GRADIENT DESCENT ON RIEMANNIAN MANIFOLDS

EVAN TAYLOR

Abstract. Optimization on Euclidean spaces, while well-established, does not directly cater to the complex data structures and manifold constraints encountered in modern machine learning applications–applications such as image processing and autonomous driving. In this paper, we extend traditional Euclidean gradient descent algorithms to Riemannian manifolds, providing a mathematical foundation and computational framework suited to the intrinsic geometry of data residing on manifolds. Initially, we define manifolds, tangent spaces, and metrics, setting the stage for understanding the Riemannian gradient and the necessary adjustments required for optimization on curved spaces. We then discuss the adaptation of the gradient descent algorithm for Riemannian manifolds, emphasizing the critical role of the exponential map in navigating the manifold's geodesic paths for optimization. This approach ensures that updates remain within the manifold, leveraging their geometric properties to potentially enhance the efficiency and effectiveness of learning algorithms. The proposed method offers a practical algorithm for implementing gradient descent in non-Euclidean domains, offering a suitable optimization technique in manifold-embedded data scenarios.

1. Introduction

The emergence of complex data structures in modern computational tasks necessitates a reevaluation of traditional optimization techniques. In fields ranging from computer vision to robotics, input data often resides on non-linear structures that are best modeled as manifolds rather than as points in Euclidean space. This distinction is critical. While traditional optimization techniques in Euclidean spaces are well-understood and have been the backbone of many machine learning algorithms, they fall short when directly applied to manifold-embedded data due to the inherent geometric complexities of these spaces. [AMS08]

The gradient descent algorithm, a cornerstone of optimization in machine learning, is typically formulated in Euclidean settings. However, the underlying assumption of linearity and flat geometry does not hold in manifold settings. For instance, the optimization of camera calibration parameters, shape analysis in medical imaging, and pose estimation in robotics often involve data intrinsically linked to spherical, symmetrical, or otherwise curved manifolds. In such cases, the straight paths prescribed by Euclidean gradient descent do not respect the curved nature of the data space, potentially leading to suboptimal convergence and inefficiency.

This paper introduces an adaptation of the Euclidean gradient descent algorithm for Riemannian manifolds. Our approach leverages the mathematical framework of differential geometry, particularly focusing on defining gradients and mapping optimization paths directly onto the manifold. By doing so, we aim to maintain the integrity of the manifold's structure, thus aligning more closely with the true nature of the data, and avoiding search in irrelevant directions.

We begin by defining the essential mathematical concepts necessary for this adaptation, including manifolds, tangent spaces, Riemannian metrics, and the Riemannian gradient. Following this foundational setup, we explore the adaptation of the gradient descent algorithm to Riemannian manifolds, emphasizing the role of the exponential map in moving along geodesics, which are the natural analogs to straight lines in curved spaces. This method not only respects the manifold's curvature but also promises more natural and efficient paths to minima, thus potentially improving convergence properties and algorithmic performance. [Smi93]

Throughout this paper, we address both the theoretical underpinnings and practical implications of our approach, illustrating how this method can be implemented effectively in various real-world applications where data inherently resides on manifolds. By extending gradient descent to Riemannian manifolds, we contribute to the broader discourse on optimization in advanced machine learning contexts.

2. Preliminary Definitions

Definition (Manifolds)**.** *A manifold $\mathcal{M}$ is a topological space that locally resembles Euclidean space near each point. Specifically, it is a space such that for all $p \in \mathcal{M}$, there exists a neighborhood U that is homeomorphic to an open subset of $\mathbb{R}^n$.*

Definition (Tangent spaces and metrics)**.** *The tangent space $T_p\mathcal{M}$ at a point p on a manifold $\mathcal{M}$ is a vector space consisting of the tangent vectors at p. The metric g at a point p, denoted g_p, defines an inner product on $T_p\mathcal{M}$. This metric allows us to define and measure distances and angles in $T_p\mathcal{M}$, enabling necessary calculations involving the lengths of curves and angles between vectors.*

Definition (Riemannian manifolds)**.** *A Riemannian manifold $(\mathcal{M}, g)$ extends the concept of a manifold by equipping the it with a Riemannian metric g, which is a smooth assignment of an inner product g_p on the tangent space $T_p\mathcal{M}$ at each point p.*

3. Gradient Descent in Euclidean Space

Before we explore the topic of optimization on Riemannian manifolds, let's look at optimization in Euclidean space.

Problem. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function. Compute*

$$\min_{x \in \mathbb{R}^n} f(x)$$

That is, we are searching for a point $x \in \mathbb{R}^n$ such that $f(x)$ is the minimum of f.

