Rieoptax: Riemannian Optimization in JAX

author names withheld

Under Review for OPT 2022

Abstract

We present Rieoptax, an open source Python library for Riemannian optimization in JAX. We show that many differential geometric primitives, such as Riemannian exponential and logarithm maps, are usually faster in Rieoptax than existing frameworks in Python, both on CPU and GPU. We support various range of basic and advanced stochastic optimization solvers like Riemannian stochastic gradient, stochastic variance reduction, and adaptive gradient methods. A distinguishing feature of the proposed toolbox is that we also support differentially private optimization on Riemannian manifolds.

1. Introduction

Riemannian geometry is a generalization of the Euclidean geometry [49, 66] and includes several nonlinear spaces such as positive definite matrices [18, 92], Grassmann manifold [5, 15, 33], Stiefel manifold [5, 25, 33], kendall shape spaces [59, 60, 71], hyperbolic spaces [94, 95], and special Euclidean and orthogonal group [38, 87], to name a few. Optimization with manifold based constraints has become increasingly popular and has been employed in various applications such as low rank matrix completion [21], learning taxonomy embeddings [76, 77], neural networks [53–55, 75, 81], optimal transport [9, 26, 46, 73, 88], shape analysis [52, 90], and topological dimension reduction [56], among others. In addition, privacy preserving machine learning [29, 31] has become crucial in real applications, which has been generalized to manifold-valued problems very recently [44, 82, 96]. Nevertheless, such feature is absent in existing Riemannian optimization libraries.

In this work, we introduce Rieoptax (**Rie**mannian **Opt**imization in **Jax**), an open source Python library for Riemannian optimization in JAX [23, 37]. The proposed library is mainly driven by the needs of efficient implementation of manifold-valued operations and optimization solvers, readily compatible with GPU and even TPU processors as well as the needs of privacy-supported Riemannian optimization. To the best of our knowledge, Rieoptax is the first library to provide privacy guarantees within the Riemannian optimization framework.

1.1. Background on Riemannian optimization, privacy, and JAX

Riemannian Optimization. Riemannian Optimization [5, 20] considers the following problem

$$\min_{w \in \mathcal{M}} f(w) \tag{1}$$

where $f : \mathcal{M} \to \mathbb{R}$, and $\mathcal{M}$ denotes a Riemannian manifold. Instead of considering (1) as a constrained problem, Riemannian optimization [5, 20] views it as an unconstrained problem on the manifold space. Riemannian (stochastic) gradient descent [19, 98] generalizes the Euclidean

gradient descent with intrinsic updates on manifold, i.e., $w_{t+1} = \mathrm{Exp}_{w_t}(-\eta_t \operatorname{grad} f(w_t))$, where $\operatorname{grad} f(w_t)$ is the Riemannian (stochastic) gradient, $\mathrm{Exp}_w(\cdot)$ is the Riemannian exponential map at w and η_t is the step size. Recent years have witnessed significant advancements for Riemannian optimization where more advanced solvers are generalized from the Euclidean space to Riemannian manifolds. These include variance reduction methods [42, 43, 57, 85, 100, 101], adaptive gradient methods [14, 58], accelerated gradient methods [7, 8, 45, 68, 99], quasi-Newton methods [51, 80], zeroth-order methods [67] and second order methods, such as trust region methods [4] and cubic regularized Newton's methods [6].

Differential privacy on Riemannian Manifolds. Differential privacy (DP) provides a rigorous treatment for data privacy by precisely quantifying the deviation in the model's output distribution under modification of a small number of data points [29–32]. Provable guarantees of DP coupled with properties like immunity to arbitrary post-processing and graceful composability have made it a de-facto standard of privacy with steadfast adoption in the real applications [3, 10, 28, 34, 74].

Recently, there is a surge of interest on differential privacy over Riemannian manifolds, which has been explored in the context of Fréchet mean computation [82, 96] and, more generally, empirical risk minimization problems where the parameters are constrained to lie on a Riemannian manifold [44].

JAX and its ecosystem. JAX [23, 37] is recently introduced machine learning framework which support automatic differentiation capabilities [13] via `grad()`. Further some of the distinguishing features of JAX are just-in-time (JIT) compilation using the accelerated linear algebra (XLA) compiler [41] via `jit()`, automatic vectorization (batch-level parallelism) support with `vmap()`, and strong support for parallel computation via `pmap()`. All the above transformations can be composed arbitrarily because JAX follows the functional programming paradigm and implements these as pure functions.

