Gradient Descent Ascent for Minimax Problems on Riemannian Manifolds

Feihu Huang and Shangqian Gao

Abstract—In this paper, we study a class of useful minimax problems on Riemannian manifolds and propose a class of effective Riemannian gradient-based methods to solve these minimax problems. Specifically, we propose an effective Riemannian gradient descent ascent (RGDA) algorithm for the deterministic minimax optimization. Moreover, we prove that our RGDA has a sample complexity of \( O(\epsilon^2) \) for finding an \( \epsilon \)-stationary solution of the Geodesically-Nonconvex Strongly-Concave (GNSC) minimax problems, where \( \epsilon \) denotes the condition number. At the same time, we present an effective Riemannian stochastic gradient descent ascent (RSGDA) algorithm for the stochastic minimax optimization, which has a sample complexity of \( O(\epsilon^3) \) for finding an \( \epsilon \)-stationary solution. To further reduce the sample complexity, we propose an accelerated Riemannian stochastic gradient descent ascent (Acc-RSGDA) algorithm based on the momentum-based variance-reduced technique. We prove that our Acc-RSGDA algorithm achieves a lower sample complexity of \( O(\epsilon^4) \) in searching for an \( \epsilon \)-stationary solution of the GNSC minimax problems. Extensive experimental results on the robust distributional optimization and robust Deep Neural Networks (DNNs) training over Stiefel manifold demonstrate efficiency of our algorithms.

Index Terms—Deep neural networks, minimax optimization, riemannian manifolds, robust optimization, stiefel manifold

1 INTRODUCTION

In this paper, we study a class of useful minimax optimization problems on the Riemannian manifold \( \mathcal{M} \), defined as:

\[
\min_{x \in \mathcal{M}} \max_{y \in \mathcal{Y}} f(x, y),
\]

where function \( f(x, y) : \mathcal{M} \times \mathcal{Y} \to \mathbb{R} \) is \( \mu \)-strongly concave in \( y \in \mathcal{Y} \subseteq \mathbb{R}^d \) but possibly (geodesically) nonconvex in \( x \in \mathcal{M} \). Here \( \mathcal{M} \) is a Riemannian manifold, and \( \mathcal{Y} \) is a convex and closed set in euclidean space. \( f(\cdot, y) : \mathcal{M} \to \mathbb{R} \) for any \( y \in \mathcal{Y} \) is a smooth but possibly (geodesically) nonconvex real-valued function on manifold \( \mathcal{M} \), and \( f(x, \cdot) : \mathcal{Y} \to \mathbb{R} \) for any \( x \in \mathcal{M} \) is a smooth and strongly-concave real-valued function. Note that a geodesically nonconvex function on Riemannian manifold also is nonconvex on euclidean space, and a geodesically convex function on Riemannian manifold may be nonconvex on euclidean space. In this paper, we also focus on the stochastic form of minimax problem (1), defined as

\[
\min_{x \in \mathcal{M}} \max_{y \in \mathcal{Y}} \mathbb{E}_{\xi \sim \mathcal{D}} [f(x, y; \xi)],
\]

where \( \xi \) is a random variable that follows an unknown distribution \( \mathcal{D} \). In fact, Problems (1) and (2) are associated to many existing machine learning applications:

1. Robust DNNs Training Over Riemannian manifold. Deep Neural Networks (DNNs) recently have been demonstrating exceptional performance on many machine learning applications such as image classification. However, they are vulnerable to the adversarial example attacks, which show that a small perturbation in the data input can significantly change the output of DNNs. Thus, the security properties of DNNs have been widely studied. One of secured DNN research topics is to enhance the robustness of DNNs under the adversarial example attacks. Given the training sample \( \mathcal{D} := \{\xi_i = (a_i, b_i)\}_{i=1}^n \), where \( a_i \in \mathbb{R}^d \) and \( b_i \in \mathbb{R} \) represent the features and label of sample \( \xi_i \) respectively. Then we train a robust DNN against a universal adversarial attack [1], [2], which can be formulated the following minimax problem:

\[
\min_{x \in \mathbb{R}^q} \max_{y \in \mathcal{Y}} \frac{1}{n} \sum_{i=1}^n \ell(h(a_i + y; x), b_i),
\]

where \( x \in \mathbb{R}^q \) denotes weight of the DNN, and \( h(\cdot; x) \) denotes the DNN parameterized by \( x \), and \( \ell(\cdot) \) is the loss function. Here \( y \) denotes a small universal perturbation in the features \( \{a_i\}_{i=1}^n \), and the constraint \( \mathcal{Y} = \{ y : \|y\|_{\infty} \leq \varepsilon \} \) indicates that the poisoned samples should not be too different from the original ones.

Recently, the orthonormality on weights of DNNs has gained much interest and has been found to be useful across different tasks such as person re-identification [3] and image classification [4]. In fact, the orthonormality constraints improve the performances of DNNs [5], [6], and...
reduce overfitting to improve generalization [7]. At the same time, the orthonormality can stabilize the distribution of activation over layers within DNNs [8]. Thus, we further consider the following robust DNN training over the Stiefel manifold $M$:

$$
\min_{x \in \mathcal{M}} \max_{y \in \mathcal{Y}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(a_i + y; x), b_i),
$$

(4)

When data are continuously coming, we can rewrite the stochastic form of Problem (4) as follows:

$$
\min_{x \in \mathcal{M}} \max_{y \in \mathcal{Y}} \mathbb{E}_{\xi}[\ell(f(\xi; x; y), \xi)],
$$

(5)

where $f(x; y; \xi) = \ell(h(a + y; x), b)$ with $\xi = (a, b)$.

2) Distributionally Robust Optimization Over Riemannian Manifold. Distributionally Robust Optimization (DRO) [9], [10] is an effective method to deal with the noisy data, adversarial data, and imbalanced data. In the paper, we consider the DRO over the Riemannian manifold that can be applied in many machine learning problems such as robust principal component analysis (PCA) and distributionally robust DNN training. To be more specific, given a set of data samples $\{\xi_i\}_{i=1}^n$, the DRO over Riemannian manifold $\mathcal{M}$ can be written as the following minimax problem:

$$
\min_{x \in \mathcal{M}} \max_{p \in \mathcal{S}} \left\{ \sum_{i=1}^{n} p_i \ell(x; \xi_i) - \|p - \frac{1}{n}\|^2 \right\},
$$

(6)

where $p = (p_1, \ldots, p_n)$, $\mathcal{S} = \{p \in \mathbb{R}^n : \sum_{i=1}^{n} p_i = 1, p_i \geq 0\}$. Here $\ell(x; \xi_i)$ denotes the loss function over Riemannian manifold $\mathcal{M}$, which applies to many machine learning problems such as PCA [11], dictionary learning [12], DNNs [8], structured low-rank matrix learning [13], [14], [15], among others. For example, the task of PCA can be cast on a Grassmann manifold.

Recently some algorithms [16], [17], [18] have been studied for variational inequalities on Riemannian manifolds, which are the implicit minimax problems on Riemannian manifolds. Meanwhile, some methods [19], [20] for computing the projection robust Wasserstein distance, which can be represented as a minimax optimization over the Stiefel manifold [21]. To the best of our knowledge, the existing explicitly minimax optimization methods such as gradient descent ascent method only focus on the minimax problems in euclidean space.

