2023-08-24

1 Optimality conditions

In an unconstrained problem with a differentiable objective function, a necessary (but not sufficient) condition for x_* to be a local minimizer is that $\phi'(x_*) = 0$. For intuition, picture a function $\phi : \mathbb{R}^n \to \mathbb{R}$; if you'd like to be concrete, let $n = 2$. Absent a computer, we might optimize ϕ by the physical experiment of dropping a tiny ball onto the surface and watching it roll downhill (in the steepest descent direction) until it reaches the minimum. The statement that $\phi'(x_*) = 0$ (or that $\nabla\phi(x_*) = 0$) basically means the function looks flat at x_* to a sufficiently near-sighted observer; if $\phi'(x_*)$ is not zero, then $x_* - \epsilon\nabla\phi(x_*)$ will be a little bit "downhill" of x_*; that is, if $\|\nabla\phi(x_*)\| \neq 0$ then

$$\phi(x_* - \epsilon\nabla\phi(x_*)) = \phi(x_*) - \epsilon\|\nabla\phi(x_*)\|^2 + o(\epsilon) < \phi(x_*)$$

for sufficiently small ϵ.

Most students learn the first-order optimality conditions for unconstrained optimization in a first course, but sometimes that course gets everyone too stuck on the idea of computing a gradient. What is really happening is that the function should be "flat in all directions," i.e. all directional derivatives are zero. This is equivalent to the statement that the gradient is zero, of course, but sometimes it is notationally easier to check that an arbitrary directional derivative is zero than to try to write down the gradient. For example, consider the quadratic objective

$$\phi(x) = \frac{1}{2}x^T Ax + x^T b + c.$$

Now, we will write an arbitrary directional derivative of ϕ in terms of "variational notation" (described in the background notes):

$$\delta\phi(x) = \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \phi(x + \epsilon\delta x) = (\delta x)^T(Ax + b).$$

At a critical point, $\delta\phi(x)$ should be zero for any choice of δx, so the stationary point occurs at $Ax_* + b = 0$. There is a unique minimizer x_* if A is positive definite. When the number of variables is not too large — up to a

few thousand, say — we might solve this system of linear equations directly using a variant of Gaussian elimination if we wanted to find the minimizer. When the number of variables is much larger, we may prefer to use an iterative method to solve the system, e.g. the method of conjugate gradients (CG). This method can be interpreted either as an iterative solver for linear equations or as an iterative optimization method.

Now let's turn to the *constrained* case. Rather than repeating the formal derivation of the first-order constrained optimality conditions that you have likely seen before, let me again give you an interpretation that involves some physical intuition. For the unconstrained case, we thought about solving the problem by rolling a tiny ball down hill until it came to rest. If we wanted to solve a constrained minimization problem, we could build a great wall between the feasible and the infeasible region. A ball rolling into the wall would still roll freely in directions tangent to the wall (or away from the wall) if those directions were downhill; at a constrained miminizer, the force pulling the ball downhill would be perfectly balanced against an opposing force pushing into the feasible region in the direction of the normal to the wall. If the feasible region is $\{x : c(x) \leq 0\}$, the normal direction pointing inward at a boundary point x_* s.t. $c(x_*) = 0$ is proportional to $-\nabla c(x_*)$. Hence, if x_* is a constrained minimum, we expect the sum of the "rolling downhill" force ($-\nabla \phi$) and something proportional to $-\nabla c(x_*)$ to be zero:

$$-\nabla \phi(x_*) - \mu \nabla c(x_*) = 0.$$

The *Lagrange multiplier* μ in this picture represents the magnitude of the restoring force from the wall balancing the tendency to roll downhill.

