# AN EFFICIENT ALTERNATING RIEMANNIAN/PROJECTED GRADIENT DESCENT ASCENT ALGORITHM FOR FAIR PRINCIPAL COMPONENT ANALYSIS

*Meng Xu*[⋆,§]*, Bo Jiang*[†]*, Wenqiang Pu*[‡]*, Ya-Feng Liu*[⋆]*, and Anthony Man-Cho So*[♯]

[⋆]LSEC, ICMSEC, AMSS, Chinese Academy of Sciences, Beijing, China
[§]School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China
[†]Ministry of Education Key Laboratory of NSLSCS,
School of Mathematical Sciences, Nanjing Normal University, Nanjing, China
[‡]Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong, Shenzhen, China
[♯]Department of Sys. Eng. & Eng. Mgmt, The Chinese University of Hong Kong, HKSAR, China
Email: xumeng22@mails.ucas.ac.cn, jiangbo@njnu.edu.cn, wenqiangpu@cuhk.edu.cn,
yafliu@lsec.cc.ac.cn, manchoso@se.cuhk.edu.hk

## ABSTRACT

Fair principal component analysis (FPCA), a ubiquitous dimensionality reduction technique in signal processing and machine learning, aims to find a low-dimensional representation for a high-dimensional dataset in view of fairness. The FPCA problem involves optimizing a non-convex and non-smooth function over the Stiefel manifold. The state-of-the-art methods for solving the problem are subgradient methods and semidefinite relaxation-based methods. However, these two types of methods have their obvious limitations and thus are only suitable for efficiently solving the FPCA problem in special scenarios. This paper aims at developing efficient algorithms for solving the FPCA problem in general, especially large-scale, settings. In this paper, we first transform FPCA into a smooth non-convex linear minimax optimization problem over the Stiefel manifold. To solve the above general problem, we propose an efficient alternating Riemannian/projected gradient descent ascent (ARPGDA) algorithm, which performs a Riemannian gradient descent step and an ordinary projected gradient ascent step at each iteration. We prove that ARPGDA can find an $\varepsilon$-stationary point of the above problem within $\mathcal{O}(\varepsilon^{-3})$ iterations. Simulation results show that, compared with the state-of-the-art methods, our proposed ARPGDA algorithm can achieve a better performance in terms of solution quality and speed for solving the FPCA problems.

***Index Terms*—** FPCA, iteration complexity, minimax problem, Riemannian optimization.

## 1. INTRODUCTION

Dimensionality reduction, which aims to identify a low-dimensional subspace to represent high-dimensional data, has attracted much attention in recent decades due to its wide applications in signal processing and machine learning. The classical principal component analysis (PCA), one of the most popular dimensional reduction techniques in practice, minimizes the single criterion of the total reconstruction error over the whole samples and may ignore the potential difference among different sensitive groups. Recently, the idea of fair machine learning has come to the stage, and applying this idea to PCA gives rise to fair principal component analysis (FPCA) [1]. It has been shown in [1–3] that compared to PCA, FPCA is more effective for not only machine learning applications such as credit scoring and image processing but also signal processing applications. There is also growing interest in alternative definitions of fairness for PCA [4–8], with various problem formulations and algorithms.

However, FPCA is generally NP-hard (to solve exactly) [2], and there are two main approaches to the problem for approximate or local solutions. One approach is the semidefinite relaxation (SDR)-based algorithms [1, 2, 9, 10]. In [1], an SDR for FPCA was first proposed, and a multiplicative-weight style algorithm was used to approximately solve the semidefinite program. Some theoretical bounds for the SDR approach were also established in [1]. Later on, the work [2] improved the results in [1] to give a tighter approximation ratio of the SDR approach and showed that when the number of groups is two, the SDR approach can recover the solution of FPCA exactly. As mentioned in [2], solving the semidefinite program is the main bottleneck of the SDR approach. Hence, the SDR approach might be inapplicable to high-dimensional datasets [3]. The other approach to solving the FPCA problem is the subgradient methods. In particular, the work [3] proposed an interesting equivalent unconstrained factorized FPCA formulation and applied the subgradient algorithm to solve the unconstrained equivalent problem. Notably, it has been shown in [3] that any local minimizer of the factorized FPCA is a global maximizer of FPCA if the dataset satisfies an orthogonality condition. Therefore, the global solution of the original problem can be obtained if the orthogonality condition holds true. Furthermore, the recently proposed Riemannian subgradient algorithm in [11] can be utilized for solving the FPCA problem. However, the Riemannian subgradient algorithm often exhibits a slow convergence rate, requiring approximately $\mathcal{O}(\varepsilon^{-4})$ iterations to return an $\varepsilon$-stationary point.