There exist a large number of algorithms that offer a solution to this problem. The most popular in the world of machine learning, and the one we will focus on in this paper, is gradient descent. [Ber16]

Algorithm (Euclidean gradient descent)**.** .

```
(1) Pick an arbitrary $x_0 \in \mathbb{R}^n$ and let $\alpha \in \mathbb{R}$ be the learning rate parameter
    with $\alpha > 0$
(2) Repeat until convergence:
    2.1 Compute $\nabla_f(x)$
    2.2 $x_{t+1} = x_t - \alpha\nabla_f(x_t)$ i.e. the update step
    2.3 $t = t + 1$
(3) Return $x_t$
```

Notice that in Euclidean space the algorithm is pretty computationally simple. We move in the direction of the steepest gradient until convergence. There is one parameter, the learning rate α, which governs the size of the step at each iteration. [Hu19]

4. Building Intuition

Before we begin exploring the Riemannian adaptation of gradient descent, consider why we might want to use such an adaptation instead of the Euclidean counterpart. Suppose you are training a neural network for autonomous driving. In self-driving cars, input data such as images or LiDAR point clouds naturally reside on manifolds due to the inherent constraints and structures of the data. For example, the space of all possible images captured by cameras can be thought of as a manifold within the higher-dimensional space of all possible pixel values. This manifold structure arises because real-world images are not arbitrary pixel combinations, but instead reflect structured and correlated features like edges, shapes, and textures. Using gradient descent on a Riemannian manifold for such data is advantageous because it respects the underlying geometric structure of the data space, leading to more efficient and robust optimization. In a case when optimizing camera calibration parameters or 3D object recognition models, maintaining the geometric constraints can help yield more accurate results and better generalization. This is because as each update is constrained to be relevant to the actual structure of the data, we avoid searching irrelevant directions that lead to poor performance.

5. Gradient Descent on Riemannian Manifolds

Now that we've defined (Euclidean) gradient descent, we can look at the problem on manifolds.

Problem. *Let $(\mathcal{M}, g)$ be a connected Riemannian manifold and $f : \mathcal{M} \to \mathbb{R}$ a real-valued function defined on $\mathcal{M}$. Compute*

$$\min_{p \in \mathcal{M}} f(p)$$

For the purposes of this analysis, we assume that the Riemannian manifold $(\mathcal{M}, g)$ is connected. This assumption ensures that any two points on $\mathcal{M}$ can be connected by a path, thereby allowing the comprehensive application of the gradient descent algorithm across the entire manifold. While some Riemannian manifolds may consist of multiple disconnected components, our focus here is restricted to scenarios where the manifold is a single, unified entity. [Lee03]

The gradient descent algorithm on a Riemannian manifold is conceptually identical to the algorithm in Euclidean space. While the two optimization problems seem similar enough, some complexities arise when adapting gradient descent to Riemannian manifolds. First, we have to tackle the gradient calculation in our new space. We also need to define an analogous way to move between points on $\mathcal{M}$ during the update step. Before we define an adapted algorithm, we must work through these complexities. [Smi93]

5.1. The Gradient.

First, let us look at the gradient calculation on a manifold.

Definition (Riemannian gradient). *Let $(\mathcal{M}, g)$ be a Riemannian manifold, and let $f : \mathcal{M} \to \mathbb{R}$ be a real-valued function over $\mathcal{M}$. The gradient of f at a point $p \in \mathcal{M}$, $\operatorname{grad} f(p)$, is the vector field that satisfies*

$$g_p(\operatorname{grad} f(p), v) = df_p(v)$$

for any vector $v \in T_p\mathcal{M}$ at p. Here, df_p is the differential of f at p, which maps tangent vectors at p to $\mathbb{R}$, and g_p is the metric tensor evaluated at p.

For the actual computation, let $\{x^1, \cdots, x^n\}$ be a local coordinate basis around point p. Then we can represent g_p by a matrix G, and df_p by the vector of partials (the gradient) ∇_f. Then $\operatorname{grad} f(p)$ is given by

$$\operatorname{grad} f(p) = G^{-1}\nabla f$$

where G^{-1} is he inverse of the metric tensor matrix G, and $\nabla f = \{\frac{\partial f}{\partial x^1}, \cdots, \frac{\partial f}{\partial x^n}\}$.

The equation above is valid because in local coordinates around p, df_p can be expressed as the vector of partial derivatives ∇_f. These these partial derivatives represent the rate of change of f in the coordinate directions. However, in order to convert these changes to something that respects the manifold's geometry, we need to adjust them according to the metric tensor G. This is where we use the inverse of the metric tensor G^{-1}.

