# On Riemannian Optimization over Positive Definite Matrices with the Bures-Wasserstein Geometry

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

In this paper, we comparatively analyze the Bures-Wasserstein (BW) geometry with the popular Affine-Invariant (AI) geometry for Riemannian optimization on the symmetric positive definite (SPD) matrix manifold. Our study begins with an observation that the BW metric has a linear dependence on SPD matrices in contrast to the quadratic dependence of the AI metric. We build on this to show that the BW metric is a more suitable and robust choice for several Riemannian optimization problems over ill-conditioned SPD matrices. We show that the BW geometry has a non-negative curvature, which further improves convergence rates of algorithms over the non-positively curved AI geometry. Finally, we verify that several popular cost functions, which are known to be geodesic convex under the AI geometry, are also geodesic convex under the BW geometry. Extensive experiments on various applications support our findings.

## 1 Introduction

Learning on symmetric positive definite (SPD) matrices is a fundamental problem in various machine learning applications, including metric and kernel learning [56, 22, 53], medical imaging [44, 43], computer vision [24, 31], domain adaptation [38], modeling time-varying data [16], and object detection [57], among others. Recent studies have also explored SPD matrix learning as a building block in deep neural networks [30, 21].

The set of SPD matrices of size $n \times n$, defined as $\mathbb{S}^n_{++} := \{\mathbf{X} : \mathbf{X} \in \mathbb{R}^{n\times n}, \mathbf{X}^\top = \mathbf{X}, \text{ and } \mathbf{X} \succ \mathbf{0}\}$, has a smooth manifold structure with a richer geometry than the Euclidean space. When endowed with a metric (inner product structure), the set of SPD matrices becomes a Riemannian manifold [10]. Hence, numerous existing works [44, 7, 33, 32, 30, 37, 21, 43] have studied and employed the Riemannian optimization framework for learning over the space of SPD matrices [2, 14].

Several Riemannian metrics on $\mathbb{S}^n_{++}$ have been proposed such as the Affine-Invariant [44, 10], the Log-Euclidean [7, 45], the Log-Det [51, 18], the Log-Cholesky [37], to name a few. One can additionally obtain different families of Riemannian metrics on $\mathbb{S}^n_{++}$ by appropriate parameterizations based on the principles of invariance and symmetry [20, 18, 54, 43]. However, to the best of our knowledge, a systematic study comparing the different metrics for optimizing generic cost functions defined on $\mathbb{S}^n_{++}$ is missing. Practically, the Affine-Invariant (AI) metric seems to be the most widely used metric in Riemannian first and second order algorithms (e.g., steepest descent, conjugate gradients, trust regions) as it is the only Riemannian SPD metric available in several manifold optimization toolboxes, such as Manopt [15], Manopt.jl [9], Pymanopt [55], ROPTLIB [28], and McTorch [40]. Moreover, many interesting problems in machine learning are found to be geodesic convex (generalization of Euclidean convexity) under the AI metric, which allows fast convergence of optimization algorithms [63, 27].

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

Recent works have studied the Bures-Wasserstein (BW) distance on SPD matrices [39, 11, 59]. It is a well-known result that the Wasserstein distance between two multivariate Gaussian densities is a function of the BW distance between their covariance matrices. Indeed, the BW metric is a Riemannian metric. Under this metric, the necessary tools for Riemannian optimization, including the Riemannian gradient and Hessian expressions, can be efficiently computed [39]. Hence, it is a promising candidate for Riemannian optimization on $\mathbb{S}^n_{++}$. In this work, we theoretically and empirically analyze the quality of optimization with the BW geometry and show that it is a viable alternative to the default choice of AI geometry. Our analysis discusses the classes of cost functions (e.g., polynomial) for which the BW metric has better convergence rates than the AI metric. We also discuss cases (e.g., log-det) where the reverse is true. In particular, our contributions are as follows.

- We observe that the BW metric has a linear dependence on SPD matrices while the AI metric has a quadratic dependence. We show this impacts the condition number of the Riemannian Hessian and makes the BW metric more suited to learning ill-conditioned SPD matrices than the AI metric.
- In contrast to the non-positively curved AI geometry, the BW geometry is shown to be non-negatively curved, which leads to a tighter trigonometry distance bound and faster convergence rates for first order optimization algorithms.
- For both metrics, we analyze the convergence rates of Riemannian steepest descent and trust region methods and highlight the impacts arising from the differences in the curvature and condition number of the Riemannian Hessian.
- We verify that common problems that are geodesic convex under the AI metric are also geodesic convex under the BW metric.
- We support our analysis with extensive experiments on applications such as weighted least squares, trace regression, metric learning, and Gaussian mixture model.

## 2 Preliminaries

The fundamental ingredients for Riemannian optimization are Riemannian metric, exponential map, Riemannian gradient, and Riemannian Hessian. We refer readers to [2, 14] for a general treatment on Riemannian optimization.

A Riemannian metric is a smooth, bilinear, and symmetric positive definite function on the tangent space $T_x\mathcal{M}$ for any $x \in \mathcal{M}$. That is, $g : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$, which is often written as an inner product $\langle \cdot, \cdot \rangle_x$. The induced norm of a tangent vector $u \in T_x\mathcal{M}$ is given by $\|u\|_x = \sqrt{\langle u, u \rangle_x}$. A geodesic on a manifold $\gamma : [0, 1] \to \mathcal{M}$ is defined as a locally shortest curve with zero acceleration. For any $x \in \mathcal{M}, u \in T_x\mathcal{M}$, the exponential map, $\mathrm{Exp}_x : T_x\mathcal{M} \to \mathcal{M}$ is defined such that there exists a geodesic curve $\gamma$ with $\gamma(0) = x$, $\gamma(1) = \mathrm{Exp}_x(u)$ and $\gamma'(0) = u$.

**First-order geometry and Riemannian steepest descent.** Riemannian gradient of a differentiable function $f : \mathcal{M} \to \mathbb{R}$ at $x$, denoted as $\mathrm{grad} f(x)$, is a tangent vector that satisfies for any $u \in T_x\mathcal{M}$, $\langle \mathrm{grad} f(x), u \rangle_x = \mathrm{D}_u f(x)$, where $\mathrm{D}_u f(x)$ is the directional derivative of $f(x)$ along $u$. The Riemannian steepest descent method [58] generalizes the standard gradient descent in the Euclidean space to Riemannian manifolds by ensuring that the updates are along the geodesic and stay on the manifolds. That is, $x_{t+1} = \mathrm{Exp}_{x_t}(-\eta_t \, \mathrm{grad} f(x_t))$ for some step size $\eta_t$.

**Second-order geometry and Riemannian trust region.** Second-order methods such as trust region and cubic regularized Newton methods are generalized to Riemannian manifolds [1, 3]. They make use of the Riemannian Hessian, $\mathrm{Hess} f(x) : T_x\mathcal{M} \to T_x\mathcal{M}$, which is a linear operator that is defined as the covariant derivative of the Riemannian gradient. Both the trust region and cubic regularized Newton methods are Hessian-free in the sense that only evaluation of the Hessian acting on a tangent vector, i.e., $\mathrm{Hess} f(x)[u]$ is required. Similar to the Euclidean counterpart, the Riemannian trust region method approximates the Newton step by solving a subproblem, i.e.,

$$\min_{u \in T_{x_t}\mathcal{M} : \|u\|_{x_t} \leq \Delta} m_{x_t}(u) = f(x_t) + \langle \mathrm{grad} f(x_t), u \rangle_{x_t} + \frac{1}{2} \langle \mathcal{H}_{x_t}[u], u \rangle_{x_t},$$

where $\mathcal{H}_{x_t} : T_{x_t}\mathcal{M} \to T_{x_t}\mathcal{M}$ is a symmetric and linear operator that approximates the Riemannian Hessian. $\Delta$ is the radius of trust region, which may be increased or decreased depending on how

Table 1: Riemannian optimization ingredients for AI and BW geometries.

