Mathematical Programming (2023) 199:525–556
https://doi.org/10.1007/s10107-022-01840-5

FULL LENGTH PAPER

Series A

Riemannian Optimization via Frank-Wolfe Methods

Melanie Weber[1] · **Suvrit Sra**[2]

Received: 15 March 2020 / Accepted: 10 May 2022 / Published online: 14 July 2022

Abstract
We study projection-free methods for constrained Riemannian optimization. In particular, we propose a Riemannian Frank-Wolfe (RFW) method that handles constraints directly, in contrast to prior methods that rely on (potentially costly) projections. We analyze non-asymptotic convergence rates of RFW to an optimum for geodesically convex problems, and to a critical point for nonconvex objectives. We also present a practical setting under which RFW can attain a linear convergence rate. As a concrete example, we specialize RFW to the manifold of positive definite matrices and apply it to two tasks: (i) computing the matrix geometric mean (Riemannian centroid); and (ii) computing the Bures-Wasserstein barycenter. Both tasks involve geodesically convex interval constraints, for which we show that the Riemannian "linear" oracle required by RFW admits a closed form solution; this result may be of independent interest. We complement our theoretical results with an empirical comparison of RFW against state-of-the-art Riemannian optimization methods, and observe that RFW performs competitively on the task of computing Riemannian centroids.

Mathematics Subject Classification 46N10 · 15A24 · 65K10 · 49Q99

SS acknowledges support from NSF-IIS-1409802. This work was partially done during a visit of MW at MIT supported by a Dean's grant from the Princeton University Graduate School.

✉ Melanie Weber
melanie.weber@maths.ox.ac.uk

Suvrit Sra
suvrit@mit.edu

[1] Mathematical Institute, University of Oxford (Work done while at Princeton University), Oxford, UK

[2] Laboratory for Information and Decision Systems, MIT, Cambridge, US

1 Introduction

We study the following constrained optimization problem

$$\min_{x \in \mathcal{X} \subseteq \mathcal{M}} \quad \phi(x), \tag{1}$$

where $\phi : \mathcal{M} \to \mathbb{R}$ is a differentiable function and $\mathcal{X}$ is a compact geodesically convex (henceforth, *g-convex*) subset of a Riemannian manifold $\mathcal{M}$. The objective ϕ may be g-convex or nonconvex. When the constraint set $\mathcal{X}$ is "simple" one may solve (1) via Riemannian projected-gradient. But in many cases, projection onto $\mathcal{X}$ can be expensive to compute, motivating us to seek *projection-free* methods.

Euclidean ($\mathcal{M} \equiv \mathbb{R}^n$) projection-free methods based on the Frank-Wolfe (FW) scheme [20] have recently witnessed a surge of interest in machine learning and related fields [29, 36]. Instead of projection, such FW methods rely on access to a *"linear" oracle,* that is, a subroutine that solves the problem

$$\min_{z \in \mathcal{X}} \quad \langle z,\ \nabla \phi(x) \rangle\ , \tag{2}$$

which can sometimes be much simpler than projection onto $\mathcal{X}$. This attractive property of FW methods has been exploited in convex [2, 29], nonconvex [35], submodular [12, 21], and stochastic [24, 52] optimization problems; among others.

But as far as we are aware, FW methods have not been studied for Riemannian manifolds. Our work fills this gap in the literature by developing, analyzing, and experimenting with Riemannian Frank-Wolfe (RFW) methods. In addition to adapting FW to the Riemannian setting, there is one more challenge that we must overcome: RFW requires access to a Riemannian analog of the linear oracle (2), which can be hard even for g-convex problems.

Therefore, to complement our theoretical analysis of RFW, we discuss in detail practical settings that admit efficient Riemannian "linear" oracles. Specifically, we discuss problems where $\mathcal{M} = \mathbb{P}_d$, the manifold of (Hermitian) positive definite matrices, and $\mathcal{X}$ is a g-convex semidefinite interval; then problem (1) assumes the form

$$\min_{X \in \mathcal{X} \subseteq \mathbb{P}_d} \quad \phi(X), \qquad \text{where } \mathcal{X} := \{X \in \mathbb{P}_d \mid L \preceq X \preceq U\}, \tag{3}$$

where L and U are positive definite matrices. An important instance of (3) is the following g-convex optimization problem (see §4 for details and notation):

$$\min_{X \in \mathbb{P}_d} \quad \sum_{i=1}^{n} w_i \delta_R^2(X, A_i), \quad \text{where } w \in \Delta_n, A_1, \ldots, A_n \in \mathbb{P}_d, \tag{4}$$

which computes the Riemannian centroid of a set of positive definite matrices (also known as the "matrix geometric mean" and the "Karcher mean") [5, 32, 37]. We will show that RFW offers a simple approach for solving (4) that performs competitively

against recently published state-of-the-art Riemannian approaches. As a second application, we show that RFW allows for an efficient computation of Bures-Wasserstein barycenters on the Gaussian density manifold.

Summary of results. The key contributions of this paper are as follows:

1. We introduce a Riemannian Frank-Wolfe (RFW) algorithm for addressing constrained g-convex optimization on Riemannian manifolds. We show that RFW attains a *non-asymptotic* $O(1/k)$ rate of convergence to the optimal objective value, k being the number of iterations (Theorem 1). Furthermore, under additional assumptions on the objective function and the constraints, we show that RFW can even attain linear convergence rates (Theorem 2). In the nonconvex case, RFW attains a *non-asymptotic* $O(1/\sqrt{k})$ rate of convergence to first-order critical points (Theorem 3). These rates are comparable to the best known guarantees for the classical Euclidean Frank-Wolfe algorithm [29, 36].
2. While the Euclidean "linear" oracle is a convex problem, the Riemannian "linear" oracle is nonconvex. Therefore, the key challenge of developing RFW lies in efficiently solving the "linear" oracle. We address this problem with the following contributions:
   - We specialize RFW for g-convex problems of the form (3) on the manifold of Hermitian positive definite (HPD) matrices. Importantly, for this problem we develop a closed-form solution to the Riemannian "linear" oracle, which involves solving a nonconvex semi-definite program (*SDP*), see Theorem 4. We then apply RFW to computing the Riemannian mean of HPD matrices. In comparison with state-of-the-art methods, we observe that RFW performs competitively. Furthermore, we implement RFW for the computation of Bures-Wasserstein barycenters on the Gaussian density manifold.
   - We show that we can recover a sublinear convergence rate, even if the Riemannian "linear" oracle can only be solved approximately, e.g., using relaxations or iterative solvers. This makes the approach applicable to a wider range of constrained optimization problems.

We believe that the closed-form solutions for the HPD "linear" oracle, which involve a nonconvex SDP, should be of wider interest too. A similar approach can be used to solve the Euclidean linear oracle, a convex SDP, in closed form (Appendix 1). More broadly, we hope that our results encourage others to study RFW as well as other examples of problems with efficient Riemannian "linear" oracles.

Related work. Riemannian optimization has a venerable history. The books [1, 58] provide a historical perspective as well as basic theory. The focus of these books and of numerous older works on Riemannian optimization, e.g.,[19, 25, 41, 53], is almost exclusively on asymptotic analysis. More recently, non-asymptotic convergence analysis quantifying the iteration complexity of Riemannian optimization algorithms has begun to be pursued [3, 10, 66]. Specifically, it is known that first-order methods, such as Riemannian Gradient Descent, achieve a sublinear iteration complexity. However, to the best of our knowledge, all these works either focus on *unconstrained* Riemannian optimization, or handle constraints via projections. In contrast, we explore constrained g-convex optimization within an abstract RFW framework, by assuming

access to a Riemannian "linear" oracle. Several applications of Riemannian optimization are known, including to matrix factorization on fixed-rank manifolds [57, 59], dictionary learning [16, 56], classical optimization under orthogonality constraints [19], averages of rotation matrices [46], elliptical distributions in statistics [54, 68], and Gaussian mixture models [27]. Explicit theory of g-convexity on HPD matrices is studied in [55]. Additional related work corresponding to the Riemannian mean of HPD matrices is discussed in Sect. 4.

2 Background

We begin by noting some background and notation from Riemannian geometry. For a deeper treatment we refer the reader to [15, 31].

A smooth *manifold* $\mathcal{M}$ is a locally Euclidean space equipped with a differential structure. At any point $x \in \mathcal{M}$, the set of tangent vectors forms the *tangent space* $T_x\mathcal{M}$. Our focus is on *Riemannian manifolds*, i.e., smooth manifolds with a smoothly varying inner product $\langle \xi,\ \eta \rangle_x$ defined on the $T_x\mathcal{M}$ at each point $x \in \mathcal{M}$. We write $\|\xi\|_x := \sqrt{\langle \xi,\ \xi \rangle_x}$ for $\xi \in T_x\mathcal{M}$; for brevity, we will drop the subscript on the norm whenever the associated tangent space is clear from context. Furthermore, we assume $\mathcal{M}$ to be *complete*, which ensures that the following map is defined on the whole tangent space: We define the exponential map as a mapping from $T_x\mathcal{M}$ to $\mathcal{M}$ by $\operatorname{Exp}_x : T_x\mathcal{M} \to \mathcal{M}$ such that $y = \operatorname{Exp}_x(g_x) \in \mathcal{M}$ along a geodesic $\gamma : [0, 1] \to \mathcal{M}$ with $\gamma(0) = x$, $\gamma(1) = y$ and $\dot{\gamma}(0) = g_x \in T_x\mathcal{M}$. We can define an *inverse* exponential map $\operatorname{Exp}_x^{-1} : \mathcal{M} \to T_x\mathcal{M}$ as a diffeomorphism from the neighborhood of $x \in \mathcal{M}$ onto the neighborhood of $0 \in T_x\mathcal{M}$ with $\operatorname{Exp}_x^{-1}(x) = 0$. Note, that the completeness of $\mathcal{M}$ ensures that both maps are well-defined.

Since tangent spaces are local notions, one cannot directly compare vectors lying in different tangent spaces. To tackle this issue, we use the concept of *parallel transport*: the idea is to transport (map) a tangent vector along a geodesic to the respective other tangent space. More precisely, let $x, y \in \mathcal{M}$ with $x \neq y$. We transport $g_x \in T_x\mathcal{M}$ along a geodesic γ (where $\gamma(0) = x$ and $\gamma(1) = y$) to the tangent space $T_y\mathcal{M}$; we denote this by $\Gamma_x^y g_x$. Importantly, the inner product on the tangent spaces is preserved under parallel transport, so that $\langle \xi_x,\ \eta_x \rangle_x = \left\langle \Gamma_x^y \xi_x,\ \Gamma_x^y \eta_x \right\rangle_y$, where $\xi_x, \eta_x \in T_x\mathcal{M}$, while $\langle \cdot,\ \cdot \rangle_x$ and $\langle \cdot,\ \cdot \rangle_y$ are the respective inner products.

2.1 Gradients, convexity, smoothness

Recall that the *Riemannian gradient* $\operatorname{grad} \phi(x)$ is the unique vector in $T_x\mathcal{M}$ such that the directional derivative

$$D\phi(x)[v] = \langle \operatorname{grad} \phi(x),\ v \rangle_x\,, \qquad \forall v \in T_x\mathcal{M}.$$

When optimizing functions using gradients, it is useful to impose some added structure. The two main properties that we require are sufficiently smooth gradients and geodesic

convexity. We say $\phi : \mathcal{M} \to \mathbb{R}$ is *L-smooth*, or that it has *L-Lipschitz* gradients, if

$$\| \operatorname{grad} \phi(y) - \Gamma_x^y \operatorname{grad} \phi(x) \| \leq L d(x, y) \qquad \forall\, x, y \in \mathcal{M}\,, \tag{5}$$

where $d(x, y)$ is the geodesic distance between x and y; equivalently,

$$\phi(y) \leq \phi(x) + \left\langle g_x, \operatorname{Exp}_x^{-1}(y) \right\rangle_x + \tfrac{L}{2} d^2(x, y) \qquad \forall x, y \in \mathcal{M};\ g_x \in T_x\mathcal{M}\,. \tag{6}$$

We say $\phi : \mathcal{M} \to \mathbb{R}$ is *geodesically convex* (*g-convex*) if

$$\phi(y) \geq \phi(x) + \left\langle g_x, \operatorname{Exp}_x^{-1}(y) \right\rangle_x \qquad \forall x, y \in \mathcal{M};\ g_x \in T_x\mathcal{M}\,, \tag{7}$$

and call it μ-*strongly g-convex* ($\mu \geq 0$) if

$$\phi(y) \geq \phi(x) + \left\langle g_x, \operatorname{Exp}_x^{-1}(y) \right\rangle_x + \tfrac{\mu}{2} d^2(x, y) \qquad \forall x, y \in \mathcal{M};\ g_x \in T_x\mathcal{M}\,. \tag{8}$$

The following observation underscores the reason why g-convexity is a valuable geometric property for optimization.

