The University of Texas at Austin
Department of Electrical and Computer Engineering

ECE 381V: Large-Scale Optimization II — Spring 2022

LECTURE 12

Caramanis & Mokhtari Monday, February 28, 2022

Goal: In this lecture, we talk about the general class of cutting-plane methods. We further discuss the center of gravity and ellipsoid algorithms.

1 Finding the description of a convex set

Suppose we have access to the following oracle: When we query a point $\hat{\mathbf{x}}$ we receive the following information

1. if $\hat{\mathbf{x}} \in \mathcal{C}$, we receive the information that $\hat{\mathbf{x}} \in \mathcal{C}$
2. if $\hat{\mathbf{x}} \notin \mathcal{C}$, we receive a separating hyperplane, i,e., we receive vector $\mathbf{a} \neq 0$ and scalar b such that
$$\mathbf{a}^\top \hat{\mathbf{x}} \geq b, \qquad \mathbf{a}^\top \mathbf{z} \leq b \ \text{ for all } \ \mathbf{z} \in \mathcal{C}$$
(We write the separating hyperplane as $\mathbf{a}^\top(\mathbf{z} - \hat{\mathbf{x}}) \geq 0$ when it contains the queried point $\hat{\mathbf{x}}$.)

Note that according to the above oracle, each time we can eliminate one side of the cutting-hyperplane and obtain come up with a better approximation of the convex set $\mathcal{C}$. A natural algorithm to improve our approximation is the following:

1. Initial Step: pick a set $\mathcal{P}_0$ that is large enough to contain the set $\mathcal{C}$
2. Main Loop:
   (a) Pick a point $\mathbf{x}_k$ in $\mathcal{P}_k$
   (b) If $\mathbf{x}_k \in \mathcal{C}$ then return $\mathbf{x}_k$ and stop.
   (c) If $\mathbf{x}_k \notin \mathcal{C}$ then use the cutting plane to update the set according to
$$\mathcal{P}_{k+1} = \mathcal{P}_k \cap \{\mathbf{z} \mid \mathbf{a}^\top \mathbf{z} \leq b_k\}$$

The above procedure improves at each iteration, i.e., $\mathcal{P}_k \supseteq \mathcal{P}_{k+1} \supseteq \mathcal{C}$.
Indeed, the procedure for selecting points $\mathbf{x}_k$ is an important question that we will answer later.

1.1 Connection to unconstrained optimization

Suppose we want to solve the unconstrained optimization problem $\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$ where f is convex. In this case, our goal is to find its optimal solution set, i.e., $\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n | f(\mathbf{x}) = f^*\}$, where f^* is the optimal objective function value.
In this case, if we query a point $\hat{\mathbf{x}}$, its subgradient $\mathbf{g}(\hat{\mathbf{x}})$ provides a cutting plane. In other words, we have
$$\mathbf{g}(\hat{\mathbf{x}})^\top (\mathbf{z} - \hat{\mathbf{x}}) \leq 0 \qquad \text{for all } \ \mathbf{z} \in \mathcal{C}$$

Why? because if $\hat{\mathbf{x}} \notin \mathcal{C}$ (it is not optimal and $f(\hat{\mathbf{x}}) > f^*$) then we know that $\mathbf{g}(\hat{\mathbf{x}})^\top \mathbf{z} > \mathbf{g}(\hat{\mathbf{x}})^\top \hat{\mathbf{x}}$ implies

$$f(\mathbf{z}) \geq f(\hat{\mathbf{x}}) + \mathbf{g}(\hat{\mathbf{x}})^\top (\mathbf{z} - \hat{\mathbf{x}}) > f(\hat{\mathbf{x}}) > f^*$$

1.2 Connection to feasibility problem

Suppose the convex set $\mathcal{C}$ is the set of points that satisfy the following inequalities

$$f_i(\mathbf{x}) \leq 0, \qquad i = 1, \dots, m$$

where f_i are convex.
In this case, if we pick an infeasible point $\hat{\mathbf{x}}$, we can easily find a cutting-plane using its subgradient. More precisely, if $\hat{\mathbf{x}} \notin \mathcal{C} := \{\mathbf{x} \in \mathbb{R}^n \mid f_i(\mathbf{x}) \leq 0, \ i = 1, \dots, m\}$, then there exists j such that $f_j(\hat{\mathbf{x}}) > 0$. Hence, the subgradient $\mathbf{g}_j(\hat{\mathbf{x}}) \in \partial f_j(\hat{\mathbf{x}})$ has the following property:

$$\mathbf{g}_j(\hat{\mathbf{x}})^\top \mathbf{z} \leq \mathbf{g}_j(\hat{\mathbf{x}})^\top \hat{\mathbf{x}} - f_j(\hat{\mathbf{x}}) \quad \text{for all } \mathbf{z} \in \mathcal{C}$$

