The subgradient method

The subgradient method is the non-smooth version of gradient descent. The basic algorithm is straightforward, consisting of the iteration

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - t_k \boldsymbol{g}^{(k-1)}, \tag{1}$$

where $\boldsymbol{g}^{(k-1)}$ is *any* subgradient at $\boldsymbol{x}^{(k-1)}$, $\boldsymbol{g}^{(k-1)} \in \partial f(\boldsymbol{x}^{(k-1)})$. Of course, there could be many choice for $\boldsymbol{g}^{(k)}$ at every step, and the progress you make at that iteration could very dramatically with this choice. Making this determination, though, is often very difficult, and whether or not it can even be done it very problem dependent. Thus the analytical results for the subgradient method just assume we have any subgradient at a particular step.

With the right choice of step sizes $\{t_k\}$, some simple analysis (which we will mostly gloss over here) shows that the subgradient method converges. The convergence rate, though, is very slow. This is also evidenced in most practical applications of this method: it can take many iterations on even a medium-sized problem to arrive at a solution that is even close to optimal.

Here is what we know about this algorithm for solving the general unconstrained program

$$\underset{\boldsymbol{x} \in \mathbb{R}^N}{\text{minimize}}\, f(\boldsymbol{x}). \tag{2}$$

We will just state the results here; for detailed derivations, see [Nes04, Chapter 3]. Along with f being convex, we will assume that it has at least one minimizer. The results also assume that f is Lipschitz:

$$|f(\boldsymbol{x}) - f(\boldsymbol{x})| \;\leq\; G\|\boldsymbol{x} - \boldsymbol{y}\|_2.$$

A direct consequence of this is that the norms of the subgradients are bounded:

$$\|\boldsymbol{g}\|_2 \leq G, \quad \text{for all } \boldsymbol{g} \in \partial f(\boldsymbol{x}), \quad \text{for all } \boldsymbol{x} \in \mathbb{R}^N. \tag{3}$$

The results below used pre-determined step sizes. Thus the iteration (1) does not necessarily decrease the functional $f(\boldsymbol{x})$ at every step. We will keep track of the best value we have up to the current iteration with

$$f_{\text{best}}^{(k)} = \min\left\{f(\boldsymbol{x}^{(i)}), \quad 0 \leq i < k\right\}.$$

We will use $f^\star$ to denote the minimal value of (2).

Our analytical results stem from a careful look at what happens during a single iteration. Let $\boldsymbol{x}^\star$ be any solution to (2). Then

$$\begin{aligned}\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^\star\|_2^2 &\leq \|\boldsymbol{x}^{(i-1)} - t_i\boldsymbol{g}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 \\ &= \|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - 2t_i\,\boldsymbol{g}^{(i-1)\mathrm{T}}(\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star) + t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2 \\ &\leq \|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - 2t_i(f(\boldsymbol{x}^{(i-1)}) - f^\star) + t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2,\end{aligned}$$

where the inequality follows from the definition of subgradient:

$$f^\star \geq f(\boldsymbol{x}^{(i-1)}) + \boldsymbol{g}^{(i-1)\mathrm{T}}(\boldsymbol{x}^\star - \boldsymbol{x}^{(i-1)}).$$

We have

$$2t_i\left(f(\boldsymbol{x}^{(i-1)}) - f^\star\right) \leq \|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^\star\|_2^2 + t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2,$$

and so of course

$$2t_i\left(f_{\text{best}}^{(i)} - f^\star\right) \leq \|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^\star\|_2^2 + t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2.$$

Since $f_{\text{best}}^{(i)}$ is monotonically decreasing, at iteration k we have

$$2t_i\left(f_{\text{best}}^{(k)} - f^\star\right) \le \|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^\star\|_2^2 + t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2,$$

for all $i \le k$. To understand what has happened after k iterations, we sum both sides of the expression above from $i = 1$ to $i = k$. Notice that the two error terms on the right hand side give us the "telescoping" sum:

$$\begin{aligned}\sum_{i=1}^k\left(\|\boldsymbol{x}^{(i-1)} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^\star\|_2^2\right) &= \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^\star\|_2^2 \\ &\le \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2\end{aligned}$$

and so

$$f_{\text{best}}^{(k)} - f^\star \le \frac{\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2 + \sum_{i=1}^k t_i^2\|\boldsymbol{g}^{(i-1)}\|_2^2}{2\sum_{i=1}^k t_i} \tag{4}$$

We can now specialize this result to general step-size strategies.