To fill this gap, in the paper, we study the explicit minimax optimization problems over the general Riemannian manifold, and propose a class of efficient Riemannian gradient-based algorithms to solve the Geodesically-Nonconvex Strongly-Concave (GNSC) minimax problem (1) via using general retraction and vector transport. When Problem (1) is deterministic, we propose a new deterministic Riemannian gradient descent ascent algorithm. When Problem (1) is stochastic (i.e., Problem (2)), we propose two efficient stochastic Riemannian gradient descent ascent algorithms. Our main contributions can be summarized as follows:

1) We propose an effective Riemannian gradient descent ascent (RGDA) algorithm for the deterministic minimax Problem (1). Moreover, we prove that the RGDA has a sample complexity of $O(\kappa^2\epsilon^{-2})$ in finding an $\epsilon$-stationary solution of Problem (1).

2) Meanwhile, we present an effective Riemannian stochastic gradient descent ascent (RSGDA) algorithm for the stochastic minimax Problem (2), which has a sample complexity of $O(\kappa^4\epsilon^{-4})$ in searching for an $\epsilon$-stationary solution of Problem (2).

3) We further propose an accelerated Riemannian stochastic gradient descent ascent (Acc-RSGDA) algorithm based on the variance-reduced technique of STORM [22]. We prove our Acc-RSGDA achieves a lower sample complexity of $O(\kappa^4\epsilon^{-3})$.

4) Extensive experimental results on the robust DNNs training and distributionally robust optimization over Stiefel manifold demonstrate the efficiency of our proposed algorithms.

2 RELATED WORKS

In this section, we briefly review the minimax optimization and Riemannian manifold optimization, respectively.

2.1 Minimax Optimization

Minimax optimization [23] recently has been widely applied in many machine learning problems such as adversarial training [24], reinforcement learning [25], and robust federated learning [26]. Meanwhile, many efficient minimax methods [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39] have been proposed for solving these minimax optimization problems. For example, [29] proposed a class of efficient dual implicit accelerated gradient algorithms to solve smooth minimax optimization. [27] studied the convergence properties of the gradient descent ascent (GDA) methods for nonconvex minimax optimization. Subsequently, the accelerated GDA algorithms [30] have been proposed for minimax optimization. Meanwhile, [33] presented a catalyst accelerated framework for minimax optimization. Moreover, [36], [39] proposed some faster stochastic variance-reduced GDA algorithms to solve the stochastic nonconvex-strongly-concave minimax problems. [32] studied the convergence properties of GDA methods for solving a class of nonconvex-nonconcave minimax problems. More recently, a class of efficient mirror descent ascent algorithms [38] have been proposed for nonconvex nonsmooth minimax optimization.

2.2 Riemannian Manifold Optimization

Riemannian manifold optimization methods have been widely applied in machine learning problems including dictionary learning [12], low-rank matrix completion [14], [15], DNNs [8] and natural language processing [40]. Many Riemannian optimization methods have been recently proposed. E.g., [41], [42] proposed some efficient first-order gradient methods for geodesically convex functions. Subsequently, [43] presented fast stochastic variance-reduced methods to Riemannian manifold optimization. More recently, [44] proposed fast first-order gradient algorithms for Riemannian manifold optimization by using general retraction and vector transport. Subsequently, based on these retraction and vector transport, some fast Riemannian gradient-based methods [11], [45], [46], [47], [48] have been
have been introduced for matrix mani-

and have been studied for variational inequalities on Rie-

Here we focus on the isometric vector transport

mapping tangent space

\( T \) of the second variable with fixing

\( y \). The

\( L \)

is commonly used in Riemannian optimi-

\( T \) denotes the derivative of \( L \), and \( \gamma \) denotes the inner

product; \( \nabla \) and the partial gradient

\( \Delta \) are tangent vectors at

\( u \). For \( f(x, y) \), \( f(x, \cdot) \), \( f(\cdot, y) \) denote function w.r.t. the second variable with fixing \( x \), and \( f(\cdot, y) \) denotes function w.r.t. the first variable with fixing \( y \). Given a convex closed set \( Y \), we define a projection operation on the set \( Y \) as \( P_Y(y) = \text{arg min}_{y \in Y} \frac{1}{2} \| y - y_0 \|^2 \).

We denote \( a = O(b) \) if \( a \leq Cb \) for some constant \( C > 0 \), and the notation \( O(\cdot) \) hides logarithmic terms. The operation \( \bigoplus \) denotes the Whitney sum that takes two vector bundles over a fixed space and produces a new vector bundle over the same space. Given function \( f(x) \), let \( grad(f(x)) \) denote its Riemannian gradients at Riemannian manifold and \( \nabla f(x) \) denote its gradients at euclidean space. Given \( B_1 = \{ \xi \}_i \) for any \( t \geq 1 \), let \( \nabla f_{B_1}(x) = \frac{1}{t} \sum_{i=1}^B \nabla f(x; \xi_i) \) and \( grad f_{B_1}(x) = \frac{1}{B} \sum_{i=1}^B grad f(x; \xi_i) \).

3 Preliminaries

In this section, we first re-visit some basic information on the Riemannian manifold \( M \). In general, the manifold \( M \) is endowed with a smooth inner product \( \langle \cdot, \cdot \rangle_\varepsilon : T_xM \cdot T_yM \rightarrow \mathbb{R} \) on tangent space \( T_xM \) for every \( x \in M \). The induced norm \( \| \cdot \|_\varepsilon \) of a tangent vector in \( T_xM \) is associated with the Riemannian metric. We first define a retraction \( R_x : T_xM \rightarrow M \) mapping tangent space \( T_xM \) onto \( M \) with a local rigidity condition that preserves the gradients at \( x \in M \) (please see Fig. 1a). The retraction \( R_x \) satisfies all of the following: 1) \( R_x(0) = x \), where \( 0 \in T_xM \); 2) \( DR_x(0) = \text{id}_{T_xM} \), where \( DR_x \) denotes the derivative of \( R_x \), and \( \text{id}_{T_xM} \) denotes an identity mapping on \( T_xM \). In fact, exponential mapping \( \exp_x \) is a special case of retraction \( R_x \) that locally approximates the exponential mapping \( \exp_x \) to the first order on the manifold.

Next, we define a vector transport \( T : T_xM \bigoplus T_yM \rightarrow T_zM \) (please see Fig. 1b) that satisfies all of the following 1) \( T \) has an associated retraction \( R_x \), i.e., for \( x \in M \) and \( w, u \in T_xM, T_uw \) is a tangent vector at \( R_x(w) \); 2) \( T(u) = v \); 3) \( T(u + bw) = aT(u) + bT_uw \) for all \( a, b \in \mathbb{R}, u, v, w \in T_xM \). Vector transport \( T^\gamma_{xv} \) or equivalently \( T_{uv} \) with \( y = R_x(u) \) transports \( v \in T_xM \) along the retraction curve defined by direction \( u \). Here we focus on the isometric vector transport \( T^\gamma_{xv} \), which satisfies \( \langle u, v \rangle_z = \langle T^\gamma_{xv} u, T^\gamma_{xv} v \rangle \) for all \( u, v \in T_xM \). Based on these definitions, we provide some standard assumptions about Problems (1) and (2).