It can be verified that if $\mathbf{g}_j(\hat{\mathbf{x}})^\top \mathbf{z} > \mathbf{g}_j(\hat{\mathbf{x}})^\top \hat{\mathbf{x}}$, then $\mathbf{z} \notin \mathcal{C}$, because:

$$f_j(\mathbf{z}) > f_j(\hat{\mathbf{x}}) + \mathbf{g}_j(\hat{\mathbf{x}})^\top (\mathbf{z} - \hat{\mathbf{x}}) > 0.$$

1.3 Connection to constrained optimization

By combining the above approaches we obtain that the cutting-plane approach can be used to solve constrained optimization of the form

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0, \qquad i = 1, \dots, m \end{aligned}$$

where the functions are convex. Basically, if the point that we query is not feasible, then we improve the set for feasibility. If the point that we query is feasible but not optimal, then we improve the set corresponding to optimality.

2 Lower bound for for localization methods

Consider the problem of finding a feasible point $\mathbf{x}$ in the set $\mathcal{C} \in \mathbb{R}^n$. We assume the following conditions hold:

1. $\mathcal{C}$ is convex
2. $\mathcal{C}$ is contained in the ℓ_∞-ball with center $\mathbf{0}$ and radius R
3. $\mathcal{C}$ contains an ℓ_2-ball with radius r
4. $\mathcal{C}$ is described by a cutting-place oracle.

Theorem 1. *No localization method can find a feasible point in a complexity lower than*

$$n \ln \left(\frac{R}{2r} \right).$$

This result has an important implication for solving the class of problems

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{Q} \end{aligned}$$

Note that solving the above problem upto accuracy ϵ is equivalent to

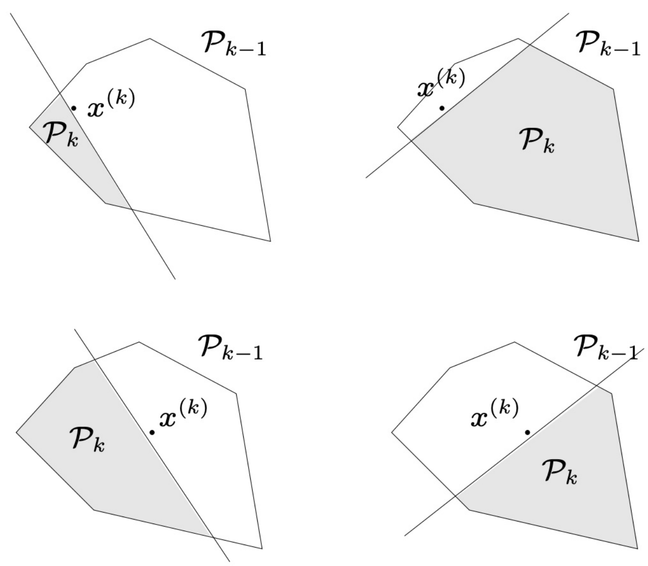
$$\hat{\mathbf{x}} \in \mathcal{Q}, \qquad f(\hat{\mathbf{x}}) - f^* \leq \epsilon$$

The above result shows that if $\mathcal{Q} \subset \{\|\mathbf{x}\|_\infty \leq R\}$ to find such a point we at least require the following number of queries to the cutting plane oracle:

$$n \ln\left(\frac{MR}{8\epsilon}\right).$$

3 The Center of Gravity algorithm

Now the main question that we need to answer is how to select the points $\mathbf{x}_k$ at each iteration to achieve a feasible point as fast as possible. Intuitively, picking a point closer to the center of the current set should be a good choice cause we decrease the volume of the set in a very efficient way.



In the enter of gravity algorithm, at each iteration k, when we have constructed the set $\mathcal{P}_k$, we select the point $\mathbf{x}_k$ according to the following scheme:

$$\mathbf{x}_k := CG(\mathcal{P}_k) = \frac{\int_{\mathcal{P}_k} \mathbf{x} d\mathbf{x}}{\int_{\mathcal{P}_k} d\mathbf{x}}$$

and hence we have

$$\mathcal{P}_{k+1} = \mathcal{P}_k \cap \{\mathbf{x} \mid \mathbf{g}^\top(\mathbf{x} - \mathbf{x}_k) \leq \mathbf{0}\}$$

where in the case of functional constrained optimization $\mathbf{g}$ is either subgradient of a constraint that is infeasible or the subgradient of the objective function.