| | Affine-Invariant | Bures-Wasserstein |
|---|---|---|
| R.Metric | $g_{\mathrm{ai}}(\mathbf{U},\mathbf{V}) = \mathrm{tr}(\mathbf{X}^{-1}\mathbf{U}\mathbf{X}^{-1}\mathbf{V})$ | $g_{\mathrm{bw}}(\mathbf{U},\mathbf{V}) = \frac{1}{2}\mathrm{tr}(\mathcal{L}_{\mathbf{X}}[\mathbf{U}]\mathbf{V})$ |
| R.Exp | $\mathrm{Exp}_{\mathrm{ai},\mathbf{X}}(\mathbf{U}) = \mathbf{X}^{1/2}\exp(\mathbf{X}^{-1}\mathbf{U})\mathbf{X}^{1/2}$ | $\mathrm{Exp}_{\mathrm{bw},\mathbf{X}}(\mathbf{U}) = \mathbf{X} + \mathbf{U} + \mathcal{L}_{\mathbf{X}}[\mathbf{U}]\mathbf{X}\mathcal{L}_{\mathbf{X}}[\mathbf{U}]$ |
| R.Gradient | $\mathrm{grad}_{\mathrm{ai}} f(\mathbf{X}) = \mathbf{X}\nabla f(\mathbf{X})\mathbf{X}$ | $\mathrm{grad}_{\mathrm{bw}} f(\mathbf{X}) = 4\{\nabla f(\mathbf{X})\mathbf{X}\}_{\mathrm{S}}$ |
| R.Hessian | $\mathrm{Hess}_{\mathrm{ai}} f(\mathbf{X})[\mathbf{U}] = \mathbf{X}\nabla^2 f(\mathbf{X})[\mathbf{U}]\mathbf{X} + \{\mathbf{U}\nabla f(\mathbf{X})\mathbf{X}\}_{\mathrm{S}}$ | $\mathrm{Hess}_{\mathrm{bw}} f(\mathbf{X})[\mathbf{U}] = 4\{\nabla^2 f(\mathbf{X})[\mathbf{U}]\mathbf{X}\}_{\mathrm{S}} + 2\{\nabla f(\mathbf{X})\mathbf{U}\}_{\mathrm{S}} + 4\{\mathbf{X}\{\mathcal{L}_{\mathbf{X}}[\mathbf{U}]\nabla f(\mathbf{X})\}_{\mathrm{S}}\}_{\mathrm{S}} - \{\mathcal{L}_{\mathbf{X}}[\mathbf{U}]\mathrm{grad}_{\mathrm{bw}} f(\mathbf{X})\}_{\mathrm{S}}$ |

model value $m_{x_t}(u)$ changes. The subproblem is solved iteratively using a truncated conjugate gradient algorithm. The next iterate is given by $x_{t+1} = \mathrm{Exp}_{x_t}(u)$ with the optimized $u$.

Next, the eigenvalues and the condition number of the Riemannian Hessian are defined as follows, which we use for analysis in Section 3.

**Definition 1.** The minimum and maximum eigenvalues of $\mathrm{Hess} f(x)$ are defined as $\lambda_{\min} = \min_{\|u\|_x^2=1}\langle \mathrm{Hess} f(x)[u], u\rangle_x$ and $\lambda_{\max} = \max_{\|u\|_x^2=1}\langle \mathrm{Hess} f(x)[u], u\rangle_x$. The condition number of $\mathrm{Hess} f(x)$ is defined as $\kappa(\mathrm{Hess} f(x)) := \lambda_{\max}/\lambda_{\min}$.

**Function classes on Riemannian manifolds.** For analyzing algorithm convergence, we require the definitions for several important function classes on Riemannian manifolds, including geodesic convexity and smoothness. Similarly, we require the definition for geodesic convex sets that generalize (Euclidean) convex sets to manifolds [52, 61].

**Definition 2** (Geodesic convex set [52, 61])**.** A set $\mathcal{X} \subseteq \mathcal{M}$ is geodesic convex if for any $x, y \in \mathcal{X}$, the distance minimizing geodesic $\gamma$ joining the two points lies entirely in $\mathcal{X}$.

Indeed, this notion is well-defined for any manifold because a sufficiently small geodesic ball is always geodesic convex.

**Definition 3** (Geodesic convexity [52, 61])**.** Consider a geodesic convex set $\mathcal{X} \subseteq \mathcal{M}$. A function $f : \mathcal{X} \to \mathbb{R}$ is called geodesic convex if for any $x, y \in \mathcal{X}$, the distance minimizing geodesic $\gamma$ joining $x$ and $y$ satisfies $\forall\, t \in [0,1], f(\gamma(t)) \le (1-t)f(x) + tf(y)$. Function $f$ is strictly geodesic convex if the equality holds only when $t = 0, 1$.

**Definition 4** (Geodesic strong convexity and smoothness [52, 29])**.** Under the same settings in Definition 3. A twice-continuously differentiable function $f : \mathcal{X} \to \mathbb{R}$ is called geodesic $\mu$-strongly convex if for any distance minimizing geodesic $\gamma$ in $\mathcal{X}$ with $\|\gamma'(0)\| = 1$, it satisfies $\frac{d^2 f(\gamma(t))}{dt^2} \ge \mu$, for some $\mu > 0$. Function $f$ is called geodesic $L$-smooth if $\frac{d^2 f(\gamma(t))}{dt^2} \le L$, for some $L > 0$.

## 3 Comparing BW with AI for Riemannian Optimization

This section starts with an observation of a linear-versus-quadratic dependency between the two metrics. From this observation, we analyze the condition number of the Riemannian Hessian. Then, we further compare the sectional curvature of the two geometries. Together with the differences in the condition number, this allows us to compare the convergence rates of optimization algorithms on the two geometries. We conclude this section by showing geodesic convexity of several generic cost functions under the BW geometry.

**AI and BW geometries on SPD matrices.** When endowed with a Riemannian metric $g$, the set of SPD matrices of size $n$ becomes a Riemannian manifold $\mathcal{M} = (\mathbb{S}^n_{++}, g)$. The tangent space at $\mathbf{X}$ is $T_{\mathbf{X}}\mathcal{M} := \{\mathbf{U} : \mathbf{U} \in \mathbb{R}^{n\times n} \text{ and } \mathbf{U}^\top = \mathbf{U}\}$. Under the AI and BW metrics, the Riemannian exponential map, Riemannian gradient, and Hessian are compared in Table 1, where we denote $\{\mathbf{A}\}_{\mathrm{S}} := (\mathbf{A} + \mathbf{A}^\top)/2$ and $\exp(\mathbf{A})$ as the matrix exponential of $\mathbf{A}$. $\mathcal{L}_{\mathbf{X}}[\mathbf{U}]$ is the solution to the matrix linear system $\mathcal{L}_{\mathbf{X}}[\mathbf{U}]\mathbf{X} + \mathbf{X}\mathcal{L}_{\mathbf{X}}[\mathbf{U}] = \mathbf{U}$ and is known as the Lyapunov operator. We use $\nabla f(\mathbf{X})$ and $\nabla^2 f(\mathbf{X})$ to represent the first-order and second-order derivatives, i.e., the Euclidean gradient and Hessian, respectively. The derivations in Table 1 can be found in [43, 11]. In the rest of

the paper, we use $\mathcal{M}_{\rm ai}$ and $\mathcal{M}_{\rm bw}$ to denote the SPD manifolds under the two metrics. From Table 1, the computational costs for evaluating the AI and BW ingredients are dominated by the matrix exponential/inversion operations and the Lyapunov operator $\mathcal{L}$ computation, respectively. Both at most cost $\mathcal{O}(n^3)$, which implies a comparable per-iteration cost of optimization algorithms between the two metric choices. This claim is validated in Section 4.

**A key observation.** From Table 1, the Affine-Invariant metric on the SPD manifold can be rewritten as for any $\mathbf{U}, \mathbf{V} \in T_{\mathbf{X}}\mathcal{M}$,

$$\langle \mathbf{U}, \mathbf{V} \rangle_{\rm ai} = {\rm tr}(\mathbf{X}^{-1}\mathbf{U}\mathbf{X}^{-1}\mathbf{V}) = {\rm vec}(\mathbf{U})^\top (\mathbf{X} \otimes \mathbf{X})^{-1} {\rm vec}(\mathbf{V}), \tag{1}$$

where ${\rm vec}(\mathbf{U})$ and ${\rm vec}(\mathbf{V})$ are the vectorizations of $\mathbf{U}$ and $\mathbf{V}$, respectively. Note that we omit the subscript $\mathbf{X}$ for inner product $\langle \cdot, \cdot \rangle$ to simplify the notation. The specific tangent space where the inner product is computed should be clear from contexts.

The Bures-Wasserstein metric is rewritten as, for any $\mathbf{U}, \mathbf{V} \in T_{\mathbf{X}}\mathcal{M}$,

$$\langle \mathbf{U}, \mathbf{V} \rangle_{\rm bw} = \frac{1}{2}{\rm tr}(\mathcal{L}_{\mathbf{X}}[\mathbf{U}]\mathbf{V}) = \frac{1}{2}{\rm vec}(\mathbf{U})^\top (\mathbf{X} \oplus \mathbf{X})^{-1} {\rm vec}(\mathbf{V}), \tag{2}$$

where $\mathbf{X} \oplus \mathbf{X} = \mathbf{X} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{X}$ is the Kronecker sum.