**Assumption 1.** \( X \subseteq M \) is compact. Each component function \( f_0(x, y) \) is twice continuously differentiable in both \( x \in X \), \( y \in Y \), and there exist constants \( L_{11}, L_{12}, L_{21}, L_{22} \), such that for every \( x, x_1, x_2 \in X \) and \( y, y_1, y_2 \in Y \), we have

\[
\| \text{grad}_x f(x_1, y; \xi) - \text{grad}_x f(x_2, y; \xi) \| \leq L_{11} \| u \|,
\]

\[
\| \text{grad}_x f(x_1, y; \xi) - \text{grad}_x f(x_2, y; \xi) \| \leq L_{12} \| y_1 - y_2 \|,
\]

\[
\| \nabla_y f(x_1, \cdot; \xi) - \nabla_y f(x_2, \cdot; \xi) \| \leq L_{21} \| u \|.
\]

\[
\| \nabla_y f(x_1, \cdot; \xi) - \nabla_y f(x_2, \cdot; \xi) \| \leq L_{22} \| y_1 - y_2 \|.
\]

where \( u \in T_{x_1}M \) and \( x_2 = R_{x_1}(u) \).

**Assumption 1** is commonly used in Riemannian optimization [11], [44], and minimax optimization [27], [36]. Here, the terms \( L_{11}, L_{12}, L_{21} \) implicitly contain the curvature information as in [11], [44]. Specifically, Assumption 1 implies the partial Riemannian gradient \( \text{grad}_x f(\cdot, y; \xi) \) for all \( y \in Y \) is \( L_{11} \)-Lipschitz continuous with respect to retraction as in [11] and the partial gradient \( \nabla_y f(x, \cdot; \xi) \) for all \( x \in X \) is \( L_{22} \)-Lipschitz continuous as in [27].

To further verify the rationality of Assumption 1, we consider the Stiefel manifold \( \mathcal{M} = \{ X \in \mathbb{R}^{d \times r} | X^T X = I_r \} \). For notational simplicity, let matrix \( X \) instead of the variable \( x \) in Assumption 1. Let \( \nabla X f(X, y) \) denote the gradient of \( f(X) \) on variable \( X \) in the euclidean space, and \( \text{grad}_X f(\cdot, y) \) denote the Riemannian gradient of \( f(X, y) \) on \( \mathcal{M} \) and \( \text{grad}_X f(X, y) \) can be seen as a projection onto the tangent space \( T_X M \) of Riemannian \( M \) at \( X \), which can be computed as follows:
\[
\text{grad}_X f(X, y) = P_{TX} (\nabla_X f(X, y)) = WX,
\]
\[
W = \bar{W} - \bar{T},
\]
\[
\bar{W} = \nabla_X f(X, y)X^T - \frac{1}{2}X^T \nabla_X f(X, y)X^T. \tag{7}
\]

Then we have for any \(X_1, X_2 \in \mathcal{M},\)
\[
\|\text{grad}_X f(X_1, y) - T_{\bar{X}_2}^2 \text{grad}_X f(X_2, y)\|
\]
\[
= \|P_{T_{\bar{X}_2}} (\nabla_X f(X_1, y)) - T_{\bar{X}_2}^2 P_{T_{\bar{X}_2}} (\nabla_X f(X_2, y))\|
\]
\[
\leq \|\nabla_X f(X_1, y) - \nabla_X f(X_2, y)\| \leq L \|X_1 - X_2\|, \tag{8}
\]
where the last inequality holds by Lipschitz continuous for gradient in the euclidean space. Let \(d(X_1, X_2)\) denote geodesic distance between \(X_1\) and \(X_2\) in \(\mathcal{M}\), then we have \(d(X_1, X_2) = \xi \|X_1 - X_2\|\), where \(\xi > 0\) denote curvature parameter of manifold \(\mathcal{M}\). In our Assumption 1, due to \(X_2 = R_{X_1}(u)\), we have \(\|d\| = d(X_1, X_2)\). According to the above (8), we have
\[
\|\text{grad}_X f(X_1, y) - T_{\bar{X}_2}^2 \text{grad}_X f(X_2, y)\|
\]
\[
\leq L \|X_1 - X_2\| = \frac{L}{\xi} d(X_1, X_2) = \frac{L}{\xi} \|u\|, \tag{9}
\]
where \(X_2 = R_{X_1}(u)\). This similarly holds for the other inequalities in our Assumption 1.

For the deterministic problem, let \(f(x, y)\) instead of \(f(x, y, \xi)\) in Assumption 1. In fact, these Lipschitz continuity assumptions are widely applicable to deep learning architectures [5]. Note that in the following experiments, given the DNNs using ReLU, the derivative of ReLU is Lipschitz continuous almost everywhere with an appropriate Lipschitz constant, except for a small neighbourhood around 0, whose measure tends to 0. Such cases do not affect either analysis in theory or training in practice.

Since \(f(x, y)\) is strongly concave in \(y \in \mathcal{Y}\), there exists a unique solution to the problem \(\max_{y \in \mathcal{Y}} f(x, y)\) for any \(x\). We define the function \(\Phi(x) = \max_{y \in \mathcal{Y}} f(x, y)\) and \(y^*(x) = \arg \max_{y \in \mathcal{Y}} f(x, y)\).

**Assumption 2.** The function \(\Phi(x) : \mathcal{M} \to \mathbb{R}\) is \(L\)-smooth. \(\)There exists a constant \(L > 0\), for all \(x \in \mathcal{X}, z = R_x(u)\) with \(u \in T_z\mathcal{M}\), such that
\[
\Phi(x) \leq \Phi(x) + \langle \text{grad}\Phi(x), u \rangle + \frac{L}{2} \|u\|^2.
\]

**Assumption 3.** The objective function \(f(x, y)\) is \(\mu\)-strongly concave w.r.t \(y\), i.e., for any \(x \in \mathcal{M}, y_1, y_2 \in \mathcal{Y}\)
\[
f(x, y_1) \leq f(x, y_2) + \langle \nabla_y f(x, y_2), y_1 - y_2 \rangle - \frac{\mu}{2} \|y_1 - y_2\|^2.
\]

**Assumption 4.** The function \(\Phi(x)\) is bounded below in \(\mathcal{M}\), i.e., \(\Phi^* = \inf_{x \in \mathcal{M}} \Phi(x)\).

**Assumption 5.** The variance of stochastic gradient is bounded, i.e., there \(\)exists a constant \(\sigma_1 > 0\) such that for all \(x\), it follows
\[
E_\xi \|\text{grad}_X f(x, y; \xi) - \text{grad}_X f(x, y)\|^2 \leq \sigma_1^2;
\]
there exists a constant \(\sigma_2 > 0\) such that for all \(x\), it follows \(E_\xi \|\nabla_y f(x, y; \xi) - \nabla_y f(x, y)\|^2 \leq \sigma_2^2\). We also define \(\sigma = \max\{\sigma_1, \sigma_2\}\).

