Lecture 4

4.1 Adaptive Search for Gradient Method

4.1.1 Gradient Method: Summary

In the previous lecture we considered the following problem class of *smooth unconstrained optimization*:

$$\min_{x \in \mathbb{R}^n} f(x),$$

where f has a Lipschitz continuous gradient, for some constant $L > 0$:

$$\|\nabla f(y) - \nabla f(x)\|_* \quad \leq \quad L\|y - x\|, \qquad x, y \in \mathbb{R}^n, \tag{4.1}$$

and we allow to use an arbitrary norm $\|\cdot\|$ for our primal space $\mathbb{R}^n$. We also assume that f is bounded from below: $f(x) \geq f^\star$ for all $x \in \mathbb{R}^n$.

Our goal is to find an approximate *stationary point* $\bar{x} \in \mathbb{R}^n$ such that, for a given $\varepsilon > 0$:

$$\|\nabla f(\bar{x})\|_* \quad \leq \quad \varepsilon. \tag{4.2}$$

We studied the gradient method, which starts from some initialization $x_0 \in \mathbb{R}^n$ and then iterates, for $k \geq 0$:

$$x_{k+1} \quad := \quad x_k^+(M_k),$$

where $\{M_k\}_{k \geq 0}$ is a sequence of regularization parameters, and $x_k^+(M)$ denotes a gradient step with respect to the given norm $\|\cdot\|$:

$$x_k^+(M) \quad = \quad \arg\min_{y \in \mathbb{R}^n} \Big\{ f(x_k) + \langle \nabla f(x_k), y - x_k \rangle + \tfrac{M}{2}\|y - x_k\|^2 \Big\}. \tag{4.3}$$

In case the norm is Euclidean, $\|\cdot\| = \|\cdot\|_2$, this simplifies to the classical update:

$$x_k^+(M) \quad = \quad x_k - \tfrac{1}{M}\nabla f(x_k),$$

and parameter $M > 0$ controls the step-size. For a non-Euclidean norm, the solution to (4.3) may be not unique, but always exists. We ensures the following progress in function value, for $M \geq L$:

$$f(x_k) - f(x_k^+(M)) \quad \geq \quad \tfrac{1}{2M}\|\nabla f(x_k)\|_*^2. \tag{4.4}$$

This is the key inequality to ensure convergence. It holds if we choose the *constant step-size*: $M_k \equiv L$, for every $k \geq 0$. But can we do better?

It appears that in practice it is much more efficient to use an *adaptive strategy* for choosing the regularization parameters, which does not require to know the Lipschitz constant, and which can adaptively adjust to a local smoothness of the objective. We study such a strategy in the next section, which is also often called a *line search*.

4.1.2 Adaptive Search

To choose $M_k \geq 0$ adaptively, we can use the following algorithm, which is very useful in practice.

We start with an initial estimate of the regularization constant $M_0 > 0$, which can be arbitrary. Then, at each iteration $k \geq 2$, we double our current estimate M_k until (4.4) is satisfied. After each iteration, we also divide it by 2 to ensure both growth and decrease of the sequence $\{M_k\}_{k \geq 0}$.

Algorithm 4.1: *Gradient Method with Adaptive Search.*

Initialization: $x_0 \in \mathbb{R}^n$, $\varepsilon > 0$, $M_0 > 0$

For $k \geq 0$ **iterate:**

1. If $\|\nabla f(x_k)\|_* \leq \varepsilon$ then
   return x_k
2. **For** $t \geq 0$ **iterate:**
   - Set $M_k^+ := 2^t \cdot M_k$
   - Try gradient step: $x_k^+ := x_k^+(M_k^+)$
   - If $f(x_k) - f(x_k^+) \geq \frac{1}{2M_k^+}\|\nabla f(x_k)\|_*^2$ then **break** and **go to** step 3
3. Set $x_{k+1} = x_k^+$ and $M_{k+1} = \frac{1}{2}M_k^+$

This algorithm is well-define since the break condition will be satisifed at least when $M_k^+ \geq L$. In addition to using gradients, at each iteration $k \geq 0$ we might need to compute several function values (at least one). However, it is easy to show that the total number of oracle calls is well-bounded.