More abstractly, and more generally, suppose that we have a mix of equality and inequality constraints. We define the *augmented Lagrangian*

$$L(x, \lambda, \mu) = \phi(x) + \sum_{i \in \mathcal{E}} \lambda_i c_i(x) + \sum_{i \in \mathcal{I}} \mu_i c_i(x).$$

The *Karush-Kuhn-Tucker (KKT) conditions* for x_* to be a constrained minimizer are

$$\begin{aligned}
\nabla_x L(x_*) &= 0 & & \\
c_i(x_*) &= 0, \quad i \in \mathcal{E} & & \text{equality constraints} \\
c_i(x_*) &\leq 0, \quad i \in \mathcal{I} & & \text{inequality constraints} \\
\mu_i &\geq 0, \quad i \in \mathcal{I} & & \text{non-negativity of multipliers} \\
c_i(x_*)\mu_i &= 0, \quad i \in \mathcal{I} & & \text{complementary slackness}
\end{aligned}$$

where the (negative of) the "total force" at x_* is

$$\nabla_x L(x_*) = \nabla\phi(x_*) + \sum_{i\in\mathcal{E}} \lambda_i \nabla c_i(x_*) + \sum_{i\in\mathcal{I}} \mu_i \nabla c_i(x_*).$$

The complementary slackness condition corresponds to the idea that a multiplier should be nonzero only if the corresponding constraint is active (a "restoring force" is only present if our test ball is pushed into a wall).

Like the critical point equation in the unconstrained case, the KKT conditions define a set of (necessary but not sufficient) nonlinear algebraic equations that must be satisfied at a minimizer. I like to think about the "rolling downhill" intuition for these necessary conditions because it suggests a way of thinking about numerical methods.

For completeness, we will say a few brief words about the second-order *sufficient* conditions for optimality. In the unconstrained case, x_* is a strong local minimizer of ϕ if $\nabla\phi(x_*) = 0$ and the Hessian matrix H_ϕ is positive definite; that is because in this case x_* is the strong minimizer of the quadratic approximation

$$\phi(x) \approx \phi(x_*) + \frac{1}{2}(x - x_*)^T H_\phi(x_*)(x - x_*).$$

In the constrained case, the Hessian only needs to be positive definite for those u that are orthogonal to $\nabla c_i(x_*)$ for each c_i that is active (has a nonzero Lagrange multiplier). We will see this idea in two weeks when we talk about kernel methods, and in particular talk about the idea of a *conditionally positive definite* kernel function.

2 Numerical methods

With our lightning review of some fundamental theory out of the way, it is time for a lightning overview of some numerical methods! We will see additional solver ideas as we move through the semester, but these are nicely prototypical examples that illustrate two running themes in the design of numerical methods for optimization.

Fixed point iterations All our optimizers and nonlinear solvers (and some of our linear solvers) will be *iterative*. We can write most as *fixed*

point iterations

$$x^{k+1} = G(x^k), \tag{1}$$

which we hope will converge to a fixed point, i.e. $x^* = G(x^*)$. We often approach convergence analysis through the *error iteration* relating the error $e^k = x^k - x^*$ at successive steps:

$$e^{k+1} = G(x^* + e^k) - G(x^*). \tag{2}$$

As a teaser for this sort of analysis, consider one of the simplest algorithms I know: gradient descent with a fixed step size h, applied to the quadratic model problem

$$\phi(x) = \frac{1}{2} x^T A x + b^T x + c$$

where A is assumed to be symmetric and positive definite. The algorithm produces iterates

$$\begin{aligned} x^{k+1} &= x^k - h\nabla\phi(x^k) \\ &= x^k - h(Ax^k + b) \\ &= (I - hA)x^k - hb. \end{aligned}$$

Now we subtract the fixed point equation for the true solution x^* in order to get an error iteration:

$$\begin{array}{r} [x^{k+1} = (I - hA)x^k - hb] \\ -[x^* = (I - hA)x^* - hb] \\ \hline =[e^{k+1} = (I - hA)e^k] \end{array}$$

where $e^k = x^k - x^*$. The error iteration converges iff the largest eigenvalue of A is less than $2h^{-1}$; if this condition is satisfied, then

$$\|e^{k+1}\| \leq (1 - h\lambda_{\max}(A))\|e^k\|$$

and so we have $\|e^{k+1}\| \leq (1 - h\lambda_{\max}(A))^{k+1}\|e^0\|$, a convergence rate which is known as (R)-linear convergence or as geometric convergence, depending on which corner of the literature one prefers to read.