Proposition 1 (Optimality) *Let $x^* \in \mathcal{X} \subset \mathcal{M}$ be a local optimum for* (1)*. Then, x^* is globally optimal, and $\left\langle \operatorname{grad} \phi(x^*), \operatorname{Exp}_{x^*}^{-1}(y) \right\rangle_{x^*} \geq 0$ for all $y \in \mathcal{X}$.*

2.2 Projection-free vs. Projection-based methods

The growing body of literature on Riemannian optimization considers mostly projection-based methods, such as *Riemannian Gradient Decent* (RGD) or *Riemannian Steepest Decent* (RSD) [1]. Such methods and their convergence guarantees typically require Lipschitz assumptions. However, the objectives of many classic optimization and machine learning tasks are not Lipschitz on the whole manifold. In such cases, an additional compactness argument is required. However, in projection-based methods, the typically used retraction back onto the manifold may not be guaranteed to land in this compact set. Thus, in each iteration, an additional and potentially expensive projection step is needed to ensure that the update remains in the compact region where the gradient is Lipschitz. On the other hand, projection-free methods, such as FW, bypass this issue, because their update is guaranteed to stay within the compact feasible region. Importantly, in some problems, the Riemannian linear oracle at the heart of FW can be less expensive than computing a projection back onto the compact set. A detailed numerical study comparing the complexity of projections with that of computing linear minimizers in the Euclidean case can be found in [18]. The efficiency of linear minimizers is especially significant for the applications highlighted in this paper, where the linear oracle even admits a closed form solution (see Sect. 4.1).

2.3 Constrained optimization in Riemannian space

A large body of literature has considered the problem of translating a constrained Euclidean optimization problem into an *unconstrained* Riemannian problem, by encoding the primary constraint in the manifold structure. However, often a problem has additional constraints, requiring a Riemannian approach to constrained optimization. We list below notable examples, including those that will be covered in the application section of the paper.

Examples on the manifold of positive definite matrices include the computation of Riemannian centroids (with interval constraints, see Sect. 4.2.1) and learning determinantal point processes (with interval constraints [44]). A related problem is that of computing Wasserstein-Barycenters on the Bures manifold (wirh interval constraints, see Sect. 4.3.1). The k-means clustering algorithm corresponds to an optimization task on the Stiefel manifold with equality and inequality constraints [14]. Non-negative PCA can be computed on the sphere with equality constraints [47]. The synchronization of data matrices can be written as an optimization task on the manifold of the orthogonal group with a determinant constraint (see [61, Sect. 5]). Computing a minimum balanced cut for graph bisection can be computed on the Oblique manifold with quadratic equality constraints [41].

3 Riemannian Frank-wolfe

The condition $\left\langle \operatorname{grad} \phi(x^*),\ \operatorname{Exp}_{x^*}^{-1}(y) \right\rangle_{x^*} \geq 0$ for all $y \in \mathcal{X}$ in Proposition 1 lies at the heart of Frank-Wolfe (also known as "conditional gradient") methods. In particular, if this condition is not satisfied, then there must be a *feasible descent direction* — FW schemes seek such a direction and update their iterates [20, 29]. This high-level idea is equally valid in the Riemannian setting. Algorithm 1 recalls the basic (Euclidean) FW method, which solves $\min_{x \in \mathcal{X}} \phi(x)$, and Algorithm 2 introduces its Riemannian version (RFW), obtained by simply replacing Euclidean objects with their Riemannian counterparts. In the following, $\langle \cdot,\ \cdot \rangle$ will denote $\langle \cdot,\ \cdot \rangle_{x_k}$ unless otherwise specified.

Algorithm 1 Euclidean Frank-Wolfe without line-search

1: Initialize with a feasible point $x_0 \in \mathcal{X} \subset \mathbb{R}^n$
2: **for** $k = 0, 1, \ldots$ **do**
3: Compute $z_k \leftarrow \operatorname*{argmin}_{z \in \mathcal{X}} \langle \nabla \phi(x_k),\ z - x_k \rangle$
4: Let $s_k \leftarrow \frac{2}{k+2}$
5: Update $x_{k+1} \leftarrow (1 - s_k) x_k + s_k z_k$
6: **end for**

Notice that to implement Algorithm 1, $\mathcal{X}$ must be compact and convex. Convexity ensures that after the update in Step 5, x_{k+1} remains feasible, while compactness ensures that the *linear oracle* in Step 3 has a solution. To obtain RFW, we first replace

the linear oracle (Step 3 in Algorithm 1) with the Riemannian "linear oracle":

$$\min_{z \in \mathcal{X}} \quad \left\langle \operatorname{grad} \phi(x_k),\ \operatorname{Exp}_{x_k}^{-1}(z) \right\rangle, \tag{9}$$

where now $\mathcal{X}$ is assumed to be a compact g-convex set. Similarly, observe that Step 5 of Algorithm 1 updates the current iterate x_k along a straight line joining x_k with z_k. Thus, by analogy, we replace this step by moving x_k along a geodesic joining x_k with z_k. The resulting RFW algorithm is presented as Alg 2.

Algorithm 2 Riemannian Frank-Wolfe (RFW) for g-convex optimization

1: Initialize $x_0 \in \mathcal{X} \subseteq \mathcal{M}$; assume access to the geodesic map $\gamma : [0, 1] \to \mathcal{M}$
2: **for** $k = 0, 1, \ldots$ **do**
3: $\quad z_k \leftarrow \operatorname*{argmin}_{z \in \mathcal{X}} \left\langle \operatorname{grad} \phi(x_k),\ \operatorname{Exp}_{x_k}^{-1}(z) \right\rangle$
4: $\quad$ Let $s_k \leftarrow \frac{2}{k+2}$
5: $\quad x_{k+1} \leftarrow \gamma(s_k)$, where $\gamma(0) = x_k$ and $\gamma(1) = z_k$
6: **end for**

While we obtained Algorithm 2 purely by analogy, we must still show that this analogy results in a valid algorithm. In particular, we need to show that Algorithm 2 converges to a solution of (1). We will in fact prove a stronger result that RFW converges globally at the rate $O(1/k)$, i.e., $\phi(x_k) - \phi(x^*) = O(1/k)$, which matches the rate of the Euclidean FW method.

3.1 Convergence analysis

We make the following smoothness assumption:

Assumption 1 *(Smoothness)* The objective ϕ has a locally Lipschitz continuous gradient on $\mathcal{X}$, that is, there exists a constant L such that for all $x, y \in \mathcal{X}$ we have

$$\|\operatorname{grad} \phi(y) - \Gamma_x^y \operatorname{grad} \phi(x)\| \leq L\, d(x, y). \tag{10}$$

Next, we introduce a quantity that will play a central role in the convergence rate of RFW, namely the *curvature constant*

$$M_\phi := \sup_{x,y,z \in \mathcal{X}} \frac{2}{\eta^2} \left[\phi(y) - \phi(x) - \left\langle \operatorname{grad} \phi(x),\ \operatorname{Exp}_x^{-1}(y) \right\rangle \right]. \tag{11}$$

An analogous quantity is used in the analysis of the Euclidean FW [29]. In the following we adapt proof techniques from [29] to the Riemannian setting. Here and in the following, $y = \gamma(\eta)$ for some $\eta \in [0, 1]$ and a geodesic map $\gamma : [0, 1] \to \mathcal{M}$ with $\gamma(0) = x$ and $\gamma(1) = z$ (denoted in the following as γ_{xz}). Lemma 1 relates the curvature constant (11) to the Lipschitz constant L.

Lemma 1 *Let $\phi : \mathcal{M} \to \mathbb{R}$ be L-smooth on $\mathcal{X}$, and let* $\operatorname{diam}(\mathcal{X}) := \sup_{x,y\in\mathcal{X}} d(x, y)$. *Then, the curvature constant M_ϕ satisfies the bound $M_\phi \leq L \operatorname{diam}(\mathcal{X})^2$.*

Proof Let $x, z \in \mathcal{X}$ and $\eta \in (0, 1)$; let $y = \gamma_{xz}(\eta)$ be a point on the geodesic joining x with z. This implies $\frac{1}{\eta^2} d(x, y)^2 = d(x, z)^2$. From (6) we know that

$$\left|\phi(y) - \phi(x) - \langle \operatorname{grad} \phi(x), \operatorname{Exp}_x^{-1}(y)\rangle\right|^2 \leq \frac{L}{2} d(x, y)^2$$

whereupon using the definition of the curvature constant we obtain

$$M_\phi \leq \sup \frac{2}{\eta^2} \frac{L}{2} d(x, y)^2 = \sup L\, d(x, z)^2 \leq L \cdot \operatorname{diam}(\mathcal{X})^2 \,. \tag{12}$$

□

We note below an analog of the Lipschitz inequality (6) using the constant M_ϕ.

Lemma 2 (Lipschitz) *Let $x, y, z \in \mathcal{X}$ and $\eta \in [0, 1]$ with $y = \gamma_{xz}(\eta)$. Then,*

$$\phi(y) \leq \phi(x) + \eta \left\langle \operatorname{grad}\ \phi(x),\ \operatorname{Exp}_x^{-1}(z) \right\rangle + \tfrac{1}{2}\eta^2 M_\phi \,.$$

Proof From definition (11) of the constant M_ϕ we see that

$$M_\phi \geq \tfrac{2}{\eta^2} \left(\phi(y) - \phi(x) - \left\langle \operatorname{grad}\ \phi(x),\ \operatorname{Exp}_x^{-1}(y) \right\rangle \right),$$

which we can rewrite as

$$\phi(y) \leq \phi(x) + \left\langle \operatorname{grad}\ \phi(x),\ \operatorname{Exp}_x^{-1}(y) \right\rangle + \tfrac{1}{2}\eta^2 M_\phi. \tag{13}$$

Furthermore, since $y = \gamma_{xz}(\eta)$, we have $\operatorname{Exp}_x^{-1}(y) = \eta \operatorname{Exp}_x^{-1}(z)$, and therefore

$$\left\langle \operatorname{grad} \phi(x),\ \operatorname{Exp}_x^{-1}(y) \right\rangle = \left\langle \operatorname{grad} \phi(x),\ \eta \operatorname{Exp}_x^{-1}(z) \right\rangle = \eta \left\langle \operatorname{grad} \phi(x),\ \operatorname{Exp}_x^{-1}(z) \right\rangle.$$

Plugging this equation into (13) the claim follows. □

We need one more technical lemma (easily verified by a quick induction).

Lemma 3 (Stepsize for RFW) *Let $(a_k)_{k\in I}$ be a nonnegative sequence that fulfills*

$$a_{k+1} \leq (1 - s_k) a_k + \tfrac{1}{2} s_k^2 M_\phi. \tag{14}$$

If $s_k = \frac{2}{(k+2)}$, then, $a_k \leq \frac{2M_\phi}{(k+2)}$.

We are now ready to state our first main convergence result, Theorem 1 that establishes a *global* iteration complexity for RFW.