Proposition 4.1.1. *For each $k \geq 0$, we have*

$$M_k \;\leq\; M_\star \;:=\; \max\Big\{M_0,\; L\Big\}. \tag{4.5}$$

Moreover, during the first $k \geq 1$ iterations of the method, the total number of first-order oracle calls N_k of the type $\mathcal{O}(x) = \{f(x), \nabla f(x)\}$ is bounded as

$$N_k \;\leq\; 2k + \max\{0,\; 1 + \log_2 \tfrac{L}{M_0}\} \tag{4.6}$$

Proof. We prove (4.5) by induction. It obviously holds for $k = 0$. Assume that it holds for some $k \geq 0$ and consider one iteration of the algorithm. Let us denote by $t_k \geq 0$ the value of parameter t at step 2 that triggers the break. Thus,

$$M_{k+1} \;=\; 2^{t_k - 1} M_k. \tag{4.7}$$

If $t_k = 0$, then $M_{k+1} = \frac{1}{2}M_k \overset{(4.5)}{\leq} \frac{1}{2}M_\star < M_\star$. Otherwise, if $t_k > 0$, it means that the break condition was not satisfied for $2^{t_k - 1}M_k \equiv M_{k+1}$, so $M_{k+1} < L \leq M_\star$. Thus, we have established (4.5) for all $k \geq 0$.

To show (4.6), we notice that

$$N_k \;=\; \sum_{i=0}^{k-1}(1 + t_i) \;\overset{(4.7)}{=}\; \sum_{i=0}^{k-1}(2 + \log_2 \tfrac{M_{i+1}}{M_i}) \;=\; 2k + \log_2 \tfrac{M_k}{M_0}$$

$$\overset{(4.5)}{\leq} \; 2k + \max\{0,\, 1 + \log_2 \tfrac{L}{M_0}\},$$

which completes the proof. □

Due to bound (4.5), and using Theorem 3.2.4 from the previous lecture, we ensure the same iteration complexity for Algorithm 4.1 as for the method with exact Lipschitz constant.

Corollary 4.1.2. *To find a point $\bar{x} \in \mathbb{R}^n$ such that $\|\nabla f(\bar{x})\|_* \leq \varepsilon$, Algorithm 4.1 needs to perform*

$$K \quad = \quad \left\lceil \frac{4 \max\{M_0, L\}(f(x_0) - f^\star)}{\varepsilon^2} \right\rceil$$

iterations, and the total number of oracle calls is bounded as in (4.6).

4.2 Stochastic Gradient Mehod

In applications, sometimes we do not have an access to the exact gradients $\nabla f(x)$, or it can be too expensive to compute them.

Example 4.2.1 (Expensive)**.** In machine learning, we are often interested in solving problems of the following form, called the *finite-sum* structure:

$$\min_{x \in \mathbb{R}^n} \Big[f(x) \quad := \quad \tfrac{1}{N} \sum_{i=1}^{N} f_i(x) \Big], \tag{4.8}$$

where each $f_i : \mathbb{R}^n \to \mathbb{R}$ is a smooth function, $1 \leq i \leq N$, and N is typically a large number (i.e., number of entries in a dataset). Then, computing the exact gradient of f would involve computing the gradient over all data samples:

$$\nabla f(x) \quad = \quad \tfrac{1}{N} \sum_{i=1}^{N} \nabla f_i(x). \tag{4.9}$$

For large-scale datasets, an $O(N)$ per-iteration cost can be prohibitive. However, we can afford to compute a small subset of gradients from the finite sum (4.9).

Example 4.2.2 (Impossible)**.** Another class of problems is *purely stochastic optimization*, where the objective function is given in the integral form:

$$\min_{x \in \mathbb{R}^n} \Big[f(x) \quad := \quad \mathbb{E}_\xi \big[F(x, \xi) \big] \Big], \tag{4.10}$$

where ξ is some random variable, and each $F(\cdot, \xi)$ is a smooth function. Of course, the finite-sum problem (4.8) can be casted as (4.10) by setting ξ to be a uniform distribution over $\{1, \ldots, N\}$. In general, it may be impossible for us to compute the full gradient $\nabla f(x)$, while we have access to $\nabla_x F(x, \xi)$ for any random sample ξ.

Example 4.2.3 (Unfavorable)**.** In some data analysis and machine learning problems, we may be restricted from using exact gradients $\nabla f(x)$ to protect user privacy. In such cases, we can *intentionally introduce noise* into the oracle information to prevent identifiability.

4.2.1 Stochastic Oracle

To cover such situations, we refine the formulation of our problem. We are still interested to minimize a smooth objective f: $\min_{x \in \mathbb{R}^n} f(x)$ that satisfies our previous conditions. However, instead of an access to exact gradients, we assume that, for any point $x \in \mathbb{R}^n$, we can sample a random variable ξ and get an access to *stochastic gradient* vector:

$$g(x; \xi) \quad \in \quad \mathbb{R}^n.$$

For example, solving finite-sum problems (4.8), ξ can be an index i of a function from the sum that we sample, and then $g(x; i) := \nabla f_i(x)$.