**Remark 1.** Comparing Eq. (1) and (2) reveals that the BW metric has a linear dependence on $\mathbf{X}$ while the AI metric has a quadratic dependence. This suggests that optimization algorithms under the AI metric should be more sensitive to the condition number of $\mathbf{X}$ compared to the BW metric.

The above observation serves as a key motivation for the further analysis.

### 3.1 Condition number of Riemannian Hessian at optimality

Throughout the rest of the paper, we make the following assumptions.

**Assumption 1.** (a). $f$ is at least twice continuously differentiable with a non-degenerate local minimizer $\mathbf{X}^*$. (b). The subset $\mathcal{X} \subseteq \mathcal{M}$ (usually as a neighbourhood of a center point) we consider throughout this paper is totally normal, i.e., the exponential map is a diffeomorphism.

Assumption 1 is easy to satisfy. Particularly, Assumption 1(b) is guaranteed for the SPD manifold under the AI metric because its geodesic is unique. Under the BW metric, for a center point $\mathbf{X}$, we can choose the neighbourhood such that $\mathcal{X} = \{{\rm Exp}_{\mathbf{X}}(\mathbf{U}) : \mathbf{I} + \mathcal{L}_{\mathbf{X}}[\mathbf{U}] \in \mathbb{S}^n_{++}\}$ as in [39]. In other words, $\mathcal{X}$ is assumed to be unique-geodesic under both the metrics.

We now formalize the impact of the linear-versus-quadratic dependency, highlighted in Remark 1. At a local minimizer $\mathbf{X}^*$ where the Riemannian gradient vanishes, we first simplify the expression for the Riemannian Hessian in Table 1.

On $\mathcal{M}_{\rm ai}$, $\langle {\rm Hess}_{\rm ai} f(\mathbf{X}^*)[\mathbf{U}], \mathbf{U} \rangle_{\rm ai} = {\rm tr}(\nabla^2 f(\mathbf{X}^*)[\mathbf{U}]\mathbf{U}) = {\rm vec}(\mathbf{U})^\top \mathbf{H}(\mathbf{X}^*){\rm vec}(\mathbf{U})$, with $\mathbf{H}(\mathbf{X}) \in \mathbb{R}^{n^2 \times n^2}$ is the matrix representation of the Euclidean Hessian $\nabla^2 f(\mathbf{X})$ and $\mathbf{U} \in T_{\mathbf{X}^*}\mathcal{M}_{\rm ai}$. The maximum eigenvalue of ${\rm Hess}_{\rm ai} f(\mathbf{X}^*)$ is then given by $\lambda^*_{\rm max} = \max_{\|\mathbf{U}\|^2_{\rm ai}=1} {\rm vec}(\mathbf{U})^\top \mathbf{H}(\mathbf{X}^*){\rm vec}(\mathbf{U})$, where $\|\mathbf{U}\|^2_{\rm ai} = {\rm vec}(\mathbf{U})^\top (\mathbf{X}^* \otimes \mathbf{X}^*)^{-1}{\rm vec}(\mathbf{U})$. This is a generalized eigenvalue problem with the solution to be the maximum eigenvalue of $(\mathbf{X}^* \otimes \mathbf{X}^*)\mathbf{H}(\mathbf{X}^*)$. Similarly, $\lambda^*_{\rm min}$ corresponds to the minimum eigenvalue of $(\mathbf{X}^* \otimes \mathbf{X}^*)\mathbf{H}(\mathbf{X}^*)$.

On $\mathcal{M}_{\rm bw}$, $\langle {\rm Hess}_{\rm bw} f(\mathbf{X}^*)[\mathbf{U}], \mathbf{U} \rangle_{\rm bw} = {\rm vec}(\mathbf{U})^\top \mathbf{H}(\mathbf{X}^*){\rm vec}(\mathbf{U})$ and the norm is $\|\mathbf{U}\|^2_{\rm bw} = {\rm vec}(\mathbf{U})^\top (\mathbf{X}^* \oplus \mathbf{X}^*)^{-1}{\rm vec}(\mathbf{U})$. Hence, the minimum/maximum eigenvalue of ${\rm Hess}_{\rm bw} f(\mathbf{X}^*)$ equals the minimum/maximum eigenvalue of $(\mathbf{X}^* \oplus \mathbf{X}^*)\mathbf{H}(\mathbf{X}^*)$.

Let $\kappa^*_{\rm ai} := \kappa({\rm Hess}_{\rm ai} f(\mathbf{X}^*)) = \kappa((\mathbf{X}^* \otimes \mathbf{X}^*)\mathbf{H}(\mathbf{X}^*))$ and $\kappa^*_{\rm bw} := \kappa({\rm Hess}_{\rm bw} f(\mathbf{X}^*)) = \kappa((\mathbf{X}^* \oplus \mathbf{X}^*)\mathbf{H}(\mathbf{X}^*))$. The following lemma bounds these two condition numbers.

**Lemma 1.** *For a local minimizer $\mathbf{X}^*$ of $f(\mathbf{X})$, the condition number of ${\rm Hess} f(\mathbf{X}^*)$ satisfies*

$$\begin{aligned} \kappa(\mathbf{X}^*)^2/\kappa(\mathbf{H}(\mathbf{X}^*)) \leq & \kappa^*_{\rm ai} \leq \kappa(\mathbf{X}^*)^2\kappa(\mathbf{H}(\mathbf{X}^*)) \\ \kappa(\mathbf{X}^*)/\kappa(\mathbf{H}(\mathbf{X}^*)) \leq & \kappa^*_{\rm bw} \leq \kappa(\mathbf{X}^*)\kappa(\mathbf{H}(\mathbf{X}^*)). \end{aligned}$$

It is clear that $\kappa^*_{\rm bw} \leq \kappa^*_{\rm ai}$ when $\kappa(\mathbf{H}(\mathbf{X}^*)) \leq \sqrt{\kappa(\mathbf{X}^*)}$. This is true for linear, quadratic, higher-order polynomial functions and in general holds for several machine learning optimization problems on the SPD matrices (discussed in Section 4).

**Case 1** (Condition number for linear and quadratic optimization)**.** For a linear function $f(\mathbf{X}) = \mathrm{tr}(\mathbf{XA})$, its Euclidean Hessian matrix is $\mathbf{H}(\mathbf{X}) = \mathbf{0}$. For a quadratic function $f(\mathbf{X}) = \mathrm{tr}(\mathbf{XAXB})$ with $\mathbf{A}, \mathbf{B} \in \mathbb{S}^n_{++}$, $\mathbf{H}(\mathbf{X}) = \mathbf{A} \otimes \mathbf{B} + \mathbf{B} \otimes \mathbf{A}$. Therefore, $\kappa(\mathbf{H}(\mathbf{X}^*))$ is a constant and for ill-conditioned $\mathbf{X}^*$, we have $\kappa(\mathbf{H}(\mathbf{X}^*)) \leq \sqrt{\kappa(\mathbf{X}^*)}$, which leads to $\kappa^*_{\mathrm{ai}} \geq \kappa^*_{\mathrm{bw}}$.

**Case 2** (Condition number for higher-order polynomial optimization)**.** For an integer $\alpha \geq 3$, consider a function $f(\mathbf{X}) = \mathrm{tr}(\mathbf{X}^\alpha)$ with derived $\mathbf{H}(\mathbf{X}) = \alpha \sum_{l=0}^{\alpha-2} (\mathbf{X}^l \otimes \mathbf{X}^{\alpha-l-2})$. We get $\kappa^*_{\mathrm{ai}} = \alpha \sum_{l=1}^{\alpha-1} ((\mathbf{X}^*)^l \otimes (\mathbf{X}^*)^{\alpha-l})$ and $\kappa^*_{\mathrm{bw}} = \alpha(\mathbf{X}^* \oplus \mathbf{X}^*)(\sum_{l=0}^{\alpha-2} (\mathbf{X}^l \otimes \mathbf{X}^{\alpha-l-2}))$. It is apparent that $\kappa^*_{\mathrm{ai}} = \mathcal{O}(\kappa(\mathbf{X}^*)^\alpha)$ while $\kappa^*_{\mathrm{bw}} = \mathcal{O}(\kappa(\mathbf{X}^*)^{\alpha-1})$. Hence, for ill-conditioned $\mathbf{X}^*$, $\kappa^*_{\mathrm{ai}} \geq \kappa^*_{\mathrm{bw}}$.