Theorem 1 (Rate) *Let $s_k = \frac{2}{k+2}$, and let X^* be a minimum of ϕ. Then, the sequence of iterates X_k generated by Algorithm 2 satisfies $\phi(X_k) - \phi(X^*) = O(1/k)$.*

Proof The proof of this claim is straightforward; indeed

$$\begin{aligned}
&\phi(X_{k+1}) - \phi(X^*) \\
&\quad \leq \phi(X_k) - \phi(X^*) + s_k \left\langle \operatorname{grad} \phi(X_k),\, \operatorname{Exp}_{X_k}^{-1}(Z_k) \right\rangle + \tfrac{1}{2} s_k^2 M_\phi \\
&\quad \leq \phi(X_k) - \phi(X^*) + s_k \left\langle \operatorname{grad} \phi(X_k),\, \operatorname{Exp}_{X_k}^{-1}(X^*) \right\rangle + \tfrac{1}{2} s_k^2 M_\phi \\
&\quad \leq \phi(X_k) - \phi(X^*) - s_k(\phi(X_k) - \phi(X^*)) + \tfrac{1}{2} s_k^2 M_\phi \\
&\quad = (1 - s_k)(\phi(X_k) - \phi(X^*)) + \tfrac{1}{2} s_k^2 M_\phi,
\end{aligned}$$

where the first inequality follows from Lemma 2, while the second one from Z_k being an argmin obtained in Step 3 of the algorithm. The third inequality follows from g-convexity of ϕ. Setting $a_k = \phi(X_k) - \phi(X^*)$ in Lemma 3 we immediately obtain

$$\phi(X_k) - \phi(X^*) \leq \frac{2M_\phi}{k+2}, \qquad k \geq 0,$$

which is the desired $O(1/k)$ convergence rate. □

Theorem 1 provides a global sublinear convergence rate for RFW. Typically, FW methods trade off their simplicity for this slower convergence rate, and even for smooth strongly convex objectives they do not attain linear convergence rates [29]. We study in Sect. 3.2 a setting that permits RFW to attain a linear rate of convergence.

3.2 Linear convergence of RFW

In general, the sublinear convergence rate that we derived in the previous section is best-possible for Frank-Wolfe methods. This is due to the following phenomenon, which has been studied extensively in the Euclidean setting [13, 62]: If the optimum lies on the boundary of the constraint set $\mathcal{X}$, then the FW updates "zig-zag", resulting in a slower convergence. In this case, the upper bound on the global convergence rate is tight. If, however, the optimum lies in the strict interior of the constraint set, Euclidean FW is known to converge at a linear rate [22, 23]. Remarkably, under a similar assumption, RFW also displays global linear convergence, which we will formally prove below (Theorem 2). Notably, for the special case of the *geometric matrix mean* that we analyze in the next section, this strict interiority assumption will always be valid, provided that not all the matrices are the same.

3.2.1 Linear convergence under strict interior assumption

We use a Riemannian extension to the well-known Polyak-Łojasiewicz (PL) inequality [42, 49], which we define below. Consider the minimization

$$\min_{x \in \mathcal{M}} f(x),$$

and let f^* be the optimal function value. We say that f satisfies the PL inequality if for some $\mu > 0$,

$$\tfrac{1}{2}\|\operatorname{grad} f(x)\|^2 \geq \mu\left(f(x) - f^*\right) \quad \forall x, y \in \mathcal{M}. \tag{15}$$

Inequality (15) is weaker than strong convexity (and is in fact implied by it). It has been widely used for establishing linear convergence rates of gradient-based methods; see [33] for several (Euclidean) examples, and [65] for a Riemannian example. We will make use of inequality (15) for obtaining linear convergence of RFW, by combining it with a strict interiority condition on the minimum.

Theorem 2 (Linear convergence RFW) *Suppose that ϕ is strongly g-convex with constant μ and that its minimum lies in a ball of radius r that strictly inside the constraint set $\mathcal{X}$. Define $\Delta_k := \phi(X_k) - \phi(X^*)$ and let the step-size $s_k = \frac{r\sqrt{\mu\Delta_k}}{\sqrt{2}M_\phi}$. Then,* RFW *converges linearly since it satisfies*

$$\Delta_{k+1} \leq \left(1 - \frac{r^2\mu}{4M_\phi}\right)\Delta_k.$$

Proof Let $\mathcal{B}_r(X^*) \subset \mathcal{X}$ be a ball of radius r containing the optimum. Let

$$W_k := \operatorname{argmax}_{\substack{\xi \in \mathcal{T}_{X_k} \\ \|\xi\| \leq 1}} \langle \xi,\ \operatorname{grad}\phi(X_k)\rangle$$

be the direction of steepest descent in the tangent space $\mathcal{T}_{X_k}$. The point $P_k = \operatorname{Exp}_{X_k}(rW_k)$ lies in $\mathcal{X}$. Consider now the following inequality

$$\left\langle -\operatorname{Exp}_{X_k}^{-1}(P_k),\ \operatorname{grad}\phi(X_k)\right\rangle = -\langle \operatorname{grad}\phi(X_k),\ rW_k\rangle = -r\|\operatorname{grad}\phi(X_k)\|, \tag{16}$$

which follows upon using the definition of W_k. Thus, we have the bound

$$\begin{aligned}\Delta_{k+1} &\leq \Delta_k + s_k\left\langle \operatorname{grad}\phi(X_k),\ \operatorname{Exp}_{X_k}^{-1}(X_{k+1})\right\rangle + \tfrac{1}{2}s_k^2 M_\phi \\ &\leq \Delta_k - s_k r\|\operatorname{grad}\phi(X_k)\| + \tfrac{1}{2}s_k^2 M_\phi \\ &\leq \Delta_k - s_k r\sqrt{2\mu}\sqrt{\Delta_k} + \tfrac{1}{2}s_k^2 M_\phi,\end{aligned}$$

where the first inequality follows from the Lipschitz-bound (Lemma 2), the second one from (16), and the third one from the PL inequality (which, in turn holds due to the μ-strong g-convexity of ϕ). Now setting the step size $s_k = \frac{r\sqrt{\mu\Delta_k}}{\sqrt{2}M_\phi}$, we obtain

$$\Delta_{k+1} \leq \left(1 - \frac{r^2\mu}{4M_\phi}\right)\Delta_k,$$

which delivers the claimed linear convergence rate. □

Theorem 2 provides a setting where RFW can converge fast, however, it uses step sizes s_k that require knowing $\phi(X^*)$[1]; in case the optimal value is not available, we can use a worse value, which will still yield the desired inequality.

Remark 1 *(Necessity of strict interior assumption)* For optimization tasks with polytope constraints that do not fulfill a strict interior assumption as the one described above, several EFW variants achieve linear convergence [36]. Notable examples include *Away-step* FW [23, 62], *Pairwise* FW [45] and *Fully-corrective* FW [26]. One may ask, whether these variants can be generalized to the Riemannian case. The first difficulty lies in finding a Riemannian equivalent of the polytope constraint set, which is defined as the convex hull of a finite set of vectors (*atoms*). Naturally, we could consider the convex hull of a finite set of points on a manifold, which is the intersection of all convex set that contain them. Unfortunately, such a set is in general not compact – compactness is only guaranteed for Hadamard manifolds under additional, restrictive conditions [39]. Even in this special case, ensuring that the resulting constraint sets have sufficiently "good" geometry is not straightforward.

3.3 RFW for nonconvex problems

Finally, we want to consider the case where ϕ in Eq. 1 may be nonconvex. In this case, we cannot hope to find the global minimum with first-order methods, such as RFW. However, we can compute first-order critical point via RFW. For the setup and analysis, we follow the Euclidean case [35].

We first introduce the *Frank-Wolfe gap* as a criterion for evaluating convergence rates. For $X \in \mathcal{X}$, we write

$$G(X) := \max_{Z \in \mathcal{X}} \left\langle \operatorname{Exp}_X^{-1}(Z),\ -\operatorname{grad}\ \phi(X) \right\rangle . \tag{17}$$

With this, we can show the following sublinear convergence guarantee:

Theorem 3 (Rate (nonconvex case)) *Let* $\tilde{G}_k := \min_{0 \le k \le K} G(X_k)$ *(where* $G(X_k)$ *denotes the Frank-Wolfe gap at* X_k*). After* K *iterations of Algorithm 2, we have* $\tilde{G}_k \le \frac{\max\{2h_0, M_\phi\}}{\sqrt{K+1}}$.

The proof utilizes techniques similar to those in [35, Theorem 1].

4 Specializing RFW for HPD matrices

In this section we study a concrete setting for RFW, namely, a class of g-convex optimization problems with Hermitian positive definite (HPD) matrices. We will show that the Riemannian linear oracle (9) admits an efficient solution for this class of problems, thereby allowing an efficient implementation of Algorithm 2. The concrete

[1] This step size choice is reminiscent of the so-called "Polyak stepsizes" used in the convergence analysis of subgradient methods [50].

class of problems that we consider is the following:

$$\min_{X \in \mathcal{X} \subseteq \mathbb{P}_d} \quad \phi(X), \qquad \text{where } \mathcal{X} := \{X \in \mathbb{P}_d \mid L \preceq X \preceq U\}, \tag{18}$$

where ϕ is a g-convex function and $\mathcal{X}$ is a "positive definite interval" (which is easily seen to be a g-convex set). Note that the set $\mathcal{X}$ actually does not admit an easy projection for matrices. Problem (18) captures several g-convex optimization problems, of which perhaps the best known is the task of computing the matrix geometric mean (also known as the the *Riemannian centroid* or *Karcher mean*)—see Sect. 4.2.1.

We briefly recall some facts about the Riemannian geometry of HPD matrices below. For a comprehensive overview, see, e.g., [5, Chapter 6]. We denote by $\mathbb{H}_d$ the set of $d \times d$ Hermitian matrices. The most common Riemannian geometry on $\mathbb{P}_d$ is induced by

$$\langle A, B \rangle_X := \operatorname{tr}(X^{-1} A X^{-1} B), \quad \text{where } A, B \in \mathbb{H}_d. \tag{19}$$

This metric induces the geodesic $\gamma : [0, 1] \to \mathbb{P}_d$ between $X, Y \in \mathbb{P}_d$ given by

$$\gamma(t) := X^{1/2}(X^{-1/2} Y X^{-1/2})^t X^{1/2}, \qquad t \in [0, 1]. \tag{20}$$

The corresponding *Riemannian distance* is

$$d(X, Y) := \|\log(X^{-1/2} Y X^{-1/2})\|_{\mathrm{F}}, \qquad X, Y \in \mathbb{P}_d. \tag{21}$$

The Riemannian gradient $\operatorname{grad} \phi$ is obtained from the Euclidean one ($\nabla \phi$) as follows

$$\operatorname{grad} \phi(X) = X \nabla^{\mathbb{H}} \phi(X) X \,, \tag{22}$$

where $\nabla^{\mathbb{H}} \phi(X) := \frac{1}{2}\big(\nabla \phi(X) + (\nabla \phi(X))^*\big)$ denotes the (Hermitian) symmetrization of the gradient. The exponential map and its inverse at a point $P \in Pd$ are given by

$$\begin{aligned} \operatorname{Exp}_X(A) &= X^{1/2} \exp(X^{-1/2} A X^{-1/2}) X^{1/2}, \qquad A \in T_X \mathbb{P}_d \equiv \mathbb{H}_d, \\ \operatorname{Exp}_X^{-1}(Y) &= X^{1/2} \log(X^{-1/2} Y X^{-1/2}) X^{1/2}, \qquad X, Y \in \mathbb{P}_d, \end{aligned}$$

where $\exp(\cdot)$ and $\log(\cdot)$ denote the matrix exponential and logarithm, respectively. Observe that using (22) we obtain the identity

$$\Big\langle \operatorname{grad} \phi(X),\ \operatorname{Exp}_X^{-1}(Y) \Big\rangle_X = \Big\langle X^{1/2} \nabla^{\mathbb{H}} \phi(X) X^{1/2},\ \log(X^{-1/2} Y X^{-1/2}) \Big\rangle. \tag{23}$$

With these details, Algorithm 2 can almost be applied to (18) — the most crucial remaining component is the Riemannian linear oracle, which we now describe.

