CSE 599: Interplay between Convex Optimization and Geometry **Winter 2018**

Lecture 6: Geometric Descent

Lecturer: Yin Tat Lee

Disclaimer: *Please tell me any mistake you noticed.*

This lecture is controversial and should be taken with a grain of salt. This lecture is mainly based on [37, 39].

6.1 Geometric Descent

6.1.1 Anecdote

In the past few lectures, we discussed various cutting plane methods, their amazing theoretical guarantees for general convex optimizations and some geometry stories behind them. However, these methods currently are not as popular as first-order methods or interior point methods due to their practical performance. Especially, ellipsoid method is infamous for its practical performance if implemented as stated.

In the past few years, I have been arguing with various researchers that ellipsoid method would work with some small changes. The geometric descent paper [37] is a record for such an argument. For me, the purpose of the geometric descent is not to give another interpretation of accelerated gradient descent, but to revive the interest of cutting plane methods.

Now, I still naively believe there is a universal algorithm for solving convex problems and hence there should be connections between all frameworks for convex optimization. Clearly, our current understanding is lacking and more researches are needed.

Problem 6.1.1. Can we have a natural algorithm for convex problems that gives the best guarantee of cutting plane methods, interior point methods and first-order methods?

I believe the answer for this problem will be one of the greatest result for optimization for a decade and that the answer is probably simple in retrospect. Unfortunately, I am both blind and distracted right now.

6.1.2 Problems of Ellipsoid Method

In practice, many convex problems are smooth and strongly convex.

Definition 6.1.2. A function f is α-strongly convex if $\nabla^2 f(x) \succeq \alpha$ for all x, equivalently,

$$f(y) \geq f(x) + \nabla f(x)^\top (y-x) + \frac{\alpha}{2}\|y-x\|_2^2 \text{ for all } x, y. \tag{6.1}$$

A function is β-smooth if $\nabla^2 f(x) \preceq \beta$ for all x, equivalently,

$$f(y) \leq f(x) + \nabla f(x)^\top (y-x) + \frac{\beta}{2}\|y-x\|_2^2 \text{ for all } x, y. \tag{6.2}$$

The key observation is that ellipsoid method is not greedy enough when the function is strongly convex and smooth. In particular, we know x is contained in a much smaller region than the halfspace. Let x^* be the

minimizer of f. Putting $y = x^*$ in (6.1) shows that

$$f(x^*) \geq f(x) + \nabla f(x)^\top (x^* - x) + \frac{\alpha}{2} \|x^* - x\|_2^2 .$$

Completing the squares, we have that

$$\left\|x^{++} - x^*\right\|_2^2 \leq \frac{\|\nabla f(x)\|_2^2}{\alpha^2} - \frac{2}{\alpha}(f(x) - f(x^*)).$$

Since $f(x^*) \leq f(x)$, this shows that x^* lies in a sphere centered at x^{++} with radius at most $\frac{\|\nabla f(x)\|_2^2}{\alpha^2}$.

To make ellipsoid method greedy, we should

1. Use cutting sphere instead of cutting plane.
2. Maintain a good estimate of $f(x^*)$.

Another issue of ellipsoid method is the n^2 cost per each iteration. To make each iteration cheaper, it is natural to use sphere instead of ellipsoid.

6.1.3 Epigraph Cutting Plane Method

In previous lectures, we discussed cutting plane methods that maintain a region that contains the minimizer x^* of f. The proof of such cutting plane method relies on the following inequality

$$f(x) > f(x^*) \geq f(x) + \langle \nabla f(x), x^* - x \rangle . \tag{6.3}$$

Notice that the left-hand side is a strict inequality (unless we already solved the problem). Therefore, as the algorithm runs, we will find a new point x_{new} such that $f(x_{\text{new}}) < f(x)$. To be more greedy, we should move the halfspace we got before. However, this makes the program complicated.

An easier way to avoid this nightmare is instead maintaining a region that contains $(x^*, f(x^*))$. Now, we can think the equation $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$ as a cutting plane of the epigraph of f. Hence, we do not need to update cutting plane anymore.