Given that JAX has so many interesting features, its ecosystem has been constantly expanding in the last couple of years. Examples include neural network modules (Flax [48], Haiku [50], Equinox [62], Jraph [39], Equivariant-MLP [35]), reinforcement learning agents (Rlax [12]), Euclidean optimization algorithms (Optax [12]), federated learning (Fedjax [83]), optimal transport toolboxes (Ott [27]), sampling algorithms (Blackjax [64]), differential equation solvers (Diffrax [61]), rigid body simulators (Brax [36]), differentiable physics (Jax-md [86]).

1.2. Rieoptax

We believe the proposed framework for Riemannian optimization in JAX is a timely contribution, bringing several benefits of JAX and new features (such as privacy support) to the manifold optimization community, which are discussed below.

- *Automatic and efficient vectorization with* `vmap()`: functions that are written for inputs of size 1 can be converted to functions that take batch of inputs by wrapping it with `vmap()`. For example, the function `def dist(point_a, point_b)` for computing distance between a single `point_a` and a single `point_b` can be converted to function that computes distance between a batch of `point_a` and/or a batch `point_b` by wrapping `dist` with `vmap()` without modifying the `dist()` function. This is useful in many cases, e.g., Fréchet mean computation $\min_{w\in\mathcal{M}} \left\{\frac{1}{n}\sum_{i=1}^n f_i(w) := \frac{1}{n}\sum_{i=1}^n \mathrm{dist}^2(w, z_i)\right\}$. Furthermore, vectorization with `vmap()` is usually faster or on par with manual vectorization [23].

- *Per-example gradient clipping* : A key process in differentially private optimization is per-example gradient clipping $\frac{1}{n}\sum_{i=1}^{n}\text{clip}_\tau(\text{grad} f_i(w))$, where clip_τ ensures norm is atmost τ. Here, the order of operations is important: the gradients are first clipped and then averaged. Popular libraries including Autograd [69], Pytorch [78] and Tensorflow [1] are heavily optimized to directly compute the mean gradient $\frac{1}{n}\sum_{i=1}^{n}\text{grad} f_i(w)$ and hence do not expose per-example gradients i.e., $\text{grad} f_i(w)$. Hence, one has to resort to ad-hoc techniques [40, 65, 84] or come up with algorithmic modifications [24] which inherently have speed versus performance trade-off. JAX, however, offers native support for handling such scenarios and JAX-based differentially private Euclidean optimization methods have been shown to be much faster than their non-JAX counterparts [91]. We observe that JAX offer similar benefits for differentially private Riemannian optimization as well.

- *Single Source Multiple Devices (SSMD) Paradigm* : JAX follows SSMD paradigm, and hence code written for CPU can be run on GPU/TPU without any modification.

Rieoptax is made available at `https://anonymous.4open.science/r/Rieoptax/` for review and will be made public.

2. Implementation Overview

The package currently implements several commonly used geometries, optimization algorithms and differentially private mechanisms on manifolds. More geometries and advanced solvers will be added in the future.

2.1. Geometries

Geometry module contains manifolds equipped with Riemannian metrics. Each Geometry contains Riemannian inner product `inp()`, and induced norm `norm()`, Riemannian exponential `exp()` and logarithm maps `log()`, induced Riemannian distance `dist()`, parallel transport `pt()`, and transformation from Euclidean gradient to Riemannian gradient `egrad_to_rgrad()`.

Manifolds include symmetric positive definite (SPD) matrices $\text{SPD}(m) := \{\mathbf{X} \in \mathbb{R}^{m\times m} : \mathbf{X} = \mathbf{X}^\top, \mathbf{X} \succ 0\}$, hyperbolic space, Grassmann manifold $\mathcal{G}(m,r) := \{[\mathbf{X}] : \mathbf{X} \in \mathbb{R}^{m\times r}, \mathbf{X}^\top\mathbf{X} = \mathbf{I}\}$ where $[\mathbf{X}] := \{\mathbf{X}\mathbf{O} : O \in O(r)\}$, $O(r)$ denotes the orthogonal group and hypersphere $\mathcal{S}(d) := \{\mathbf{x} \in \mathbb{R}^d : \mathbf{x}^\top\mathbf{x} = 1\}$. We use $T_x\mathcal{M}$ to represent the tangent space at x and $\langle u, v\rangle_x$ to represent the Riemannian inner product. For more detailed treatment on these geometries, we refer to [5, 20, 95].