4.1 Solving the Riemannian linear oracle

For solving (18), the Riemannian linear oracle (see (9)) requires solving

$$\min_{L \preceq Z \preceq U} \quad \left\langle X_k^{1/2} \nabla^{\mathbb{H}} \phi(X_k) X_k^{1/2},\ \log\left(X_k^{-1/2} Z X_k^{-1/2}\right) \right\rangle. \tag{24}$$

Problem (24) is a non-convex optimization problem over HPD matrices. However, remarkably, it turns out to have a closed form solution. Theorem 4 presents this solution and is our main technical result for Sect. 4.

Theorem 4 *Let $L, U \in \mathbb{P}_d$ such that $L \prec U$. Let $S \in \mathbb{H}_d$ and $X \in \mathbb{P}_d$ be arbitrary. Then, the solution to the optimization problem*

$$\min_{L \preceq Z \preceq U} \quad \operatorname{tr}(S \log(XZX)), \tag{25}$$

is given by $Z = X^{-1} Q \left(P^* [-\operatorname{sgn}(D)]_+ P + \hat{L}\right) Q^* X^{-1}$, where $S = QDQ^*$ is a diagonalization of S, $\hat{U} - \hat{L} = P^* P$ with $\hat{L} = Q^* X L X Q$ and $\hat{U} = Q^* X U X Q$.

For the proof of Theorem 4, we need a fundamental lemma about eigenvalues of Hermitian matrices (Lemmas 4). First, we need to introduce some additional notation. For $x \in \mathbb{R}^d$, let $x^{\downarrow} = \left(x_1^{\downarrow}, \ldots, x_d^{\downarrow}\right)$ denote the vector with entries of x in decreasing order, i.e., $x_1^{\downarrow} \geq \ldots \geq x_d^{\downarrow}$. For $x, y \in \mathbb{R}^d$ we say that x is weakly majorized by y ($x \prec_w y$), if $\sum_{i=1}^k x_i^{\downarrow} \leq \sum_{i=1}^k y_i^{\downarrow}$ for $1 \leq k \leq d$. We can now recall the following lemma on eigenvalues of Hermitian matrices, which can be found, e.g., in [4, Problem III.6.14]:

Lemma 4 ([4]) *Let X, Y be HPD matrices. Then*

$$\lambda^{\downarrow}(X) \cdot \lambda^{\uparrow}(Y) \prec_w \lambda(XY) \prec_w \lambda^{\downarrow}(X) \cdot \lambda^{\downarrow}(Y),$$

where $\lambda^{\downarrow}(X)$ ($\lambda^{\uparrow}$) denote eigenvalues of X arranged in decreasing (increasing) order, $\prec_w$ denotes the weak majorization order and $\cdot$ denotes the elementwise product. If X, Y are Hermitian, we have

$$\left\langle \lambda^{\downarrow}(X),\ \lambda^{\uparrow}(Y) \right\rangle \leq \operatorname{tr}(XY) \leq \left\langle \lambda^{\downarrow}(X),\ \lambda^{\downarrow}(Y) \right\rangle,$$

with equality, if the the product XY is symmetric.

Remark 2 We recall a well-known fact on solving the trace minimization problem $\min_Y \operatorname{tr} XY$. Diagonalizing X, Y allows for rewriting the minimization problem as $\min_S \operatorname{tr} DQ^T SQ$, where D, S are diagonal and Q orthogonal. To see for which S the minimum is attained, note that $\min \sum_{ij} d_i s_j q_{ij}^2 \geq \min \sum_{ij} d_i s_j r_{ij}$, where (r_{ij}) is doubly-stochastic. By Birkhoff's theorem the minimum occurs for permutation matrices, i.e., $\min \sum_i d_i s_{\sigma(i)}$. By the rearrangement inequality this is minimized, if the order of the indices is reversed.

Proof *(Theorem 4)* First, introduce the variable $Y = XZX$; then (25) becomes

$$\min_{L' \preceq Y \preceq U'} \quad \operatorname{tr}(S \log Y), \tag{26}$$

where the constraints have also been modified to $L' \preceq Y \preceq U'$, where $L' = XLX$ and $U' = XUX$. Diagonalizing S as $S = QDQ^*$, we see that $\operatorname{tr}(S \log Y) = \operatorname{tr}(D \log W)$, where $W = Q^*YQ$. Thus, instead of (26) it suffices to solve

$$\min_{L'' \preceq W \preceq U''} \quad \operatorname{tr}(D \log W), \tag{27}$$

where $L'' = Q^*L'Q$ and $U'' = Q^*U'Q$. We have strict inequality $U'' \succ L''$, thus, our constraints are $0 \prec W - L'' \preceq U'' - L''$, which we may rewrite as $0 \prec R \preceq I$, where $R = (U'' - L'')^{-1/2}(W - L'')(U'' - L'')^{-1/2}$. Notice that

$$W = (U'' - L'')^{1/2} R (U'' - L'')^{1/2} + L''.$$

Thus, problem (27) now turns into

$$\min_{0 \prec R \preceq I} \quad \operatorname{tr}(D \log(P^*RP + L'')) \ , \tag{28}$$

where $U'' - L'' = P^*P$. We want to construct an R that attains the minimum. Note that using Lemma 4 we see that

$$\operatorname{tr}(D \log(P^*RP + L'')) \geq \lambda^{\uparrow}(D) \cdot \lambda^{\downarrow}\left(\log(P^*RP + L'')\right) \ .$$

Remark 2 ensures that the minimum attains the lower bound, i.e., the minimum is attained by matching the eigenvalues of D and $\log(P^*RP + L'')$ in reverse orders. Note that the matrix logarithm $\log(\cdot)$ and the map $R \mapsto P^*RP + L''$ are operator monotone. Now, without loss of generality, assume that R is diagonal and recall that, by construction $0 \prec R \preceq I$. Consider

$$r_{ii} = \begin{cases} 0 & d_{ii} \geq 0 \\ 1 & d_{ii} < 0. \end{cases} \tag{29}$$

This ensures that

1. if $d_{ii} > 0$, the corresponding element of $\lambda^{\downarrow}(\log(P^*RP + L''))$ is minimized,
2. if $d_{ii} \leq 0$, the corresponding element of $\lambda^{\downarrow}(\log(P^*RP + L''))$ is maximized.

With that, the minimum of the trace is attained. Thus, we see that

$$Y = Q(P^*RP + L'')Q^* = Q(P^*\left[-\operatorname{sgn}(D)\right]_+ P + L'')Q^*$$

and we immediately obtain the optimal $Z = X^{-1}YX^{-1}$. □

Remark 3 Computing the optimal direction Z_k takes one Cholesky factorization, two matrix square roots (Schur method), eight matrix multiplications, and one eigenvalue decomposition. This gives a complexity of $O(N^3)$. On our machines, we report $\approx \frac{1}{3}N^3 + 2 \times 28.\bar{3}N^3 + 8 \times 2N^3 + 20N^3 \approx 93N^3$.

4.2 Application to the Riemannian mean

4.2.1 Computing the Riemannian mean

Statistical inference frequently involves computing averages of input quantities. Typically encountered data lies in Euclidean spaces where arithmetic means are the "natural" notions of averaging. However, the Euclidean setting is not always the most natural or efficient way to represent data. Many applications involve non-Euclidean data such as graphs, strings, or matrices [38, 48]. In such applications, it is often beneficial to represent the data in its natural space and adapt classical tools to the specific setting. In other cases, a problem might be very hard to solve in Euclidean space but may become more accessible when viewed through a different geometric lens. Specifically, with a suitable choice of the Riemannian metric, a Euclidean non-convex problem may be geodesically convex. Hence, we can give *global* convergence guarantees for solving such problems with Riemannian optimization methods.

This section considers one of these cases, namely the problem of determining the *geometric matrix mean* (*Karcher mean problem*). While there exists an intuitive notion for the geometric mean of sets of positive real numbers, this notion does not immediately generalize to sets of positive definite matrices due to the lack of commutativity on matrix spaces. Over a collection of Hermitian, positive definite matrices the geometric mean can be viewed as the geometric optimization problem

$$G := \underset{X \succ 0}{\operatorname{argmin}} \quad \left[\phi(X) = \sum\nolimits_{i=1}^{m} w_i \delta_R^2(X, A_i) \right], \tag{30}$$

where δ_R denotes the Riemannian metric. In a Euclidean setting, the problem is *non-convex*. However, one can view Hermitian, positive matrices as points on a Riemannian manifold and compute the geometric mean as the Riemannian centroid. The corresponding optimization problem (Eq. 30) is *geodesically convex* [51]. In this section, we look at the problem through both geometric lenses and provide efficient algorithmic solutions while illustrating the benefits of switching geometric lenses in geometric optimization problems.

There exists a large body of work on the problem of computing the geometric matrix means [30]. Classic algorithms like *Newton's method* or *Gradient Decent* (*GD*) have been successfully applied to the problem. Standard toolboxes implement efficient variations of *GD* like *Steppest Decent* or *Conjugate Gradient* (*Manopt* [11]) or Richardson-like linear-gradient decent (*Matrix Means Toolbox* [9]). Recent work by Yuan et al. [64] analyzes condition numbers of Hessians in Riemannian and Euclidean steepest-decent approaches that provide theoretical arguments for the good performance of Riemannian approaches.

Recently, T. Zhang developed a majorization-minimization approach with asymptotic linear convergence [67]. In this section, we use the Riemannian Frank-Wolfe algorithm for computing the geometric matrix mean. Here, we exploit the strong geodesic convexity of the problem to achieve global linear convergence. To complement this analysis, we show that recent advances in nonconvex analysis [35] can be used to develop a Frank-Wolfe scheme (EFW) for the nonconvex Euclidean case (see Appendix 1).

In the simple case of two PSD matrices, X and Y, one can view the geometric mean as their metric midpoint computed by [34]

$$G(X, Y) = X \#_t Y = X^{\frac{1}{2}} \left(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}} \right)^t X^{\frac{1}{2}} . \tag{31}$$

More generally, for a collection of M matrices, the geometric mean can be seen as a minimization problem of the sum of squares of distances [6],

$$G(A_1, \ldots, A_M) = \underset{X \succ 0}{\operatorname{argmin}} \sum_{i=1}^{M} \delta_R^2(X, A_i) , \tag{32}$$

where $\delta_R(X, Y) = \|\log(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}})\|_F$ as introduced earlier. Here we consider the more general *weighted* geometric mean:

$$G(A_1, \ldots, A_M) = \underset{X \succ 0}{\operatorname{argmin}} \sum_{i=1}^{M} w_i \delta_R^2(X, A_i) . \tag{33}$$

E. Cartan showed in a Riemannian setting that a global minimum exists, which led to the term *Cartan mean* frequently used in the literature. In this setting, one can view the collection of matrices as points on a Riemannian manifold. H. Karcher associated the minimization problem with that of finding centers of masses on these manifolds [32], hence motivating a second term to describe the geometric matrix mean (*Karcher mean*).

