Functions and Differentiability

This section will provide you with a brief overview of the concepts of functions and differentiability. A function *f* is a rule that takes an input to an output. There are functions in all computer programming languages, and the mathematical definition of a function isn't really much different. However, mathematical functions commonly used in physics and engineering have other important properties such as continuity and differentiability. A continuous function, loosely speaking, is one that can be drawn without lifting your pencil from the paper, as shown in Figure 3-1. (This is of course not the technical definition, but it captures the spirit of the continuity condition.)

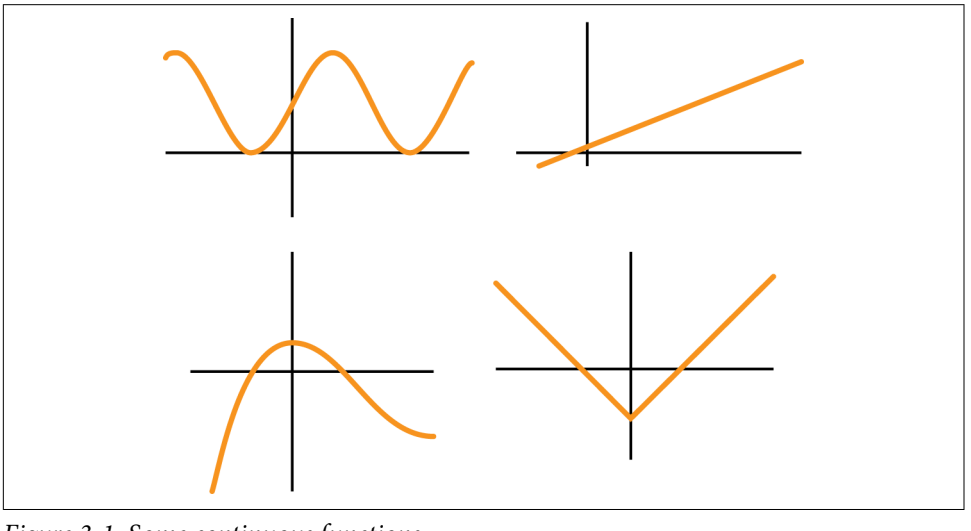

Figure 3-1. Some continuous functions.

Differentiability is a type of smoothness condition on functions. It says no sharp corners or turns are allowed in the function (Figure 3-2).

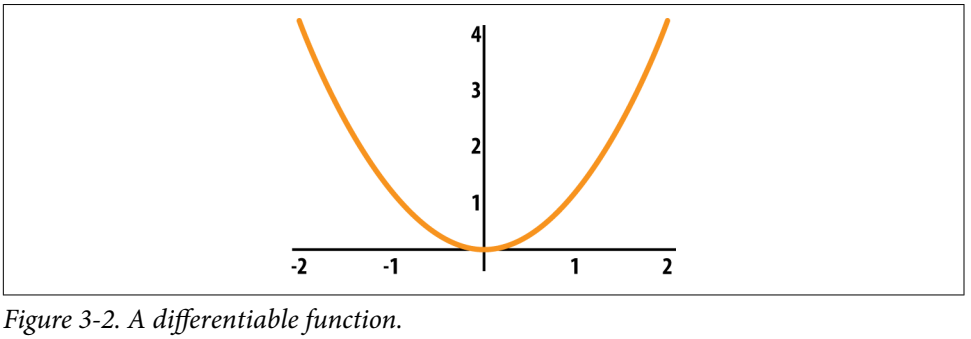

Figure 3-2. A differentiable function.

The key advantage of differentiable functions is that we can use the slope of the function at a particular point as a guide to find places where the function is higher or lower than our current position. This allows us to find the *minima* of the function. The *derivative* of differentiable function f, denoted f', is another function that provides the slope of the original function at all points. The conceptual idea is that the derivative of a function at a given point gives a signpost pointing to directions where the function is higher or lower than its current value. An optimization algorithm can follow this signpost to move closer to a minima of f. At the minima itself, the function will have derivative zero.

The power of derivative-driven optimization isn't apparent at first. Generations of calculus students have suffered through stultifying exercises minimizing tiny functions on paper. These exercises aren't useful since finding the minima of a function with only a small number of input parameters is a trivial exercise best done graphically. The power of derivative-driven optimization only becomes evident when there are hundreds, thousands, millions, or billions of variables. At these scales, understanding the function analytically is nigh impossible, and all visualizations are fraught exercises that may well miss the key attributes of the function. At these scales, the *gradient* of the function ∇f, a generalization of f' to multivariate functions, is likely the most powerful mathematical tool to understand the function and its behavior. We will dig into gradients in more depth later in this chapter. (Conceptually that is; we won't cover the technical details of gradients in this work.)

At a very high level, machine learning is simply the act of function minimization: learning algorithms are nothing more than minima finders for suitably defined functions. This definition has the advantage of mathematical simplicity. But, what are these special differentiable functions that encode useful solutions in their minima and how can we find them?

Loss Functions

In order to solve a given machine learning problem, a data scientist must find a way of constructing a function whose minima encode solutions to the real-world problem at hand. Luckily for our hapless data scientist, the machine learning literature has built up a rich history of *loss functions* that perform such encodings. Practical machine learning boils down to understanding the different types of loss functions available and knowing which loss function should be applied to which problems. Put another way, the loss function is the mechanism by which a data science project is transmuted into mathematics. All of machine learning, and much of artificial intelligence, boils down to the creation of the right loss function to solve the problem at hand. We will give you a whirlwind tour of some common families of loss functions.

We start by noting that a loss function $\mathscr{L}$ must satisfy some mathematical properties to be meaningful. First $\mathscr{L}$ must use both datapoints x and labels y. We denote this by