- `rieoptax.geometry.spd.SPDAffineInvariant` : SPD matrices with the affine-invariant metric [79]: $\text{SPD}(m)$ with $\langle \mathbf{U}, \mathbf{V}\rangle_{\mathbf{X}} = \text{tr}(\mathbf{X}^{-1}\mathbf{U}\mathbf{X}^{-1}\mathbf{V})$ for $\mathbf{U}, \mathbf{V} \in T_{\mathbf{X}}\text{SPD}(m)$.

- `rieoptax.geometry.spd.SPDLogEuclidean` : SPD matrices with the log-Euclidean metric [11]: $\text{SPD}(m)$ with $\langle \mathbf{U}, \mathbf{V}\rangle_{\mathbf{X}} = \text{tr}\big(\text{D}_{\mathbf{U}}\text{logm}(\mathbf{X})\text{D}_{\mathbf{V}}\text{logm}(\mathbf{X})\big)$ where $\text{D}_{\mathbf{U}}\text{logm}(\mathbf{X})$ is the directional derivative of matrix logarithm at $\mathbf{X}$ along $\mathbf{U}$.

- `rieoptax.geometry.hyperbolic.PoincareBall` : Poincare-ball model of Hyperbolic space with Poincare metric [95], i.e., $\mathbb{D}(d) := \{\mathbf{x} \in \mathbb{R}^d : \mathbf{x}^\top\mathbf{x} < 1\}$ with $\langle \mathbf{u}, \mathbf{v}\rangle_{\mathbf{x}} = 4\mathbf{u}^\top\mathbf{v}/(1-\mathbf{x}^\top\mathbf{x})^2$ for $\mathbf{u}, \mathbf{v} \in T_{\mathbf{x}}\mathbb{D}(d)$.

- `rieoptax.geometry.hyperbolic.LorentzHyperboloid` Lorentz Hyperboloid model of Hyperbolic space [95], i.e., $\mathbb{H}(d) = \{\mathbf{x} \in \mathbb{R}^d : \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = -1\}$ with $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbf{x}} = \langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{L}}$ for $\mathbf{u}, \mathbf{v} \in T_{\mathbf{x}}\mathbb{H}(d)$, where $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{L}} := -u_0 v_0 + u_1 v_1 + \cdots u_{d-1} v_{d-1}$.
- `rieoptax.geometry.grassmann.GrassmannCanonicalMetric` : Grassmann manifold with the canonical metric [33], i.e., $\mathcal{G}(m, r)$ with $\langle \mathbf{U}, \mathbf{V} \rangle_{\mathbf{X}} = \mathrm{tr}\big(\mathbf{U}^T \mathbf{V}\big)$ for $\mathbf{U}, \mathbf{V} \in T_{\mathbf{X}}\mathcal{G}(m, r)$.
- `rieoptax.geometry.hypersphere.HypersphereCanonicalMetric` : Hypersphere manifold which canonical metric [5, 20], i.e., $\mathcal{S}(d)$ with $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathbf{x}} = \mathbf{u}^\top \mathbf{v}$ for $\mathbf{u}, \mathbf{v} \in T_{\mathbf{x}}\mathcal{S}(d)$.

2.2. Optimizers

Optimizers module contains Riemannian optimization algorithms.

- `riepotax.optimizers.first_order.rsgd` : Riemannian stochastic gradient descent [19].
- `riepotax.optimizers.first_order.rsvrg` : Riemannian stochastic variance reduced gradient descent [100].
- `riepotax.optimizers.first_order.rsrg` : Riemannian stochastic recursive gradient descent [57].
- `riepotax.optimizers.first_order.rasa` : Riemannian adaptive stochastic gradient algorithm [58].
- `riepotax.optimizers.zeroth_order.zo_rgd` : Zeroth-order Riemannian gradient descent [67].

2.3. Mechanism

Mechanism module contains differential private mechanisms on Riemannian manifolds.