Motivated by the features of the existing approaches, it is of interest to develop more efficient as well as easy-to-implement methods for large-scale FPCA. By observing that FPCA can be rewritten as a non-convex linear minimax problem over the Stiefel manifold, in this paper, we propose an efficient alternating Riemannian/projected gradient descent ascent (ARPGDA) algorithm for solving the FPCA problem and the more general non-convex linear minimax problem over a Riemannian manifold. Our proposed ARPGDA algorithm only needs to perform a simple Riemannian gradient descent step and a projected gradient ascent step at each iteration, making it easy to implement and scalable to large-scale problems. In addition, we prove that our proposed ARPGDA al-

gorithm has an iteration complexity of $\mathcal{O}(\varepsilon^{-3})$ for finding an $\varepsilon$-stationary point (defined in Definition 1), which matches the best known complexity for its Euclidean counterpart. Finally, simulation results show that, when applied to solve the FPCA problems from signal processing and machine learning applications, our proposed ARPGDA algorithm is more efficient in terms of both solution quality and speed compared to the subgradient and SDR methods.

Our proposed ARPGDA algorithm can be seen as an extension of the alternating gradient projection algorithm [12] for solving the non-convex linear minimax problem from the Euclidean space to the Riemannian manifold setting. To the best of our knowledge, the proposed algorithm is the first one with an iteration complexity guarantee for general non-convex linear minimax optimization over a Riemannian manifold. Existing works on minimax optimization over a Riemannian manifold either focus on different problem settings or make strong assumptions that do not apply to the problems of our interest, including FPCA. Specifically, the works [13] and [14, 15] considered the settings of non-convex strongly concave and geodesically convex geodesically concave minimax problems over Riemannian manifolds, respectively. However, their settings do not apply to the FPCA problem. Most recently, the work [16] introduced a Riemannian Hamiltonian gradient method for the Riemannian manifold constrained minimax problem, but it relies on a strong Riemannian Polyak-Łojasiewicz condition that is unlikely to hold for the FPCA problem.

## 2. FPCA AND MINIMAX REFORMULATION

Let the columns of a $d \times N$ matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ denote $N$ data samples, each with $d$ attributes. Suppose that the $N$ data samples belong to $n$ groups, according to demographics or some other semantically meaningful clustering, and each group $i$ corresponds to a $d \times n_i$ submatrix $\mathbf{X}_i$ with $\sum_{i=1}^n n_i = N$. Classical PCA aims to find a subspace with dimension $r < d$ to minimize the total reconstruction error or maximize the total variance equivalently. However, this may cause certain sensitive groups to suffer a higher reconstruction error than the others [1]. To reduce such disparity, FPCA minimizes the maximum reconstruction error among the $n$ groups, which is equivalent to maximizing the minimum variance among the $n$ groups. Formally, FPCA can be formulated as [2]

$$\text{FPCA:} \quad \max_{\mathbf{U}\in\mathcal{S}} \min_{i=1,2,\ldots,n} f_i(\mathbf{U}) := \langle \mathbf{X}_i\mathbf{X}_i^T, \mathbf{U}\mathbf{U}^T\rangle, \tag{1}$$

where $\langle\cdot,\cdot\rangle$ is the standard Euclidean inner product and $\mathcal{S} := \{\mathbf{U} \in \mathbb{R}^{d\times r} \mid \mathbf{U}^T\mathbf{U} = \mathbf{I}_r\}$ is known as the Stiefel manifold. Note that if $n = 1$, problem (1) reduces to classical PCA. If $n \ll N$, problem (1) is called block FPCA [3], which was first introduced in the context of fair machine learning [1]. If $n = N$, problem (1) becomes

$$\max_{\mathbf{U}\in\mathcal{S}} \min_{i=1,2,\ldots,n} \langle \mathbf{x}_i\mathbf{x}_i^T, \mathbf{U}\mathbf{U}^T\rangle, \tag{2}$$

which arises in many signal processing applications, such as multicast downlink beamforming [17–20] and filter design [3]. In the context of filter design, $\{\mathbf{x}_i\}_{i=1}^n$ denote the high-resolution digital signals of $n$ kinds of materials, and the parameters always satisfy $r < n = N \ll d$, which is sharply different from the settings in fair machine learning, in which $d$ might be smaller than $N$.

Let $\Delta := \left\{\mathbf{y} \in \mathbb{R}^n \mid \sum_{i=1}^n y_i = 1,\ y_i \geq 0,\ i = 1, 2, \ldots, n\right\}$ be the standard simplex. The FPCA problem (1) can be equivalently rewritten as follows:

$$\min_{\mathbf{U}\in\mathcal{S}} \max_{\mathbf{y}\in\Delta} \sum_{i=1}^n y_i(-f_i(\mathbf{U})). \tag{3}$$

Hence, we generalize (3) to consider the non-convex linear minimax problem over a Riemannian submanifold $\mathcal{M}$ of a finite-dimensional Euclidean space $\mathcal{E}_1$, i.e.,

$$\min_{\mathbf{U}\in\mathcal{M}} \max_{\mathbf{y}\in\mathcal{Y}} f(\mathbf{U}, \mathbf{y}), \tag{4}$$

where $f(\mathbf{U}, \mathbf{y}) : \mathcal{M} \times \mathcal{Y} \to \mathbb{R}$ is a smooth function that is linear with respect to $\mathbf{y}$ but possibly non-convex with respect to $\mathbf{U}$, and $\mathcal{Y}$ is a non-empty compact convex set in a finite-dimensional Euclidean space $\mathcal{E}_2$. In the rest of this paper, we shall develop an efficient algorithm for solving the general non-convex linear minimax problem (4) with the Riemannian manifold constraint, which includes our interested FPCA problem as a special case.