Thus the computation $\operatorname{grad} f(p) = G^{-1}\nabla_f$ is essentially transforming the coordinate representation of df_p, which is in the cotangent space, into a vector v in the tangent space. The transformation by G^{-1} ensures that the inner product $g_p(\operatorname{grad} f(p), v)$ gives the correct directional derivates $df_p(v)$, for all v. This works because

$$g_p(G^{-1}\nabla_f, v) = (G^{-1}\nabla_f)^T G v = (\nabla_f)^T v = df_p(v)$$

The equality above ensures that the Riemannian gradient $\operatorname{grad} f(p)$ points in the direction of the steepest gradient of f at p in a way that is consistent with the geometry defined by g. Thus, the Euclidean gradient, when appropriately transformed by the metric tensor, serves as the correct generalization to Riemannian manifolds, ensuring that the manifold's curvature and other geometric properties are respected in the computation of the gradient. [AMS08]

5.2. The Descent.

Now that we've found an analogue to the gradient computation step, we need to define how we will move between points on $\mathcal{M}$ at each iteration. Notice that in Euclidean space, we move between points by following a straight line in the direction of the steepest gradient. This is done because the straight line is the path with shortest distance between points in $\mathbb{R}^n$. Meanwhile, on Riemannian manifolds, we need to utilize an analogous way to move between points on $\mathcal{M}$. The Euclidean method is no longer guaranteed to work on manifolds as there is no guarantee that the update step will provide us with a new point that lies on the manifold.

In Riemannian manifolds, geodesics serve as the counterparts to straight lines found in Euclidean spaces. These curves on a Riemannian manifold $\mathcal{M}$ represent the shortest routes between points locally and generalize the notion of straight lines to the context of manifolds. [Lee03]

Definition (Geodesics)**.** *Let $(\mathcal{M}, g)$ be a Riemannian manifold. A curve*

$$\gamma : [a, b] \to \mathcal{M}$$

is called a geodesic if it is locally a distance-minimizing path between any two points on the curve. More formally, γ is a geodesic if for each $t \in [a, b]$, there exists an $\epsilon > 0$ such that for all $s, t \in [t - \epsilon, t + \epsilon] \subset [a, b]$, the segment of γ from $\gamma(s)$ to $\gamma(t)$ minimizes the distance compared to any other sufficiently smooth curve connecting these points. Additionally, the γ has no acceleration in the manifold, preserving the direction and magnitude of its tangent vector along its length.

Now that we've defined the shortest path, we need a method to traverse this path during the update step in gradient descent. This traversal is facilitated by the exponential map, which converts updates expressed as tangent vectors at a point into movements along geodesics on the manifold. [GN14]

Definition (Exponential map)**.** *For a point $p \in \mathcal{M}$ and a tangent vector $v \in T_p\mathcal{M}$ at p, the exponential map*

$$\exp_p : T_p\mathcal{M} \to \mathcal{M}$$

is defined such that $\exp_p(v) = \gamma(1)$, where $\gamma : [0, 1] \to \mathcal{M}$ is the unique geodesic starting at p with initial velocity $\dot{\gamma}(0) = v$. The map $\exp_p(v)$ provides a way to 'step' from p in the direction of v along the manifold, adhering to its intrinsic geometric structure.

These adaptations allow the gradient descent algorithm to operate within the intrinsic geometry of the manifold, respecting its curvature and topological structure. Specifically, let $\mathcal{M}$ be a Riemannian manifold and let $p \in \mathcal{M}$. Suppose we are performing gradient descent to minimize a smooth function $f : \mathcal{M} \to \mathbb{R}$. At each iteration of the gradient descent, we need to update our point p on the manifold $\mathcal{M}$ in the direction that most decreases f. For a point $x \in \mathbb{R}^n$, this update would simply be

$$x_{t+1} = x_t - \alpha \nabla_f(x_t)$$

as defined above. [Car92], [AMS08]

However, on a Riemannian manifold, we cannot directly subtract vectors from points, and the gradient $\nabla f(p)$ in Euclidean space translates to the Riemannian gradient $\operatorname{grad} f(p)$ on the manifold. This Riemannian gradient is a vector in the tangent space $T_p\mathcal{M}$ of the manifold at point p, indicating the direction of steepest ascent of f. To perform a gradient descent, we move in the direction of the negative Riemannian gradient, $-\operatorname{grad} f(p)$.