- `rieoptax.mechanism.output_perturbation.RieLaplaceMechanism` : Implements Riemannian Laplace mechanism [82] which is used for privatizing Fréchet mean computation.
- `rieoptax.mechanism.output_perturbation.LogEuclideanMechanism`: Implements log-Euclidean mechanism [96] which is used for differentially private Fréchet mean on SPD matrices with log-Euclidean metric.
- `rieoptax.mechanism.gradient_perturbation.DPRGDMechanism`: Implements noise calibration for Differentially private Riemannian gradient descent [44] based on moments accountant [2] in autodp library [97].
- `rieoptax.mechanism.gradient_perturbation.DPRSGDMechanism` : Implements noise calibration for Differentially private Riemannian stochastic gradient descent [44] based on moments accountant [2] in autodp library [97].

3. Benchmarking Rieoptax

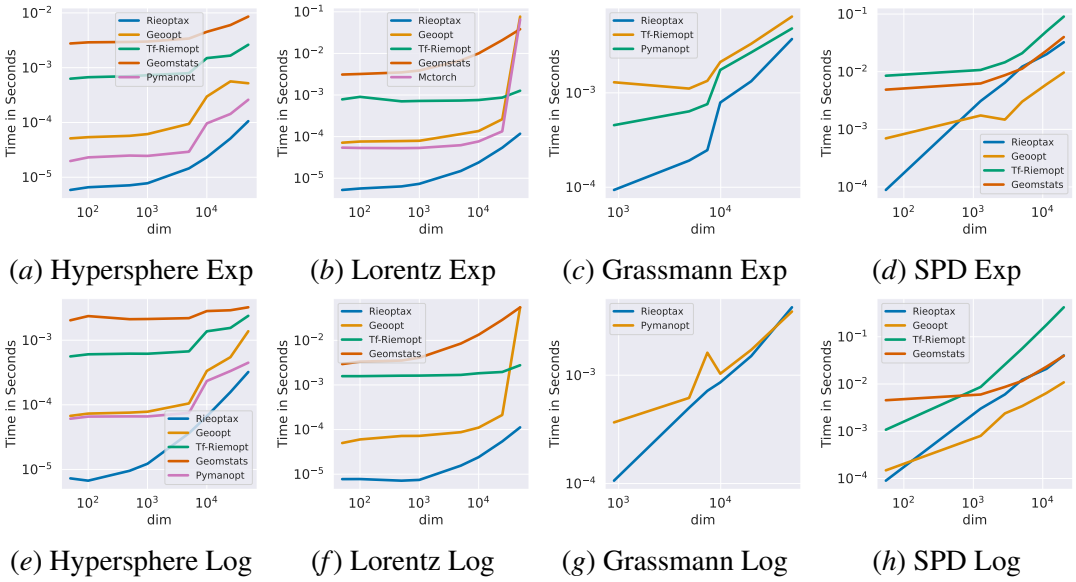
In this section, we benchmark the proposed Rieoptax against existing Riemannian optimization libraries in Python. These include Pytorch [78] based Mctorch [70] and Geoopt [63], Tensorflow [89] based Tensorflow-Riemopt (Tf-Riemopt) [89], Numpy [47] based Pymanopt [93], and Tensorflow based Geomstats [72]. While Geomstats supports Numpy, Pytorch, and Tensorflow as backend, currently only Tensorflow backend provides support for both CPU and GPU. Other non-Python based libraries include Manopt [22] in Matlab and Manopt.jl [16] in Julia [17]. We benchmark the Riemannian exponential (Exp) and logarithm (Log) maps with the proposed Rieoptax against the aforementioned Python libraries whenever available with `64bitfloat` precision. For CPU benchmarking we use Processor AMD EPYC 7B1 with 2 cores and 16GB RAM. For GPU benchmarking, we use CUDA version 11.4 on 16GB Tesla P100.