One counter-example where $\kappa^*_{\mathrm{bw}} \geq \kappa^*_{\mathrm{ai}}$ is the log-det function.

**Case 3** (Condition number for log-det optimization)**.** For the log-det function $f(\mathbf{X}) = -\log\det(\mathbf{X})$, its Euclidean Hessian is $\nabla^2 f(\mathbf{X})[\mathbf{U}] = \mathbf{X}^{-1}\mathbf{U}\mathbf{X}^{-1}$ and $\mathbf{H}(\mathbf{X}) = \mathbf{X}^{-1} \otimes \mathbf{X}^{-1}$. At a local minimizer $\mathbf{X}^*$, $\mathrm{Hess}_{\mathrm{ai}} f(\mathbf{X}^*)[\mathbf{U}] = \mathbf{U}$ with $\kappa^*_{\mathrm{ai}} = 1$. While on $\mathcal{M}_{\mathrm{bw}}$, we have $\kappa^*_{\mathrm{bw}} = \kappa((\mathbf{X}^* \oplus \mathbf{X}^*)((\mathbf{X}^*)^{-1} \otimes (\mathbf{X}^*)^{-1})) = \kappa((\mathbf{X}^*)^{-1} \oplus (\mathbf{X}^*)^{-1}) = \kappa(\mathbf{X}^*)$. Therefore, $\kappa^*_{\mathrm{ai}} \leq \kappa^*_{\mathrm{bw}}$.

### 3.2 Sectional curvature and trigonometry distance bound

To study the curvature of $\mathcal{M}_{\mathrm{bw}}$, we first show in Lemma 2, the existence of a matching geodesic between the Wasserstein geometry of zero-centered non-degenerate Gaussian measures and the BW geometry of SPD matrices. Denote the manifold of such Gaussian measures under the $L^2$-Wasserstein distance as $(\mathcal{N}_0(\mathbf{\Sigma}), \mathcal{W}_2)$ with $\mathbf{\Sigma} \in \mathbb{S}^n_{++}$.

**Lemma 2.** *For any $\mathbf{X}, \mathbf{Y} \in \mathbb{S}^n_{++}$, a geodesic between $\mathcal{N}_0(\mathbf{X})$ and $\mathcal{N}_0(\mathbf{Y})$ on $(\mathcal{N}_0(\mathbf{\Sigma}), \mathcal{W}_2)$ is given by $\mathcal{N}_0(\gamma(t))$, where $\gamma(t)$ is the geodesic between $\mathbf{X}$ and $\mathbf{Y}$ on $\mathcal{M}_{\mathrm{bw}}$.*

The following lemma builds on a result from the Wasserstein geometry [6] and uses Lemma 2 to analyze the sectional curvature of $\mathcal{M}_{\mathrm{bw}}$.

**Lemma 3.** *$\mathcal{M}_{\mathrm{bw}}$ is an Alexandrov space with non-negative sectional curvature.*

It is well-known that $\mathcal{M}_{\mathrm{ai}}$ is a non-positively curved space [19, 43] while, in Lemma 3, we show that $\mathcal{M}_{\mathrm{bw}}$ is non-negatively curved. The difference affects the curvature constant in the trigonometry distance bound of Alexandrov space [64]. This bound is crucial in analyzing convergence for optimization algorithms on Riemannian manifolds [64, 63]. In Section 3.3, only local convergence to a minimizer $\mathbf{X}^*$ is analyzed. Therefore, it suffices to consider a neighbourhood $\Omega$ around $\mathbf{X}^*$. In such a compact set, the sectional curvature is known to be bounded and we denote the lower bound as $K^-$.

The following lemma compares the trigonometry distance bounds under the AI and BW geometries. This bound was originally introduced for Alexandrov space with lower bounded sectional curvature [64]. The result for non-negatively curved spaces has been applied in many work [63, 47, 23] though without a formal proof. We show the proof in the supplementary material, where it follows from the Toponogov comparison theorem [42] on the unit hypersphere and Assumption 1.

**Lemma 4.** *Let $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \in \Omega$, which forms a geodesic triangle on $\mathcal{M}$. Denote $x = d(\mathbf{Y}, \mathbf{Z}), y = d(\mathbf{X}, \mathbf{Z}), z = d(\mathbf{X}, \mathbf{Y})$ as the geodesic side lengths and let $\theta$ be the angle between sides $y$ and $z$ such that $\cos(\theta) = \langle \mathrm{Exp}^{-1}_{\mathbf{X}}(\mathbf{Y}), \mathrm{Exp}^{-1}_{\mathbf{X}}(\mathbf{Z}) \rangle / (yz)$. Then, we have*

$$x^2 \leq \zeta y^2 + z^2 - 2yz\cos(\theta),$$

*where $\zeta$ is a curvature constant. Under the AI metric, $\zeta = \zeta_{\mathrm{ai}} = \frac{\sqrt{|K^-_{\mathrm{ai}}|}D}{\tanh(\sqrt{|K^-_{\mathrm{ai}}|}D)}$ with $D$ as the diameter bound of $\Omega$, i.e. $\max_{\mathbf{X}_1, \mathbf{X}_2 \in \Omega} d(\mathbf{X}, \mathbf{Y}) \leq D$. Under the BW metric, $\zeta = \zeta_{\mathrm{bw}} = 1$.*

It is clear that $\zeta_{\mathrm{ai}} > \zeta_{\mathrm{bw}} = 1$, which leads to a tighter bound under the BW metric.

### 3.3 Convergence analysis

We now analyze the local convergence properties of the Riemannian steepest descent and trust region methods under the two Riemannian geometries. Convergence is established in terms of the Riemannian distance induced from the geodesics. We begin by presenting a lemma that shows in a neighbourhood of $\mathbf{X}^*$, the second-order derivatives of $f \circ \mathrm{Exp}_{\mathbf{X}}$ are both lower and upper bounded.

**Lemma 5.** *In a totally normal neighbourhood $\Omega$ around a non-degenerate local minimizer $\mathbf{X}^*$, for any $\mathbf{X} \in \Omega$, it satisfies that $\lambda^*_{\min}/\alpha \leq \frac{d^2}{dt^2} f(\mathrm{Exp}_{\mathbf{X}}(t\mathbf{U})) \leq \alpha\lambda^*_{\max}$, for some $\alpha \geq 1$ and $\|\mathbf{U}\| = 1$. $\lambda^*_{\max} > \lambda^*_{\min} > 0$ are the largest and smallest eigenvalues of $\mathrm{Hess} f(\mathbf{X}^*)$.*

For simplicity of the analysis, we assume such an $\alpha$ is universal under both the Riemannian geometries. We, therefore, can work with a neighbourhood $\Omega$ with diameter uniformly bounded by $D$, where we can choose $D := \min\{D_{\mathrm{ai}}, D_{\mathrm{bw}}\}$ such that $\alpha$ is universal.

One can readily check that under Lemma 5 the function $f$ is both $\mu$-geodesic strongly convex and $L$-geodesic smooth in $\Omega$ where $\mu = \lambda^*_{\min}/\alpha$ and $L = \alpha\lambda^*_{\max}$. We now present the local convergence analysis of the two algorithms, which are based on results in [64, 1].