For simplicity, let us fix the standard Euclidean norm $\| \cdot \| := \| \cdot \|_2$ for the rest of this lecture. We assume that $g(x; \xi)$ satisfy the following properties.

1. $g(x; \xi)$ is an **unbiased estimator** of the true gradient $\nabla f(x)$:

$$\mathbb{E}_\xi \big[g(x; \xi) \big] \quad = \quad \nabla f(x), \qquad x \in \mathbb{R}^n.$$

2. $g(x; \xi)$ has a bounded **variance**:

$$\mathbb{E}_\xi \Big[\| g(x; \xi) - \nabla f(x) \|^2 \Big] \quad \leq \quad \sigma^2, \qquad x \in \mathbb{R}^n,$$

where $\sigma > 0$ is now a parameter of our problem class. Note that

$$\| g(x; \xi) - \nabla f(x) \|^2 \quad = \quad \| g(x; \xi) \|^2 - 2 \langle g(x; \xi), \nabla f(x) \rangle + \| \nabla f(x) \|^2.$$

Thus, taking the expectation and rearranging the gradient norm, we obtain the bound for the second moment:

$$\mathbb{E}_\xi \Big[\| g(x; \xi) \|^2 \Big] \quad \leq \quad \sigma^2 + \| \nabla f(x) \|^2. \tag{4.11}$$

4.2.2 Algorithm

We consider the following algorithm, which is gradient descent with substituted stochastic gradient instead of the full one. It is also often called *stochastic gradient descent* (SGD).

Algorithm 4.2: *Stochastic Gradient Method*

Initialization: $x_0 \in \mathbb{R}^n$, regularization parameter $M > 0$, number of iterations $K \geq 1$.

For $k = 0 \dots K - 1$ **iterate:**

1. Sample ξ_k.
2. Compute stochastic gradient $g_k := g(x_k; \xi_k)$.
3. Update $x_{k+1} := x_k - \frac{1}{M} g_k$.

Sample $j \in \{0, \dots, K - 1\}$ uniformly at random and **return** $\bar{x}_K$.

We consider a constant parameter $M > 0$ and the key questions is *how we choose it?*

Note that we do not have a clear stopping condition because we lack access to $\| \nabla f(x) \|$ or even to the function value $f(x)$. Instead, we fix the number of iterations $K \geq 0$ for the method to perform and in the end return one of the points from the past, sampled uniformly at random.

4.2.3 Convergence Analysis

First, we investigate the progress of one random step.

Proposition 4.2.4. *Let $M > 0$. Then, for $x_{k+1} = x_k - \frac{1}{M} g_k$ with $g_k = g(x_k; \xi_k)$ we have*

$$\mathbb{E}_{\xi_k}\Big[f(x_k) - f(x_{k+1})\Big] \;\geq\; \tfrac{1}{M}\|\nabla f(x_k)\|^2 \cdot \Big(1 - \tfrac{L}{2M}\Big) - \tfrac{L}{2M^2}\sigma^2. \tag{4.12}$$

Proof. Using Lipschitzness of the gradient, we have

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k\rangle + \tfrac{L}{2}\|x_{k+1} - x_k\|^2 \\ &= f(x_k) - \tfrac{1}{M}\langle \nabla f(x_k), g_k\rangle + \tfrac{L}{2M^2}\|g_k\|^2. \end{aligned}$$

Rearranging the terms and taking the expectation, we obtain

$$\begin{aligned} \mathbb{E}_{\xi_k}\Big[f(x_k) - f(x_{k+1})\Big] &\geq \mathbb{E}_{\xi_k}\Big[\tfrac{1}{M}\langle \nabla f(x_k), g_k\rangle - \tfrac{L}{2M^2}\|g_k\|^2\Big] \\ &= \tfrac{1}{M}\|\nabla f(x_k)\|^2 - \tfrac{L}{2M^2}\mathbb{E}_{\xi_k}\Big[\|g_k\|^2\Big] \\ &\geq \tfrac{1}{M}\|\nabla f(x_k)\|^2 - \tfrac{L}{2M^2}\|\nabla f(x_k)\|^2 - \tfrac{L}{2M^2}\sigma^2 \\ &= \tfrac{1}{M}\|\nabla f(x_k)\|^2 \cdot \Big(1 - \tfrac{L}{2M}\Big) - \tfrac{L}{2M^2}\sigma^2, \end{aligned}$$

which is the required bound. □

Corollary 4.2.5. *Let $M \geq L$. We obtain*

$$\mathbb{E}_{\xi_k}\Big[f(x_k) - f(x_{k+1})\Big] \;\geq\; \tfrac{1}{2M}\|\nabla f(x_k)\|^2 - \tfrac{L}{2M^2}\sigma^2. \tag{4.13}$$

In case of no randomness ($\sigma = 0$), we recover the progress bound from the previous lecture. However, when $\sigma > 0$, we cannot longer guarantee a positive progress of each iteration.