## 3. PROPOSED ARPGDA ALGORITHM

In this section, we propose an efficient and scalable ARPGDA algorithm for solving the non-convex linear minimax problem (4) and establish its iteration complexity.

### 3.1. Proposed Algorithm

Before presenting the ARPGDA algorithm, we introduce some basic objects associated with the Riemannian manifold $\mathcal{M}$. Let $T_{\mathbf{U}}\mathcal{M}$ denote the tangent space to $\mathcal{M}$ at $\mathbf{U} \in \mathcal{M}$. The Riemannian manifold $\mathcal{M}$ is endowed with a smooth inner product $\langle\cdot,\cdot\rangle_{\mathbf{U}} : T_{\mathbf{U}}\mathcal{M} \times T_{\mathbf{U}}\mathcal{M} \to \mathbb{R}$. Let $\mathcal{R}_{\mathbf{U}} : T_{\mathbf{U}}\mathcal{M} \to \mathcal{M}$ be the retraction at $\mathbf{U}$, which is a smooth map satisfying i) $\mathcal{R}_{\mathbf{U}}(\mathbf{0}_{\mathbf{U}}) = \mathbf{U}$, where $\mathbf{0}_{\mathbf{U}}$ is the zero element in $T_{\mathbf{U}}\mathcal{M}$; ii) the differential of $\mathcal{R}_{\mathbf{U}}(\mathbf{0}_{\mathbf{U}})$ is the identity map. For a smooth function $f : \mathcal{M} \to \mathbb{R}$, the Riemannain gradient $\mathrm{grad} f(\mathbf{U})$ is the unique tangent vector at $\mathbf{U}$ satisfying $\langle \mathrm{grad} f(\mathbf{U}), \mathbf{D}\rangle_{\mathbf{U}} = \mathrm{D}f(\mathbf{U})[\mathbf{D}]$, where $\mathrm{D}f(\mathbf{U})[\mathbf{D}]$ denotes the directional derivative of $f$ along the direction $\mathbf{D}$. For the Stiefel manifold, the endowed norm $\|\cdot\|_{\mathbf{U}}$ is taken as the usual Euclidean norm $\|\cdot\|$ and the Riemannian gradient is

$$\mathrm{grad} f(\mathbf{U}) = \nabla f(\mathbf{U}) - \mathbf{U}(\mathbf{U}^T\nabla f(\mathbf{U}) + \nabla f(\mathbf{U})^T\mathbf{U})/2. \tag{5}$$

Moreover, we adopt the polar decomposition-based retraction [21], which satisfies $\|\mathcal{R}_{\mathbf{U}}(\mathbf{D}) - \mathcal{R}_{\mathbf{U}}(\mathbf{0}_{\mathbf{U}})\| \leq \|\mathbf{D}\|$ and $\|\mathcal{R}_{\mathbf{U}}(\mathbf{D}) - \mathcal{R}_{\mathbf{U}}(\mathbf{0}_{\mathbf{U}}) - \mathbf{D}\| \leq 1/2\|\mathbf{D}\|^2$; see [22, Lemma A.2] for details.

Recall that (4) is a non-convex linear minimax problem with a non-convex function in $\mathbf{U}$ and a manifold constraint $\mathcal{M}$, which is difficult to solve. To handle the large-scale problem, we adopt an alternating first-order approach to update $\mathbf{U}$ and $\mathbf{y}$, which has a very low cost at each iteration. Inspired by the regularization techniques in [12, 23–25], we introduce a regularized version of the original objective function in (4) at the $k$-th iteration, i.e.,

$$f_k(\mathbf{U}, \mathbf{y}) := f(\mathbf{U}, \mathbf{y}) - \frac{\lambda_k}{2}\|\mathbf{y}\|^2,$$

where $\lambda_k \geq 0$ is a regularization parameter. Our proposed algorithm is based on the above regularized function $f_k$ at the $k$-th iteration. In particular, the $k$-th iteration of the proposed algorithm performs a Riemannian gradient descent step to update $\mathbf{U}$, followed by a projected gradient ascent step to update $\mathbf{y}$. Hence, we name it the alternating Riemannian/projected gradient descent ascent (ARPGDA) algorithm. Let $\mathrm{grad}_{\mathbf{U}} f(\mathbf{U}, \mathbf{y})$ be the Riemannian gradient with respect to $\mathbf{U}$, which can be calculated according to (5). Noting that $\mathrm{grad}_{\mathbf{U}} f_k(\mathbf{U}_k, \mathbf{y}_k) = \mathrm{grad}_{\mathbf{U}} f(\mathbf{U}_k, \mathbf{y}_k)$, we update $\mathbf{U}_{k+1}$ as follows:

$$\mathbf{U}_{k+1} = \mathcal{R}_{\mathbf{U}_k}(-\zeta_k \mathrm{grad}_{\mathbf{U}} f(\mathbf{U}_k, \mathbf{y}_k)), \tag{6}$$

where $\zeta_k > 0$ is the stepsize. With the identity $\nabla_{\mathbf{y}} f_k(\mathbf{U}_{k+1}, \mathbf{y}_k) = \nabla_{\mathbf{y}} f(\mathbf{U}_{k+1}, \mathbf{y}_k) - \lambda_k \mathbf{y}_k$, we update $\mathbf{y}_{k+1}$ as follows:

$$\mathbf{y}_{k+1} = \mathcal{P}_{\mathcal{Y}}\Big(\mathbf{y}_k + \frac{1}{\lambda_k + \beta_k}\big(\nabla_{\mathbf{y}} f(\mathbf{U}_{k+1}, \mathbf{y}_k) - \lambda_k \mathbf{y}_k\big)\Big), \quad (7)$$

where $\mathcal{P}_{\mathcal{Y}}(\mathbf{z})$ denotes the projection of $\mathbf{z}$ onto the set $\mathcal{Y}$ and $\frac{1}{\lambda_k+\beta_k} > 0$ is the stepsize. The ARPGDA algorithm is formally presented in Algorithm 1.

**Algorithm 1** ARPGDA Algorithm for Solving Problem (4)

Step 1 Input $\mathbf{U}_1, \mathbf{y}_1, \varepsilon, \{\lambda_k\}, \{\beta_k\}, \{\zeta_k\}$; set $k = 1$.
Step 2 Calculate $\mathbf{U}_{k+1}$ via (6).
Step 3 Calculate $\mathbf{y}_{k+1}$ via (7).
Step 4 If $\mathcal{E}(\mathbf{U}_{k+1}, \mathbf{y}_{k+1}) \le \varepsilon$ (see (8) further ahead), stop; otherwise, set $k \leftarrow k + 1$ and go to Step 2.

Our proposed ARPGDA algorithm is a first-order algorithm, as it only uses the current gradient information of $f(\mathbf{U}, \mathbf{y})$ to generate the next point. Hence, it is suitable to solve our interested large-scale FPCA problem (1). When applied to FPCA, each iteration of the ARPGDA algorithm essentially performs a Riemannian gradient step to inexactly solve a classical PCA problem with an adaptive weighted matrix $\sum_{i=1}^n y_i \mathbf{X}_i \mathbf{X}_i^T$ (where $y_i$ can be seen as the weight of the data in group $i$), and the per-iteration cost of updating $\mathbf{U}$ and $\mathbf{y}$ is $\mathcal{O}(Nrd + r^2 d)$ and $\mathcal{O}(n \log n)$ [26,27], respectively. Compared with the SDR approach for FPCA, ARPGDA has a much lower computational complexity since the SDR approach works with a high-dimensional matrix variable in $\mathbb{R}^{d\times d}$. Compared with the subgradient-type methods [3] and [11], ARPGDA has a similar per-iteration cost. However, as shown later in the simulation, we can see that ARPGDA always converges significantly faster than the subgradient methods.

### 3.2. Iteration Complexity

Before presenting the iteration complexity result for the ARPGDA algorithm, we first define the notion of an $\varepsilon$-stationary point of problem (4) as follows, which is motivated by the definition in [12, 23] for the convex constrained counterpart of problem (4).

**Definition 1** *For any given $\varepsilon > 0$, we say that $(\tilde{\mathbf{U}}, \tilde{\mathbf{y}})$ is an $\varepsilon$-stationary point of problem (4) if $\mathcal{E}(\tilde{\mathbf{U}}, \tilde{\mathbf{y}}) \le \varepsilon$, where*

$$\mathcal{E}(\tilde{\mathbf{U}}, \tilde{\mathbf{y}}) := \max\big\{\|\mathrm{grad}_{\mathbf{U}} f(\tilde{\mathbf{U}}, \tilde{\mathbf{y}})\|_{\tilde{\mathbf{U}}}, \max_{\mathbf{y}\in\mathcal{Y}} \langle \nabla_{\mathbf{y}} f(\tilde{\mathbf{U}}, \tilde{\mathbf{y}}), \mathbf{y} - \tilde{\mathbf{y}}\rangle\big\}. \quad (8)$$

We also need to make the following (somewhat standard) assumption about the smoothness of the function $f$.