Fixed step size. Suppose that $t_k = t > 0$ for all k. Then (4) becomes

$$f_{\text{best}}^{(k)} - f^\star \le \frac{\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2}{2kt} + \frac{G^2t}{2},$$

where we have also used the Lipschitz property (3). Note that in this case, no matter how small we choose t, **the subgradient algorithm is not guaranteed to converge**. This is, in fact, standard in practice as well. The problem is that, unlike gradients for smooth functions, the subgradients do not have to vanish as we approach the solution. Even at the solution, there can be subgradients that are large.

Picture:

Fixed step length. A similar result holds if we always move the same amount, taking

$$t_k = s/\|\boldsymbol{g}^{(k-1)}\|_2.$$

This means that

$$\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\|_2 = s.$$

Of course, with this strategy it is self-evident that it will never converge, since we move some fixed amount at every step. We can bound the suboptimality at step k as

$$f_{\text{best}}^{(k)} - f^\star \leq \frac{G\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2}{2ks} + \frac{Gs}{2},$$

which is not necessarily worse than the fixed step size result. In fact, notice that even though you are moving some fixed amount, you will never move to far from an optimal point.

Decreasing step size. The results above suggest that we might want to decrease the step size as k increases, so we can get rid of this constant offset term. To make the terms in (4) work out, we let $t_k \to 0$, but not too fast. Specifically, we choose a sequence $\{t_k\}$

such that

$$t_k \to 0 \text{ as } k \to \infty, \quad \text{and} \quad \sum_{k=1}^{\infty} t_k = \infty. \tag{5}$$

It is an exercise (but a nontrivial one) to show that under these conditions

$$\frac{\sum_{i=1}^k t_i^2}{\sum_{i=1}^k t_i} \to 0 \quad \text{as} \quad k \to \infty.$$

Thus we do get guaranteed convergence of the subgradient method when the stepsizes obey (5).

To get an idea of the tradeoffs involved here, suppose that $t_k = \beta/k$. Then for large k, we have the approximations

$$\sum_{i=1}^k t_i \sim \beta \log k, \quad \text{and} \quad \sum_{i=1}^k t_i^2 \sim \text{Const} = \beta^2\pi^2/6$$

that are good as upper and lower bounds to within constants. In this case, the convergence result (4) becomes

$$f_{\text{best}}^{(k)} - f^\star \lesssim \frac{\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2}{\beta \log k} + \text{Const} \cdot \frac{\beta G^2}{\log k}.$$

So the convergence is extraordinarily slow.

You can get much better rates than this, but still not good, by decreasing the stepsize more slowly. Consider now $t_k = \alpha/\sqrt{k}$. Then for large k

$$\sum_{i=1}^k t_i \sim (\alpha + 1)\sqrt{k}, \quad \text{and} \quad \sum_{i=1}^k t_i^2 \sim \alpha^2 \log k,$$

and so

$$f_{\text{best}}^{(k)} - f^\star \lesssim \frac{\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^\star\|_2^2}{(\alpha+1)\sqrt{k}} + \text{Const} \cdot \frac{\alpha G^2 \log k}{\sqrt{k}}.$$

This is something like $O(1/\sqrt{k})$ convergence. This means that if we want to guarantee $f_{\text{best}}^{(k)} - f^\star \leq \epsilon$, we need $k = O(1/\epsilon^2)$ iterations.

In [Nes04, Chapter 3], it is shown that there is no better rate of convergence than $O(1/\sqrt{k})$ that holds uniformly across all problems.

Example. The following example comes from [Van16]. We consider the standard ℓ_1 approximation problem

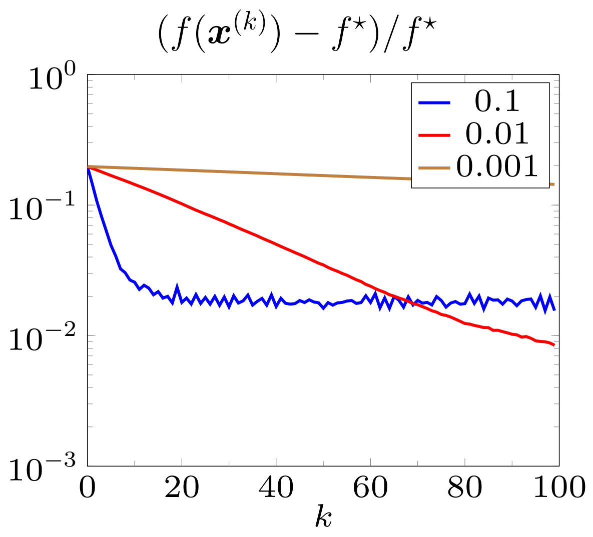
$$\underset{\boldsymbol{x}\in\mathbb{R}^N}{\text{minimize}} \ \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_1$$