6.1.4 How to Make Ellipsoid Method Great Again?

The algorithm is an epigraph cutting plane method. Instead of using cutting sphere, we simply use (6.1) as the cutting plane. More precisely, it is a cutting spherical parabola. For notation convenience, we define

$$\texttt{line\_search}(x, y) = \text{argmin}_{z = x + t(x-y) \text{ with } t \in \mathbb{R}} f(z).$$

The algorithm is the following:

- Input: $x_0 \in \mathbb{R}^n$ and strong convexity α.
- Set $Q_0 = \left\{(y, t) \in \mathbb{R}^{n+1} : \ t \geq f(x) + \nabla f(x)^\top (y - x) + \frac{\alpha}{2} \|y - x\|^2 \right\}$.
- Set $x_0^+ = \texttt{line\_search}(x_0, \nabla f(x_0))$.
- For $k = 1, 2, \cdots, T$
  - Let $x_k = \texttt{line\_search}(c_{k-1}, x_{k-1}^+)$ where c_{k-1} be the minimizer of Q_{k-1}.
  - Let $x_k^+ = \texttt{line\_search}(x_k, \nabla f(x_0))$.
  - Set $Q = \left\{(y, t) \in \mathbb{R}^{n+1} : \ t \geq f(x_k) + \nabla f(x_k)^\top (y - x_k) + \frac{\alpha}{2} \|y - x_k\|^2 \right\}$.
  - Let $Q_k = \texttt{min\_para}(Q \cap Q_{k-1})$ where $\texttt{min\_para}(K)$ is the smallest spherical parabola covering K.

The difference between this algorithm and ellipsoid is the following:

1. It is an epigraph cutting plane method.
2. It uses the line search between c_k and x_k to make the point we evaluate ∇f has small function value.
3. It does a line search from x_k on the direction $\nabla f(x_k)$.
4. It uses smallest spherical parabola instead of smallest ellipsoid.

Except for the last points, all modifications are to make the algorithm more greedy. The last point is to make the cost per each iteration is $O(n)$. To see this, we note that there is an explicit formula for smallest spherical parabola.

Lemma 6.1.3 ([39]). *Consider* $Q_1 = \left\{(y,t) : \ t \geq v_A + \frac{\alpha}{2}\|y - x_A\|_2^2\right\}$ *and* $Q_2 = \left\{(y,t) : \ t \geq v_B + \frac{\alpha}{2}\|y - x_B\|_2^2\right\}$. *Let* $Q = \texttt{min\_para}(Q_1 \cap Q_2)$. *Then,* $Q = \left\{(y,t) : \ t \geq \overline{v} + \frac{\alpha}{2}\|y - \overline{x}\|^2\right\}$ *Furthermore, we have that*

- *If* $|v_A - v_B| \leq \frac{\alpha}{2}\|x_A - x_B\|_2^2$, *then*

$$\overline{v} = \frac{\alpha}{8}\|x_A - x_B\|^2 + \frac{1}{2}(v_A + v_B) + \frac{1}{2\alpha}\left(\frac{v_A - v_B}{\|x_A - x_B\|_2}\right)^2,$$
$$\overline{x} = (\frac{1}{2} + \frac{v_A - v_B}{\alpha\|x_A - x_B\|_2^2})x_A + (\frac{1}{2} - \frac{v_A - v_B}{\alpha\|x_A - x_B\|_2^2})x_A.$$

- *If* $v_A - v_B \geq \frac{\alpha}{2}\|x_A - x_B\|_2^2$, *then* $\overline{v} = v_A$ *and* $\overline{x} = x_A$.
- *If* $v_B - v_A \geq \frac{\alpha}{2}\|x_A - x_B\|_2^2$, *then* $\overline{v} = v_B$ *and* $\overline{x} = x_B$.

However, to highlight the key point of the analysis, we would use the following lemma instead.

Lemma 6.1.4 ([37]). *For any vector* $a \in \mathbb{R}^n$ *with* $\|a\|_2 \geq g$ *and any* $0 < \varepsilon \leq 1$, *the set*

$$\{x : \ \|x\|_2^2 \leq R^2 - \varepsilon g^2\} \cap \{x : \ \|x - a\|_2^2 \leq g^2 - \varepsilon g^2\}$$

contains in a sphere of radius square $(1 - \sqrt{\varepsilon})R^2$.

The analysis of this algorithm is basically the same as ellipsoid method; we simply analyze how much the volume decrease every iteration.

Theorem 6.1.5. *Let* V_k *be the volume of* $Q_k \cap \{(y,t) : t \leq f(x_k^+)\}$. *Then, we have that*

$$\text{vol}V_{k+1} \leq \left(1 - \sqrt{\frac{\alpha}{\beta}}\right)^{\frac{n+1}{2}} \text{vol}V_k.$$

Proof. Without loss of generality, $\alpha = 1$.