**Theorem 1** (Local convergence of Riemannian steepest descent)**.** *Under Assumption 1 and consider a non-degenerate local minimizer $\mathbf{X}^*$. For a neighbourhood $\Omega \ni \mathbf{X}^*$ with diameter bounded by $D$ on two Riemannian geometries $\mathcal{M}_{\mathrm{ai}}, \mathcal{M}_{\mathrm{bw}}$, running Riemannian steepest descent from $\mathbf{X}_0 \in \Omega$ with a fixed step size $\eta = \frac{1}{\alpha\lambda^*_{\max}}$ yields for $t \geq 2$, $d^2(\mathbf{X}_t, \mathbf{X}^*) \leq \alpha^2 D^2 \kappa^* \left(1 - \min\{\frac{1}{\zeta}, \frac{1}{\alpha^2\kappa^*}\}\right)^{t-2}$.*

**Theorem 2** (Local convergence of Riemannian trust region)**.** *Under the same settings as in Theorem 1, assume further in $\Omega$, it holds that (1) $\|\mathcal{H}_{\mathbf{X}_t} - \mathrm{Hess} f(\mathbf{X}_t)\| \leq \ell \|\mathrm{grad} f(\mathbf{X}_t)\|$ and (2) $\|\nabla^2(f \circ \mathrm{Exp}_{\mathbf{X}_t})(\mathbf{U}) - \nabla^2(f \circ \mathrm{Exp}_{\mathbf{X}_t})(\mathbf{0})\| \leq \rho\|\mathbf{U}\|$ for some $\ell, \rho$ universal on $\mathcal{M}_{\mathrm{ai}}, \mathcal{M}_{\mathrm{bw}}$. Then running Riemannian trust region from $\mathbf{X}_0 \in \Omega$ yields, $d(\mathbf{X}_t, \mathbf{X}^*) \leq (2\sqrt{\rho} + \ell)(\kappa^*)^2 d^2(\mathbf{X}_{t-1}, \mathbf{X}^*)$.*

Theorems 1 and 2 show that $\mathcal{M}_{\mathrm{bw}}$ has a clear advantage compared to $\mathcal{M}_{\mathrm{ai}}$ for learning ill-conditioned SPD matrices where $\kappa^*_{\mathrm{bw}} \leq \kappa^*_{\mathrm{ai}}$. For first-order algorithms, $\mathcal{M}_{\mathrm{bw}}$ has an additional benefit due to its non-negative sectional curvature. As $\zeta_{\mathrm{ai}} > \zeta_{\mathrm{bw}} = 1$, the convergence rate degrades on $\mathcal{M}_{\mathrm{ai}}$. Although the convergence is presented in Riemannian distance, it can be readily converted to function value gap by noticing $\frac{\mu}{2} d^2(\mathbf{X}_t, \mathbf{X}^*) \leq f(\mathbf{X}_t) - f(\mathbf{X}^*) \leq \frac{L}{2} d^2(\mathbf{X}_t, \mathbf{X}^*)$. Additionally, we note that these local convergence results hold regardless of whether the function is geodesic convex or not, and similar comparisons also exist for other Riemannian optimization methods.

### 3.4 Geodesic convexity under BW metric for cost functions of interest

Finally we show geodesic convexity of common optimization problems on $\mathcal{M}_{\mathrm{bw}}$. Particularly, we verify that linear, quadratic, log-det optimization, and also certain geometric optimization problems, that are geodesic convex under the AI metric, are also geodesic convex under the BW metric.

**Proposition 1.** *For any $\mathbf{A} \in \mathbb{S}^n_+$, where $\mathbb{S}^n_+ := \{\mathbf{Z} : \mathbf{Z} \in \mathbb{R}^{n\times n}, \mathbf{Z}^\top = \mathbf{Z}$, and $\mathbf{Z} \succeq \mathbf{0}\}$, the functions $f_1(\mathbf{X}) = \mathrm{tr}(\mathbf{X}\mathbf{A})$, $f_2(\mathbf{X}) = \mathrm{tr}(\mathbf{X}\mathbf{A}\mathbf{X})$, and $f_3(\mathbf{X}) = -\log\det(\mathbf{X})$ are geodesic convex on $\mathcal{M}_{\mathrm{bw}}$.*

Based on the result in Proposition 1, we also prove geodesic convexity of a reparameterized version of the Gaussian density estimation and mixture model on $\mathcal{M}_{\mathrm{bw}}$ (discussed in Section 4). Similar claims on $\mathcal{M}_{\mathrm{ai}}$ can be found in [27].

We further show that monotonic functions on sorted eigenvalues are geodesic convex on $\mathcal{M}_{\mathrm{bw}}$. This is an analogue of [52, Theorem 2.3] on $\mathcal{M}_{\mathrm{ai}}$.

**Proposition 2.** *Let $\lambda^\downarrow : \mathbb{S}^n_{++} \to \mathbb{R}^n_+$ be the decreasingly sorted eigenvalue map and $h : \mathbb{R}_+ \to \mathbb{R}$ be an increasing and convex function. Then $f(\mathbf{X}) = \sum_{j=1}^k h(\lambda^\downarrow_j(\mathbf{X}))$ for $1 \leq k \leq n$ is geodesic convex on $\mathcal{M}_{\mathrm{bw}}$. Examples of such functions include $f_1(\mathbf{X}) = \mathrm{tr}(\exp(\mathbf{X}))$ and $f_2(\mathbf{X}) = \mathrm{tr}(\mathbf{X}^\alpha)$, $\alpha \geq 1$.*

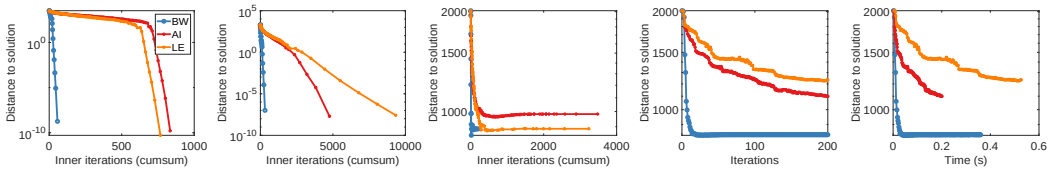
## 4 Experiments

In this section, we compare the empirical performance of optimization algorithms under different Riemannian geometries for various problems. In addition to AI and BW, we also include the Log-Euclidean (LE) geometry [7] in our experiments.

The LE geometry explores the the linear space of symmetric matrices where the matrix exponential acts as a global diffeomorphism from the space to $\mathbb{S}^n_{++}$. The LE metric is defined as

$$\langle \mathbf{U}, \mathbf{V} \rangle_{\mathrm{le}} = \mathrm{tr}(\mathrm{D}_{\mathbf{X}} \log(\mathbf{U}) \mathrm{D}_{\mathbf{X}} \log(\mathbf{V})) \tag{3}$$

for any $\mathbf{U}, \mathbf{V} \in T_{\mathbf{X}}\mathcal{M}$, where $\mathrm{D}_{\mathbf{X}} \log(\mathbf{U})$ is the directional derivative of matrix logarithm at $\mathbf{X}$ along $\mathbf{U}$. Following [56, 41, 45], for deriving various Riemannian optimization ingredients under the LE




(a) `Dense`, `LowCN` (`RTR:iter`) (b) `Dense`, `HighCN` (`RTR:iter`) (c) `Sparse`, `HighCN` (`RTR:iter`) (d) `Sparse`, `HighCN` (`RSD:iter`) (e) `Sparse`, `HighCN` (`RSD:time`)

Figure 1: Weighted least squares problem.

metric (3), we consider the parameterization $\mathbf{X} = \exp(\mathbf{S})$, where $\mathbf{S} \in \mathbb{S}^n$, i.e., the space of $n \times n$ symmetric matrices. Equivalently, optimization on the SPD manifold with the LE metric is identified with optimization on $\mathbb{S}^n$ and the function of interest becomes $f(\exp(\mathbf{S}))$ for $\mathbf{S} \in \mathbb{S}^n$. While the Riemannian gradient can be computed efficiently by exploiting the directional derivative of the matrix exponential [4], deriving the Riemannian Hessian is tricky and we rely on finite-difference Hessian approximations [13].

We present convergence mainly in terms of the distance to the solution $\mathbf{X}^*$ whenever applicable. The distance is measured in the Frobenius norm, i.e., $\|\mathbf{X}_t - \mathbf{X}^*\|_{\mathrm{F}}$. When $\mathbf{X}^*$ is not known, convergence is shown in the modified Euclidean gradient norm $\|\mathbf{X}_t \nabla f(\mathbf{X}_t)\|_{\mathrm{F}}$. This is comparable across different metrics as the optimality condition $\mathbf{X}^* \nabla f(\mathbf{X}^*) = \mathbf{0}$ arises from problem structure itself [34]. We initialize the algorithms with the identity matrix for the AI and BW metrics and zero matrix for the LE metric (i.e., the matrix logarithm of the identity).

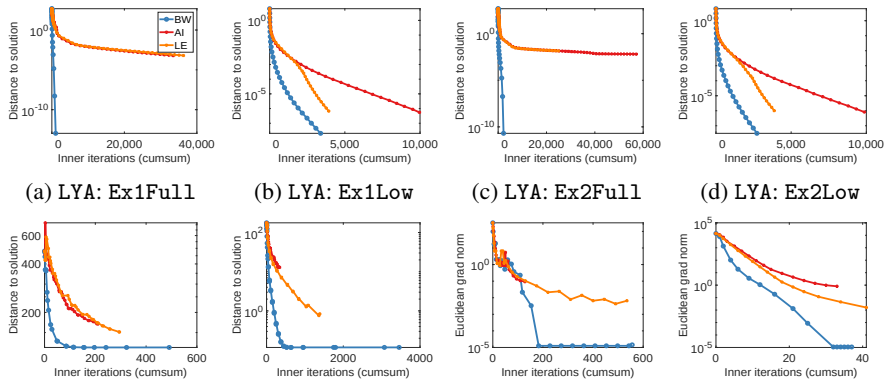
We mainly present the results on the Riemannian trust region (RTR) method, which is the method of choice for Riemannian optimization. Note for RTR, the results are shown against the cumulative sum of inner iterations (which are required to solve the trust region subproblem at every iteration). We also include the Riemannian steepest descent (RSD) and Riemannian stochastic gradient (RSGD) [12] methods for some examples. The experiments are conducted in Matlab using the Manopt toolbox [15] on a i5-10500 3.1GHz CPU processor.

