**Chapter 4**

**NUMERICAL COMPUTATION**

* 1. **Overflow and Underflow**

Rounding error is problematic, especially when it compounds across many operations, and can cause algorithms that work in theory to fail in practice if they are not designed to minimize the accumulation of rounding error.

One form of rounding error that is particularly devastating is *underflow*. Underflow occurs when numbers near zero are rounded to zero.

Another highly damaging form of numerical error is *overflow*. Overflow occurs when numbers with large magnitude are approximated as ∞ or −∞.

One example of a function that must be stabilized against underflow and overflow is the softmax function. The softmax function is often used to predict the probabilities associated with a multinoulli distribution. The softmax function is defined to be

* 1. **Poor Conditioning**

Conditioning refers to how rapidly a function changes with respect to slight changes in its inputs. Functions that change rapidly when their inputs are perturbed slightly can be problematic for scientific computation because rounding errors in the inputs can result in substantial changes in the output.

* 1. **Gradient-Based Optimization**
* Most deep learning algorithms involve optimization of some sort. Optimization refers to the task of either minimizing or maximizing some function f (x ) by altering x.
* The function we want to minimize or maximize is called the objective function or criterion. When we are minimizing it, we may also call it the cost function, loss function, or error function.
* We often denote the value that minimizes or maximizes a function with a superscript ∗. For example, we might say x\* = argmin f(x).
* The derivative f’(x) gives the slope of f (x) at the point x. In other words, it specifies how to scale a slight change in the input to obtain the corresponding change in the output:

f(x + €) ≈ f(x) + € f’(x).

* The derivative is therefore useful for minimizing a function because it tells us how to change x to make a small improvement in y. This technique is called **gradient descent.**
* When f’(x) = 0, the derivative provides no information about which direction to move. Points where f’(x) = 0 are known as **critical points or stationary points**. A **local minimum** is a point where f(x) is lower than at all neighboring points, so it is no longer possible to decrease f (x) by making infinitesimal steps. A **local maximum** is a point where f(x) is higher than at all neighboring points, so it is not possible to increase f(x) by making infinitesimal steps. Some critical points are neither maxima nor minima. These are known as **saddle points**. A point that obtains the absolute lowest value of f ( x) is a **global minimum**.
* For functions with multiple inputs, we must make use of the concept of partial derivatives. The partial derivative ∂/∂x f( x) measures how f changes as only the variable xi increases at point x. The gradient generalizes the notion of derivative to the case where the derivative is with respect to a vector: the gradient of f is the vector containing all of the partial derivatives, denoted ∇xf (x). Element i of the gradient is the partial derivative of f with respect to xi. In multiple dimensions, critical points are points where every element of the gradient is equal to zero.
* The **directional derivative**in direction u (a unit vector) is the slope of the function f in direction u. In other words, the directional derivative is the derivative of the function f (x + αu) with respect to α , evaluated at α = 0. Using the chain rule, we can see that
* To minimize f , we would like to find the direction in which f decreases the fastest. We can do this using the directional derivative:
* where θ is the angle between u and the gradient. Substituting in ||u||2 = 1 and ignoring factors that do not depend on u, this simplifies to minu cos θ.
* the gradient points directly uphill, and the negative gradient points directly downhill. We can decrease f by moving in the direction of the negative gradient. This is known as the method of steepest descent or gradient descent. Steepest descent proposes a new point:

x’ = x − €∇xf(x)

* where € is the learning rate, a positive scalar determining the size of the step. We can choose € in several ways.

1. A popular approach is to set € to a small constant. Sometimes, we can solve for the step size that makes the directional derivative vanish.
2. Another approach is to evaluate f (x − €∇xf(x)) for several values of € and choose the one that results in the smallest objective function value. This last strategy is called a line search.

* Steepest descent converges when every element of the gradient is zero (or, in practice, very close to zero). In some cases, we may be able to avoid running this iterative algorithm, and just jump directly to the critical point by solving the equation ∇xf(x) = 0 for x.
  + 1. **Beyond the Gradient: Jacobian and Hessian Matrices**