Lemma 6.1.3 shows that Q_k is always given in the form $\left\{(y,t) : \ t \geq v_k + \frac{1}{2}\|y - c_k\|^2\right\}$ for some $v_k \in \mathbb{R}$ and $c_k \in \mathbb{R}^n$. Let $\gamma_k = 2(f(x_k^+) - v_k)$. Note that γ_k is the radius square of the base of the truncated parabola $Q_k \cap \{(y,t) : t \leq f(x_k^+)\}$. Therefore, $\text{vol}V_k \propto \gamma_k^{\frac{n+1}{2}}$. Therefore, it suffices to prove

$$\gamma_k \leq \left(1 - \sqrt{\frac{1}{\beta}}\right)\gamma_{k-1}.$$

For any spherical parabola P, we define $[P]_s$ be the section of P at $\{t = s\}$. It is easy to see (for example by Lemma 6.1.3) that it suffices to prove that

$$[Q_{k-1}]_{f(x_k^+)} \cap [Q]_{f(x_k^+)} \subset \{y : \ \|x - c\|_2^2 \le (1 - \sqrt{\frac{1}{\beta}})\gamma_{k-1}\}$$

for some vector c.

For the sphere $[Q_{k-1}]_{f(x_k^+)}$, we note that

$$[Q_{k-1}]_{f(x_k^+)} = \left\{y : \ f(x_k^+) \ge v_{k-1} + \frac{1}{2}\|y - c_{k-1}\|_2^2\right\}. \tag{6.4}$$

The smoothness of f and the line search $x_k = \texttt{line\_search}(c_{k-1}, x_{k-1}^+)$ shows that

$$f(x_k^+) \le f(x_k) - \frac{1}{2\beta}\|\nabla f(x_k)\|_2^2 \le f(x_{k-1}^+) - \frac{1}{2\beta}\|\nabla f(x_k)\|_2^2. \tag{6.5}$$

Without loss of generality, we can assume $c_{k-1} = 0$. Combining this, (6.4) and (6.5), we have that

$$[Q_{k-1}]_{f(x_k^+)} \subset \left\{y : \ \gamma_{k-1} - \frac{1}{\beta}\|\nabla f(x_k)\|_2^2 \ge \|y\|^2\right\} \tag{6.6}$$

For the sphere $[Q]_{f(x_k^+)}$, we note that

$$\begin{aligned}[Q]_{f(x_k^+)} &= \left\{y : \ f(x_k^+) \ge f(x_k) + \nabla f(x_k)^\top (y - x_k) + \frac{1}{2}\|y - x_k\|_2^2\right\} \\ &= \left\{y : \ f(x_k^+) \ge f(x_k) - \frac{1}{2}\|\nabla f(x_k)\|_2^2 + \frac{1}{2}\|y - a\|_2^2\right\}\end{aligned}$$

where $a = x_k - \nabla f(x_k)$. Using (6.5), we have that

$$[Q]_{f(x_k^+)} \subset \left\{y : \ (1 - \frac{1}{\beta})\|\nabla f(x_k)\|_2^2 \ge \|y - a\|_2^2\right\}. \tag{6.7}$$

To apply Lemma 6.1.4, we note that $\nabla f(x_k)$ is perpendicular to $c_{k-1} = 0$ and x_{k-1}^+ and hence $\left\|x_k^{++}\right\|_2^2 \ge \left\|x_k - x_k^{++}\right\|_2^2 = \|\nabla f(x_k)\|_2^2$. Therefore, we have

$$[Q_{k-1}]_{f(x_k^+)} \cap [Q]_{f(x_k^+)} \subset \{y : \ \|x - c\|_2^2 \le (1 - \sqrt{\frac{1}{\beta}})\gamma_{k-1}\}.$$

□

6.1.5 Discussions

The use of line search: Often, convex functions are given in the form $\sum_i \phi(a_i^T x)$, this includes many problems such as least squares, linear programs, logistic regression and many problems in empirical risk minimization. When such problem is slightly dense, computing $a_i^T x$ is the computational bottleneck. If we reuse the calculations carefully, each iteration of all mentioned methods requires exactly one calculation of $a_i^T x$ for some x. Therefore, the cost of exact line searches is negligible compares with the cost of computing $a_i^T x$. For example, if each rows of a_i has say $\ge k$ non-zeros, then the gradient computation costs C while the line search costs only $O(C \log(1/\varepsilon)/k) \ll C$ if implemented correctly.

In general, we believe that one can usually design a specialized fast algorithm for the linear search by studying the problem closely. If it is not possible, we note that [38] recently generalized this algorithm and studied how to avoid the exact line search.

Strongly convex assumption: We note that there is a general reduction from the convex case to the strongly convex case [35]. Basically, this involves adding a small quadratic term into f, runs the algorithm for this modified function and gradually removes the quadratic term. Their reduction shows if one obtain an optimal (up to constant) algorithm for the strongly convex case, then one can recover an optimal (up to constant) algorithm for non-strongly convex case. Even surprisingly, such reduction algorithm seems to be pretty efficient compare to a direct algorithm. Due to this, we only consider the strongly convex case here. Certainly, by incorporating the reduction inside our algorithm, one can obtain a direct algorithm.

