Steepest Descent

MA 348

Kurt Bryan

A Calc III Refresher

A one-dimensional curve C in n-dimensional space can be described parametrically by giving each coordinate x_k as a function of a single variable, say t, as $(x_1(t), x_2(t), \ldots, x_n(t))$. If we let boldface variables denote vectors, so $\mathbf{x} = (x_1, \ldots, x_n)$, then a curve is described parametrically as $\mathbf{x} = \mathbf{x}(t)$. Any curve has infinitely many parameterizations. You can think of $\mathbf{x}(t)$ as describing the motion of a particle along the curve C as a function of time t.

Now consider a function $f(\mathbf{x})$ of n variables and a curve C parameterized by $\mathbf{x} = \mathbf{x}(t)$. Along the curve C the function f itself becomes a function of t, as the composition $f(\mathbf{x}(t)) = f(x_1(t), x_2(t), \ldots, x_n(t))$. If we differentiate $f(\mathbf{x}(t))$ with respect to t we obtain, by the chain rule,

$$\frac{d}{dt}f(\mathbf{x}(t)) = \frac{\partial f}{\partial x_1}\frac{dx_1}{dt} + \cdots + \frac{\partial f}{\partial x_n}\frac{dx_n}{dt},$$

which is just the rate at which f changes along C with respect to t. This can also be written as a dot product

$$\frac{d}{dt}f(\mathbf{x}(t)) = \nabla f(\mathbf{x}(t)) \cdot \frac{d\mathbf{x}}{dt} \tag{1}$$

where the vectors ∇f and $\frac{d\mathbf{x}}{dt}$ are defined as

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \left(\frac{dx_1}{dt}, \ldots, \frac{dx_n}{dt}\right) \\ \nabla f &= \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right). \end{aligned}$$

We'll also use the notation $\mathbf{x}'$ for $\frac{d\mathbf{x}}{dt}$. Geometrically, $\mathbf{x}'$ is a tangent vector to the curve parameterized by $\mathbf{x}(t) = (x_1(t), \ldots, x_n(t))$, or $\mathbf{x}'(t)$ can be thought of as the velocity of a particle moving according to $\mathbf{x}(t)$ (with t as time). The gradient vector ∇f at any point $\mathbf{x}$ points in the direction in which f increases most rapidly near $\mathbf{x}$ (more this below). The quantity $\nabla f \cdot \mathbf{x}'$ is called the *directional derivative of f in the direction* $\mathbf{x}'$, and is simply a measured of how fast f changes along the curve C as a function of t.

Example: Let $f(x_1, x_2, x_3) = x_1 x_2 - x_3^2$. Consider a curve parameterized by $x_1 = t^2$, $x_2 = t + 3$, $x_3 = t^2 + t$, or just $\mathbf{x}(t) = (t^2, t+3, t^2+t)$ in vector notation. Then we have $\mathbf{x}(1) = (1, 4, 2)$ and $\mathbf{x}'(1) = (2, 1, 3)$. Also, $\nabla f(1, 4, 2) = (4, 1, -4)$. Thus when $t = 1$ the function $f(\mathbf{x}(t))$ is changing at a rate of $(4, 1, -4) \cdot (2, 1, 3) = -3$. You can also check this directly by noting that $f(\mathbf{x}(t)) = t^2(t+3) - (t^2+t)^2 = -t^4 - t^3 + 2t^2$, so $\frac{d}{dt}(f(\mathbf{x}(t))) = -4t^3 - 3t^2 + 4t$ and at $t = 1$ this is -3.

Recall that for vectors $\mathbf{v}$ and $\mathbf{w}$ we have

$$\mathbf{v} \cdot \mathbf{w} = |\mathbf{v}||\mathbf{w}|\cos(\theta)$$

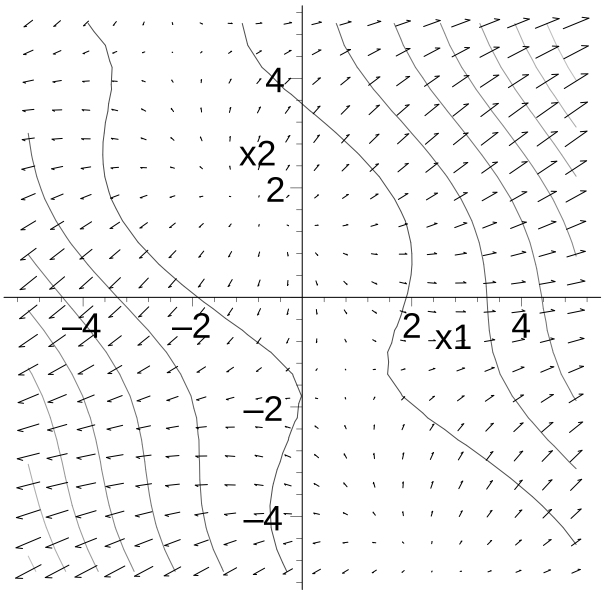
where $|\mathbf{v}|$ is the length of $\mathbf{v}$, $|\mathbf{w}|$ is the length of $\mathbf{w}$, and θ is the angle between the vectors $\mathbf{v}$ and $\mathbf{w}$. From this it follows that