The geometric matrix mean enjoys several key properties. We list below the ones of crucial importance to our paper and refer the reader to [37, 40] for a more extensive list. To state these results, we recall the general form of the two other basic means of operators: the (weighted) harmonic and arithmetic means, denoted by H and A respectively.

$$H := \left(\sum\nolimits_{i=1}^{M} w_i A_i^{-1} \right)^{-1}, \qquad A := \sum\nolimits_{i=1}^{M} w_i A_i .$$

Then, one can show the following well-known operator inequality that relates H, G, and A:

Lemma 5 (Means Inequality, [5]) *Let $A_1, \ldots, A_M > 0$, and let H, G, and A denote their (weighted) harmonic, geometric, and arithmetic means, respectively. Then,*

$$H \preceq G \preceq A \,. \tag{34}$$

The key computational burden of all our algorithms lies in computing the gradient of the objective function (30). A short calculation (see e.g., [5, Ch. 6]) shows that if $f(X) = \delta_R^2(X, A)$, then $\nabla f(X) = X^{-1} \log(XA^{-1})$. Thus, we immediately obtain

$$\nabla \phi(X) = \sum\nolimits_i w_i X^{-1} \log(XA_i^{-1}).$$

4.2.2 Implementation

We compute the *geometric matrix mean* with Algorithm 2. For the PSD manifold, line 3 can be written as

$$Z_k \leftarrow \operatorname*{argmin}_{H \preceq Z \preceq A} \left\langle X_k^{1/2} \nabla \phi(X_k) X_k^{1/2},\ \log\left(X_k^{-1/2} Z X_k^{-1/2}\right) \right\rangle \,. \tag{35}$$

Note that the operator inequality $H \preceq G \preceq A$ given by Lemma 5 plays a crucial role: It shows that the optimal solution lies in a compact set so we may as well impose this compact set as a constraint to the optimization problem (i.e., we set $\mathcal{X} = \{H \preceq Z \preceq A\}$). We implement the linear oracles as discussed above: In the Euclidean case, a closed-form solution is given by

$$Z = H + P^* Q [-\operatorname{sgn}(\Lambda)]_+ Q^* P, \tag{36}$$

where $A - H = P^* P$ and $P \nabla \phi(X_k) P^* = Q \Lambda Q^*$ (see Thm. 7). Analogously, for the Riemannian case, the "linear" oracle

$$\min_{H \preceq Z \preceq A} \quad \langle \nabla \phi(X_k),\ \log(Z) \rangle \,, \tag{37}$$

is well defined and solved by

$$Z = Q \left(P^* [-\operatorname{sgn}(\Lambda)]_+ P + \hat{H} \right) Q^* \,, \tag{38}$$

with $\hat{A} - \hat{H} = P^* P$, $\hat{A} = Q^* A Q$, $\hat{H} = Q^* H Q$ and $\nabla \phi(X_k) = Q \Lambda Q^*$ (see Thm. 4). The resulting Frank-Wolfe method for the geometric matrix mean is given by Algorithm 3.

Algorithm 3 FW for fast Geometric mean (GM)/ Wasserstein mean (WM)

$(A_1, \ldots, A_N)$, $\boldsymbol{w} \in \mathbb{R}^N_+$
$\bar{X} \approx \operatorname{argmin}_{X \succ 0} \sum_i w_i \delta_R^2(X, A_i)$
$\beta = \min_{1 \le i \le N} \lambda_{\min}(A_i)$
for $k = 0, 1, \ldots$ **do**
  Compute gradient.
    GM: $\nabla\phi(X_k) = X_k^{-1}\left(\sum_i w_i \log(X_k A_i^{-1})\right)$
    WM: $\nabla\phi(X_k) = \sum_i w_i \left(I - (A_i X_k)^{-1/2} A_i\right)$
  Compute Z_k:
    GM: $Z_k \leftarrow \operatorname{argmin}_{H \preceq Z \preceq A} \left\langle X_k^{1/2} \nabla\phi(X_k) X_k^{1/2},\ \log(X_k^{-1/2} Z X_k^{-1/2}) \right\rangle$
    WM: $Z_k \leftarrow \operatorname{argmin}_{\beta I \preceq Z \preceq A} \left\langle X_k^{1/2} \nabla\phi(X_k) X_k^{1/2},\ \log(X_k^{-1/2} Z X_k^{-1/2}) \right\rangle$
  Let $\alpha_k \leftarrow \frac{2}{k+2}$.
  Update X:
    $X_{k+1} \leftarrow X_k \#_{\alpha_k} Z_k$
end for
return $\bar{X} = X_k$

4.3 Application to Bures-wasserstein barycenters

4.3.1 Computing Bures-Wasserstein barycenters on the Gaussian density manifold

As a second application of RFW, we consider the computation of Bures-Wasserstein barycenters of multivariate (centered) Gaussians. This application is motivated by optimal transport theory; in particular, the Bures-Wasserstein barycenter is the solution to the multi-marginal transport problem [8, 43]: Let $\{A_i\}_{i=1}^m \in \mathbb{P}_d$ and $w = (w_1, \ldots, w_m)$ a vector of weights ($\sum_i w_i = 1;\ w_i > 0, \forall i$). The minimization task

$$\min_{\substack{X \in \mathbb{P}_d \\ X \succ 0}} \sum_{i=1}^m w_i d_W^2(X, A_i) \tag{39}$$

is called *multi-marginal transport problem*. Its unique minimizer

$$\Omega(w; \{A_i\}) = \operatorname*{argmin}_{X \succ 0} \sum_{i=1}^m w_i d_W^2(X, A_i) \in \mathbb{P}_d$$

is the *Bures-Wasserstein barycenter*. Here, d_W denotes the *Wasserstein distance*

$$d_W(X, Y) = \left[\operatorname{tr}(X + Y) - 2\operatorname{tr}\left(X^{1/2} Y X^{1/2}\right)^{1/2}\right]^{1/2}. \tag{40}$$

To see the connection to optimal transport, note that Eq. 39 corresponds to a least-squares optimization over a set of multivariate center Gaussians with respect to the Wasserstein distance. The Gaussian density manifold is isomorphic to the manifold of symmetric positive definite matrices, which allows for a direct application of RFW and

the setup in the previous section to Eq. 39, albeit with a different set of constraints. In the following, we discuss a suitable set of constraints and adapt RFW for the computation of the Bures-Wasserstein barycenter (short: Wasserstein mean).

First, note that as for the Karcher mean, one can show that the Wasserstein mean of two matrices is given in closed form; namely as

$$X \diamondsuit_t Y = (1-t)^2 X + t^2 Y + t(1-t)\left((XY)^{1/2} + (YX)^{1/2}\right) . \tag{41}$$

However, the computation of the barycenter of m matrices ($m > 2$) requires solving a quadratic optimization problem. Note, that Eq. 41 defines a geodesic map from X to Y. Unfortunately, an analog of the *means inequality* does not hold in the Wasserstein case. However, one can show, that the arithmetic matrix mean always gives an upper bound, providing one-sided constraints [7]:

$$A_i \diamondsuit_t A_j \preceq \frac{A_i + A_j}{2} ,$$

and similarly for the arithmetic mean $A(w; \{A_i\})$ of m matrices. Moreover, one can show that $\alpha I \preceq X$ [8], where α is the minimal eigenvalue over $\{A_i\}_{i=1}^m$, i.e.

$$\alpha = \min_{1 \le j \le m} \lambda_{\min}(A_j) .$$

In summary, this gives the following constraints on the Wasserstein mean:

$$\alpha I \preceq \Omega(w; \{A_i\}) \preceq A(w; \{A_i\}) . \tag{42}$$

Next, we will derive the gradient of the objective function. According to Eq. 39, the objective is given as

$$\phi(X) := \sum\nolimits_j w_j \left[\operatorname{tr}(A_j) + \operatorname{tr}(X) - 2\operatorname{tr}\left(A_j^{1/2} X A_j^{1/2}\right)^{1/2}\right]^{1/2} . \tag{43}$$

Two expressions for the gradient of (43) are derived in the following.

Lemma 6 *Let ϕ be given by (43). Then, its gradient is*

$$\nabla\phi(X, \mathcal{A}) = \sum_j \omega_j \left(I - A_j \# X^{-1}\right) . \tag{44}$$

Proof

$$\begin{aligned}\nabla\phi(X, \mathcal{A}) &= \underbrace{\frac{d}{dX}\sum_j w_j \operatorname{tr} A_j}_{\text{vanishes}} + \frac{d}{dX}\sum_j w_j \left(\operatorname{tr}(X) - 2\operatorname{tr}\left(A_j^{1/2} X A_j^{1/2}\right)^{1/2}\right) \\ &= \sum_j w_j \frac{d}{dX}\operatorname{tr}(X) - \sum_j w_j 2 \frac{d}{dX}\operatorname{tr}\left(A_j^{1/2} X A_j^{1/2}\right)^{1/2} .\end{aligned}$$

Note that $\frac{d}{dX}\operatorname{tr}(X) = I$. Consider

$$\begin{aligned}
2\frac{d}{dX}\operatorname{tr}\left(A^{1/2}XA^{1/2}\right)^{1/2} &\overset{(1)}{=} 2\frac{d}{dX}\left[\left(\left(A^{1/2}XA^{1/2}\right)^{1/2}\right)^{*}\left(\left(A^{1/2}XA^{1/2}\right)^{1/2}\right)\right]^{1/2} \\
&\overset{(2)}{=} 2\frac{d}{dX}\left(A^{1/2}XA^{1/2}\right)^{1/2} \\
&= A^{1/2}\left(A^{1/2}XA^{1/2}\right)^{-1/2}A^{1/2} \\
&= A^{1/2}\left(A^{-1/2}X^{-1}A^{-1/2}\right)^{1/2}A^{1/2} \\
&= A\#X^{-1}\,,
\end{aligned}$$

where (1) follows from the chain rule and (2) from A and X being symmetric and positive definite: If A and X are symmetric and positive definite, then A has a unique positive definite root $A^{1/2}$ and therefore

$$\left(A^{1/2}XA^{1/2}\right)^{*} = \left(A^{1/2*}X^{*}A^{1/2*}\right) = A^{1/2}XA^{1/2}\,.$$

Putting everything together, the desired expression follows as

$$\nabla\phi(X, \mathcal{A}) = \sum_{j}\omega_j\left(I - A_j\#X^{-1}\right)\,. \tag{45}$$

□

4.3.2 Implementation

Given a set of constraints and an expression for the gradient, we can solve Eq. 39 with RFW using the following setup: We write the log-linear oracle as

$$Z_k \in \operatorname*{argmin}_{\alpha I \preceq Z \preceq A} \langle \operatorname{grad}\phi(X), \operatorname{Exp}_X^{-1}(Z)\rangle\,, \tag{46}$$

where $\operatorname{grad}\phi(X)$ is the Riemannian gradient of ϕ and $\operatorname{Exp}_X^{-1}(Z)$ the exponential map with respect to the Riemannian metric. The constraints are given by Eq. 42. Since the constraint set is given by an interval of the form $L \preceq Z \preceq U$, the oracle can be solved in closed form using Theorem 4. The resulting algorithm is given in Algorithm 3.

5 Approximately solving the Riemannian "linear" oracle

In the previous section, we have given examples of applications where the Riemannian "linear" oracle can be solved in closed form – rendering the resulting RFW algorithm into a practical method. Unfortunately, the "linear" oracle is in general a nonconvex subroutine. Therefore, it can be challenging to find efficient solutions for constrained problems in practice.

One remedy for such situations is to solve the "linear" oracle only approximately. In the following, we show that we can recover sublinear convergence rates for RFW, even if we solve the "linear" oracle only approximately. This extension greatly widens the range of possible applications for RFW. For instance, while we currently do not have closed-form solutions for the"linear" oracle in some of the examples in Sect. 2.3, we could find approximate solutions via relaxations or iteratively solving the respective subroutine.

In the following, we say that $Z' \in \mathcal{X}$ is a δ-*approximate linear minimizer*, if

$$\left\langle \text{grad } \phi(X_k),\ \text{Exp}_{X_k}^{-1}(Z') \right\rangle \leq \min_{Z \in \mathcal{X}} \left\langle \text{Exp}_{X_k}^{-1}(Z),\ \text{grad } \phi(X_k) \right\rangle + \frac{1}{2}\delta\eta M_\phi \ .$$

We give the following sublinear convergence guarantee:

Theorem 5 *Let X^* be a minimum of a geodesically convex function ϕ and $\delta \geq 0$ the accuracy to which the "linear" oracle is solved in each round. Then, the sequence of iterates X_k generated by Algorithm 2 satisfies*

$$\phi(X_k) - \phi(X^*) \leq \frac{2M_\phi}{k+2}(1+\delta) \ .$$

The proof relies on adapting proof techniques from [17, 29] to the Riemannian setting. It utilizes the following auxiliary lemma:

Lemma 7 *For a steps size $\eta \in (0, 1)$ and accuracy δ, we have*

$$\phi(X_{k+1}) \leq \phi(X_k) - \eta \left\langle \text{grad } \phi(X_k),\ \text{Exp}_{X_k}(Z') \right\rangle + \frac{1}{2}\eta^2 M_\phi (1+\delta) \ .$$

A proof of the Lemma can be found in Appendix 1. The proof of Theorem 5 follows then from Lemma 3 and 8 similar to Theorem 1.