Assumption 2 imposes the smooth of function \(\Phi(x)\) over Riemannian manifold \(\mathcal{M}\), as in [11], [44], [48]. Assumption 3 imposes the strongly concave of \(f(x, y)\) on variable \(y\), as in [27], [36]. Assumption 4 guarantees the feasibility of the GNNS minimax problem (1), as the nonconex-strongly-concave minimax optimization on euclidean space used in [27], [36]. Assumption 5 imposes the bounded variance of stochastic (Riemannian) gradients, which is commonly used in the stochastic optimization [27], [36], [48].

**Algorithm 1.** RGDA and RSGDA Algorithms
\(\)
1: **Input:** \(T,\) parameters \(\{\gamma, \lambda, \eta_i\}_{i=1}^T\), mini-batch size \(B\), and initial input \(x_1 \in \mathcal{M}, y_1 \in \mathcal{Y}\);
2: for \(t = 1, 2, \ldots, T\) do
3: \hspace{10pt} (RGDA) Compute deterministic gradients
\(
\begin{align*}
\n v_t &= \text{grad}_x f(x_t, y_t), \quad w_t = \nabla_y f(x_t, y_t);
\end{align*}
\)
4: \hspace{10pt} (RSGDA) Draw \(B\) i.i.d. samples \(\{\xi_i\}_{i=1}^B\), then compute stochastic gradients
\(
\begin{align*}
\n v_t &= \frac{1}{B} \sum_{i=1}^B \text{grad}_x f(x_t, y_t, \xi_i), \\
 w_t &= \frac{1}{B} \sum_{i=1}^B \nabla_y f(x_t, y_t, \xi_i);
\end{align*}
\)
5: Update: \(x_{t+1} = R_{x_t}(-\gamma \eta_t v_t);\)
6: Update: \(y_{t+1} = P_Y (y_t + \lambda w_t)\) and \(y_{t+1} = y_t + \eta_t (\tilde{y}_{t+1} - y_t);\)
7: end for
8: Output: \(x_T\) and \(y_T\) chosen uniformly random from \(\{x_t, y_t\}_{t=1}^T\).

**4 RIEMANNIAN GRADIENT-BASED METHODS**

In this section, we propose a class of Riemannian gradient-based methods to solve the deterministic and stochastic GNNS minimax problems (1) and (2), respectively.

**4.1 RGDA and RSGDA Algorithms**

In this subsection, we propose an efficient Riemannian gradient descent ascent (RGDA) algorithm to solve the deterministic minimax Problem (1). At the same time, we propose a standard Riemannian stochastic gradient descent ascent (RSGDA) algorithm to solve the stochastic minimax Problem (2). Algorithm 1 summarizes the algorithmic framework of our RGDA and RSGDA algorithms.

At the line 3 of Algorithm 1, we calculate the deterministic Riemannian gradient in variable \(x \in \mathcal{M}\), and calculate the deterministic gradient in variable \(y \in \mathcal{Y}\). At the line 4 of Algorithm 1, we calculate the stochastic Riemannian gradient for \(x \in \mathcal{M}\), and calculate the stochastic gradient for variable \(y \in \mathcal{Y}\).

At the line 5 of Algorithm 1, we use the Riemannian gradient descent to update variable \(x\) based on the retraction operator \(R_{x_t}()\), which guarantees the variable \(x_t\) for all \(t \geq 1\) in the manifold \(\mathcal{M}\). Here \(R_{x_t}()\) can be seen as a generalized projection operator, which can be competent to the general Riemannian manifolds. For example, we consider the popular Stiefel manifold \(\mathcal{M} = S(t, d) = \{X \in \mathbb{R}^{d \times n} : X^TX = I_n\}\) that is a nonconvex constraint set in the euclidean space. Given \(q_t = -\gamma \eta_t v_t \in T_{x_t} \mathcal{M}\), we can define a standard QR-based retraction: \(R_{x_t}(q_t) = \mathcal{Q}H,\) where the matrices \(Q\) and \(H\) can be obtained from the QR
decomposition of matrix $x_i + g_i \in \mathbb{R}^{d \times r}$, i.e., $x_i + g_i = QR$, and $H = \text{diag}((\text{sign}(R_{i1}))_{i=1}^r)$. It is well known that the standard projected gradient methods with convergence guarantee require the convex constraint sets belonging to euclidean space [51], while our Riemannian gradient-based methods with convergence guarantee do not need the convex constraint sets (Please see the following convergence analysis).

At the line 6 of Algorithm 1, we simultaneously use a projection iteration and a momentum iteration to update the variable $y_t$, where we use $0 < \eta_t \leq 1$ to ensure the variable $y_t$ for all $t \geq 1$ in convex constraint $\mathcal{Y}$. Note that we use two learning rates $\gamma$ and $\eta_t$ at the line 5, where $\gamma$ is a constant learning rate and $\eta_t$ is a dynamic or constant learning rate with iteration $t$. Under this case, we can flexibly choose learning rates in practice, and can easily analyze the convergence properties of our algorithms, where simultaneously Riemannian gradient descent on the variable $x \in M$ and gradient ascent on the variable $y \in \mathcal{Y}$.

4.2 ACC-RSGDA Algorithm

In this subsection, we propose an accelerated stochastic Riemannian gradient descent ascent (Acc-RSGDA) algorithm to solve the stochastic minimax Problem (2), which builds on the momentum-based variance reduction technique of STORM [22]. Algorithm 2 describes the algorithmic framework of Acc-RSGDA method.

Algorithm 2. Acc-RSGDA Algorithm

1: Input: $T$, parameters $\{\gamma, \lambda, b, m, c_1, c_2\}$ and initial input $x_1, y_1 \in M$ and $y_1 \in \mathcal{Y}$;
2: Draw $B$ i.i.d. samples $B_t = \{x_{i1}\}_{i=1}^B$, then compute $v_1 = \text{grad}_f f_{B_1}(x_1, y_1)$ and $w_1 = \nabla_y f_{B_1}(x_1, y_1)$;
3: for $t = 1, 2, \ldots, T$ do
4: Update: $x_{t+1} = R_{x_t}(\gamma v_t w_t)$ with $\eta_t = \frac{b}{(m+\delta)^2}$;
5: Update: $\tilde{y}_{t+1} = P_y (y_t + \lambda w_t)$ and $y_{t+1} = y_t + \eta_t (\tilde{y}_{t+1} - y_t)$;
6: Draw $B$ i.i.d. samples $B_{t+1} = \{x_{i1}\}_{i=1}^B$, then compute
7: $v_{t+1} = \text{grad}_f f_{B_1}(x_{t+1}, y_{t+1}) + (1 - \alpha_{t+1})$
8: $\cdot T_{x_{t+1}}^{\frac{1}{2}} [v_t - \text{grad}_f f_{B_1}(x_t, y_t)]$, \hspace{1cm} (10)
9: $w_{t+1} = \nabla_y f_{B_1}(x_{t+1}, y_{t+1}) + (1 - \beta_{t+1})$
10: $\cdot [w_t - \nabla_y f_{B_1}(x_t, y_t)]$, \hspace{1cm} (11)
11: where $\alpha_{t+1} = c_1 \eta_t^2$ and $\beta_{t+1} = c_2 \eta_t^2$.
12: end for
13: Output: $x_T$ and $y_T$, chosen uniformly random from $\{x_t, y_t\}_{t=1}^T$.