$$\nabla f \cdot \mathbf{x}' = |\nabla f||\mathbf{x}'|\cos(\theta) \tag{2}$$

where θ is the angle between ∇f and $\mathbf{x}'$. This makes it easy to see that of all possible curves C parameterized by $\mathbf{x}(t)$ and passing though a given point $\mathbf{p}$ with $|\mathbf{x}'|$ fixed (which describes a particle moving at some fixed speed), the choice for $\mathbf{x}$ that maximizes the rate of change of f along C at $\mathbf{p}$ is obtained when $\theta = 0$ (so $\cos(\theta) = 1$), i.e., $\mathbf{x}'$ should be parallel to $\nabla f(\mathbf{p})$. The direction parallel to $\nabla f(\mathbf{p})$ is called the *direction of steepest ascent.*

On the other hand, if $\mathbf{x}'$ is opposed to ∇f (so $\theta = \pi$) we see from equation (2) that the directional derivative is minimized. The direction of $-\nabla f$ is the *direction of steepest descent.* If $\mathbf{x}'$ is orthogonal to ∇f at some point then from equation (2) the function f is (at least momentarily) constant along the curve C at this point.

The gradient ∇f has a nice relation to the level curves or surfaces of f (curves or surfaces of the form $f(\mathbf{x}) = c$ for a constant c): the gradient is always orthogonal to the level curves. This fact is illustrated by overlaying a contour plot of f and a plot of ∇f, in this case for $f(x_1, x_2) = x_1x_2 + x_1^2 + x_2^2/5 - 3\sin(x_2)$:



The Method of Steepest Descent

The method of steepest descent is a very simple algorithm that you yourself would have invented for optimization, if only you'd been born before Cauchy. A notational note: I'll use boldface $\mathbf{x}_k$ to refer to a point in n dimensions, while normal typeface x_k mean the kth component of a point $\mathbf{x}$.

Here's how steepest descent works for a differentiable function $f(\mathbf{x})$:

1. Make an initial guess $\mathbf{x}_0$ at the location of a minimum; set a loop counter $k = 0$.

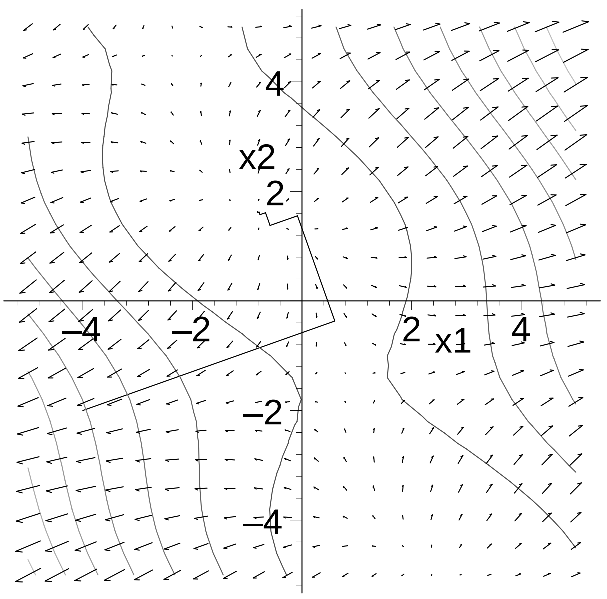
2. Compute a vector $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$. The vector $\mathbf{d}_k$ is called the *search direction.* Note that it points downhill from the point $\mathbf{x}_k$ in the steepest possible direction.

3. Define a line $\mathbf{x}(t) = \mathbf{x}_k + t\mathbf{d}_k$ passing through $\mathbf{x}_k$ in the direction $\mathbf{d}_k$. Minimize $g(t) = f(\mathbf{x}(t))$; note that this is a one-dimensional optimization problem, so you can apply any 1D technique you like. This step is called a *line search.*

4. If the minimum of g occurs at $t = t^*$ then set $\mathbf{x}_{k+1} = \mathbf{x}(t^*)$.

5. If some termination criteria is met, stop. Otherwise, increment $k = k + 1$ and return to step 2.

The figure below shows a contour plot of the function $f(x_1, x_2) = x_1^2 + x_1x_2 - 3\sin(x_2) + x_2^2/5$, together with a plot of the progress of steepest descent started at $(-4, -2)$. The 1D line searches were done with Golden Section and the algorithm terminated after 12 iterations, at the point $(-0.812, 1.625)$. This is a local minimum of f, to 3 significant figures.