6 Computational experiments

In this section, we will remark on the implementation of Algorithm 3 and show numerical results for computing Riemannian centroids for different parameter choices. To evaluate the efficiency of our method, we compare the performance of RFW against that of a selection of state-of-the-art methods. Additionally, we use Algorithm 3 to compute Wasserstein barycenters of positive definite matrices. All computational experiments are performed using MATLAB.

6.1 Computational considerations

When implementing the algorithm we can take advantage of the positive definiteness of the input matrices. For example, if using MATLAB, rather than computing

$X^{-1}\log(XA_i^{-1})$, it is more preferable to compute

$$X^{-1/2}\log(X^{1/2}A_i^{-1}X^{1/2})X^{1/2},$$

because both $X^{-1/2}$ and $\log(X^{1/2}A_i^{-1}X^{1/2})$ can be computed by suitable eigendecomposition. In contrast, computing $\log(XA_i^{-1})$ invokes the matrix logarithm (`logm` in MATLAB), which can be much slower.

To save on computation time, we prefer to use a diminishing scalar as the stepsize in Algorithm 3. In principle, this simple stepsize selection could be replaced by a more sophisticated Armijo-like line-search or even exact minimization by solving

$$\alpha_k \leftarrow \operatorname*{argmin}_{\alpha\in[0,1]} \quad \phi(X_k + \alpha_k(Z_k - X_k)) \ . \tag{47}$$

This stepsize tuning may accelerate the convergence speed of the algorithm, but it must be combined with a more computational intensive strategy of "away" steps [23] to obtain a geometric rate of convergence. However, we prefer Algorithm 3 for its simplicity and efficiency.

Theorems 1 and 3 show that Algorithm 2 converges at the global (non-asymptotic) rates $O(1/\epsilon)$ (g-convex RFW) and $O(1/\epsilon^2)$ (nonconvex RFW). However, by further exploiting the simple structure of the constraint set and the "curvature" of the objective function, we might obtain a stronger convergence result.

6.2 Numerical results

We present numerical results for computing the Riemannian and Wasserstein means for sets of positive definite matrices. We test our methods on both well- and ill-conditioned matrices; the generation of sample matrices is described in the appendix. An extended arXiv version of the paper [61] contains additional examples and numerical experiments.

6.2.1 Computing the Riemannian Mean

To test the efficiency of our method, we implement RFW (Algorithm 3) in MATLAB and compare its performance on the problem of computing the Riemannian mean against related state-of-the-art Riemannian optimization methods:

1. *Riemannian L-BFGS* (R- LBFGS[2]) is a quasi-Newton method that iteratively approximates the Hessian for evaluating second-order information [63].
2. *Riemannian Barzilai–Borwein* (BB) is a first-order gradient-based method for constrained and unconstrained optimization. It evaluates second-order information from an approximated Hessian to choose the stepsize [28]. We use the *Manopt* version of RBB.

[2] Also known as LRBFGS.

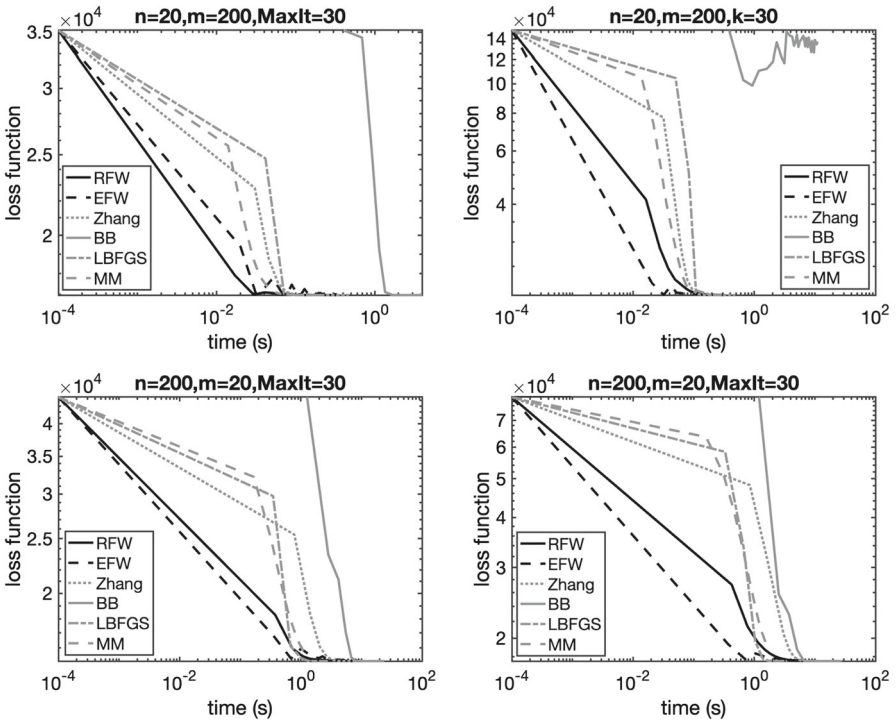

Fig. 1 Performance of EFW and RFW in comparison with state-of-the-art methods for well-conditioned (left) and ill-conditioned inputs (right) of different sizes (n: size of matrices, m: number of matrices, $MaxIt$: maximum number of iterations). All tasks are initialized with the harmonic mean $X_0 = HM$

3. *Matrix Means Toolbox* (MM) [9] is an efficient MATLAB toolbox for matrix optimization. Its implementation of the geometric mean problem uses a Richardson-like iteration of the form

$$X_{k+1} \leftarrow X_k - \alpha X_k \sum_{i=1}^{n} \log(A_i^{-1} X_k) \ ,$$

   with a suitable $\alpha > 0$.
4. ZHANG [67] is a recently published majorization-minimization method for computing the geometric matrix mean.

Note that this selection reflects a broad spectrum of commonly used methods for Riemannian optimization. It ranges from highly specialized approaches that are targeted to the Karcher mean (MM), to more general and versatile methods (e.g., R- LBFGS). A careful evaluation should take both computing time and algorithmic features, such as the number of calls to oracles and loss functions, into account. We perform and further discuss such an evaluation below. For completeness, we also compare against RFW's Euclidean counterpart EFW (Algorithm 4).

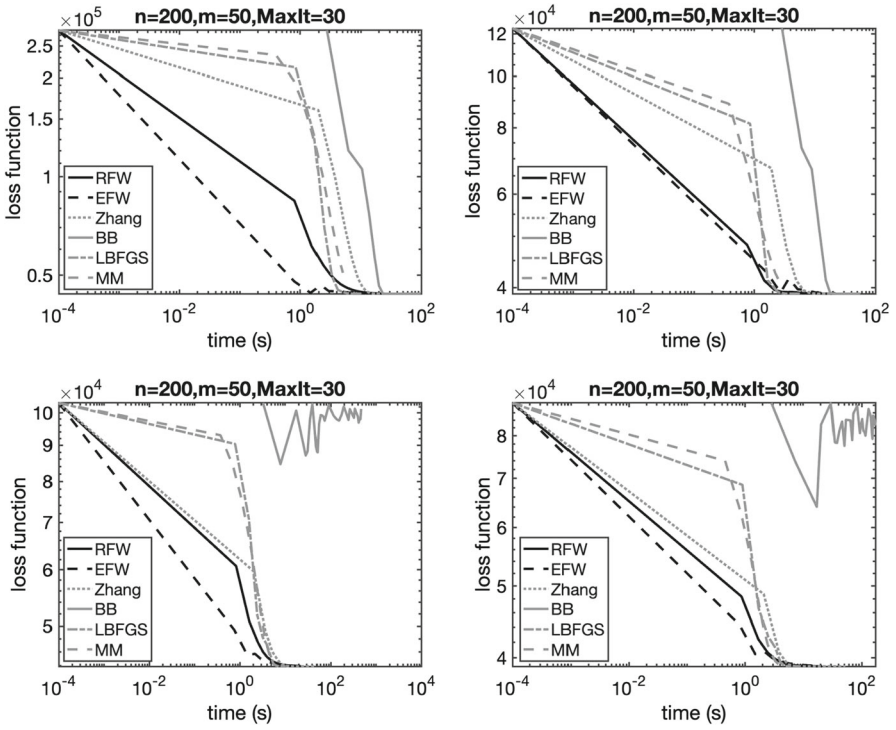

Fig. 2 Performance of EFW and RFW in comparison with state-of-the-art methods for well-conditioned (left) and ill-conditioned inputs (right) and different initializations: $X_0 = HM$ (top) and $X_0 = A_1$ (bottom)

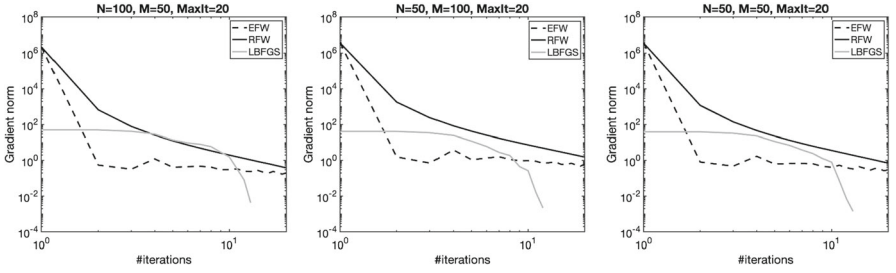

Fig. 3 Gradient norms at each iteration for EFW and RFW in comparison with R-LBFGS

We generate a set A of m positive definite matrices of size n and compute the geometric mean with EFW and RFW as specified in Algorithms 3 and 4. To evaluate the performance of the algorithm, we compute the cost function

$$f(X, A) = \sum_{i=1}^{M} \|\log\left(X^{-\frac{1}{2}} A_i X^{-\frac{1}{2}}\right)\|_F^2 , \tag{48}$$

Table 1 Number of calls to the gradient and cost functions until reaching convergence, averaged over ten experiments with $N = 40$, $M = 10$ and $K = 30$. Note that the competitive performance of EFW/ RFW is partially due to avoiding internal calls to the cost function, significantly increasing the efficiency of both methods compared to the other methods

Method	# calls to grad	# calls to cost
EFW	30	0
RFW	30	0
RBB	17	35
R-LBFGS	30	49
MM	30	0
ZHANG	30	0

after each iteration. We further include an experiment, where we report the gradient norm $\|\text{grad } f\|_F$.

Figure 1 shows performance comparisons of all methods for different parameter choices and condition numbers. In a second experiment (Fig. 2) we compare inputs with different condition numbers for two initialization, the harmonic mean and a matrix $A_i \in A$. We observe that RFW outperforms BB for all inputs and initializations. Furthermore, RFW performs competitively in comparison with R- LBFGS, ZHANG'S METHOD and MM. The competetive performance of RFW is consistent across different initializations. In a third experiment we compared the accuracy reached by RFW with R- LBFGS as the (in our experiments) most accurate Riemannian state-of-the-art method (Fig. 3). We observe that RFW reaches a medium accuracy fast; however, ultimately R-LBFGS reaches a higher accuracy.

In comparison with EFW, we observe that RFW reaches a similar performance and accuracy. The numerical advantage of EFW may be due to implementation differences between Riemannian and Euclidean methods.

In addition, we include a comparison of the number of internal calls to cost and gradient functions (Fig. 1) for all methods. Note that this comparison evaluates algorithmic features and is therefore machine-independent. The reported numbers were obtained by averaging counts over ten experiments with identical parameter choices. We observe that RFW (and EFW) avoid internal calls to the gradient and cost function, requiring only a single gradient evaluation per iteration. This results in the acceleration (i.e., faster convergence) observed in the plots.