At the line 4 of Algorithm 2, we use two learning rates $\gamma$ and $\eta_t$, where $\gamma$ is a constant learning rate and $\eta_t = \frac{b}{(m+\delta)^2}$ is a decreasing learning rate with iteration $t$. Similarly, we can flexibly choose learning rates in practice, and can easily analyze the convergence properties of our algorithms, where simultaneously Riemannian gradient descent on the variable $x \in M$ and gradient ascent on the variable $y \in \mathcal{Y}$.

At the line 6 of Algorithm 2, we use the momentum-based variance-reduced technique of STORM to estimate stochastic Riemannian gradient $v_t$ defined in (10), where $\alpha_{t+1} \in (0, 1)$. When $\alpha_{t+1} = 1$, $v_{t+1} = \text{grad}_f f_{B_1}(x_{t+1}, y_{t+1})$ will degenerate a vanilla stochastic Riemannian gradient estimator; When $\alpha_{t+1} = 0$, $v_{t+1} = \text{grad}_f f_{B_1}(x_{t+1}, y_{t+1}) - T_{x_{t+1}}^{\frac{1}{2}}(\text{grad}_f f_{B_1}(x_t, y_t) - v_t)$ will degenerate a stochastic Riemannian gradient estimator based on variance-reduced technique of SPIDER [52]. Since our Acc-RSGDA algorithm uses variance-reduced technique of STORM to estimate the stochastic gradients, it does not rely on large mini-batch size to guarantee its convergence (Please see the following convergence analysis).

Riemannian gradient $\text{grad}_f f_{B_1}(x_{t+1}, y_{t+1})$ is over the tangent space $T_{x_t} M$, while the Riemannian gradient estimator $\text{grad}_f f_{B_1}(x_t, y_t) - v_t$ is over the tangent space $T_x M$. In order to feasibility of $v_{t+1}$, we use the vector transport $T_{x_{t+1}}$ to project the Riemannian gradient estimator $\text{grad}_f f_{B_1}(x_t, y_t) - v_t$ into the tangent space $T_{x_{t+1}} M$. Thus, we can add the term $\text{grad}_f f_{B_1}(x_{t+1}, y_{t+1})$ and the term $(1 - \alpha_{t+1})T_{x_{t+1}}^{\frac{1}{2}}[v_t - \text{grad}_f f_{B_1}(x_t, y_t)]$.

4.3 Novelities of Our Algorithms

Compared with the existing Riemannian gradient algorithms [11, 45, 46] and minimax optimization algorithms [27, 36], our algorithms have the following main differences:

1) Compared with the existing Riemannian gradient algorithms, our algorithms simultaneously use a constant learning rate $\gamma$ and a dynamic or constant learning rate $\eta_t$ at each iteration. This dynamic/constant learning rate $\eta_t$ is the same tuning parameter of the momentum iteration in updating variable $y$ (i.e., $y_{t+1} = y_t + \eta_t (\tilde{y}_{t+1} - y_t)$). In other words, the learning rate in updating the variable $x \in M$ depends on the tuning parameter of the momentum iteration in updating dual variable $y$.

2) Compared with the existing minimax optimization algorithms, our algorithms simultaneously use a projection iteration and a momentum iteration to update the variable $y$. Meanwhile, our algorithms use the Riemannian gradients and retraction operator to update variable $x \in M$ instead of the standard gradients and projection operator used in the existing minimax algorithms.

5 Convergence Analysis

In this section, we study the convergence properties of our RGDA, RSGDA, and Acc-RSGDA algorithms, respectively. The basic idea of our convergence analysis is given in Fig. 2. We first give some useful lemmas.

Lemma 1. Under the above assumptions, the gradient of function $\Phi(x) = \max_{y \in \mathcal{Y}} f(x, y)$ is $G$-Lipschitz with respect to retraction, and the mapping or function $y^*(x) = \arg \max_{y \in \mathcal{Y}} f(x, y)$ is $\kappa$-Lipschitz with respect to retraction. Given any $x_1, x_2 \in X \subseteq M$ and $u \in T_x M$, we have:

$$\|\text{grad}\Phi(x_1) - T_{x_2}^{\frac{1}{2}}\text{grad}\Phi(x_2)\| \leq G \|u\|,$$

$$\|y^*(x_1) - y^*(x_2)\| \leq \kappa \|u\|,$$

where $x_2 = R_{x_1}(u)$, and $G = \kappa L_{12} + L_{11}$, and $\kappa = L_{21}/\mu$ denotes the number condition of function $f(x, y)$.\]

Lemma 2. Suppose the sequence $(x_t, y_t)_{t=1}^T$ is generated from Algorithm 1 or 2. Given $0 < \eta_t \leq \mu/2\Gamma$, we have
The basic idea of our convergence analysis

### Investigating the variances of estimated gradients on the variables $x$ and $y$:
1) RGDA, No Applied
2) RSGDA, By Assumption 5
3) Acc-RSGDA, In Lemma 4

### Tracking the error terms of the variables $x$ and $y$ in RGDA, RSGDA, Acc-RSGDA:
1) For $x$, In Lemma 2
2) For $y$, In Lemma 3

### Establishing useful Lyapunov functions for RGDA, RSGDA and Acc-RSGDA algorithms based on variances of estimated gradients and error terms of variables.

Based on Lyapunov functions, choosing the suitable tuning parameters such as learning rates to obtain convergence results.

**Fig. 2. The basic idea of our convergence analysis.**

\[ \Phi(x_{t+1}) \leq \Phi(x_t) + \gamma L_2 y_t \|g'(x_t) - y_t\|^2 - \frac{\gamma L_2}{2} \|y_t\|^2 \]
\[ + \gamma y_t \|\nabla f(x_t, y_t) - v_t\|^2 - \frac{\gamma y_t}{2} \|\nabla \Phi(x_t)\|^2. \quad (14) \]

**Lemma 3.** Suppose the sequence \{x_t, y_t\} is generated from Algorithm 1 or 2. Under the above assumptions, and set \( 0 < \eta_t < 1 \) and \( 0 < \lambda \leq \frac{1}{6\lambda_0} \), we have

\[ \|y_{t+1} - g'(x_{t+1})\|^2 \leq \left( 1 - \frac{\eta_t \mu \lambda}{4} \right) \|y_t - g'(x_t)\|^2 - \frac{3 \eta_t}{4} \|y_{t+1} - y_t\|^2 \]
\[ + \frac{25 \eta_t \lambda}{6 \mu} \|\nabla g(x_t, y_t) - w_t\|^2 + \frac{25 \eta_t^2 \lambda^2}{6 \mu} \|v_t\|^2, \quad (15) \]

where \( \lambda = \frac{L_2}{\mu} \) and \( L = \max(1, L_1, L_2), L_1, L_2). \)

Although Problems (1) and (2) are nonconvex, following [53], there exists a local solution or stationary point \((x^*, y^*)\) satisfies the Nash Equilibrium, i.e., \( f(x^*, y^*) \leq f(x^*, y_t) \leq f(x, y^*) \) for all \( t \geq 1 \). Hence, the nonconvex minimax problem (1) is equivalent to minimizing the nonconvex function \( \Phi(x) = \max_{y \in Y} f(x, y) \) for any \( x \in X \). It is NP hard to find the global minimum of \( \Phi(x) \) in general since \( \Phi(x) \) is nonconvex in \( x \in X \). Thus, we will find the stationary points of function \( \Phi(x) \), which is equal to the stationary points of the minimax problem (1). Next we define an \( \epsilon \)-stationary point of \( \Phi(x) \) in \( x \in X \).