**Assumption 1** *The function $f$ is continuously differentiable, and there exist positive constants $L_1$ and $L_2$ such that for any $\mathbf{U} \in \mathcal{M}, \mathbf{D} \in T_{\mathbf{U}}\mathcal{M}$, and $\bar{\mathbf{U}} = \mathcal{R}_{\mathbf{U}}(\mathbf{D})$, we have*

$$f(\bar{\mathbf{U}}, \mathbf{y}) \le f(\mathbf{U}, \mathbf{y}) + \langle \mathrm{grad}_{\mathbf{U}} f(\mathbf{U}, \mathbf{y}), \mathbf{D}\rangle_{\mathbf{U}} + \frac{L_1}{2}\|\mathbf{D}\|_{\mathbf{U}}^2, \quad (9a)$$

$$\|\mathrm{grad}_{\mathbf{U}} f(\mathbf{U}, \mathbf{y}) - \mathrm{grad}_{\mathbf{U}} f(\mathbf{U}, \bar{\mathbf{y}})\|_{\mathbf{U}} \le L_2 \|\mathbf{y} - \bar{\mathbf{y}}\|. \quad (9b)$$

Condition (9a) is a standard assumption in manifold optimization [28], and condition (9b) is widely used in the minimax optimization literature [12, 13, 25]. For our interested FPCA problem (3), by some calculations, we can show that Assumption 1 holds with $L_1 = 2\max_i \|\mathbf{X}_i \mathbf{X}_i^T\|$ and $L_2 = 2\sqrt{\|\|\sum_{i=1}^n \mathbf{X}_i \mathbf{X}_i^T \mathbf{X}_i \mathbf{X}_i^T\|\|_{(r)}}$. Here, $\|\|\cdot\|\|_{(r)}$ is the Ky Fan $r$-norm, which is the sum of the $r$ largest singular values of a matrix.

Now we are ready to present the iteration complexity result of the ARPGDA algorithm for solving a general non-convex linear minimax problem over a Riemannian manifold, namely, problem (4).

**Theorem 1** *Suppose that Assumption 1 holds and the parameters in Algorithm 1 satisfy $\lambda_k = \frac{\varepsilon}{8R^2}$, $\beta_k = \mu_k k^{-\rho}, 0 \le \mu_k \le \mu, \zeta_k = \theta/(L_1 + \frac{L_2^2}{\lambda_k+\beta_k+\beta_{k+1}})$, where $\rho > 1, \theta \in (0, 2)$ are constants, and $R = \max_{\mathbf{y}\in\mathcal{Y}} \|\mathbf{y}\|$. Then, Algorithm 1 is guaranteed to return an $\varepsilon$-stationary point of problem (4) within $\mathcal{O}(\varepsilon^{-3})$ iterartions.*

Our iteration complexity result of $\mathcal{O}(\varepsilon^{-3})$ in Theorem 1 matches the best known complexity result of general non-convex linear minimax optimization, in which the Riemannian manifold is replaced by a non-empty compact convex set [12]. Due to the space limitation, instead of giving a rigorous proof of Theorem 1, we just give some key steps of the proof. First, based on the update schemes (6) and (7), we can estimate the sufficient reduction of $f_k(\mathbf{U}, \mathbf{y})$ at each iteration as follows:

$$\begin{aligned} &f_{k+1}(\mathbf{U}_{k+1}, \mathbf{y}_{k+1}) - f_k(\mathbf{U}_k, \mathbf{y}_k) \\ \le& -\frac{2-\theta}{2\theta}\zeta_k \|\mathrm{grad}_{\mathbf{U}} f(\mathbf{U}_k, \mathbf{y}_k)\|_{\mathbf{U}_k}^2 + \frac{1}{2}(\lambda_k - \lambda_{k+1} + 4\beta_k)R \\ &-\frac{1}{2}(\beta_k \|\mathbf{y}_k - \mathbf{y}_{k-1}\|^2 - \beta_{k+1}\|\mathbf{y}_{k+1} - \mathbf{y}_k\|^2). \end{aligned}$$

Second, due to the fact that $f$ is linear in $\mathbf{y}$, we have

$$\max_{\mathbf{y}\in\mathcal{Y}} \langle \nabla_{\mathbf{y}} f(\mathbf{U}_k, \mathbf{y}_k), \mathbf{y} - \mathbf{y}_k\rangle \le 4R^2(\lambda_k + \beta_k).$$

Finally, by carefully choosing the parameters, we can complete the proof of Theorem 1. It should be mentioned that our key proof process is motivated by and similar to that in [12]. The main difference lies in the first step, where we use conditions (9a) and (9b) to bound the change of the function value. Even in the Euclidean setting, our condition (9a) is still weaker than the first Lipschitz continuous gradient assumption of Assumption 1 in [12].