6.2.2 Computing Wasserstein barycenters

To demonstrate the versatility of our approach, we use RFW to compute Bures-Wasserstein barycenters. For this, we adapt the above-described setup to implement Algorithm 3. Fig. 4 shows the performance of RFW on both well- and ill-conditioned matrices. We compare three different initializations in each experiment: In (1), we choose an arbitrary element of A as starting point ($X_0 \sim \mathcal{U}(A)$). The other experiments start at the lower and upper bounds of the constraint set, i.e., (2) X_0 is set to the arithmetic mean of A and (3) $X_0 = \alpha I$, where α is the smallest eigenvalue over A. Our results suggest that RFW performs well when initialized from any point in the feasible region, even on sets of ill-conditioned matrices.

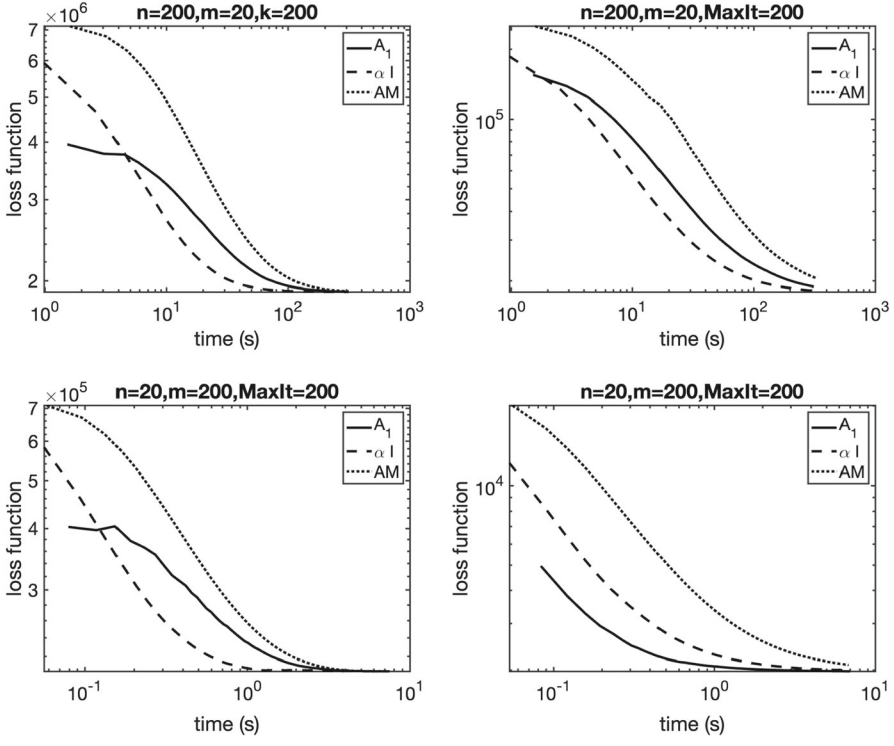

Fig. 4 Performance of RFW for computing the Wasserstein mean for m well- (left) and M ill-conditioned matrices (right) of size n for $MaxIt$ iteration and three initializations: (1) X_0 is set to one of the matrices in the set A, i.e., $A_i \in A$; (2) X_0 is initialized as the arithmetic mean of A, i.e., the upper bound of the constraint set; and (3) X_0 is set to the lower bound αI of the constraint, where α is the minimal eigenvalue over A

7 Discussion

We presented a Riemannian version of the classical Frank-Wolfe method that enables constrained optimization on Riemannian (more precisely, on Hadamard) manifolds. Similar to the Euclidean case, we recover sublinear convergence rates for Riemannian Frank-Wolfe for both geodesically convex and nonconvex objectives. Under the stricter assumption of μ-strongly g-convex objectives and a strict interiority condition on the constraint set, we show that even linear convergence rates can be attained by Riemannian Frank-Wolfe. To our knowledge, this work represents the first extension of Frank-Wolfe methods to a manifold setting.

In addition to the general results, we present an efficient algorithm for optimization on Hermitian positive definite matrices. A key highlight of this specialization is a closed-form solution to the Riemannian "linear" oracle needed by Frank-Wolfe (this oracle involves solving a nonconvex problem). While we focus on the specific problem of computing the Karcher mean (also known as the Riemannian centroid or geometric matrix mean), the derived closed-form solutions apply to more general objective functions and should be of wider interest for related nonconvex and g-convex problems. To demonstrate this versatility, we also included an application of RFW to the computation of Wasserstein barycenters on the Gaussian density manifold. In future work, we hope to explore other constraint sets and matrix manifolds that admit an efficient solution to the Riemannian "linear" oracle.

Our algorithm is shown to be competitive against a variety of established and recently proposed approaches [28, 63, 67] providing evidence for its applicability to large-scale statistics and machine learning problems. In follow-up work [60], we show that RFW extends to nonconvex stochastic settings, further increasing the efficiency and versatility of Riemannian Frank-Wolfe methods. Exploring the use of this class of algorithms in large-scale machine learning applications is a promising avenue for future research.

Acknowledgements The authors thank Charles Fefferman for his helpful comments on the manuscript.

A Approximately solving the Riemannian "linear" oracle

We want to prove the following lemma, stated in the main text:

Lemma 8 *For a steps size $\eta \in (0, 1)$ and accuracy δ, we have*

$$\phi(X_{k+1}) \leq \phi(X_k) - \eta \left\langle \operatorname{grad} \phi(X_k), \operatorname{Exp}_{X_k}(z') \right\rangle + \frac{1}{2}\eta^2 M_\phi (1+\delta) \,.$$

Proof We use again the notation $Y = \gamma_{XZ'}(\eta)$ for a point on the geodesic joining X and Z'. By Lemma 2, we have

$$\phi(Y) \leq \phi(X) + \eta \left\langle \text{grad } \phi(X),\ \text{Exp}_X^{-1}(Z') \right\rangle + \frac{1}{2}\eta^2 M_\phi.$$

By construction, we have

$$\begin{aligned}\left\langle \text{grad } \phi(X),\ \text{Exp}_X^{-1}(Z') \right\rangle &\leq \min_{Y \in \mathcal{X}} \left\langle \text{grad } \phi(X),\ \text{Exp}_X^{-1}(Y) \right\rangle + \frac{1}{2}\delta\eta M_\phi \\ &\leq -(\phi(X) - \phi(X^*)) + \frac{1}{2}\delta\eta M_\phi\ .\end{aligned}$$

Inserting this above, the claim follows as

$$\begin{aligned}\phi(Y) &\leq \phi(X) - \eta(\phi(X) - \phi(X^*)) + \frac{1}{2}\delta\eta^2 M_\phi + \frac{1}{2}\eta^2 M_\phi \\ &\leq \phi(X) - \eta(\phi(X) - \phi(X^*)) + \frac{1}{2}\delta\eta^2 M_\phi(1+\delta)\ .\end{aligned}$$

□

B Non-convex Euclidean Frank-Wolfe

We make a short digression here to mention nonconvex Euclidean Frank-Wolfe (EFW) as a potential alternative approach to solving (18). Indeed, the constraint set therein is not only g-convex, it is also convex in the usual sense. Thus, one can also apply an EFW scheme to solve (18), albeit with a slower convergence rate. In general, a g-convex set $\mathcal{X}$ need not be Euclidean convex, so this observation does not always apply.

EFW was recently analyzed in [35]; the convergence rate reported below adapts one of its main results. The key difference, however, is that due to g-convexity, we can translate the local result of [35] into a global one for problem (18).

Theorem 6 (Convergence FW-gap ([35])) *Define* $\tilde{G}_k := \min_{0 \leq k \leq K} G(X_k)$, *where* $G(X_k) = \max_{Z_k \in \mathcal{X}} \langle Z_k - X_k,\ -\nabla\phi(X_k)\rangle$ *is the* FW*-gap (i.e., measure of convergence) at* X_k. *Define the curvature constant*

$$M_\phi := \sup_{\substack{X,Y,Z \in \mathcal{X} \\ Y = X + s(Z-X)}} \frac{2}{s^2}\left[\phi(Y) - \phi(X) - \langle \nabla\phi(X),\ Y - X\rangle\right].$$

Then, after K *iterations,* EFW *satisfies* $\tilde{G}_K \leq \frac{\max\{2h_0, M_\phi\}}{\sqrt{K+1}}$.

The proof (a simple adaption of [35]) is similar to that of Theorem 3; therefore, we omit it here. Finally, to implement EFW, we need to also efficiently implement its linear oracle. Theorem 7 below shows how to; the proof is similar to that of Theorem 4. It is important to note that this linear oracle involves solving a simple SDP, but it is unreasonable to require the use of an SDP solver at each iteration. Theorem 7 thus proves to be crucial, because it yields an easily computed closed-form solution.

Theorem 7 *Let $L, U \in \mathbb{P}_d$ such that $L \prec U$ and $S \in \mathbb{H}_d$ is arbitrary. Let $U - L = P^*P$, and $PSP^* = Q\Lambda Q^*$. Then, the solution to*

$$\min_{L \preceq Z \preceq U} \quad \operatorname{tr}(SZ), \tag{49}$$

is given by $Z = L + P^*Q[-\operatorname{sgn}(\Lambda)]_+ Q^*P$.

Proof First, shift the constraint to $0 \preceq X - L \preceq U - L$; then factorize $U - L = P^*P$, and introduce a new variable $Y = X - L$. Therewith, problem (49) becomes

$$\min_{0 \preceq Y \preceq U - L} \quad \operatorname{tr}(S(Y + L)) \ .$$

If $L = U$, then clearly $P = 0$, and $X = L$ is the solution. Assume thus, $L \prec U$, so that P is invertible. Thus, the above problem further simplifies to

$$\min_{0 \preceq (P^*)^{-1} Y P^{-1} \preceq I} \quad \operatorname{tr}(SY) \ .$$

Introduce another variable $Z = P^*YP$ and use circularity of trace to now write

$$\min_{0 \preceq Z \preceq I} \quad \operatorname{tr}(PSP^*Z) \ .$$

To obtain the optimal Z, first write the eigenvalue decomposition

$$PSP^* = Q\Lambda Q^*. \tag{50}$$

Lemma 4 implies that the trace will be maximized when the eigenvectors of PSP^* and Z align and their eigenvalues match up. Since $0 \preceq Z \preceq I$, we therefore see that $Z = QDQ^*$ is an optimal solution, where D is diagonal with entries

$$d_{ii} = \begin{cases} 1 & \text{if } \lambda_i(Y) < 0 \\ 0 & \text{if } \lambda_i(Y) \geq 0 \end{cases} \quad \Longrightarrow \ D = [-\operatorname{sgn}(\Lambda)]_+. \tag{51}$$

Undoing the variable substitutions we obtain $X = L + P^*Q[-\operatorname{sgn}(\Lambda)]_+ Q^*P$ as desired. □

Remark 4 Computing the optimal X requires 1 Cholesky factorization, 5 matrix multiplications, and 1 eigenvector decomposition. The theoretical complexity of the Euclidean Linear Oracle can therefore be estimated as $O(N^3)$. On our machine, eigenvector decomposition is approximately 8–12 times slower than matrix multiplication. So the total flop count is approximately $\approx \frac{1}{3}N^3 + 5 \times 2N^3 + 20N^3 \approx 33N^3$.

An implementation of EFW for the computation of Riemannian centroids is shown in Algorithm 4. Experimental results for EFW in comparison with RFW and state-of-the-art Riemannian optimization methods can be found in the main text (see Sect. 6.2.1).