**Definition 1.** A point \( x \in X \) is an \( \epsilon \)-stationary point \((\epsilon > 0)\) of a differentiable function \( \Phi(x) \) if \( \|\nabla \Phi(x)\| \leq \epsilon \). If \( \epsilon = 0 \), then \( x \) is a stationary point.

### 5.1 Convergence Analysis of Both RGDA and RSGDA Algorithms

In this subsection, we study the convergence properties of our RGDA and RSGDA algorithms, respectively.

Suppose the sequence \{x_t, y_t\} is generated from our RGDA Algorithm, we establish a useful Lyapunov function (i.e., potential function) \( \Lambda_t \) for convergence analysis of RGDA, defined as

\[ \Lambda_t = \Phi(x_t) + \frac{6 \gamma L_2}{\lambda \mu} \|y_t - g'(x_t)\|^2, \quad \forall t \geq 1. \quad (16) \]

**Theorem 1.** Suppose the sequence \{x_t, y_t\} is generated from Algorithm 1 by using deterministic gradients. Given \( y_t = g'(x_t), \eta = \eta_t \) for all \( t \geq 1, 0 < \eta \leq \min(1, \frac{1}{L_2}), 0 < \lambda \leq \frac{1}{6\lambda_0} \), we have

\[ \frac{1}{T} \sum_{t=1}^{T} \|\nabla \Phi(x_t)\| \leq \frac{2 \sqrt{\Phi(x_1) - \Phi^*}}{\sqrt{\eta T}}. \quad (17) \]

**Remark 1.** Since \( 0 < \eta \leq \min(1, \frac{1}{L_2}) \) and \( 0 < \gamma \leq \frac{\mu}{10 L_4} \), we have \( 0 < \eta \gamma \leq \min(\frac{\mu}{10 L_4}, \frac{1}{2\gamma}). \) Let \( \eta \gamma = \min(\frac{\mu}{10 L_4}, \frac{1}{2\gamma}) \), we have \( \eta \gamma = O(\frac{1}{\sqrt{T}}) \). The RSGDA algorithm has convergence rate of \( O(\frac{1}{\sqrt{T}}) \). By \( \gamma \leq \frac{1}{2\gamma} \), \( \|\nabla \Phi(x_t)\| \leq \epsilon \), we choose \( T \geq \epsilon^2 \). When our RSGDA Algorithm solves the deterministic minimax Problem (1), we only need one sample to execute the gradients \( v_t \) and \( w_t \) at each iteration, and need \( T \) iterations. Thus, our RSGDA reaches a sample complexity of \( T = O(\epsilon^2) \) for finding an \( \epsilon \)-stationary point of Problem (1). Note that since the function \( f(x, y) \) is \( \mu \)-strongly concave in \( y \in \mathcal{Y} \), given any initial input \( x_1 \), we can easily obtain \( y_1 \approx y^*(x_1) \). So we can assume \( y_1 = y^*(x_1) \).

Suppose the sequence \{x_t, y_t\} is generated from our RSGDA Algorithm, we establish a useful Lyapunov function \( \Theta_t \) for convergence analysis of RSGDA, defined as

\[ \Theta_t = \mathbb{E} \left[ \Phi(x_t) + \frac{6 \gamma L_2}{\lambda \mu} \|y_t - g'(x_t)\|^2 \right], \quad \forall t \geq 1. \quad (18) \]

**Theorem 2.** Suppose the sequence \{x_t, y_t\} is generated from Algorithm 1 by using stochastic gradients. Given \( y_t = g'(x_t), \eta = \eta_t \) for all \( t \geq 1, 0 < \eta \leq \min(1, \frac{1}{L_2}), 0 < \lambda \leq \frac{1}{6\lambda_0} \),
and $0 < \gamma \leq \frac{\mu \lambda}{10L\alpha}$, we have

$$1 \frac{T}{T} \sum_{t=1}^{T} \mathbb{E}\|\| \Phi(x_t)\| \leq 2\sqrt{\Phi(x_1) - \Phi^*} \sqrt{\gamma \mu T} + \left(1 + \frac{5L}{\mu} \right) \sqrt{2 \sigma} \sqrt{B}$$

(19)

**Remark 2.** Since $0 < \eta \leq \min(1, \frac{\mu \lambda}{10L\alpha})$ and $0 < \gamma \leq \frac{\mu \lambda}{10L\alpha}$, we have $0 < \eta \gamma \leq \min(\frac{\mu \lambda}{10L\alpha}, \frac{1}{10L\alpha})$. Let $\eta \gamma \leq \min(\frac{\mu \lambda}{10L\alpha}, \frac{1}{10L\alpha})$, we have $\gamma = O(\frac{1}{\mu \lambda})$. Let $B = T$, the RSGDA algorithm has convergence rate of $O(\frac{1}{\sqrt{T \gamma \mu}})$. By $\frac{\mu \lambda}{10L\alpha} \leq \epsilon$, i.e., $\mathbb{E}\|\| \Phi(x_t)\| \leq \epsilon$, we choose $T \geq k^4 e^{-3}$. When our RGDGALgorithm solves the stochastic minimax Problem (2), we need $B$ samples to estimate the stochastic gradients $v_t$ and $w_t$ at each iteration, and need $T$ iterations. Thus, the RSGDA reaches a sample complexity of $BT = O(k^4 e^{-3})$ for finding an $\epsilon$-stationary point of Problem (2).

### 5.2 Convergence Analysis of ACC-RSGDA Algorithm

In the subsection, we provide the convergence properties of our ACC-RSGDA algorithm.

**Lemma 4.** Suppose the stochastic gradients $v_t$ and $w_t$ is generated from Algorithm 2, given $0 < \alpha_{t+1} \leq 1$ and $0 < \beta_{t+1} \leq 1$, we have

$$\mathbb{E}\|\| \| \Phi(x_t + 1, y_t + 1) - v_t\| \leq 4(1 - \alpha_{t+1})^2 L^2 \| y_t \|^2 E\|\| v_t\| \leq \left(1 - \alpha_{t+1}\right)^2 E\|\| \Phi(x_t, y_t) - v_t\|$$

$$+ \left(1 - \alpha_{t+1}\right)^2 E\|\| \Phi(x_t, y_t) - v_t\| + (1 - \alpha_{t+1})^2 E\|\| \Phi(x_t, y_t) - v_t\|$$

$$+ 4(1 - \alpha_{t+1})^2 L^2 \| y_t \|^2 E\|\| v_t - w_t\|^2 + \frac{2\sigma^2}{\epsilon \lambda T} \cdot B$$

(20)

$$\|\| \nabla_y f(x_t, y_t + 1) - w_t\| \leq 4(1 - \beta_{t+1})^2 L^2 \| y_t \|^2 E\|\| v_t\|$$

$$+ (1 - \beta_{t+1})^2 E\|\| \nabla_y f(x_t, y_t) - w_t\|$$

$$+ 4(1 - \beta_{t+1})^2 L^2 \| y_t \|^2 E\|\| v_t - w_t\|^2 + \frac{2\sigma^2}{\epsilon \lambda T} \cdot B$$