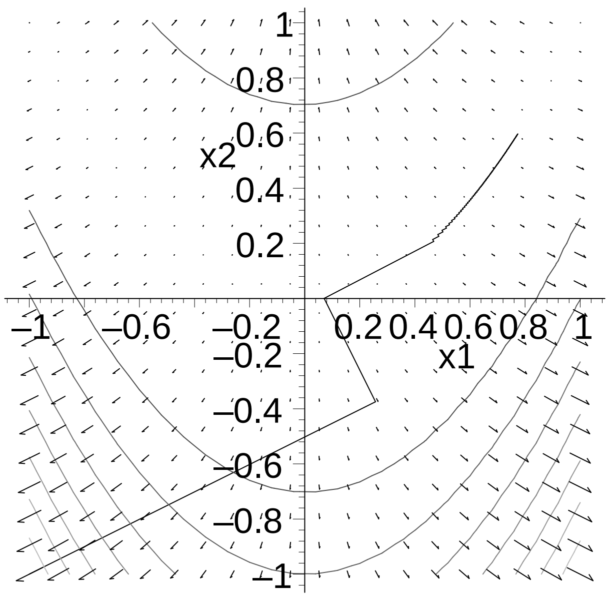


There is one key feature of steepest descent that is worth noticing: at the end of each line search the algorithm ALWAYS make a right angle turn—the new search direction $\mathbf{d}_{k+1}$ is always orthogonal to the previous search direction $\mathbf{d}_k$. It's easy to see why. To obtain $\mathbf{x}_{k+1}$ we searched along a line of the form $\mathbf{x}(t) = \mathbf{x}_k + t\mathbf{d}_k$, where $\mathbf{x}_k$ was our previous "base point". At a minimum $t = t^*$ of the function $f(\mathbf{x}(t))$ we must have that $d(f(\mathbf{x}(t)))/dt = 0$, which by equation (2) (and note $\mathbf{x}'(t) = \mathbf{d}_k$) means that

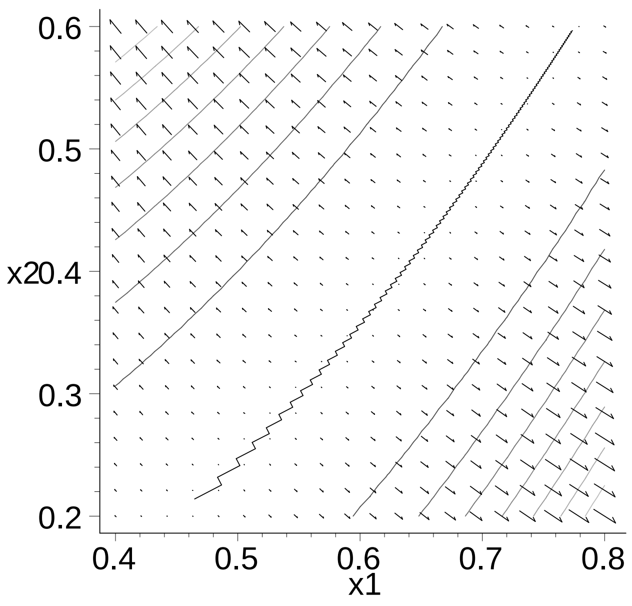
$$\mathbf{x}' \cdot \nabla f(\mathbf{x}(t^*)) = \mathbf{d}_k \cdot \nabla f(\mathbf{x}_{k+1}) = 0,$$

where $\mathbf{x}_{k+1} = \mathbf{x}(t^*)$ is the new base point. Thus the new search direction $\mathbf{d}_{k+1} = -\nabla f(\mathbf{x}_{k+1})$ must be orthogonal to the old search direction $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$.

Steepest might seem to be "optimal". After all, what could be more efficient than repeatedly going downhill in the steepest possible direction? Unfortunately, this turns out to be strangely inefficient. Consider Rosenbrock's function $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ with steepest descent initial guess $\mathbf{x}_0 = (-1, -1)$. Below is a plot of 200 iterations of the progress of the algorithm toward the true minimum at $(1, 1)$:



Doesn't look too bad—the first few line searches seem to make good progress. But let's zoom in on the line searches once we're in the valley leading to the point $(1, 1)$:



Now you can see that steepest descent wasn't so efficient in its progress toward the minimum; it zigzags wastefully back and forth.

This is somewhat similar to the first algorithm we looked at for minimizing functions of n variables: minimize in one coordinate at a time. The problem with that algorithm was that being forced to move at all times parallel to a coordinate axis can lead to very slow convergence if we need to move diagonally. Something similar is happening here: We need to move in a certain direction to make progress toward the minimum, but we must always make a right angle turn at the conclusion of each line search. This narrows our choice of search directions—in effect, the direction of the first line search limits our choice for all future line searches, and may exclude the choices that lead rapidly to a minimum. In the figure above we want to move diagonally toward $(1, 1)$, but our choices for the search direction do not include the directions we need. As a result we have to zigzag back and forth across the proper direction, slowing our progress. This inefficiency is why the method of steepest descent is rarely used as is. There are methods which require little more computation than steepest descent and yet which converge much more rapidly.

Still, steepest descent is the basic model algorithm for what is to follow. The key to improving steepest descent is to make a small change in how we pick the search direction for each line search.