CS 6241: Numerics for Data Science

Stochastic grdients, scaling, and Neston

David Bindel

2025-02-04

Stochastic gradient methods

In the last half of the last lecture, we discussed the gradient descent iteration

$$x^{k+1} = x^k - \alpha_k \nabla \phi(x^k).$$

For small enough fixed α and nice enough ϕ, we can guarantee that the error scales like $\|e^k\| = O(\rho^k)$ for some $\rho < 1$. This type of convergence is known by optimizers as *(R)-linear convergence*, and in machine learning it is sometimes called *geometric convergence*. We also saw last time that we can sometimes still obtain convergence results even when $\nabla \phi(x^k)$ is not computed exactly, as long as the errors in the gradient computation are controlled in some way.

The *stochastic gradient* methods replace $\nabla \phi(x)$ by a randomized estimator. These methods are typically applied to objective functions that consist of a large number of independent terms, e.g.

$$\phi(x) = \frac{1}{N} \sum_{i=1}^{N} \phi_i(x)$$

In this case, we can *randomly sample* the ϕ_i in order to obtain an unbiased estimate of the gradient, e.g.

$$\nabla \phi(x) = \mathbb{E}_i[\nabla \phi_i(x)].$$

This estimator is unbiased, but the variance is high; in order to reduce the variance, one sometimes uses randomly-selected "minibatches" of points

$$\nabla \phi(x) = \mathbb{E}_{\mathcal{J}} \left[\frac{1}{|\mathcal{J}|} \sum_{i \in \mathcal{J}} \nabla \phi_i(x) \right].$$

Let's call such estimators $g(x, \xi)$, where ξ is a random variable that determines the selection of data used in the estimator. Then the stochastic gradient algorithm is

$$x^{k+1} = x^k - \alpha_k g(x^k, \xi_k).$$

What does the convergence of the stochastic gradient algorithm look like? For nice enough functions and a sufficiently small fixed step size α, the expected values optimality gap behaves like

$$\mathbb{E}[\phi(x^k) - \phi(x^*)] \leq c_1\alpha + (1 - c_2\alpha)^{-k} \left(\phi(x^0) - \phi(x^*)\right)$$

That is, the expected optimality gap converges linearly, but not to zero! To get closer to the true optimal value, we have to reduce the step size. Unfortunately, reducing the step size also reduces the rate of convergence! We can balance the two effects by taking n_0 steps with an initial size of α_0 to get the error down to $O(\alpha_0)$, $2n_0$ steps of size $2^{-1}\alpha_0$ to get the error down to $O(2^{-1}\alpha)$, and so forth. This gives us a convergence rate of $O(1/k)$. More generally, we can get convergence with any (sufficiently small) schedule of step sizes such that

$$\sum_{k=1}^{\infty} \alpha_k = \infty, \quad \sum_{k=1}^{\infty} \alpha_k^2 < \infty.$$

There are a wide variety of methods for choosing the step sizes ("learning rate"), sometimes in conjunction with methods to choose a better search direction than the (approximate) steepest descent direction.

The $O(1/k)$ rate of convergence for stochastic gradient descent is quite slow compared to the rate of convergence for ordinary gradient descent. However, each step of a stochastic gradient method may be much cheaper, so there is a tradeoff. The slow rate of the stochastic gradient method comes from a combination of two effects: variance in the gradient estimates, and the slow rate of gradient descent when the problem is ill-conditioned.

Scaling Steepest Descent

Let us put aside, for now, the stochastic methods and instead return to gradient descent. We saw last time that with an optimal step size, the convergence of gradient descent on a positive definite quadratic model problem behaves like

$$\|e^k\| \leq \rho^k \|e^0\|, \quad \text{where } \rho = 1 - O(\kappa(A)^{-1}),$$

where $\kappa(A) = \lambda_{\max}(A)/\lambda_{\min}(A)$ is the condition number of A. Hence, if $\kappa(A)$ is large (the problem is *ill-conditioned*), then convergence can be quite slow. Sometimes slow convergence is a blessing in disguise, as we saw last time, but sometimes we really do want a faster method. What can we do?