The exponential map, $\exp_p : T_p\mathcal{M} \to \mathcal{M}$, then plays a crucial role. It projects a tangent vector at p back to the manifold, tracing along a geodesic in the direction of the vector for a unit time, giving us our update step

$$p_{t+1} = \exp_{p_t}(-\alpha \operatorname{grad} f(p_t))$$

Refer to the figure below: a visualization of the role of the tangent space and exponential map when performing the update step.

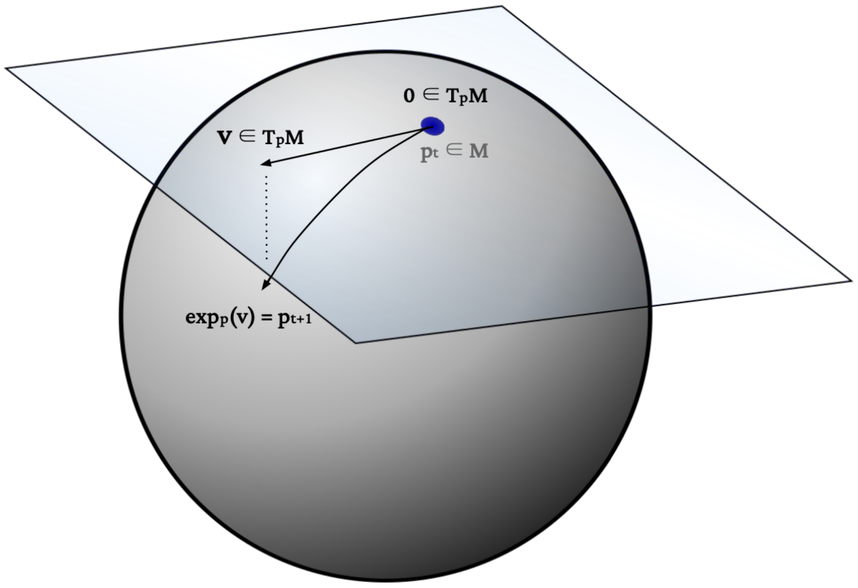

FIGURE 1. Visualization of the update step

Before we proceed, lets explore why the exponential map provides a valid update point for any input. Specifically, why the possibility of moving along geodesics for any required distance is guaranteed.

Theorem (Hopf-Rinow)**.** *Let $(\mathcal{M}, g)$ be a connected Riemannian manifold. The following statements are equivalent:*

1. *$\mathcal{M}$ is complete as a metric space.*
2. *$\mathcal{M}$ is geodesically complete, i.e. every geodesic $\gamma : [0, a) \to \mathcal{M}$ can be extended to $[0, \infty)$.*
3. *Any two points in $\mathcal{M}$ can be joined by a minimizing geodesic.*

Proof. *To prove this theorem, we will show that $1 \Rightarrow 2$, $2 \Rightarrow 3$, and $3 \Rightarrow 1$.*

(1 ⇒ 2) *Assume $\mathcal{M}$ is metrically complete. Consider a maximal geodesic*

$$\gamma : [0, a) \to \mathcal{M}$$

for some $a > 0$. If a is finite, we show γ can be extended beyond a, contradicting its maximality unless $a = \infty$.

If γ cannot be extended beyond a, consider a sequence $\{t_n\}$ in $[0, a)$ converging to a. Since $\mathcal{M}$ is complete, the sequence $\{\gamma(t_n)\}$ must converge to some point $p \in \mathcal{M}$. By the Picard-Lindelöf theorem, we can extend γ to a neighborhood around a by *setting $\gamma(a) = p$ and solving the geodesic equation with this initial condition. This contradicts the assumption that a was the upper bound.*

(2 $\Rightarrow$ 3) *Assume every geodesic can be extended indefinitely. Take any two points $p, q \in \mathcal{M}$. The set of all lengths of piecewise smooth curves connecting p and q has a lower bound (non-negative), so we consider a minimizing sequence of curves whose lengths approach the infimum.*

By the Arzelà-Ascoli theorem and the completeness of $\mathcal{M}$, a subsequence of these curves converges uniformly to a limit curve, which is a geodesic by the properties of the Riemannian metric and the limit process. This geodesic is necessarily minimizing because its length is the limit of the lengths of the curves in the minimizing sequence.

(3 $\Rightarrow$ 1) *Assume every two points in $\mathcal{M}$ can be joined by a minimizing geodesic. If $\mathcal{M}$ were not metrically complete, there would exist a Cauchy sequence $\{p_n\}$ in $\mathcal{M}$ that does not converge. However, the existence of minimizing geodesics between points of $\{p_n\}$ would imply that the sequence remains bounded and must have a convergent subsequence, contradicting the assumption that $\{p_n\}$ does not converge. Thus, $\mathcal{M}$ must be complete.* □

[Car92]. For further details on the Arzelà-Ascoli and Picard-Lindelöf theorems, refer to [Sha99] and [Nem13], respectively.