## 4. SIMULATION RESULTS

In this section, we present numerical results to show both the effectiveness and efficiency of the proposed ARPGDA algorithm when solving the FPCA problem. We mainly compare ARPGDA with the subgradient (SG) method, which is designed to solve the unconstrained factorized FPCA problem [3], and the Riemannian subgradient (RSG) method [11], which solves the following equivalent formulation of problem (1):

$$\max_{\mathbf{U}\in\mathcal{S}} \Phi(\mathbf{U}) := \min_{i=1,2,\ldots,n} f_i(\mathbf{U}).$$

The stepsize of the two subgradient methods is $c/k^{1/2}$, where $c$ is chosen carefully in each test. We stop SG or RSG when $\Phi(\mathbf{U}) \ge (1 - 10^{-4})\Phi_{\text{ARPGDA}}$ or the iteration number hits the maximum number of iterations $10^5$. Here, $\Phi(\mathbf{U})$ and $\Phi_{\text{ARPGDA}}$ denote the objective values returned by SG or RSG and ARPGDA, respectively. When the dimension $d$ of the problem is not high, we also compare the SDR-based approach [2]. All results in this section are averaged over 10 runs with different randomly generated initial points.

**Numerical Results on FPCA Problem (1) with $n = N$.** The FPCA problem (1) with $n = N$, namely, problem (2), arises in many

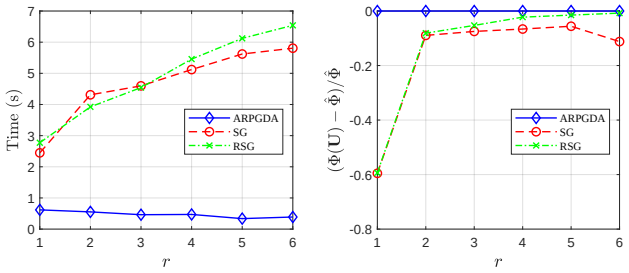


**Fig. 1**. Average performance comparison on the synthetic data.

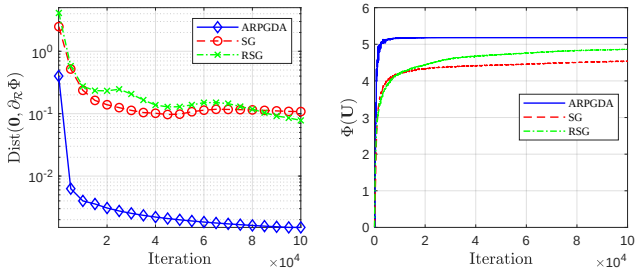


**Fig. 2**. Performance comparison of an instance on the synthetic data when $r = 2$.

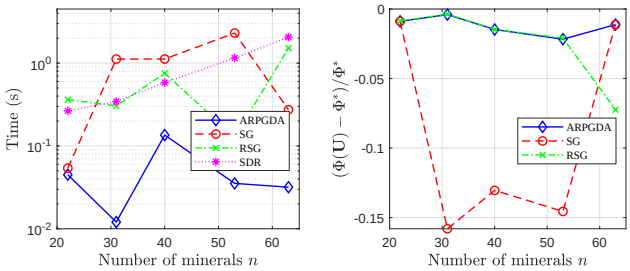


**Fig. 3**. Average performance comparison on the mineral dataset.

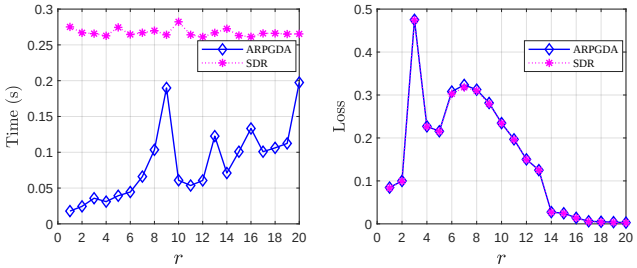


**Fig. 4**. Average performance comparison on the credit dataset.

signal processing applications [3, 17]. For ARPGDA, we choose $\varepsilon = 10^{-3}\max_i \|\mathbf{x}_i\|^2$, $\rho = 1.1$, $\theta = 1.5$, and $\mu_k \equiv 30n^2\sqrt{r}$. Note that for FPCA, the constant $R = 1$.