(21)

Assume the sequence $\{x_t, y_t\}_t$ is generated from our ACC-RSGDA Algorithm, we establish a useful *Lyanunov* function $\Omega_t$ for convergence analysis of ACC-RSGDA, defined as

$$\Omega_t = \mathbb{E}\left[\Phi(x_t) + \frac{\gamma}{2 \lambda \mu \eta_t} \left(\|\| \Phi(x_t, y_t) - v_t\| \right)^2$$

$$+ \|\| \nabla_y f(x_t, y_t) - w_t\| \right] + \frac{6\gamma L^2}{\lambda \mu} \| y_t - y^* \left(\|\| x_t\| \right)^2$$

(22)

**Theorem 3.** Suppose the sequence $\{x_t, y_t\}_t$ is generated from Algorithm 2. Given $y_t = \eta_t(x_t), c_1 \geq \frac{2}{\lambda \mu} + 2 \mu \lambda, c_2 \geq \frac{2}{\lambda \mu} + \frac{50mL^2}{\mu^2}, b > 0$, $m \geq \max(2, \sqrt{\hat{d}})$, $0 < \gamma \leq \frac{\mu \lambda}{2L \sqrt{25 + 4\mu}}$ and $0 < \lambda \leq \frac{1}{bL}$, we have

$$1 \frac{T}{T} \sum_{t=1}^{T} \mathbb{E}\|\| \Phi(x_t)\| \leq \frac{\sqrt{2T \mu m/1 \theta}}{\sqrt{T^{1/2}}} + \frac{\sqrt{2M}}{T^{1/3}} \cdot \sqrt{\frac{\gamma^2}{\mu \lambda \gamma T}}$$

(23)

where $\hat{d} = \max(1, c_1, c_2, 2\mu L)$ and $M' = \frac{2\Phi(\Phi(x_1) - \Phi^*)}{\theta b} + \frac{2\sigma^2}{\epsilon \lambda T} + \frac{\gamma L^2}{\epsilon \lambda T} \cdot \log(m)$. We have $\lambda = O(1), \mu = O(\frac{1}{\theta}), c_1 = O(1), c_2 = O(\theta), m = O(\beta^3)$ and $\eta_0 = O(\frac{1}{\theta})$. Without loss of generality, let $T \geq m = O(\beta^3)$, we have $M' = O(\beta^2 + \lambda \| x^* \|^2 + \frac{\lambda \| x^* \|^2}{\theta b \lambda T})$. When $B = \lambda$, we have $M' = O(\beta^2 + \lambda \| x^* \|^2)$. Thus, the ACC-RSGDA algorithm has a convergence rate of $O(\frac{1}{\sqrt{T \gamma \mu}})$. By $\frac{\mu \lambda}{10L\alpha} \leq \epsilon$, i.e., $\mathbb{E}\|\| \Phi(x_t)\| \leq \epsilon$, we choose $T \geq k^4 e^{-3}$. When Algorithm 2, we require $B$ samples to estimate the stochastic gradients $v_t$ and $w_t$ at each iteration, and need $T$ iterations. Thus, the ACC-RSGDA has a sample complexity of $BT = O(k^4 e^{-3})$ for finding an $\epsilon$-stationary point of Problem (2). Since our ACC-RSGDA algorithm uses variance-reduced technique of STORM to estimate the stochastic gradients, it does not rely on large mini-batch size to guarantee its convergence. When $B = 1$, our ACC-RSGDA algorithm has a convergence rate of $O(\frac{1}{\sqrt{T \gamma \mu}})$, and has a sample complexity of $BT = O(k^4 e^{-3})$ for finding an $\epsilon$-stationary point.

**Remark 4.** In the above theoretical analysis, we only assume the convexity of constraint set $\mathcal{Y}$, while [27] not only assume the convexity of set $\mathcal{Y}$, but also assume and use its bounded (i.e., $|y| \leq D$, where $D$ is a positive constant.) to guarantee convergence of the GDA and SGD algorithms in [27] (Please see Assumption 4.2 in [27]). Clearly, our assumption is milder than [27]. When there does not exist a constraint set on parameter $y$, i.e., $\mathcal{Y} = \mathbb{R}^d$, our RGDGA and RSGDA algorithms and theoretical results still work, while [27] can not work.

### 6 EXPERIMENTS

In this section, we conduct experiments on two tasks: 1) robust DNNs training over Riemannian manifold and distributionally robust optimization over Riemannian manifold. In the experiment, we use the SGD [27] and Acc-MDA [39] as the comparison baselines. Since the SGD and Acc-MDA methods are not designed for optimization on Riemannian manifolds, we add the retraction operation (projection-like) at the end of parameter updates.

#### 6.1 Robust DNNs Training

In this subsection, we focus on the robust DNNs training over Riemannian manifold defined in Problem (4), which is a nonconvex and nonconcave minimax problem. Following [28], we cast the original robust training problem into the following nonconvex-(strongly)-concave problem:

$$\min_{x \in \mathcal{M}} \max_{u \in \mathcal{U}} \frac{1}{n} \sum_{i=1}^{n} \sum_{i=1}^{C} u_i f(b(h(x_i^k; x), b_i) - r(u),$$

$$s.t. \mathcal{U} = \{u \in \mathbb{R}^C \mid u \geq 0, \|u\|_1 = 1\}.$$  

(24)

where $a^k$ is the permuted sample after $K$ iterations of Projected Gradient Descent (PGD) [54] attack, and $C$ is the number of classes for the dataset. Here $r(u)$ is a (strongly) convex regularization term, e.g., $r(u) = a\|u - 1/C\|^2$ or KL divergence $r(u) = a\sum_{i=1}^{C} u_i \log(u_i/C)$, where $a \geq 0$ is a tuning parameter. In the experiment, we use Stiefel manifold $\mathcal{M} = \text{St}(r, d) = \{X \in \mathbb{R}^{d \times T} : X^TX = L\}$ on parameters $x$ of DNNs (convolution layers and linear layers).
For robust training, we choose five datasets for this experiment: MNIST, FashionMNIST, CIFAR10, CIFAR100, and STL10. We use a 5-layer DNN as the target model, whose architecture is given in Table 2. For five datasets, we use \( f \) and \( h \) for RSGDA. For SGDA, we set the learning rates of both maximization and minimization as 0.01. For Acc-RSGDA, we set \( f \) and \( h \) for RSGDA. For Acc-MDA, we set \( K = 3 \) for all datasets, and \( \varepsilon \) for the robust training is set to 0.4, 0.2, 0.02, 0.02 and 0.02 for MNIST, FashionMNIST, CIFAR10, CIFAR100, and STL10, respectively. We further set the mini-batch size as 512, and the model is trained for 200 epochs.

The training progress for robust training is shown in Fig. 3. From Fig. 3, we can see that our Acc-RSGDA method converges faster than the other comparison baselines, and it can achieve the best test accuracy with natural images for both datasets. RSGDA does not use momentum terms, but it reaches lower training loss compared to SGDA and Acc-MDA. This observation implies that our framework better utilizes the property of Riemannian manifold for robust DNN training. On the other hand, simply adding the retraction operation (Acc-MDA and SGDA) cannot achieve the same effect.