5.3. **Gradient Descent.** Now, we are ready to put it all together.

Algorithm (Riemannian gradient descent)**.** .

```
(1) Pick an arbitrary $p_0 \in \mathcal{M}$ and let $\alpha \in \mathbb{R}$ be the learning rate parameter with $\alpha > 0$
(2) Repeat until convergence:
    2.1 Compute $\nabla_f(p)$
    2.2 $p_{t+1} = \exp_{p_t}(-\alpha \operatorname{grad} f(p_t))$ i.e. the update step
    2.3 $t = t + 1$
(3) Return $p_t$
```

In this algorithm, $\exp_{p_t}$ denotes the exponential map at point p_t, which moves a small distance along the geodesic starting from p_t in the direction of $\operatorname{grad} f(p_t)$. This adjustment ensures that each update stays on the manifold and respects its curvature, unlike in Euclidean space where updates are unrestricted. [Smi93]

6. Challenges

In adapting gradient descent to Riemannian manifolds, several challenges arise that impact both the theoretical foundation and practical implementation of the algorithm.

6.1. **Computational Cost.** One of the key adaptations in extending gradient descent from Euclidean spaces to Riemannian manifolds is the use of the exponential map to define the update steps. While theoretically sound, the exponential map can be computationally expensive to evaluate, particularly for manifolds with complex geometries or high dimensions. [Ber16] The exponential map requires solving a system of ordinary differential equations (ODEs) to trace geodesics on the manifold, which can be computationally intensive and numerically challenging. [Smi93]

To mitigate the computational burden, practical applications often employ retraction maps as approximations to the exponential map. A retraction

$$R_p : T_p\mathcal{M} \to \mathcal{M}$$

is a smooth mapping that approximates the exponential map but is typically simpler to compute. Retractions still satisfy the basic requirement of moving along a direction in the tangent space while maintaining adherence to the manifold structure, though they may not strictly follow geodesics. The use of retractions provides a balance between computational efficiency and the need to respect the manifold's geometry, making them a viable alternative in applications where performance is a critical concern. [AMS08]

6.2. **Assumption of Connectivity.** The assumption that the Riemannian manifold $\mathcal{M}$ is connected is crucial for the theoretical aspects of Riemannian gradient descent, particularly in ensuring the global applicability of geodesics and the exponential map. This assumption simplifies many aspects of the analysis and ensures that the algorithm can theoretically converge from any starting point to a global minimum.

While this assumption is mathematically convenient, it may not always hold in practical scenarios, where the manifold might consist of several disconnected components. In such cases, the gradient descent algorithm may fail to find a global minimum if it starts in a component that does not contain any such minima. Moreover, if the manifold is not connected, the behavior of paths and the definition of distance can become more complex, affecting the convergence and efficiency of the algorithm. To address this, it may be necessary to analyze each component separately or to implement mechanisms that allow for jumping between components, though the latter approach can significantly complicate the algorithm and its analysis.

References

[Lee03] John M. Lee. *Introduction to Smooth Manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, New York, NY

[Lee18] John M. Lee. *Introduction to Riemannian manifolds*, volume 176 of *Graduate Texts in Mathematics*. Springer, Cham, second edition, 2018.

[AMS08] P.-A. Absil, R. Mahoney, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds. Princeton University Press, 2008.*

[Smi93] Steven Thomas Smith. Geometric Optimization Methods for Adaptive Filtering arXiv:1305.1886, 1993.

[Car92] Manfredo Perdigão do Carmo. *Riemannian Geometry*. Birkhäuser, 1992.

[Hu19] Jiang Hu, Xin Liu, Zaiwen Wen, Yaxiang Yuan. *A Brief Introduction to Manifold Optimization arXiv:1906.05450, 2019.*

[GN14] Leonor Godinho, José Natário. An Introduction to Riemannian Geometry: With Applications to Mechanics and Relativity*. Springer, 2014.*

[Ber16] Dimitri Bertsekas. Nonlinear Programming: 3rd Edition*. Athena Scientific, 2016.*

[Sha99] Joel H. Shapiro. The Arzelà-Ascoli Theorem, lecture notes*. Michigan State University, 1999.*

[Nem13] Martin Nemer. Picard-Lindelof Theorem*. University of New Mexico, 2015*