In the supplementary material, we include additional experiments comparing convergence in objective function values for the three geometries. We also present results for the Riemannian conjugate gradient method, and results with different initializations (other than the identity and zero matrices) to further support our claims.

**Weighted least squares.** We first consider the weighted least squares problem with the symmetric positive definite constraint. The optimization problem is $\min_{\mathbf{X} \in \mathbb{S}^n_{++}} f(\mathbf{X}) = \frac{1}{2}\|\mathbf{A} \odot \mathbf{X} - \mathbf{B}\|_{\mathrm{F}}^2$, which is encountered in for example, SPD matrix completion [50] where $\mathbf{A}$ is a sparse matrix. The Euclidean gradient and Hessian are $\nabla f(\mathbf{X}) = (\mathbf{A} \odot \mathbf{X} - \mathbf{B}) \odot \mathbf{A}$ and $\nabla^2 f(\mathbf{X})[\mathbf{U}] = \mathbf{A} \odot \mathbf{U} \odot \mathbf{A}$, respectively. Hence, at optimal $\mathbf{X}^*$, the Euclidean Hessian in matrix representation is $\mathbf{H}(\mathbf{X}^*) = \mathrm{diag}(\mathrm{vec}(\mathbf{A} \odot \mathbf{A}))$. We experiment with two choices of $\mathbf{A}$, i.e. $\mathbf{A} = \mathbf{1}_n \mathbf{1}_n^\top$ (`Dense`) and $\mathbf{A}$ as a random sparse matrix (`Sparse`). The former choice for $\mathbf{A}$ leads to well-conditioned $\mathbf{H}(\mathbf{X}^*)$ while the latter choice leads to an ill-conditioned $\mathbf{H}(\mathbf{X}^*)$. Also note that when $\mathbf{A} = \mathbf{1}_n \mathbf{1}_n^\top$, $\kappa^*_{\mathrm{ai}} = \kappa(\mathbf{X}^*)^2$ and $\kappa^*_{\mathrm{bw}} = \kappa(\mathbf{X}^*)$.

We generate $\mathbf{X}^*$ as a SPD matrix with size $n = 50$ and exponentially decaying eigenvalues. We consider two cases with condition numbers $\kappa(\mathbf{X}^*) = 10$ (`LowCN`) and $10^3$ (`HighCN`). The matrix $\mathbf{B}$ is generated as $\mathbf{B} = \mathbf{A} \odot \mathbf{X}^*$. Figure 1 compares both RSD and RTR for different metrics. When $\mathbf{A}$ is either dense or sparse, convergence is significantly faster on $\mathcal{M}_{\mathrm{bw}}$ than both $\mathcal{M}_{\mathrm{ai}}$ and $\mathcal{M}_{\mathrm{le}}$. The advantage of using $\mathcal{M}_{\mathrm{bw}}$ becomes more prominent in the setting when condition number of $\mathbf{X}^*$ is high. Figure 1(e) shows that $\mathcal{M}_{\mathrm{bw}}$ is also superior in terms of runtime.

**Lyapunov equations.** Continuous Lyapunov matrix equation, $\mathbf{A}\mathbf{X} + \mathbf{X}\mathbf{A} = \mathbf{C}$ with $\mathbf{X} \in \mathbb{S}^n_{++}$, are commonly employed in analyzing optimal control systems and differential equations [46, 36]. When $\mathbf{A}$ is stable, i.e., $\lambda_i(\mathbf{A}) > 0$ and $\mathbf{C} \in \mathbb{S}^n_{++}$, the solution $\mathbf{X}^* \succ \mathbf{0}$ and is unique [35]. When $\mathbf{C} \in \mathbb{S}^n_{+}$ and is low rank, $\mathbf{X}^* \in \mathbb{S}^n_{+}$ is also low rank. We optimize the following problem for solving the Lyapunov equation [60]: $\min_{\mathbf{X} \in \mathbb{S}^n_{++}} f(\mathbf{X}) = \mathrm{tr}(\mathbf{X}\mathbf{A}\mathbf{X}) - \mathrm{tr}(\mathbf{X}\mathbf{C})$. The Euclidean gradient and Hessian are respectively $\nabla f(\mathbf{X}) = \mathbf{A}\mathbf{X} + \mathbf{X}\mathbf{A} - \mathbf{C}$ and $\nabla^2 f(\mathbf{X})[\mathbf{U}] = \mathbf{A}\mathbf{U} + \mathbf{U}\mathbf{A}$ with $\mathbf{H}(\mathbf{X}) = \mathbf{A} \oplus \mathbf{A}$. At optimal $\mathbf{X}^*$, the condition number $\kappa(\mathbf{H}(\mathbf{X}^*)) = \kappa(\mathbf{A})$.




(a) LYA: Ex1Full (b) LYA: Ex1Low (c) LYA: Ex2Full (d) LYA: Ex2Low

(e) TR: SynFull (f) TR: SynLow (g) DML: glass (h) DML: phoneme

Figure 2: Lyapunov equation (a, b, c, d), trace regression (e, f), and metric learning (g, h) problems.

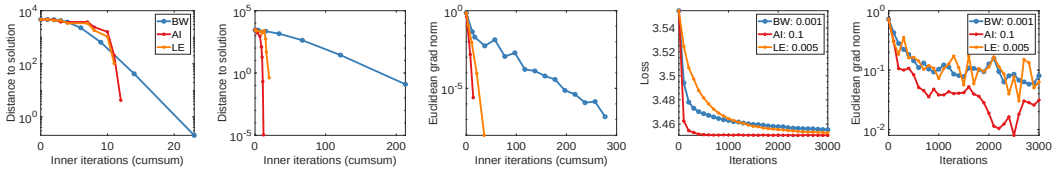
We experiment with two settings for the matrix $\mathbf{A}$, i.e. $\mathbf{A}$ as the Laplace operator on the unit square where we generate 7 interior points so that $n = 49$ (Ex1), and $\mathbf{A}$ is a particular Toeplitz matrix with $n = 50$ (Ex2). The generated $\mathbf{A}$ matrices are ill-conditioned. The above settings correspond to Examples 7.1 and 7.3 in [36]. Under each setting, $\mathbf{X}^*$ is set to be either full or low rank. The matrix $\mathbf{C}$ is generated as $\mathbf{C} = \mathbf{A}\mathbf{X}^* + \mathbf{X}^*\mathbf{A}$. The full rank $\mathbf{X}^*$ is generated from the full-rank Wishart distribution while the low rank $\mathbf{X}^*$ is a diagonal matrix with $r = 10$ ones and $n - r$ zeros in the diagonal. We label the four cases as Ex1Full, Ex1Low, Ex2Full, and Ex2Low. The results are shown in Figures 2(a)-(d), where we observe that in all four cases, the BW geometry outperforms both AI and LE geometries.

**Trace regression.** We consider the regularization-free trace regression model [49] for estimating covariance and kernel matrices [48, 17]. The optimization problem is $\min_{\mathbf{X}\in\mathbb{S}^d_{++}} f(\mathbf{X}) = \frac{1}{2m}\sum_{i=1}^m(\mathbf{y}_i - \mathrm{tr}(\mathbf{A}_i^\top\mathbf{X}))^2$, where $\mathbf{A}_i = \mathbf{a}_i\mathbf{a}_i^\top$, $i = 1, ..., m$ are some rank-one measurement matrices. Thus, we have $\nabla f(\mathbf{X}) = \sum_{i=1}^m(\mathbf{a}_i^\top\mathbf{X}\mathbf{a}_i - \mathbf{y}_i)\mathbf{A}_i$ and $\nabla^2 f(\mathbf{X})[\mathbf{U}] = \sum_{i=1}^m(\mathbf{a}_i^\top\mathbf{U}\mathbf{a}_i)\mathbf{A}_i$. We create $\mathbf{X}^*$ as a rank-$r$ Wishart matrix and $\{\mathbf{A}_i\}$ as rank-one Wishart matrices and generate $\mathbf{y}_i = \mathrm{tr}(\mathbf{A}_i\mathbf{X}^*) + \sigma\epsilon_i$ with $\epsilon_i \sim \mathcal{N}(0, 1)$, $\sigma = 0.1$. We consider two choices, $(m, d, r) = (1000, 50, 50)$ and $(1000, 50, 10)$, which are respectively labelled as SynFull and SynLow. From Figures 2(e)&(f), we also observe that convergence to the optimal solution is faster for the BW geometry.