Algorithm 4 EFW for fast Geometric mean

1: $(A_1, \ldots, A_N)$, $\boldsymbol{w} \in \mathbb{R}^N_+$
2: $\bar{X} \approx \operatorname{argmin}_{X \succ 0} \sum_i w_i \delta_R^2(X, A_i)$
3: $\beta = \min_{1 \le i \le N} \lambda_{\min}(A_i)$
4: **for** $k = 0, 1, \ldots$ **do**
5: Compute gradient: $\nabla\phi(X_k) = X_k^{-1}\big(\sum_i w_i \log(X_k A_i^{-1})\big)$
6: Compute Z_k: $Z_k \leftarrow \operatorname{argmin}_{H \preceq Z \preceq A} \langle \nabla\phi(X_k),\, Z - X_k \rangle$
7: Let $\alpha_k \leftarrow \frac{2}{k+2}$.
8: Update X: $X_{k+1} \leftarrow X_k + \alpha_k(Z_k - X_k)$.
9: **end for**
10: **return** $\bar{X} = X_k$

C Generating Positive Definite Matrices

For testing our methods in the well-conditioned regime, we generate matrices $\{A_i\}_{i=1}^n \in \mathbb{P}_d$ by sampling real matrices of dimension d uniformly at random $A_i \sim \mathcal{U}(\mathbb{R}^{d \times d})$ and multiplying each with its transpose $A_i \leftarrow A_i A_i^T$. This gives well-conditioned, positive definite matrices. Furthermore, we sample m matrices $U_i \sim \mathcal{U}(\mathbb{R}^{d \times d})$ with a rank deficit, i.e. $\operatorname{rank}(U) < d$. Then, setting $B_i \leftarrow \delta I + U_i U_i^T$ with δ being small yields ill-conditioned matrices.

References

1. Absil, P.A., Mahony, R., Sepulchre, R.: Optimization algorithms on matrix manifolds. Princeton University Press Princeton, NJ (2009)
2. Bach, F.: Duality between subgradient and conditional gradient methods. SIAM J. Optim. **25**(1), 115–129 (2015)
3. Bento, G.C., Ferreira, O.P., Melo, J.G.: Iteration-complexity of gradient, subgradient and proximal point methods on Riemannian manifolds. J. Optim. Theory Appl. **173**(2), 548–562 (2017)
4. Bhatia, R.: Matrix Analysis. Springer, Berlin (1997)
5. Bhatia, R.: Positive Definite Matrices. Princeton University Press, NJ (2007)
6. Bhatia, R., Holbrook, J.: Riemannian geometry and matrix geometric means. Linear Algebra Appl. **413**, 594–618 (2006)
7. Bhatia, R., Jain, T., Lim, Y.: On the bures-wasserstein distance between positive definite matrices. Expo. Math. **37**(2), 165–191 (2018)
8. Bhatia, R., Jain, T., Lim, Y.: Strong convexity of sandwiched entropies and related optimization problems. Rev. Math. Phys. **30**(09), 1850014 (2018)
9. Bini, D.A., Iannazzo, B.: Computing the Karcher mean of symmetric positive definite matrices. Linear Algebra Appl. **438**(4), 1700–10 (2013)
10. Boumal, N., Absil, P.A., Cartis, C.: Global rates of convergence for nonconvex optimization on manifolds. arXiv preprint arXiv:1605.08101 (2016)
11. Boumal, N., Mishra, B., Absil, P.A., Sepulchre, R.: Manopt, a Matlab toolbox for optimization on manifolds. Journal of Machine Learning Research **15**, 1455–1459 (2014). http://www.manopt.org
12. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint. SIAM J. Computing **40**(6), 1740–1766 (2011)
13. Canon, M., Cullum, C.: A tight upper bound on the rate of convergence of frank-wolfe algorithm. SIAM J. Control **6**, 509–516 (1968)

14. Carson, T., Mixon, D.G., Villar, S.: Manifold optimization for k-means clustering. In: 2017 International Conference on Sampling Theory and Applications (SampTA), pp. 73–77 (2017). https://doi.org/10.1109/SAMPTA.2017.8024388
15. Chavel, I.: Riemannian Geometry: A modern introduction, vol. 98. Cambridge University Press, Cambridge (2006)
16. Cherian, A., Sra, S.: Riemannian dictionary learning and sparse coding for positive definite matrices. arXiv:1507.02772 (2015)
17. Clarkson, K.L.: Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. ACM Trans. Algorithms **6**(4), 1–30 (2010)
18. Combettes, C.W., Pokutta, S.: Complexity of linear minimization and projection on some sets. Oper. Res. Lett. **49**(4), 565–571 (2021)
19. Edelman, A., Arias, T.A., Smith, S.T.: The geometry of algorithms with orthogonality constraints. SIAM J. Matrix Analysis Applications (SIMAX) **20**(2), 303–353 (1998)
20. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval Research Logistics Quarterly **3**(95), 95–110 (1956)
21. Fujishige, S., Isotani, S.: A submodular function minimization algorithm based on the minimum-norm base. Pacific Journal Optimization **7**, 3–17 (2011)
22. Garber, D., Hazan, E.: Faster rates for the Frank-Wolfe method over strongly-convex sets. In: International Conference on Machine Learning, pp. 541–549 (2015)
23. GuéLat, J., Marcotte, P.: Some comments on Wolfe's 'away step'. Math. Program. **35**(1), 110–119 (1986)
24. Hazan, E., Luo, H.: Variance-reduced and projection-free stochastic optimization. In: International Conference on Machine Learning, pp. 1263–1271 (2016)
25. Helmke, U., Hüper, K., Lee, P.Y., Moore, J.: Essential matrix estimation using Gauss-Newton iterations on a manifold. Int. J. Comput. Vision **74**(2), 117–136 (2007)
26. Holloway, C.A.: An extension of the frank and wolfe method of feasible directions. Math. Program. **6**, 14–27 (1974)
27. Hosseini, R., Sra, S.: Matrix manifold optimization for Gaussian mixtures. In: NIPS (2015)
28. Iannazzo, B., Porcelli, M.: The riemannian barzilai-borwein method with nonmonotone line search and the matrix geometric mean computation. IMA J. Numer. Anal. **38**(1), 495–517 (2018)
29. Jaggi, M.: Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In: International Conference on Machine Learning (ICML), pp. 427–435 (2013)
30. Jeuris, B., Vandebril, R., Vandereycken, B.: A survey and comparison of contemporary algorithms for computing the matrix geometric mean. Electron. Trans. Numer. Anal. **39**, 379–402 (2012)
31. Jost, J.: Riemannian Geometry and Geometric Analysis. Springer, Berlin (2011)
32. Karcher, H.: Riemannian center of mass and mollifier smoothing. Comm. Pure Appl. Math. **30**(5), 509–541 (1977)
33. Karimi, H., Nutini, J., Schmidt, M.W.: Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition. CoRR arXiv:1608.04636 (2016)
34. Kubo, F., Ando, T.: Means of positive linear operators. Math. Ann. **246**, 205–224 (1979)
35. Lacoste-Julien, S.: Convergence rate of Frank-Wolfe for non-convex objectives. arXiv preprint arXiv:1607.00345 (2016)
36. Lacoste-Julien, S., Jaggi, M.: On the global linear convergence of Frank-Wolfe optimization variants. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, pp. 496–504. MIT Press, Cambridge, MA, USA (2015)
37. Lawson, J., Lim, Y.: Karcher means and Karcher equations of positive definite operators. Trans. Amer. Math. Soc. Ser. B **1**, 1–22 (2014)
38. Le Bihan, D., Mangin, J.F., Poupon, C., Clark, C.A., Pappata, S., Molko, N., Chabriat, H.: Diffusion Tensor Imaging: Concepts and Applications. J. Magn. Reson. Imaging **13**(4), 534–546 (2001)
39. Ledyaev, Y.S., Treiman, J.S., Zhu, Q.J.: Helly's intersection theorem on manifolds of nonpositive curvature. J. Convex Anal. **13**(3/4), 785 (2006)
40. Lim, Y., Pálfia, M.: Matrix power means and the Karcher mean. J. Funct. Anal. **262**(4), 1498–1514 (2012)
41. Liu, C., Boumal, N.: Simple algorithms for optimization on riemannian manifolds with constraints. Applied Mathematics & Optimization **82**(3), 949–981 (2019)
42. Lojasiewicz, S.: Une propriété topologique des sous-ensembles analytiques réels. Les équations aux dérivées partielles **117**, 87–89 (1963)

43. Malagò, L., Montrucchio, L., Pistone, G.: Wasserstein riemannian geometry of positive-definite matrices ? (2018)
44. Mariet, Z.E., Sra, S.: Fixed-point algorithms for learning determinantal point processes. In: ICML (2015)
45. Mitchell, B.F., Dem'yanov, V.F., Malozemov, V.N.: Finding the point of a polyhedron closest to the origin. SIAM J. Control **12**(1), 19–26 (1974)
46. Moakher, M.: Means and averaging in the group of rotations. SIAM J. Matrix Anal. Appl. **24**(1), 1–16 (2002)
47. Montanari, A., Richard, E.: Non-negative principal component analysis: Message passing algorithms and sharp asymptotics. IEEE Trans. Inf. Theory **62**(3), 1458–1484 (2016)
48. Nielsen, F., Bhatia, R. (eds.): Matrix Information Geometry. Springer (2013)
49. Polyak, B.T.: Gradient methods for minimizing functionals (in Russian). Zh. Vychisl. Mat. Mat. Fiz. **3**(4), 643–653 (1963)
50. Polyak, B.T.: Introduction to Optimization. Optimization Software Inc. (1987). Nov 2010 revision
51. Pálfia, M.: Operator means of probability measures and generalized karcher equations. Adv. Math. **289**, 951–1007 (2016)
52. Reddi, S.J., Sra, S., Póczos, B., Smola, A.: Stochastic Frank-Wolfe methods for nonconvex optimization. In: Communication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference on, pp. 1244–1251. IEEE (2016)
53. Ring, W., Wirth, B.: Optimization methods on Riemannian manifolds and their application to shape space. SIAM J. Optim. **22**(2), 596–627 (2012)
54. Sra, S., Hosseini, R.: Geometric optimisation on positive definite matrices for elliptically contoured distributions. In: Advances in Neural Information Processing Systems, pp. 2562–2570 (2013)
55. Sra, S., Hosseini, R.: Conic geometric optimization on the manifold of positive definite matrices. SIAM J. Optim. **25**(1), 713–739 (2015)
56. Sun, J., Qu, Q., Wright, J.: Complete Dictionary Recovery over the Sphere II: Recovery by Riemannian Trust-region Method. arXiv:1511.04777 (2015)
57. Tan, M., Tsang, I.W., Wang, L., Vandereycken, B., Pan, S.J.: Riemannian pursuit for big matrix recovery. In: International Conference on Machine Learning (ICML-14), pp. 1539–1547 (2014)
58. Udriste, C.: Convex functions and optimization methods on Riemannian manifolds, vol. 297. Springer Science & Business Media, Berlin (1994)
59. Vandereycken, B.: Low-rank matrix completion by Riemannian optimization. SIAM J. Optim. **23**(2), 1214–1236 (2013)
60. Weber, M., Sra, S.: Projection-free nonconvex stochastic optimization on Riemannian manifolds. IMA J. Numer. Anal. (2021). https://doi.org/10.1093/imanum/drab066
61. Weber, M., Sra, S.: Riemannian optimization via frank-wolfe methods arXiv:1710.10770 (2021)
62. Wolfe, P.: Convergence theory in nonlinear programming. Integer and Nonlinear Programming (1970)
63. Yuan, X., Huang, W., Absil, P.A., Gallivan, K.: A riemannian limited-memory bfgs algorithm for computing the matrix geometric mean. Procedia Computer Science **80**, 2147–2157 (2016)
64. Yuan, X., Huang, W., Absil, P.A., Gallivan, K.A.: A Riemannian quasi-Newton method for computing the Karcher mean of symmetric positive definite matrices. Florida State University (FSU17-02) (2017)
65. Zhang, H., Reddi, S., Sra, S.: Fast stochastic optimization on Riemannian manifolds. In: Advances in Neural Information Processing Systems (NIPS) (2016)
66. Zhang, H., Sra, S.: First-order methods for geodesically convex optimization. In: Conference on Learning Theory (COLT) (2016)
67. Zhang, T.: A majorization-minimization algorithm for computing the Karcher mean of positive definite matrices. SIAM J. Matrix Anal. Appl. **38**(2), 387–400 (2017)
68. Zhang, T., Wiesel, A., Greco, M.S.: Multivariate generalized Gaussian distribution: Convexity and graphical models. Signal Processing, IEEE Transactions on **61**(16), 4141–4148 (2013)