1 Introduction

Optimization is a staple of mathematical modeling. In this rich framework, we consider a set S called the *search space*—it contains all possible answers to our problem, good and bad—and a *cost function* $f\colon S \to \mathbb{R}$ which associates a cost $f(x)$ to each element x of S. The goal is to find $x \in S$ such that $f(x)$ is as small as possible, that is, a best answer. We write

$$\min_{x \in S} f(x)$$

to represent both the optimization problem and the minimal cost (if it exists). Occasionally, we wish to denote specifically the subset of S for which the minimal cost is attained; the standard notation is

$$\arg\min_{x \in S} f(x),$$

bearing in mind that this set might be empty. We will discuss a few simple applications which can be modeled in this form.

Rarely, optimization problems admit an analytical solution. Typically, we need numerical algorithms to (try to) solve them. Often, the best algorithms exploit mathematical structure in S and f.

An important special case arises when S is a linear space such as $\mathbb{R}^n$. Minimizing a function f in $\mathbb{R}^n$ is called *unconstrained optimization* because the variable x is free to move around $\mathbb{R}^n$, unrestricted.

If f is sufficiently differentiable and $\mathbb{R}^n$ is endowed with an inner product (i.e., if we make it into a Euclidean space), then we have a notion of gradient and perhaps even a notion of Hessian for f. These objects give us a firm understanding of how f behaves locally around any given point. Famous algorithms such as gradient descent and Newton's method exploit these objects to move around $\mathbb{R}^n$ efficiently in search of a solution.

Notice, however, that the Euclidean structure of $\mathbb{R}^n$ and the smoothness of f are irrelevant to the definition of the optimization problem itself: they are merely structures that we may (and as experience shows, we should) use algorithmically to our advantage.

Subsuming linearity, we focus on *smoothness* as the key structure to exploit: we assume the set S is a *smooth manifold* and the function f is smooth on S. This calls for precise definitions, constructed first in Chapter 3. For a first intuition, one can think

https://doi.org/10.1017/9781009166164.002 Published online by Cambridge University Press

of smooth manifolds as surfaces in $\mathbb{R}^n$ that do not have kinks or boundaries, such as a plane, a sphere, a torus, or a hyperboloid.

We could think of optimization over such surfaces as *constrained*, in the sense that x is not allowed to move freely in $\mathbb{R}^n$: it is constrained to remain on the surface. Alternatively, and this is the viewpoint favored here, we can think of this as unconstrained optimization, in a world where the smooth surface is the only thing that exists: like an ant walking on a large ball might feel unrestricted in its movements, aware only of the sphere it lives on; or like the two-dimensional inhabitants of Flatland [Abb84] who find it hard to imagine that there exists such a thing as a third dimension, feeling thoroughly free in their own subspace.

A natural question then is: can we generalize the Euclidean algorithms from unconstrained optimization to handle the broader class of optimization over smooth manifolds? The answer is essentially yes, going back to the 1970s [Lue72, Lic79], the 1980s [Gab82] and the 1990s [Udr94, Smi94, HM96, Rap97, EAS98], and sparking a significant amount of research in the past two decades.

To generalize algorithms such as gradient descent and Newton's method, we need a proper notion of gradient and Hessian on smooth manifolds. In the linear case, this required the introduction of an inner product: a Euclidean structure. In our more general setting, we leverage the fact that smooth manifolds can be linearized locally around every point. The linearization at x is called the *tangent space* at x. By endowing each tangent space with its own inner product (varying smoothly with x, in a sense to be made precise), we construct what is called a *Riemannian structure* on the manifold: it becomes a *Riemannian manifold*.

A Riemannian structure is sufficient to define gradients and Hessians on the manifold, paving the way for optimization. There exist several Riemannian structures on each manifold: our choice may impact algorithmic performance. In that sense, identifying a useful structure is part of the algorithm design—as opposed to being part of the problem formulation, which ended with the definition of the search space (as a crude set) and the cost function.

Chapter 2 covers a few simple applications, mostly to give a sense of how manifolds come up. We then go on to define smooth manifolds in a restricted[1] setting in Chapter 3, where manifolds are *embedded* in a linear space, much like the unit sphere in three-dimensional space. In this context, we define notions of smooth functions, smooth vector fields, gradients and *retractions* (a means to move around on a manifold). These tools are sufficient to design and analyze a first optimization algorithm in Chapter 4: Riemannian gradient descent. As readers progress through these chapters, it is the intention that they also read bits of Chapter 7 from time to time: useful embedded manifolds are studied there in detail. Chapter 5 provides more advanced geometric tools for embedded manifolds, including the notions of Riemannian *connections* and

[1] Some readers may know Whitney's celebrated embedding theorems, which state that any smooth manifold can be embedded in a linear space [BC70, p. 82]. The mere existence of an embedding, however, is of little use for computation.

https://doi.org/10.1017/9781009166164.002 Published online by Cambridge University Press

Hessians. These are put to good use in Chapter 6 to design and analyze Riemannian versions of Newton's method and the trust-region method.

The linear *embedding space* is useful for intuition, to simplify definitions, and to design tools. Notwithstanding, all the tools and concepts we define in the restricted setting are *intrinsic*, in the sense that they are well defined regardless of the embedding space. We make this precise much later, in Chapter 8, where all the tools from Chapters 3 and 5 are redefined in the full generality of standard treatments of differential geometry. This is also the time to discuss topological issues to some extent. Generality notably makes it possible to discuss a more abstract class of manifolds called *quotient manifolds* in Chapter 9. They offer a beautiful way to harness symmetry, so common in applications.

In closing, Chapter 10 offers a limited treatment of more advanced geometric tools such as the Riemannian distance, geodesics, the exponential map and its inverse, parallel transports and transporters, notions of Lipschitz continuity, finite differences, and covariant differentiation of tensor fields. Then, Chapter 11 covers elementary notions of convexity on Riemannian manifolds with simple implications for optimization. This topic has been around since the 1990s, and has been gaining traction in research lately.

More than 150 years ago, Riemann invented a new kind of geometry for the abstract purpose of understanding curvature in high-dimensional spaces. Today, this geometry plays a central role in the development of efficient algorithms to tackle technological applications Riemann himself—arguably—could have never envisioned. Through this book, I invite you to enjoy this singularly satisfying success of mathematics, with an eye to turning geometry into algorithms.

https://doi.org/10.1017/9781009166164.002 Published online by Cambridge University Press