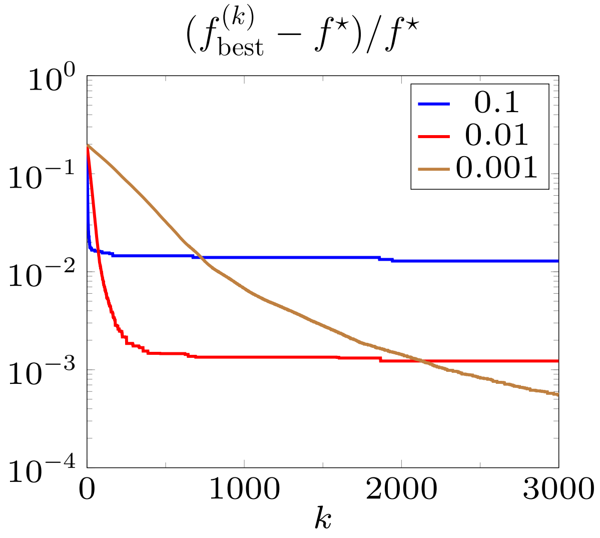
for a randomly generated example with $\boldsymbol{A} \in \mathbb{R}^{500\times 100}$ and $\boldsymbol{b} \in \mathbb{R}^{1000}$. In this case, a subgradient can be computed quickly using

$$\boldsymbol{g}^{(k)} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{z}, \quad \text{where} \quad \boldsymbol{z} = \text{sign}(\boldsymbol{A}\boldsymbol{x}^{(k)} - \boldsymbol{b}).$$

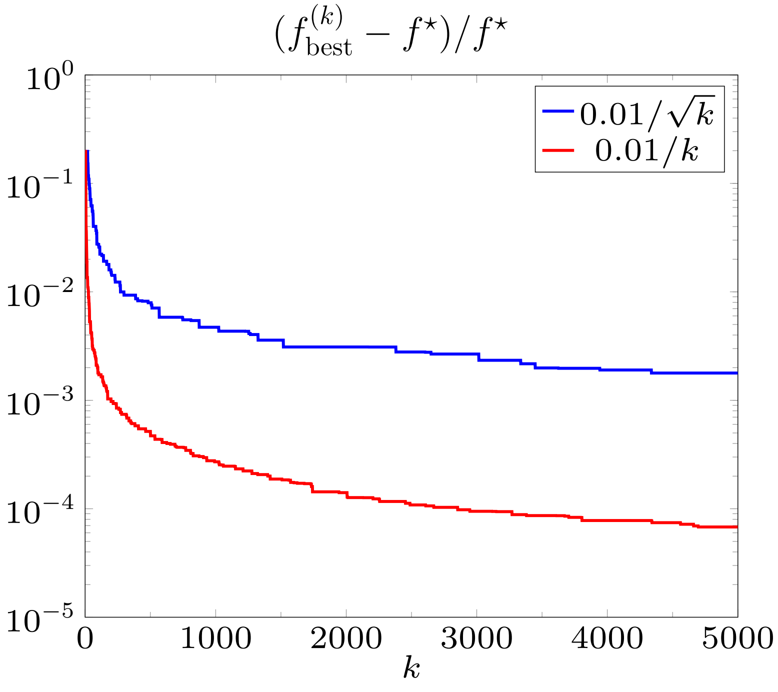
For indexes where $\boldsymbol{A}\boldsymbol{x}^{(k)} - \boldsymbol{b}$ is zero, we can use any number with magnitude less than 1 for the corresponding entry of $\boldsymbol{z}$.

For three different sizes of fixed step length, $s = 0.1, 0.01, 0.001$, we make quick progress at the beginning, but then saturate, just as the theory predicts:





Here is a run using two different decreasing step size strategies: $t_k = .01/\sqrt{k}$ and $t_k = .01/k$.



As you can see, even though the theoretical worst case bound makes a stepsize of $\sim 1/\sqrt{k}$ look better, in this particular case, a stepsize $\sim 1/k$ actually performs better.

Qualitatively, the takeaways for the subgradient method are:

1. It is a natural extension of the gradient descent formulation
2. In general, it does not converge for fixed stepsizes.
3. If the stepsizes decrease, you can guarantee convergence.
4. Theoretical convergence rates are slow.
5. Convergence rates in practice are also very slow, but depend a lot on the particular example.

References

[Nes04] Y. Nesterov. *Introductory Lectures on Convex Optimization.* Springer Science+Business Media, 2004.

[Van16] L. Vandenberghe. Lecture notes for EE236C, UCLA, Spring 2016.