- **Hypersphere canonical metric:** Hypersphere $\mathcal{S}(d)$ is supported in Geoopt, Tf-Riemopt, Geomstats, and Pyamanopt. On GPU however, Geomstats raises an error. we benchmark for dimensions $d \in \{50, 100, 500, 1000, 5000, 10000, 25000, 50000\}$.
- **Loretnz hyperboloid model:** Loretnz hyperboloid model $\mathbb{H}(d)$ is supported in Geoopt, Tf-Riemopt, Geomstats, and Mctorch. While Riemannian exponential map is available in Mctorch, it does not implement the logarithm map. We benchmark for dimensions $d \in \{50, 100, 500, 1000, 5000, 10000, 25000, 50000\}$.
- **Grassmann with canonical metric** : Grassmann manifold $\mathcal{G}(m, r)$ with canonical metric is supported in Tf-Riemopt, Pymanopt, Geomstats. However, we notice that the logarithm map in Tf-Riemopt is incorrectly implemented and Geomstats represents Grassmann in projector matrix $\mathbf{X}\mathbf{X}^\top \in \mathbb{R}^{m\times m}$ instead of $\mathbf{X} \in \mathbb{R}^{m\times r}$, which is prohibitively expensive. We thus exclude these two libraries from benchmarking. We benchmark for matrix size $(m, r) \in \{(100, 10), (500, 10), (750, 10), (1000, 10), (2000, 10), (5000, 10)\}$.
- **SPD with affine invariant metric** : SPD manifold $\mathrm{SPD}(m)$ with affine invariant metric is supported in Geoopt, Tf-Riemopt, Geomstats. We benchmark for matrix size $m \in \{10, 50, 75, 100, 150, 200\}$.

Figures 1 and 2 present the timing results with CPU- and GPU-based computations, respectively. Overall, we observe that Rieoptax offers significant time improvement, especially on GPU. For SPDAffineInvariant Case, Rieoptax is slightly slower than Geoopt because `eigh` which provides eigen decomposition is slightly slower in JAX compared to Pytorch. Given that JAX is a relatively new framework, we believe it will be faster even in this case in the near future.

4. Conclusion and future roadmap

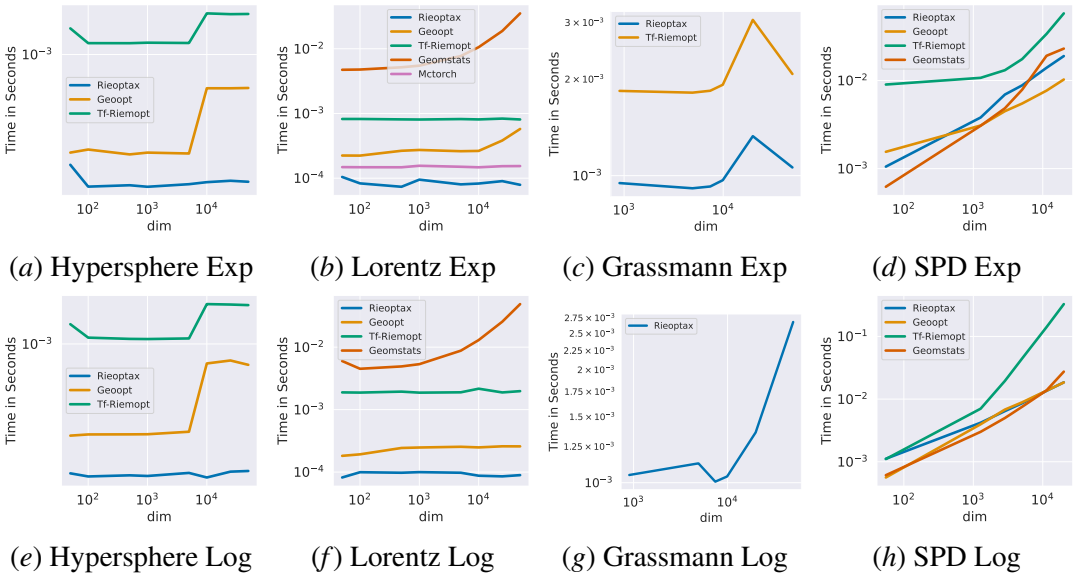
In this work, we present a Python library for (privacy-supported) Riemannian optimization, Rieoptax, and illustrate its efficacy on both CPU and GPU evaluation. Our roadmap includes adding support for more manifold geometries, optimization algorithms, and a collection of example codes showcasing the usage of Rieoptax in various applications.



(*a*) Hypersphere Exp (*b*) Lorentz Exp (*c*) Grassmann Exp (*d*) SPD Exp

(*e*) Hypersphere Log (*f*) Lorentz Log (*g*) Grassmann Log (*h*) SPD Log

Figure 1: Benchmarking of Geometric Primitives on CPU.



(*a*) Hypersphere Exp (*b*) Lorentz Exp (*c*) Grassmann Exp (*d*) SPD Exp

(*e*) Hypersphere Log (*f*) Lorentz Log (*g*) Grassmann Log (*h*) SPD Log

Figure 2: Benchmarking of Geometric Primitives on GPU.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. In *USENIX Conference on Operating Systems Design and Implementation*, 2016.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[3] John M Abowd. The US Census Bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2867–2867, 2018.