Performance: Assuming line search can be implemented efficiently, which requires work. This algorithm seems to be better than that accelerated gradient descent in general [37, 39, 38] and especially better when the problem is less smooth. For more difficult problems, there is a memory version [39, 36]. They are sometimes even better than L-BFGS, a widely used algorithm in industry for unconstrained optimization. For problems closer to quadratic functions, there are a variant that combined this algorithm with conjugate gradient [40].

Currently, the main drawback of this algorithm is that it seems difficult to extended to stochastic case, which is very important of machine learning type problems. For this reason, I currently do not recommend use this algorithm to minimize convex problems unless the problem is close to non-smooth.

Problem 6.1.6. Can we have a stochastic version of this algorithm similar to stochastic gradient descent?

6.2 Summary

In the past few lectures, we have covered many cutting plane methods. To summarize, given a convex function f, if we can compute the gradient in time $\mathcal{T}$, then we can minimize the function in $O^*(n\mathcal{T} + n^3)$ time. If $\alpha \preceq \nabla^2 f \preceq \beta$, then we can do $O^*(\sqrt{\frac{\beta}{\alpha}}\mathcal{T})$ instead. Note that two running time are not comparable. One is faster to another depending on if $\frac{\beta}{\alpha}$ is large or not.

One common belief is that dimension independent algorithm is better when the problem is large. However, this belief is only grounded if for example, β/α is independent to the dimension. For example, consider the following problem

$$f(x) = \sum_{k=0}^{n-1}(x_k - x_{k+1})^2.$$

This problem appears everywhere from physical systems, graph problems, signal denoising to statistics. This problem does not involve any very large or very small number and yet $\beta/\alpha = \Theta(n^2)$. Another fun example is

$$f(x) = \|Ax - b\|_2^2$$

where A is a random $n \times n$ matrix with independent Gaussian entries. Again, we have β/α is roughly n^2. For those problems, $O^*(n\mathcal{T})$ does not sound that terrible given that β/α could be $+\infty$ whenever you have a tiny non-smooth term like $|x|$ in the function.

In terms of oracle calls, we know that $O^*(\min(n, \sqrt{\frac{\beta}{\alpha}}))$ is optimal. The lower bound $\sqrt{\frac{\beta}{\alpha}}$ simply follows from the function

$$-x_1 + \beta\sum_{k=0}^{n-1}(x_k - x_{k+1})^2 + \alpha\sum_{k=0}^{n} x_k^2. \tag{6.1}$$

Imagine you start with the initial point $x^{(0)} = (0, 0, 0, \cdots, 0)$. The gradient at $x^{(0)}$ is of the form $(?, 0, 0, 0, \cdots)$. Note that all cutting plane methods we covered satisfies the invariant

$$x^{(k)} = \text{span}(x^{(0)}, g^{(0)}, g^{(1)}, \cdots, g^{(k-1)})$$

where $g^{(i)}$ are the gradient given by the oracle. In that case, we would have $x^{(1)} = (?, 0, 0, 0, \cdots)$ and $g^{(1)} = (?, ?, 0, 0, \cdots)$. By induction, only the first k coordinates of $x^{(k)}$ are non-zero. Using this, it is easy to prove that we need $\Omega(\sqrt{\frac{\beta}{\alpha}})$ oracle call to get an approximate solution to the "worst function" (6.1). Note however that this "worst function" naturally appears in many problems. So, it is a problem we need to address. In some sense, this points out a common problem of cutting plane methods and first-order methods is that it can only communicate the information from one vertex to next via gradient. Therefore, the running time is always lower bounded by the "diameter" of the problem.

For this particular problem (6.1), it is a Laplacian system and we can solve it in nearly linear time. However, the general question of solving problems of those types faster is still an open problem.

References

[35] Zeyuan Allen-Zhu and Elad Hazan. Optimal black-box reductions between optimization objectives. In *Advances in Neural Information Processing Systems*, pages 1614–1622, 2016.

[36] Sébastien Bubeck and Yin-Tat Lee. Black-box optimization with a politician. *arXiv preprint arXiv:1602.04847*, 2016.

[37] Sébastien Bubeck, Yin Tat Lee, and Mohit Singh. A geometric alternative to nesterov's accelerated gradient descent. *arXiv preprint arXiv:1506.08187*, 2015.

[38] Shixiang Chen, Shiqian Ma, and Wei Liu. Geometric descent method for convex composite minimization. In *Advances in Neural Information Processing Systems*, pages 636–644, 2017.

[39] Dmitriy Drusvyatskiy, Maryam Fazel, and Scott Roy. An optimal first order method based on optimal quadratic averaging. *arXiv preprint arXiv:1604.06543*, 2016.

[40] Sahar Karimi and Stephen Vavasis. A single potential governing convergence of conjugate gradient, accelerated gradient and geometric descent. *arXiv preprint arXiv:1712.09498*, 2017.