A natural generalization of steepest descent is *scaled* steepest descent. In this iteration, we choose a positive definite matrix M, and use the iteration

$$\begin{aligned} p^k &= -M^{-1}\nabla\phi(x^k) \\ x^{k+1} &= x^k + \alpha_k p^k. \end{aligned}$$

The *search direction* p^k is no longer the direction of steepest descent, but it *is* still a descent direction; that is, if $\nabla\phi(x^k) \neq 0$, then for small enough ϵ,

$$\phi(x^k + \epsilon p^k) = \phi(x^k) + \epsilon \nabla\phi(x^k)^T p^k + O(\epsilon^2) < \phi(x^k)$$

since

$$\nabla\phi(x^k)^T p^k = -\nabla\phi(x^k)^T M^{-1} \nabla\phi(x^k) < 0$$

by positive definiteness of M^{-1}.

The convergence for our quadratic model function

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

is determined by the error iteration

$$e^{k+1} = (I - \alpha_k M^{-1} A) e^k.$$

In this case, the optimal choice of M and α would be $M = A$ and $\alpha = 1$; in this case, the iteration converges in a single step! Of course, it is too much to ask for convergence in one step when our objective is more complicated. Still, the quadratic model tells us a lot. If x^* is a local strong minimum and ϕ is sufficiently smooth, we have the local Taylor approximation

$$\phi(x^* + z) = \phi(x^*) + \frac{1}{2} z^T H_\phi(x^*) z + O(\|z\|^3)$$

where $H_\phi(x^*)$ is the Hessian matrix

$$\left[H_\phi(x^*)\right]_{ij} = \frac{\partial^2 \phi(x^*)}{\partial x_i \partial x_j}.$$

For initial points x^0 near enough to x^*, we have

$$e^{k+1} = e^k - H_\phi(x^*)^{-1} \left[\nabla\phi(x^* + e^k) - \nabla\phi(x^*)\right],$$

and substituting

$$\nabla\phi(x^* + e^k) - \nabla\phi(x^*) = H_\phi(x^*) e^k + O(\|e^k\|^2),$$

we have

$$\|e^{k+1}\| = \|e^k - H_\phi(x^*)^{-1} H_\phi(x^*) e^k\| + O(\|e^k\|^2) = O(\|e^k\|^2).$$

This is known as *quadratic* convergence.

Newton's Method and Line Search

One problem with using $H_\phi(x^*)$ as a scaling matrix is that we don't know where x^* is — if we did, we would have no need for an optimization algorithm! However, we can approximate this optimal scaling. One natural choice is to scale with $H_\phi(x^k)$; this gives us *Newton's method*. This method is equivalent to at every step solving the quadratic optimization problem

$$p^k = \text{argmin}_p \, \phi(x^k) + \nabla\phi(x^k)^T p + \frac{1}{2} p^T H_\phi(x^k) p.$$

Newton's method has the disadvantage that we have a new scaling matrix at every step (and so may need to do a new factorization), but it has the advantage that the approximation to $H_\phi(x^*)$ gets better and better as $x^k \to x^*$. Indeed, this method is also locally quadratically convergent.

Newton's method converges very quickly once we are close to a strong minimizer x^* such that the Hessian is positive definite. But far away from x^*, the problem may run into trouble in two ways: we could have an indefinite Hessian matrix, so that the Newton direction is not a descent direction; or a full Newton step might go too far, causing the iteration to diverge. To deal with the issue of indefiniteness, we use an alternate scaling matrix when indefiniteness is detected. For example, we might use $H_\phi(x^k) + \lambda_k I$, which can always be made positive definite with a sufficiently positive choice of λ_k. To deal with the issue of not taking too long a step, we use a *globalization* technique; the two common approaches are *trust regions* or *line search*. For simplicity, we will focus on the latter.

For step size choices for Newton and related iterations, we want to satisfy two conditions. First, the step sizes should not go to zero, or at least they should not go to zero so quickly that the iteration can misconverge. Second, there should be "sufficient decrease" at each step, i.e. we want to satisfy the condition

$$\phi(x^k + \alpha_k p^k) \leq \phi(x^k) + c\alpha_k \nabla\phi(x^k)^T p^k,$$

for some $c < 1$. We typically choose c quite small, so usually this condition is equivalent to just making sure that $\phi(x^{k+1})$ is less than $\phi(x^k)$. The simplest method to choose the step size to satisfy these conditions is a *backtracking line search*: we start with a step size of one, then repeatedly cut the step in half until the sufficient decrease condition holds.

