GB SSD storage. The code was first compiled using Xcode and then visualized the results with MATLAB R2025b.

Parameter Settings
In all experiments, we adopted the following stopping criterion

$$\left| F_t - \max_{t-p \le i \le t} \{F_i\} \right| \le tol \cdot |F_0|,$$

Here, F_t refers to the value of the objective function at the t-th epoch. The stopping criterion means that if the objective function value does not decrease beyond a certain threshold over p epochs, the algorithm stops. We use mini-batch with fixed $batchsize$ to balance memory efficiency and gradient stability. Samples are randomly shuffled per epoch. Then split into batches of size $batchsize$ via index slicing.

In the first two applications, we set $batchsize = 10$, $tol = 10^{-8}$, $p = 5$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\lambda = 0.99$ and $b = 0.8$. Additionally, in the RAMSGrad algorithm, the bound $\phi(t)$ for $\sqrt{\hat{v}_t}$ is set to $1/t^a$, where $a = 0.99$.

Results and Analysis of Karcher Mean
We compare the performance of the RSGD, RASA, RAdam, and RAMSGrad on the Karcher mean problem. For each algorithm's stepsize, we tried different values from the set $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3\}$. The experimental results are shown in Figure 3, and the objective function values at the stopping point of the algorithms are listed in Table 5.

In Figure 3, we present the results of each algorithm at their lowest objective function values. For RSGD, the best performance is achieved when the step size is set to 0.01. To provide a comprehensive perspective, we also show the results for step sizes slightly lower and slightly higher than 0.01. Combining this with the data from Table 5, we can see that RASA achieves the lowest objective function value, RAMSGrad achieves the second lowest, and RSGD the third. Moreover, they approach the optimal point quickly along with RAdam.

Results and Analysis of Joint Diagonalization
For the stepsizes of each algorithm, we tried different values from the set $\{10^{-6}, 3\times10^{-6}, 10^{-5}, 3\times10^{-5}, \cdots, 10^{-1}, 3\times10^{-1}\}$. The experimental results are shown in Figure 4, and the objective function values at the stopping point of the algorithms are listed in Table 6.

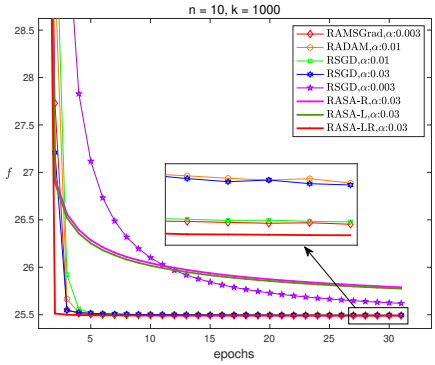


Figure 3: For Karcher mean problem, the objective function value of each algorithm varies with the epoch.

Table 5: For Karcher mean problem, the maximum epoch is set to 30. The objective function values at the stopping point for each algorithm.

Algorithms	RAMSGrad	RAdam	RSGD	RSGD
α	0.003	0.01	0.01	0.03
F	25.4907	25.4942	25.4909	25.4941
Algorithms	RSGD	RASA-R	RASA-L	RASA-LR
α	0.003	0.03	0.03	0.03
F	25.6194	25.7888	25.7748	25.4897

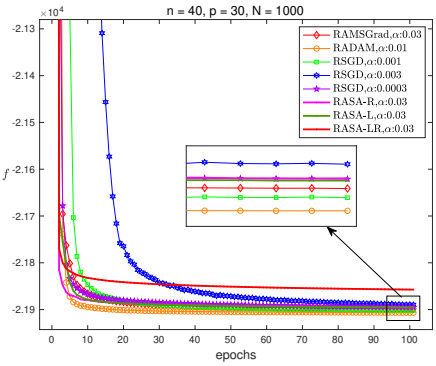


Figure 4: For JD problem, the objective function value of each algorithm varies with the epoch.

Table 6: For JD problem, the maximum epoch is set to 100. The objective function values at the stopping point for each algorithm.

Algorithms	RAMSGrad	RAdam	RSGD	RSGD
α	0.03	0.01	0.001	0.003
F	-2.1899×10^4	-2.1907×10^4	-2.1902×10^4	-2.1890×10^4
Algorithms	RSGD	RASA-R	RASA-L	RASA-LR
α	0.0003	0.03	0.03	0.03
F	-2.1895×10^4	-2.1895×10^4	-2.1896×10^4	-2.1857×10^4

Combining Figure 4 with Table 6, we can clearly see that the RAdam achieves the lowest objective function value, followed by the RSGD and RAMSGrad. The RASA, regardless of preconditioning variant, consistently has higher objective function values compared to RAdam and RAMSGrad.

6.3.2 POINCARÉ EMBEDDING

Metrics
We use the same two settings as (Nickel & Kiela, 2017), namely reconstruction and link prediction. The former evaluates the representation capability of the model, while the latter assesses the model's generalization ability. For reconstruction, we evaluate the algorithm's performance based on loss value, mean average precision (MAP) (Nickel & Kiela, 2017), and training time. For link prediction, we primarily rely on MAP. We use 2% of the dataset as the test set, and the objects in the test set do not contain root nodes and leaf nodes.

Testing Environment
The last experiment was conducted on a Linux system with an Intel(R) Xeon(R) Gold 6338 CPU (2.00GHz). We first compiled the C++ source code using gcc version 6.3.0 and then imported the dataset and called the compiled binary files through MATLAB R2022b.

Parameter Settings
We set the embedding dimension to 5 and initialized all embeddings randomly from a uniform distribution $\mathcal{U}(-0.0001, 0.0001)$. All methods employ the "burn-in" phase from (Nickel & Kiela, 2017). During this phase, the Alias algorithm (Li et al., 2014) is used for sampling. The learning rate is α/c, with c set to 100. This phase lasts for 20 epochs. The $batchsize$ is set to 10, and the number of negative samples $|\mathcal{N}(u)|$ is set to 4. The bound $\phi(t)$ in the RAMSGrad algorithm is $1/t^a$, where a is 0.1.