[4] P-A Absil, Christopher G Baker, and Kyle A Gallivan. Trust-region methods on Riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

[5] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. Optimization algorithms on matrix manifolds. In *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2009.

[6] Naman Agarwal, Nicolas Boumal, Brian Bullins, and Coralia Cartis. Adaptive regularization with cubics on manifolds. *Mathematical Programming*, 188(1):85–134, 2021.

[7] Kwangjun Ahn and Suvrit Sra. From Nesterov's estimate sequence to Riemannian acceleration. In *Conference on Learning Theory*, pages 84–118. PMLR, 2020.

[8] Foivos Alimisis, Antonio Orvieto, Gary Bécigneul, and Aurelien Lucchi. A continuous-time perspective for modeling acceleration in Riemannian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 1297–1307. PMLR, 2020.

[9] Jason Altschuler, Sinho Chewi, Patrik R Gerber, and Austin Stromme. Averaging on the Bures-Wasserstein manifold: dimension-free convergence of gradient descent. *Advances in Neural Information Processing Systems*, 34:22132–22145, 2021.

[10] D Apple. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017.

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

[12] Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, John Quan, George Papamakarios, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Luyu Wang, Wojciech Stokowiec, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/deepmind.

[13] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

[14] Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*, 2019.

[15] Thomas Bendokat, Ralf Zimmermann, and P-A Absil. A Grassmann manifold handbook: Basic geometry and computational aspects. *arXiv preprint arXiv:2011.13699*, 2020.

[16] Ronny Bergmann. Manopt. jl: Optimization on manifolds in julia. *Journal of Open Source Software*, 7(70):3866, 2022.

[17] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[18] Rajendra Bhatia. Positive definite matrices. In *Positive Definite Matrices*. Princeton university press, 2009.

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

[20] Nicolas Boumal. An introduction to optimization on smooth manifolds. To appear with Cambridge University Press, Jun 2022. URL https://www.nicolasboumal.net/book.

[21] Nicolas Boumal and Pierre-antoine Absil. RTRMC: A Riemannian trust-region method for low-rank matrix completion. *Advances in neural information processing systems*, 24, 2011.

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

[23] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[24] Zhiqi Bu, Sivakanth Gopi, Janardhan Kulkarni, Yin Tat Lee, Hanwen Shen, and Uthaipon Tantipongpipat. Fast and memory efficient differentially private-sgd via jl projections. *Advances in Neural Information Processing Systems*, 34:19680–19691, 2021.

[25] Rudrasis Chakraborty and Baba C Vemuri. Statistics on the Stiefel manifold: theory and applications. *The Annals of Statistics*, 47(1):415–438, 2019.

[26] Sinho Chewi, Tyler Maunu, Philippe Rigollet, and Austin J Stromme. Gradient descent algorithms for Bures-Wasserstein barycenters. In *Conference on Learning Theory*, pages 1276–1304. PMLR, 2020.

[27] Marco Cuturi, Laetitia Meng-Papaxanthos, Yingtao Tian, Charlotte Bunne, Geoff Davis, and Olivier Teboul. Optimal transport tools (ott): A jax toolbox for all things Wasserstein. *arXiv preprint arXiv:2201.12324*, 2022.

[28] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. *Advances in Neural Information Processing Systems*, 30, 2017.

[29] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[30] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 486–503. Springer, 2006.

[31] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[32] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[33] Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

[34] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

[35] Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *International Conference on Machine Learning*, pages 3318–3328. PMLR, 2021.

[36] C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021. URL http://github.com/google/brax.

[37] Roy Frostig, Matthew James Johnson, and Chris Leary. Compiling machine learning programs via high-level tracing. *Systems for Machine Learning*, 4(9), 2018.

[38] Jean Gallier and Jocelyn Quaintance. *Differential geometry and Lie groups: a computational perspective*, volume 12. Springer Nature, 2020.

[39] Jonathan Godwin*, Thomas Keck*, Peter Battaglia, Victor Bapst, Thomas Kipf, Yujia Li, Kimberly Stachenfeld, Petar Veličković, and Alvaro Sanchez-Gonzalez. Jraph: A library for graph neural networks in jax., 2020. URL http://github.com/deepmind/jraph.

[40] Ian Goodfellow. Efficient per-example gradient computations. *arXiv preprint arXiv:1510.01799*, 2015.