Gauss-Newton

We turn now to another popular iterative solver: the Gauss-Newton method for nonlinear least squares problems. Given $f : \mathbb{R}^n \to \mathbb{R}^m$ for $m > n$, we seek to minimize the objective function

$$\phi(x) = \frac{1}{2}\|f(x)\|^2.$$

The Gauss-Newton approach to this optimization is to approximate f by a first order Taylor expansion in order to obtain a proposed step:

$$p_k = \text{argmin}_p \frac{1}{2}\|f(x_k) + f'(x_k)p\|^2 = -f'(x_k)^\dagger f(x_k).$$

Writing out the pseudo-inverse more explicitly, we have

$$\begin{aligned} p_k &= -[f'(x_k)^T f'(x_k)]^{-1} f'(x_k)^T f(x_k) \\ &= -[f'(x_k)^T f'(x_k)]^{-1} \nabla \phi(x_k). \end{aligned}$$

The matrix $f'(x_k)^T f'(x_k)$ is positive definite if $f'(x_k)$ is full rank; hence, the direction p_k is always a descent direction provided x_k is not a stationary point and $f'(x_k)$ is full rank. However, the Gauss-Newton step is *not* the same as the Newton step, since the Hessian of ϕ is

$$H_\phi(x) = f'(x)^T f'(x) + \sum_{j=1}^m f_j(x) H_{f_j}(x).$$

Thus, the Gauss-Newton iteration can be seen as a modified Newton in which we drop the inconvenient terms associated with second derivatives of the residual functions f_j.

Assuming f' is Lipschitz with constant L, an error analysis about a minimizer x_* yields

$$\|e_{k+1}\| \;\leq\; L\|f'(x_*)^\dagger\|^2 \|f(x_*)\| \|e_k\| + O(\|e_k\|^2).$$

Thus, if the optimal residual norm $\|f(x_*)\|$ is small, then from good initial guesses, Gauss-Newton converges nearly quadratically (though the linear term will eventually dominate). On the other had, if $\|f(x_*)\|$ is larger than $\|f'(x_*)^\dagger\|$, then the iteration may not even be locally convergent unless we apply some type of globalization strategy.

Regularization and Levenberg-Marquardt

While we can certainly apply line search methods to globalize Gauss-Newton iteration, an alternate proposal due to Levenberg and Marquardt is to solve a *regularized* least squares problem to compute the step; that is,

$$p_k = \text{argmin}_p \frac{1}{2}\|f(x_k) + f'(x_k)p\|^2 + \frac{\lambda}{2}\|Dp\|^2.$$

The scaling matrix D may be an identity matrix (per Levenberg), or we may choose $D^2 = \text{diag}(f'(x_k)^T f'(x_k))$ (as suggested by Marquardt).

For $\lambda = 0$, the Levenberg-Marquardt step is the same as a Gauss-Newton step. As λ becomes large, though, we have the (scaled) gradient step

$$p_k = -\frac{1}{\lambda} D^{-2} f(x_k) + O(\lambda^{-2}).$$

Unlike Gauss-Newton with line search, changing the parameter λ affects not only the distance we move, but also the direction.

In order to get both ensure global convergence (under sufficient hypotheses on f, as usual) and to ensure that convergence is not too slow, a variety of methods have been proposed that adjust λ dynamically. To judge whether λ has been chosen too aggressively or conservatively, we monitor the *gain ratio*, or the ratio of actual reduction in the objective to the reduction predicted by the (Gauss-Newton) model:

$$\rho = \frac{\|f(x_k)\|^2 - \|f(x_k + p_k)\|^2}{\|f(x_k)\|^2 - \|f(x_k) + f'(x_k)p_k\|^2}.$$

If the step decreases the function value enough (ρ is sufficiently positive), then we accept the step; otherwise, we reject it. For the next step (or the next attempt), we may increase or decrease the damping parameter λ depending on whether ρ is close to one or far from one.