Now, we want to telescope (4.12) across K iterations to obtain the convergence rate.

Theorem 4.2.6. *Let $M \geq L$. Consider $K \geq 1$ iterations of Algorithm 4.2. Then,*

$$\tfrac{2M(f(x_0) - f^\star)}{K} + \tfrac{L}{M}\sigma^2 \;\geq\; \mathbb{E}\Big[\|\nabla f(\bar{x}_K)\|^2\Big]. \tag{4.14}$$

Proof. We denote by $\mathbb{E}[\cdot]$ the expectation w.r.t all $\{\xi_0, \ldots, \xi_{K-1}\}$ and the random choice of j for the output $\bar{x}_K$ and $\mathbb{E}_\xi[\cdot]$ with respect to all $\{\xi_0, \ldots, \xi_{K-1}\}$ only. Taking it for the both left and right hand sides of (4.13), we get

$$\mathbb{E}_\xi\Big[f(x_k) - f(x_{k+1})\Big] \;\geq\; \tfrac{1}{2M}\mathbb{E}_\xi\Big[\|\nabla f(x_k)\|^2\Big] - \tfrac{L}{2M^2}\sigma^2.$$

Telecsoping this bound for the first K iterations, we get

$$\begin{aligned} f(x_0) - f^\star &\geq f(x_0) - \mathbb{E}_\xi[f(x_k)] \;\geq\; \tfrac{1}{2M}\sum_{i=0}^{K-1}\mathbb{E}_\xi[\,\|\nabla f(x_i)\|^2\,] - \tfrac{L}{M^2}\sigma^2 K \\ &= \tfrac{1}{2M}\mathbb{E}\Big[\|\nabla f(\bar{x}_K)\|^2\Big] - \tfrac{L}{M^2}\sigma^2 K. \end{aligned}$$

Rearranging the terms we obtain the required bound (4.14). □

4.2.4 Stepsize Tuning.

Now, we need to choose $M > 0$ and $K \geq 1$ to ensure the following guarantee:

$$\mathbb{E}\Big[\|\nabla f(\bar{x}_K)\|^2\Big] \quad \leq \quad \varepsilon^2.$$

For that, using (4.14) it is enough to ensure that

1. $\frac{2M(f(x_0)-f^\star)}{K} \leq \frac{\varepsilon^2}{2}$ and
2. $\frac{L}{M}\sigma^2 \leq \frac{\varepsilon^2}{2}$.

The last inequality, together with our condition $M \geq L$ suggest the following choice of the regularization parameter:

$$\boxed{M \quad := \quad L \cdot \max\Big\{1, \ \tfrac{2\sigma^2}{\varepsilon^2}\Big\}.} \tag{4.15}$$

Then, to ensure the first inequality, it is enough to choose the number of iterations sufficiently large:

$$\boxed{K \quad := \quad 1 + \Big\lfloor \tfrac{4MF_0}{\varepsilon^2} \Big\rfloor.} \tag{4.16}$$

Combining these two choices together, and using Jensen's inequality, that is $(\mathbb{E}[\tau])^2 \leq \mathbb{E}[\tau^2]$, we obtain the following complexity bound.

Corollary 4.2.7. *To find a random point $\bar{x} \in \mathbb{R}^n$ such that $\mathbb{E}\Big[\|\nabla f(\bar{x})\|\Big] \leq \varepsilon$, it is enough to perform*

$$K \quad = \quad O\Big(L(f(x_0) - f^\star) \cdot \Big[\tfrac{1}{\varepsilon^2} + \tfrac{\sigma^2}{\varepsilon^4}\Big]\Big) \tag{4.17}$$

stochastic first-order oracle calls.

The upper bound (4.17) is matched, up to a numerical constant, by a lower complexity bound for our problem class in both for stochastic and deterministic cases. Consequently, the gradient method is *optimal* for finding a stationary point for functions with Lipschitz continuous gradients.

Literature

For additional reading on stochastic gradient methods, see [GL13] and [Lan20]. For the lower complexity bound on finding stationary points of smooth functions, see [CDHS20], which establishes that the bound (4.17) is *optimal* for both deterministic and stochastic optimization within our problem class.

[CDHS20] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points I. *Mathematical Programming*, 184(1):71–120, 2020.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

[Lan20] Guanghui Lan. *First-order and stochastic optimization methods for machine learning*, volume 1. Springer, 2020.