The numeric results against different attacks (i.e., PGD attack [54] and Fast Gradient Sign Method (FGSM) attack [55]) are shown in Tables 3, 4, 5, 6, and 7. Specifically, in the training progress, we report the numeric results against PGD attack of 40 steps and FGSM attacks. For all settings, our Acc-RSGDA method achieves the best accuracy against PGD and FGSM attacks.
FGSM attacks. Interestingly, the Acc-MDA method performs worse than SGDA under PGD and FGSM attacks, which suggests that the momentum may be not functional properly without considering the property of Riemannian manifold.

### 6.2 Distributionally Robust Optimization

In the subsection, we focus on distributionally robust optimization over Riemannian manifold defined in Problem (6). CIFAR-10 and STL-10 are selected as the datasets for this task. We use the same DNN architecture from the above robust DNN training for this task. We also apply Stiefel manifold $\mathcal{M} = \text{St}(r, d) = \{ X \in \mathbb{R}^{d \times r} : X^T X = I_r \}$ to the parameters of the DNN. We use the same hyper-parameter setting for RSGDA, Acc-RSGDA, SGDA and Acc-MDA from this task. The mini-batch size is also set 512, and the model is trained for 200 epochs. We report mean and variance across 3 runs for this experiment.

The results are reported in Fig. 4, and shaded areas represent variance. From the figure, we can see that our Acc-RSGDA achieves the best test accuracy and converges fastest. The difference between Acc-RSGDA and Acc-MDA is small, but due to using the property of Riemannian manifold, Acc-RSGDA is more stable compared to Acc-MDA.

### 7 Conclusion

In the paper, we investigated a class of useful minimax optimization problems on Riemannian manifolds. Meanwhile, we proposed a class of effective and efficient Riemannian gradient descent ascent algorithms to solve these minimax problems. Moreover, we studied convergence properties of our
proposed algorithms. To the best of our knowledge, our Riemannian gradient-based methods are the first to study the minimax optimization over the general Riemannian manifolds.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. We also thank so much for the help of Prof. Heng Huang.

REFERENCES

[1] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 1765–1773.
[2] N. Akhtar, J. Liu, and A. Mian, “Defense against universal adversarial perturbations,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 3389–3398.
[3] Y. Sun, L. Zheng, W. Deng, and S. Wang, “SVDNet for pedestrian retrieval,” in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 3800–3808.
[4] D. Xie, J. Xiong, and S. Pu, “All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 6176–6185.
[5] J. Li, L. Fuxin, and S. Todorovic, “Efficient Riemannian optimization on the stiefel manifold via the cayley transform,” in Proc. Int. Conf. Learn. Representations, 2020, pp. 1–16.
[6] N. Bansal, X. Chen, and Z. Wang, “Can we gain more from orthogonality regularizations in training deep networks?”, in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 4261–4271.
[7] M. Cogswell, F. Ahmed, R. Gannick, L. Zitnick, and D. Batra, “Reducing overfitting in deep networks by decorrelating representations,” 2015, arXiv:1511.06068.
[8] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li, “Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks,” in Proc. 32nd AAAI Conf. Artif. Intell., 2018, pp. 3271–3278.
[9] R. S. Chen, B. Lucier, Y. Singer, and S. Syrgkanis, “Robust optimization for non-convex objectives,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 4705–4714.
[10] H. Rahimian and S. Mehrrotra, “Distributionally robust optimization: A review,” 2019, arXiv:1908.05659.
[11] A. Han and J. Gao, “Riemannian stochastic recursive momentum method for non-convex optimization,” 2020, arXiv:2008.04555.
[12] J. Sun, Q. Qu, and J. Wright, “Complete dictionary recovery over the simplex II: R codes by Riemannian trust-region method,” IEEE Trans. Inf. Theory, vol. 63, no. 2, pp. 885–914, Feb. 2017.
[13] J. Yang, Z. Wang, C. Guzman, N. Kiyavash, and N. He, “The complexity of nonconvex-strongly-concave minimax optimization,” in Proc. Conf. Uncertainty Artif. Intell., 2021, pp. 482–492.
[14] X. Xu, Y. Lu, W. Liu, and T. Yang, “Sharp analysis of echo stochastic gradient descent ascent methods for min-max optimization,” 2020, arXiv:2002.05309.
[15] H. Huang and S. Mehrotra, “Distributionally robust optimization: A review,” 2019, arXiv:1908.05659.
[16] A. Han and J. Gao, “Riemannian stochastic recursive momentum method for non-convex optimization,” 2020, arXiv:2008.04555.
[17] J. Sun, Q. Qu, and J. Wright, “Complete dictionary recovery over the simplex II: R codes by Riemannian trust-region method,” IEEE Trans. Inf. Theory, vol. 63, no. 2, pp. 885–914, Feb. 2017.
[18] J. Yang, Z. Wang, C. Guzman, N. Kiyavash, and N. He, “The complexity of nonconvex-strongly-concave minimax optimization,” in Proc. Conf. Uncertainty Artif. Intell., 2021, pp. 482–492.
[19] X. Xu, Y. Lu, W. Liu, and T. Yang, “Sharp analysis of echo stochastic gradient descent ascent methods for min-max optimization,” 2020, arXiv:2002.05309.
[20] H. Huang and S. Mehrotra, “Distributionally robust optimization: A review,” 2019, arXiv:1908.05659.
[47] P. Zhou, X.-T. Yuan, S. Yan, and J. Feng, “Faster first-order methods for stochastic non-convex optimization on Riemannian manifolds,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 2, pp. 459–472, Feb. 2021.

[48] A. Han and J. Gao, “Improved variance reduction methods for Riemannian non-convex optimization,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 11, pp. 7610–7623, Nov. 2022.

[49] H. Kasai, P. Jawanpuria, and B. Mishra, “Riemannian adaptive stochastic gradient algorithms on matrix manifolds,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 3262–3271.

[50] F. Huang and S. Gao, “Riemannian gradient methods for stochastic composition problems,” Neural Netw., vol. 153, pp. 224–234, 2022.

[51] W. Huang, “Optimization algorithms on Riemannian manifolds with applications,” Ph.D. dissertation, The Florida State University, Tallahassee, Florida, 2013.

[52] C. Fang, C. J. Li, Z. Lin, and T. Zhang, “Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 689–699.

[53] J. Von Neumann and O. Morgenstern, Theory of Games and Economic Behavior, Princeton, NJ, USA: Princeton Univ. Press, 2007.

[54] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial machine learning at scale,” in Proc. Int. Conf. Learn. Representations, 2017, pp. 1–17.

[55] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2014, arXiv:1412.6572.

Feihu Huang received the PhD degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in December 2017. He was a postdoctoral researcher with the University of Pittsburgh, USA, from September 2018 to July 2022. He is currently a full professor with the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His current research interests include machine learning and optimization.

Shangqian Gao received the BEng degree in electrical engineering from Xidian University, China, in 2015 and the MS degree in computer system engineering from Northeastern University, USA, in 2017. He is currently working toward the PhD degree with the University of Pittsburgh. His research interests include computer vision, machine learning, and deep learning.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.