**Metric learning.** Distance metric learning (DML) aims to learn a distance function from samples and a popular family of such distances is the Mahalanobis distance, i.e. $d_{\mathbf{M}}(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^\top\mathbf{M}(\mathbf{x} - \mathbf{y})}$ for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. The distance is parameterized by a symmetric positive semi-definite matrix $\mathbf{M}$. We refer readers to this survey [53] for more discussions on this topic. We particularly consider a logistic discriminant learning formulation [22]. Given a training sample $\{\mathbf{x}_i, y_i\}_{i=1}^N$, denote the link $t_{ij} = 1$ if $y_i = y_j$ and $t_{ij} = 0$ otherwise. The objective is given by $\min_{\mathbf{M}\in\mathbb{S}^d_{++}} f(\mathbf{M}) = -\sum_{i,j}\left(t_{ij}\log p_{ij} + (1 - t_{ij})\log(1 - p_{ij})\right)$, with $p_{ij} = (1 + \exp(d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)))^{-1}$. We can derive the matrix Hessian as $\mathbf{H}(\mathbf{M}) = \sum_{i,j} p_{ij}(1 - p_{ij})(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top \otimes (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^\top$. Notice $\kappa(\mathbf{H}(\mathbf{M}^*))$ depends on $\mathbf{M}^*$ only through the constants $p_{ij}$. Thus, the condition number will not be much affected by $\kappa(\mathbf{M}^*)$.

We consider two real datasets, glass and phoneme, from the Keel database [5]. The number of classes is denoted as $c$. The statistics of these two datasets are $(N, d, c) = (241, 9, 7)$ for glass $(5404, 5, 2)$ for phoneme. In Figures 2(g)&(h), we similarly see the advantage of using the BW metric compared to the other two metrics that behave similarly.

**Log-det maximization.** As discussed in Section 3.1, log-det optimization is one instance where $\kappa^*_{\mathrm{bw}} \geq \kappa^*_{\mathrm{ai}}$. We first consider minimizing negative log-determinant along with a linear function as studied in [62]. That is, for some $\mathbf{C} \in \mathbb{S}^n_{++}$, the objective is $\min_{\mathbf{X}\in\mathbb{S}^n_{++}} f(\mathbf{X}) = \mathrm{tr}(\mathbf{X}\mathbf{C}) - \log\det(\mathbf{X})$. The Euclidean gradient and Hessian are given by $\nabla f(\mathbf{X}) = \mathbf{C} - \mathbf{X}^{-1}$ and $\nabla^2 f(\mathbf{X})[\mathbf{U}] =$




(a) LD: LowCN (RTR) (b) LD: HighCN (RTR) (c) GMM: (RTR) (d) GMM: (RSGD) (e) GMM: (RSGD)

Figure 3: Log-det maximization (a, b) and Gaussian mixture model (c, d, e) problems.

$\mathbf{X}^{-1}\mathbf{U}\mathbf{X}^{-1}$. This problem is geodesic convex under both AI and BW metrics. We generate $\mathbf{X}^*$ the same way as in the example of weighted least square with $n = 50$ and set $\mathbf{C} = (\mathbf{X}^*)^{-1}$. We consider two cases with condition number cn = 10 (`LowCN`) and $10^3$ (`HighCN`). As expeted, we observe faster convergence of AI and LE metrics over the BW metric in Figures 3(a)&(b). This is even more evident when the condition number increases.

**Gaussian mixture model.** Another notable example of log-det optimization is the Gaussian density estimation and mixture model problem. Following [27], we consider a reformulated problem on augmented samples $\mathbf{y}_i^\top = [\mathbf{x}_i^\top; 1], i = 1, ..., N$ where $\mathbf{x}_i \in \mathbb{R}^d$ are the original samples. The density is parameterized by the augmented covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{d+1}$. Notice that the log-likelihood of Gaussian is geodesic convex on $\mathcal{M}_{\text{ai}}$, but not on $\mathcal{M}_{\text{bw}}$. We, therefore, define $\mathbf{S} = \mathbf{\Sigma}^{-1}$ and the reparameterized log-likelihood is $p_{\mathcal{N}}(\mathbf{Y}; \mathbf{S}) = \sum_{i=1}^N \log\left((2\pi)^{1-d/2} \exp(1/2) \det(\mathbf{S})^{1/2} \exp(-\frac{1}{2}\mathbf{y}_i^\top \mathbf{S}\mathbf{y}_i)\right)$, which is now geodesic convex on $\mathcal{M}_{\text{bw}}$ due to Proposition 1. Hence, we can solve the problem of Gaussian mixture model similar as in [27]. More details are given in supplementary material.

Here, we test on a dataset included in the MixEst package [26]. The dataset has $1580$ samples in $\mathbb{R}^2$ with 3 Gaussian components. In Figure 3(c), we observe a similar pattern with RTR as in the log-det example. We also include performance of RSGD, which is often preferred for large scale problems. We set the batch size to be $50$ and consider a decaying step size, with the best initialized step size shown in Figures 3(d)&(e). Following [8], the algorithms are initialized with `kmeans++`. We find that the AI geometry still maintains its advantage under the stochastic setting.

## 5 Conclusion and Discussion

In this paper, we show that the less explored Bures-Wasserstein geometry for SPD matrices is often a better choice than the Affine-Invariant geometry for optimization, particularly for learning ill-conditioned matrices. Also, a systematic analysis shows that the BW metric preserves geodesic convexity of some popular cost functions and leads to better rates for certain function classes.

Our comparisons are based on optimization over generic cost functions. For specific problems, however, there may exist other alternative metrics that potentially work better. This is an interesting research direction to pursue. We also remark that optimization is not the only area where the AI and BW geometries can be compared. It would be also useful to compare the two metrics for other learning problems on SPD matrices, such as barycenter learning. In addition, kernel methods have been studied on the SPD manifold [33, 25, 65] that embed SPD matrices to a high dimensional feature space, known as the Reproducing Kernel Hilbert Space (RKHS). Such representations are used for subsequent learning tasks such as clustering or classification. But only a positive definite kernel provides a valid RKHS. We show (in the supplementary material) that the induced Gaussian kernel based on the BW distance is a positive definite kernel unlike the case for the AI metric. This difference highlights a potential advantage of the BW metric for representation learning on SPD matrices.

## Broader Impact

This paper is of a theoretical nature and does not foresee any immediate societal impacts.

# References


[1] 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.

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

[3] Naman Agarwal, Nicolas Boumal, Brian Bullins, and Coralia Cartis. Adaptive regularization with cubics on manifolds. *Mathematical Programming*, pages 1–50, 2020.

[4] Awad H Al-Mohy and Nicholas J Higham. Computing the Fréchet derivative of the matrix exponential, with an application to condition number estimation. *SIAM Journal on Matrix Analysis and Applications*, 30(4):1639–1657, 2009.

[5] Jesús Alcalá-Fdez, Luciano Sánchez, Salvador Garcia, Maria Jose del Jesus, Sebastian Ventura, Josep Maria Garrell, José Otero, Cristóbal Romero, Jaume Bacardit, Victor M Rivas, et al. Keel: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3):307–318, 2009.

[6] Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2008.

[7] Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 29(1):328–347, 2007.

[8] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[9] Ronny Bergmann. Optimisation on Manifolds in Julia, 2019. `https://github.com/kellertuer/Manopt.jl`.

[10] Rajendra Bhatia. *Positive definite matrices*. Princeton university press, 2009.

[11] Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the Bures-Wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2019.

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

[13] Nicolas Boumal. Riemannian trust regions with finite-difference Hessian approximations are globally convergent. In *International Conference on Geometric Science of Information*, pages 467–475. Springer, 2015.

[14] Nicolas Boumal. An introduction to optimization on smooth manifolds. *Available online, May*, 2020.

[15] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research*, 15(1):1455–1459, 2014.