Theorem 2. *If $\mathcal{P} \in \mathbb{R}^n$ is convex and $\mathbf{g} \neq \mathbf{0}$ we have*

$$\boldsymbol{vol}(\mathcal{P} \cap \{\mathbf{x} \mid \mathbf{g}^\top(\mathbf{x} - CG(\mathcal{P})) \leq \mathbf{0}\}) \leq (1 - 1/e)\boldsymbol{vol}(\mathcal{P})$$

This result shows that by picking the center of gravity as next point we reduce the volume of the set by factor $1 - 1/e$. Hence, we have the following result.

Theorem 3. *If we select a set $\mathcal{P}_0 \subseteq \{\mathbf{x} | \|\mathbf{x}\|_\infty \leq R\}$ which contains the set of interest $\mathcal{C}$, i.e., $\mathcal{P}_0 \supset \mathcal{C}$, and $\mathcal{C}$ contains a ball of size r, then we can find a feasible point after at most the following number of iterations*

$$1.51n \log(R/r)$$

Proof. Note that after k iterations we have

$$\mathbf{vol}(\mathcal{P}_k) \leq (1 - 1/e)^n \mathbf{vol}(\mathcal{P}_0) \leq (1 - 1/e)^k (2R)^n$$

We also know that

$$\mathbf{vol}(\mathcal{P}_k) \geq \mathbf{vol}(\mathcal{C}) \geq (2r)^n$$

Hence, if k is large enough such that the following condition holds $(2r)^n \geq (1 - 1/e)^k (2R)^n$ we are done. This is equivalent to

$$k = \left\lceil \frac{n \log(R/r)}{-\log(1 - 1/e)} \right\rceil = 1.51n \log(R/r)$$

□

This algorithm is near optimal, however, each iteration of that could be very costly as it requires finding the center of gravity for a convex set at each iteration.

4 Ellipsoid Method

Instead of finding the center of gravity, one can find the center of the ellipsoid that contains the set. More precisely, in the ellipsoid method, start by an ellipsoid $\mathcal{E}_0$ which contains the set of interest. Then, at each iteration k, when we have constructed the ellipsoid $\mathcal{E}_k$, we select the point $\mathbf{x}_k$ as the center of the $\mathcal{E}_k$

$$\mathbf{x}_k := \text{center of } \ \mathcal{E}_k$$

and hence we have

$$\mathcal{E}_{k+1} = \text{minimum volume ellipsoid covering } \mathcal{P}_k \cap \{\mathbf{x} \mid \mathbf{g}^\top(\mathbf{x} - \mathbf{x}_k) \leq \mathbf{0}\}$$

localization set is always an ellipsoid. Each step is a convex program is a convex optimization that can be easily solved! In fact it has a simple update rule:
suppose the current ellipsoid is given by

$$\mathcal{E} := \{\mathbf{z} | (\mathbf{z} - \mathbf{x})^\top \mathbf{P}^{-1} (\mathbf{z} - \mathbf{x}) \leq 1\}$$

where $\mathbf{x}$ is its center. Then, the next ellipsoid is given by

$$\mathcal{E}^+ := \{\mathbf{z} | (\mathbf{z} - \mathbf{x}^+)^\top (\mathbf{P}^+)^{-1} (\mathbf{z} - \mathbf{x}^+) \leq 1\}$$

where

$$\mathbf{x}^+ = \mathbf{x} - \frac{1}{n+1}\mathbf{P}\tilde{\mathbf{g}}, \qquad \mathbf{P}^+ = \frac{n^2}{n^2 - 1}\left(\mathbf{P} - \frac{2}{n+1}\mathbf{P}\tilde{\mathbf{g}}\tilde{\mathbf{g}}^\top \mathbf{P}\right)$$

where $\tilde{\mathbf{g}} = \mathbf{g}/(\sqrt{\mathbf{g}^\top \mathbf{P} \mathbf{g}})$.

Theorem 4. *It can be shown that at each round the volume is decreasing by a factor of* $e^{-1/2n}$*, i.e.,*

$$\boldsymbol{vol}(\mathcal{E}^+) \leq \left(1 - \left(\frac{1}{(n+1)^2}\right)\right)^{n/2} \boldsymbol{vol}(\mathcal{E})$$

The above bound can be simplified as

$$\mathbf{vol}(\mathcal{E}^+) < e^{-\frac{1}{2n}} \mathbf{vol}(\mathcal{E})$$

Hence, after k rounds the volume has reduced by $e^{-\frac{k}{2n}}$. Therefore, we can find the set after at most

$$k \approx 2n^2 \log(R/r)$$

this algorithm is not optimal but it can be easily implemented.