The results on synthetic datasets, wherein the samples $\{\mathbf{x}_i\}_{i=1}^n$ with $d = n = 200$ are independently generated according to the standard Gaussian distribution [3], are plotted in Fig. 1, where $\hat{\Phi}$ denotes the maximum objective value obtained by the three algorithms. From the figure, we can see that i) our proposed ARPGDA is not only much faster than RSG and SG but also can return much better solutions; ii) RSG can return better solutions than SG. More interestingly, the speed advantage of ARPGDA becomes larger when $r$ increases. In our tests, we also observe that both of the subgradient methods often reach the preset maximum number of iterations. To further understand the convergence behavior of the three algorithms, we take an instance with $r = 2$ and run all three algorithms till $10^5$ iterations. The results are plotted in Fig. 2. From this figure, we see that ARPGDA can reduce the approximate minimum norm of the (Riemannian) subgradient, denoted by $\mathrm{Dist}(\mathbf{0}, \partial_{\mathcal{R}}\Phi)$, and improve the objective value much faster than the subgradient-type methods. Here, $\mathrm{Dist}(\mathbf{0}, \partial_{\mathcal{R}}\Phi) := \min_{\sum_{i\in\mathcal{A}} y_i = 1, y_i \geq 0} \| \sum_{i\in\mathcal{A}} y_i \mathrm{grad} f_i \|$, where $\mathcal{A} := \{i \in \{1, 2, \ldots, n\} \mid f_i - \min_i f_i \leq 0.1 \min_i f_i\}$.

To get more convincing and realistic results, we consider the spectral signatures of minerals from the Spectral Library of United States Geological Survey (USGS) as in [3]. We select a total of $n = 63$ kinds of minerals, each with a reflectance signal $\{\mathbf{x}_i\}_{i=1}^{63}$ with $d = 421$ after dropping the signal vectors, whose norms are so small that they may cause a trivial degenerate problem. The vectors are normalized and centered. We also present the performance of the SDR approach, where the corresponding semidefinite program is solved via CVX [29] as in [3]. Let $\Phi^*$ denote the optimal value of the SDR. By [1, Theorem 5.2], we know that this value is equal to the optimal value of the FPCA problem (1). The averaged results are presented in Fig. 3. Note that our ARPGDA algorithm is the fastest among the four algorithms. In particular, except for $n = 20$ and $n = 40$, ARPGDA is generally 10 times faster than the subgradient-type methods and the SDR approach. Besides, the objective values returned by ARPGDA are still much better than those returned by SG and RSG on average. We also observe that the local maxima vary dramatically for this dataset, which causes the choppy curves.

**Numerical Results on Block FPCA Problem** (1) **with $n \ll N$.** The block FPCA problem (1) with $n \ll N$ arises in a fair machine learning application [1, 2]. We choose the Default Credit dataset [30], which contains $N = 30000$ client samples, each with $d = 23$ features including payments and other personal details. The dataset is partitioned into $n = 4$ sensitive groups according to education and gender. Since the dimension $d$ is not high and the solutions of the SDR approach are shown to be optimal at all ranks except $r = 8$ [2], the SDR approach is more competitive in this case than the subgradient-type methods. Therefore, we only compare the performance of ARPGDA and the SDR approach. The parameters of ARPGDA are $\varepsilon = 10^{-3}$, $\rho = 1.01$, $\theta = 1.99$, and $\mu_k \equiv 200n^2\sqrt{r}$. We present the average computational time and marginal loss value defined in [2] of SDR and ARPGDA in Fig. 4. The left subfigure of Fig. 4 illustrates that ARPGDA is always faster than SDR. More specifically, it is always more than 2 times faster than SDR except for the cases when $r = 9, 20$. For some cases when $r \leq 6$ , it is even more than 5 times faster than SDR. As shown in the right subfigure of Fig. 4, ARPGDA reaches the SDR upper bound with high accuracy for most of the rank cases, indicating that ARPGDA is effective for block FPCA as well.

At last, we can conclude from the simulation results that our proposed ARPGDA algorithm is quite efficient in solving various FPCA problems and exhibits a good capability of finding high-quality solutions compared to existing state-of-the-art algorithms [2, 3, 11].

## 5. CONCLUSION

In this paper, we proposed a novel alternating Riemannian/projected gradient descent ascent algorithm for solving the FPCA problem and the more general non-convex linear minimax problem over a Riemannian manifold. We proved that the proposed algorithm can find an $\varepsilon$-stationary point of the above problems within $\mathcal{O}(\varepsilon^{-3})$ iterations. We showed via simulation results that the proposed algorithm has a better performance in terms of solution quality and speed when solving the FPCA problems arising in signal processing and machine learning applications.

## 6. REFERENCES


[1] S. Samadi, U. Tantipongpipat, J. H. Morgenstern, M. Singh, and S. Vempala, "The price of fair PCA: One extra dimension," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 10 999–11 010.

[2] U. Tantipongpipat, S. Samadi, M. Singh, J. H. Morgenstern, and S. Vempala, "Multi-criteria dimensionality reduction with applications to fairness," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 15 161–15 171.

[3] G. Zalcberg and A. Wiesel, "Fair principal component analysis and filter design," *IEEE Transactions on Signal Processing*, vol. 69, pp. 4835–4842, 2021.

[4] M. M. Kamani, F. Haddadpour, R. Forsati, and M. Mahdavi, "Efficient fair principal component analysis," *Machine Learning*, vol. 111, pp. 3671–3702, 2022.