[16] Daniel A Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Exploring complex time-series representations for Riemannian machine learning of radar data. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3672–3676. IEEE, 2019.

[17] T Tony Cai and Anru Zhang. ROP: Matrix recovery via rank-one projections. *Annals of Statistics*, 43(1):102–138, 2015.

[18] Zeineb Chebbi and Maher Moakher. Means of Hermitian positive-definite matrices based on the log-determinant $\alpha$-divergence function. *Linear Algebra and its Applications*, 436(7):1872–1889, 2012.

[19] Calin Cruceru, Gary Bécigneul, and Octavian-Eugen Ganea. Computationally tractable Riemannian manifolds for graph embeddings. *arXiv:2002.08665*, 2020.

[20] Ian L. Dryden, Alexey Koloydenko, and Diwei Zhou. Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging. *The Annals of Applied Statistics*, 3(3):1102–1123, 2009.

[21] Zhi Gao, Yuwei Wu, Yunde Jia, and Mehrtash Harandi. Learning to Optimize on SPD Manifolds. In *Conference on Computer Vision and Pattern Recognition*, pages 7700–7709, 2020.

[22] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? Metric learning approaches for face identification. In *International Conference on Computer Vision*, pages 498–505. IEEE, 2009.

[23] Andi Han and Junbin Gao. Variance reduction for Riemannian non-convex optimization with batch size adaptation. *arXiv:2007.01494*, 2020.

[24] Mehrtash T Harandi, Mathieu Salzmann, and Richard Hartley. From manifold to manifold: Geometry-aware dimensionality reduction for SPD matrices. In *European conference on computer vision*, pages 17–32. Springer, 2014.

[25] Mehrtash T Harandi, Conrad Sanderson, Richard Hartley, and Brian C Lovell. Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach. In *European Conference on Computer Vision*, pages 216–229. Springer, 2012.

[26] Reshad Hosseini and Mohamadreza Mash'al. MixEst: An estimation toolbox for mixture models. *arXiv:1507.06065*, 2015.

[27] Reshad Hosseini and Suvrit Sra. An alternative to EM for Gaussian mixture models: batch and stochastic Riemannian optimization. *Mathematical Programming*, 181(1):187–223, 2020.

[28] Wen Huang, Pierre-Antoine Absil, Kyle A. Gallivan, and Paul Hand. Roptlib: an object-oriented C++ library for optimization on Riemannian manifolds. Technical Report FSU16-14.v2, Florida State University, 2016.

[29] Wen Huang, Kyle A Gallivan, and P-A Absil. A Broyden class of quasi-Newton methods for Riemannian optimization. *SIAM Journal on Optimization*, 25(3):1660–1685, 2015.

[30] Zhiwu Huang and Luc Van Gool. A Riemannian network for SPD matrix learning. In *AAAI*, 2017.

[31] Zhiwu Huang, Ruiping Wang, Xianqiu Li, Wenxian Liu, Shiguang Shan, Luc Van Gool, and Xilin Chen. Geometry-aware similarity learning on SPD manifolds for visual recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2513–2523, 2017.

[32] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-Euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *International Conference on Machine Learning*, pages 720–729. PMLR, 2015.

[33] 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*, pages 73–80, 2013.

[34] Michel Journée, Francis Bach, P-A Absil, and Rodolphe Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

[35] Peter Lancaster. Explicit solutions of linear matrix equations. *SIAM review*, 12(4):544–566, 1970.

[36] Yiding Lin and Valeria Simoncini. A new subspace iteration method for the algebraic riccati equation. *Numerical Linear Algebra with Applications*, 22(1):26–47, 2015.

[37] Zhenhua Lin. Riemannian geometry of symmetric positive definite matrices via cholesky decomposition. *SIAM Journal on Matrix Analysis and Applications*, 40(4):1353–1370, 2019.

[38] Sridhar Mahadevan, Bamdev Mishra, and Shalini Ghosh. A unified framework for domain adaptation using metric learning on manifolds. In *ECML-PKDD*, 2019.

[39] Luigi Malagò, Luigi Montrucchio, and Giovanni Pistone. Wasserstein Riemannian geometry of Gaussian densities. *Information Geometry*, 1(2):137–179, 2018.

[40] Mayank Meghwanshi, Pratik Jawanpuria, Anoop Kunchukuttan, Hiroyuki Kasai, and Bamdev Mishra. Mctorch, a manifold optimization library for deep learning. Technical report, arXiv:1810.01811, 2018.

[41] Gilles Meyer, Silvère Bonnabel, and Rodolphe Sepulchre. Regression on fixed-rank positive semidefinite matrices: a Riemannian approach. *The Journal of Machine Learning Research*, 12:593–625, 2011.

[42] Wolfgang Meyer. Toponogov's theorem and applications. *Lecture Notes, Trieste*, 1989.

[43] Xavier Pennec. Manifold-valued image processing with SPD matrices. In *Riemannian Geometric Statistics in Medical Image Analysis*, pages 75–134. Elsevier, 2020.

[44] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A Riemannian framework for tensor computing. *International Journal of computer vision*, 66(1):41–66, 2006.

[45] Minh Ha Quang, Marco San Biagio, and Vittorio Murino. Log-hilbert-schmidt metric between positive definite operators on hilbert spaces. In *Advances in neural information processing systems*, pages 388–396, 2014.

[46] D Rothschild and A Jameson. Comparison of four numerical algorithms for solving the Liapunov matrix equation. *International Journal of Control*, 11(2):181–198, 1970.

[47] 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.

[48] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[49] Martin Slawski, Ping Li, and Matthias Hein. Regularization-free estimation in trace regression with symmetric positive semidefinite matrices. In *Advances in neural information processing systems*, 2015.

[50] Ronald L. Smith. The positive definite completion problem revisited. *Linear Algebra and its Applications*, 429(7):1442–1452, 2008.

[51] Suvrit Sra. A new metric on the manifold of kernel matrices with application to matrix geometric means. *Advances in Neural Information Processing Systems*, 25:144–152, 2012.

[52] Suvrit Sra and Reshad Hosseini. Conic geometric optimization on the manifold of positive definite matrices. *SIAM Journal on Optimization*, 25(1):713–739, 2015.

[53] Juan Luis Suárez, Salvador García, and Francisco Herrera. A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges. *Neurocomputing*, 425:300–322, 2021.

[54] Yann Thanwerdas and Xavier Pennec. Is affine-invariance well defined on SPD matrices? a principled continuum of metrics. In *International Conference on Geometric Science of Information*, pages 502–510. Springer, 2019.

[55] James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation. *The Journal of Machine Learning Research*, 17(1):4755–4759, 2016.

[56] Koji Tsuda, Gunnar Rätsch, and Manfred K. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projection. *Journal of Machine Learning Research*, 6(34):995–1018, 2005.

[57] Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on Riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1713–1727, 2008.

[58] Constantin Udriste. *Convex functions and optimization methods on Riemannian manifolds*, volume 297. Springer Science & Business Media, 2013.

[59] Jesse van Oostrum. Bures-Wasserstein geometry. *arXiv:2001.08056*, 2020.

[60] Bart Vandereycken and Stefan Vandewalle. A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2553–2579, 2010.

[61] Nisheeth K Vishnoi. Geodesic convex optimization: Differentiation on manifolds, geodesics, and convexity. *arXiv:1806.06373*, 2018.

[62] Chengjing Wang, Defeng Sun, and Kim-Chuan Toh. Solving log-determinant optimization problems by a Newton-CG primal proximal point algorithm. *SIAM Journal on Optimization*, 20(6):2994–3013, 2010.

[63] Hongyi Zhang, Sashank J Reddi, and Suvrit Sra. Riemannian SVRG: Fast stochastic optimization on Riemannian manifolds. *arXiv:1605.07147*, 2016.

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

[65] Jianjia Zhang, Lei Wang, Luping Zhou, and Wanqing Li. Learning discriminative stein kernel for SPD matrices and its applications. *IEEE Transactions on Neural Networks and Learning Systems*, 27(5):1020–1033, 2015.

# 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] The paper is limited to an optimization viewpoint for comparing metrics. And also see the conclusion section.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Because this paper is of a theoretical nature and does not foresee any immediate societal impacts.
   (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 and Section 2 for definitions.
   (b) Did you include complete proofs of all theoretical results? [Yes] See supplementary sections.
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] See supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See supplementary material for sensitivity analysis to random seeds.

(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] See Section 4.

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]
   (b) Did you mention the license of the assets? [Yes] The license is included in code.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A] All assets have been properly cited in the main text.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] All data are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
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? [N/A]
   (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]