[41] Google. Xla : Compiling machine learning for peak performance, 2020.

[42] Andi Han and Junbin Gao. Improved variance reduction methods for Riemannian non-convex optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[43] Andi Han and Junbin Gao. Riemannian stochastic recursive momentum method for non-convex optimization. In *International Joint Conference on Artificial Intelligence*, pages 2505–2511, 8 2021.

[44] Andi Han, Bamdev Mishra, Pratik Jawanpuria, and Junbin Gao. Differentially private Riemannian optimization. *arXiv preprint arXiv:2205.09494*, 2022.

[45] Andi Han, Bamdev Mishra, Pratik Jawanpuria, and Junbin Gao. Riemannian accelerated gradient methods via extrapolation. *arXiv preprint arXiv:2208.06619*, 2022.

[46] Andi Han, Bamdev Mishra, Pratik Jawanpuria, and Junbin Gao. Riemannian block SPD coupling manifold and its application to optimal transport. *arXiv preprint arXiv:2201.12933*, 2022.

[47] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[48] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020. URL http://github.com/google/flax.

[49] Sigurdur Helgason. *Differential geometry, Lie groups, and symmetric spaces*. Academic press, 1979.

[50] Tom Hennigan, Trevor Cai, Tamara Norman, and Igor Babuschkin. Haiku: Sonnet for JAX, 2020. URL http://github.com/deepmind/dm-haiku.

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

[52] Wen Huang, Kyle A Gallivan, Anuj Srivastava, and Pierre-Antoine Absil. Riemannian optimization for registration of curves in elastic shape analysis. *Journal of Mathematical Imaging and Vision*, 54(3):320–343, 2016.

[53] Zhiwu Huang and Luc Van Gool. A Riemannian network for SPD matrix learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[54] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6099–6108, 2017.

[55] Zhiwu Huang, Jiqing Wu, and Luc Van Gool. Building deep networks on grassmann manifolds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[56] Oleg Kachan. Persistent homology-based projection pursuit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 856–857, 2020.

[57] Hiroyuki Kasai, Hiroyuki Sato, and Bamdev Mishra. Riemannian stochastic recursive gradient algorithm. In *International Conference on Machine Learning*, pages 2516–2524. PMLR, 2018.

[58] Hiroyuki Kasai, Pratik Jawanpuria, and Bamdev Mishra. Riemannian adaptive stochastic gradient algorithms on matrix manifolds. In *International Conference on Machine Learning*, pages 3262–3271. PMLR, 2019.

[59] David G Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London mathematical society*, 16(2):81–121, 1984.

[60] David G Kendall. A survey of the statistical theory of shape. *Statistical Science*, 4(2):87–99, 1989.

[61] Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

[62] Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021.

[63] Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in PyTorch. *arXiv preprint arXiv:2005.02819*, 2020.

[64] Junpeng Lao and Rémi Louf. Blackjax: A sampling library for JAX, 2020. URL http://github.com/blackjax-devs/blackjax.

[65] Jaewoo Lee and Daniel Kifer. Scaling up differentially private deep learning with fast per-example gradient clipping. *Proceedings on Privacy Enhancing Technologies*, 2021(1), 2021.

[66] John M Lee. *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media, 2006.

[67] Jiaxiang Li, Krishnakumar Balasubramanian, and Shiqian Ma. Stochastic zeroth-order Riemannian derivative estimation and optimization. *Mathematics of Operations Research*, 2022.

[68] Yuanyuan Liu, Fanhua Shang, James Cheng, Hong Cheng, and Licheng Jiao. Accelerated first-order methods for geodesically convex optimization on Riemannian manifolds. *Advances in Neural Information Processing Systems*, 30, 2017.

[69] Dougal Maclaurin, David Duvenaud, and Ryan P Adams. Autograd: Effortless gradients in numpy. In *ICML 2015 AutoML workshop*, 2015.

[70] Mayank Meghwanshi, Pratik Jawanpuria, Anoop Kunchukuttan, Hiroyuki Kasai, and Bamdev Mishra. McTorch, a manifold optimization library for deep learning. *arXiv preprint arXiv:1810.01811*, 2018.

[71] Nina Miolane, Susan Holmes, and Xavier Pennec. Template shape estimation: correcting an asymptotic bias. *SIAM Journal on Imaging Sciences*, 10(2):808–844, 2017.