[5] J. Lee, G. Kim, M. Olfat, M. Hasegawa-Johnson, and C. D. Yoo, "Fast and efficient MMD-based fair PCA via optimization over Stiefel manifold," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 7363–7371.

[6] G. D. Pelegrina, R. D. Brotto, L. T. Duarte, R. Attux, and J. M. Romano, "Analysis of trade-offs in fair principal component analysis based on multi-objective optimization," in *Proceedings of 2022 International Joint Conference on Neural Networks*, 2022, pp. 1–8.

[7] G. D. Pelegrina and L. T. Duarte, "A novel approach for fair principal component analysis based on eigendecomposition," *IEEE Transactions on Artificial Intelligence (accepted)*, pp. 1–12, 2023.

[8] H. Vu, T. Tran, M.-C. Yue, and V. A. Nguyen, "Distributionally robust fair principal components via geodesic descents," in *Proceedings of International Conference on Learning Representations (accepted)*, 2022, pp. 1–23.

[9] J. Morgenstern, S. Samadi, M. Singh, U. Tantipongpipat, and S. Vempala, "Fair dimensionality reduction and iterative rounding for SDPs," *arXiv preprint arXiv:2010.06097*, 2019.

[10] M. Olfat and A. Aswani, "Convex formulations for fair principal component analysis," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, 2019, pp. 663–670.

[11] X. Li, S. Chen, Z. Deng, Q. Qu, Z. Zhu, and A. M.-C. So, "Weakly convex optimization over Stiefel manifold using Riemannian subgradient-type methods," *SIAM Journal on Optimization*, vol. 31, no. 3, pp. 1605–1634, 2021.

[12] W. Pan, J. Shen, and Z. Xu, "An efficient algorithm for nonconvex-linear minimax optimization problem and its application in solving weighted maximin dispersion problem," *Computational Optimization and Applications*, vol. 78, pp. 287–306, 2021.

[13] F. Huang and S. Gao, "Gradient descent ascent for minimax problems on Riemannian manifolds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 7, pp. 8466–8476, 2023.

[14] M. Jordan, T. Lin, and E.-V. Vlatakis-Gkaragkounis, "First-order algorithms for min-max optimization in geodesic metric spaces," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 6557–6574.

[15] P. Zhang, J. Zhang, and S. Sra, "Minimax in geodesic metric spaces: Sion's theorem and algorithms," *arXiv preprint arXiv:2202.06950*, 2022.

[16] A. Han, B. Mishra, P. Jawanpuria, P. Kumar, and J. Gao, "Riemannian Hamiltonian methods for min-max optimization on manifolds," *SIAM Journal on Optimization*, vol. 33, no. 3, pp. 1797–1827, 2023.

[17] N. D. Sidiropoulos, T. N. Davidson, and Z.-Q. Luo, "Transmit beamforming for physical-layer multicasting," *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 2239–2251, 2006.

[18] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 20–34, 2010.

[19] C. Lu and Y.-F. Liu, "An efficient global algorithm for single-group multicast beamforming," *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3761–3774, 2017.

[20] E. Chen, M. Tao, and Y.-F. Liu, "Joint base station clustering and beamforming for non-orthogonal multicast and unicast transmission with backhaul constraints," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6265–6279, 2018.

[21] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[22] B. Jiang, S. Ma, A. M.-C. So, and S. Zhang, "Vector transport-free SVRG with general retraction for Riemannian optimization: Complexity analysis and practical implementation," *arXiv preprint arXiv:1705.09059*, 2017.

[23] M. Nouiehed, M. Sanjabi, T. Huang, J. D. Lee, and M. Razaviyayn, "Solving a class of non-convex min-max games using iterative first order methods," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 14 934–14 942.

[24] S. Lu, I. Tsaknakis, M. Hong, and Y. Chen, "Hybrid block successive approximation for one-sided non-convex min-max problems: Algorithms and applications," *IEEE Transactions on Signal Processing*, vol. 68, pp. 3676–3691, 2020.

[25] Z. Xu, H. Zhang, Y. Xu, and G. Lan, "A unified single-loop alternating gradient projection algorithm for nonconvex-concave and convex-nonconcave minimax problems," *Mathematical Programming*, vol. 201, pp. 635–706, 2023.

[26] M. Held, P. Wolfe, and H. P. Crowder, "Validation of subgradient optimization," *Mathematical Programming*, vol. 6, pp. 62–88, 1974.

[27] L. Condat, "Fast projection onto the simplex and the $l_1$ ball," *Mathematical Programming*, vol. 158, no. 1-2, pp. 575–585, 2016.

[28] N. Boumal, P.-A. Absil, and C. Cartis, "Global rates of convergence for nonconvex optimization on manifolds," *IMA Journal of Numerical Analysis*, vol. 39, no. 1, pp. 1–33, 2019.

[29] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," 2014.

[30] I.-C. Yeh and C.-h. Lien, "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2473–2480, 2009.