[72] Nina Miolane, Nicolas Guigui, Alice Le Brigant, Johan Mathe, Benjamin Hou, Yann Thanwerdas, Stefan Heyder, Olivier Peltre, Niklas Koep, Hadi Zaatiti, et al. Geomstats: a Python package for Riemannian geometry in machine learning. *Journal of Machine Learning Research*, 21(223):1–9, 2020.

[73] Bamdev Mishra, NTV Satyadev, Hiroyuki Kasai, and Pratik Jawanpuria. Manifold optimization for non-linear optimal transport problems. *arXiv preprint arXiv:2103.00902*, 2021.

[74] Joe Near. Differential privacy at scale: Uber and Berkeley collaboration. In *Enigma 2018 (Enigma 2018)*, 2018.

[75] Xuan Son Nguyen, Luc Brun, Olivier Lézoray, and Sébastien Bougleux. A neural network based on spd manifold learning for skeleton-based hand gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12036–12045, 2019.

[76] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017.

[77] Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pages 3779–3788. PMLR, 2018.

[78] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

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

[80] Chunhong Qi, Kyle A Gallivan, and P-A Absil. Riemannian BFGS algorithm with applications. In *Recent Advances in Optimization and its Applications in Engineering*, pages 183–192. Springer, 2010.

[81] Guodong Qi, Huimin Yu, Zhaohui Lu, and Shuzhao Li. Transductive few-shot classification on the oblique manifold. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8412–8422, 2021.

[82] Matthew Reimherr, Karthik Bharath, and Carlos Soto. Differential privacy over Riemannian manifolds. *Advances in Neural Information Processing Systems*, 34:12292–12303, 2021.

[83] Jae Hun Ro, Ananda Theertha Suresh, and Ke Wu. FedJAX: Federated learning simulation with JAX. *arXiv preprint arXiv:2108.02117*, 2021.

[84] Gaspar Rochette, Andre Manoel, and Eric W Tramel. Efficient per-example gradient computations in convolutional neural networks. *arXiv preprint arXiv:1912.06015*, 2019.

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

[86] Samuel S. Schoenholz and Ekin D. Cubuk. Jax m.d. a framework for differentiable physics. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

[87] Jon M Selig. *Geometric fundamentals of robotics*, volume 128. Springer, 2005.

[88] Dai Shi, Junbin Gao, Xia Hong, ST Boris Choy, and Zhiyong Wang. Coupling matrix manifolds assisted optimization for optimal transport problems. *Machine Learning*, 110(3):533–558, 2021.

[89] Oleg Smirnov. TensorFlow RiemOpt: a library for optimization on Riemannian manifolds. *arXiv preprint arXiv:2105.13921*, 2021.

[90] Anuj Srivastava, Eric Klassen, Shantanu H Joshi, and Ian H Jermyn. Shape analysis of elastic curves in Euclidean spaces. *IEEE transactions on pattern analysis and machine intelligence*, 33(7):1415–1428, 2010.

[91] Pranav Subramani, Nicholas Vadivelu, and Gautam Kamath. Enabling fast differentially private SGD via just-in-time compilation and vectorization. *Advances in Neural Information Processing Systems*, 34:26409–26421, 2021.

[92] Yann Thanwerdas and Xavier Pennec. O (n)-invariant Riemannian metrics on SPD matrices. *arXiv preprint arXiv:2109.05768*, 2021.

[93] James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation. *Journal of Machine Learning Research*, 17(137):1–5, 2016.

[94] Abraham Albert Ungar. *Analytic hyperbolic geometry and Albert Einstein's special theory of relativity*. World Scientific, 2008.

[95] Abraham Albert Ungar. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics*, 1(1):1–194, 2008.

[96] Saiteja Utpala, Praneeth Vepakomma, and Nina Miolane. Differentially private Fr\'echet mean on the manifold of symmetric positive definite (SPD) matrices. *arXiv preprint arXiv:2208.04245*, 2022.

[97] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

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

[99] Hongyi Zhang and Suvrit Sra. An estimate sequence for geodesically convex optimization. In *Conference On Learning Theory*, pages 1703–1723. PMLR, 2018.

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

[101] Pan Zhou, Xiao-Tong Yuan, and Jiashi Feng. Faster first-order methods for stochastic nonconvex optimization on Riemannian manifolds. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 138–147. PMLR, 2019.