Week 5. Convex Optimization

Lecturer: Prof. Santosh Vempala   Scribe: Xin Wang, Zihao Li

Feb. 9 and 11, 2016

1 Week 5. Convex Optimization

1.1 The convex optimization formulation

A general optimization problem is defined as follows: given a function $f : \mathbb{R}^n \to \mathbb{R}$, find the minimum $f(x)$ such that $x \in S$. We write the problem as:

$$\begin{aligned} &\min f(x) \\ &\text{s.t. } x \in S \end{aligned}$$

In this formulation S is a set that could be continuous such as all reals $\mathbb{R}^n$, discrete such as binary $\{0,1\}^n$ or integer $\mathbb{Z}^n$, or a mixed set of continuous and discrete values (e.g., $\mathbb{Z}^n \cap \{x_i \geq 10, \forall i = 1, ..., n\}$). Optimization problems are intractable in general. For example, consider a function which is 0 everywhere except at x_0, and $f(x_0) = -1$. To find the minimum, we will have to search for the whole space. However, there is an important special case for which we have found polynomial-time algorithms to solve the problem: convex optimization. We define the notion of *convexity* for functions and sets.

Definition 1.1 (Convexity). *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $\forall x, y \in \mathbb{R}^n$, and $\forall \lambda \in [0,1]$, the following inequality holds:*

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y). \tag{1}$$

A set $S \subseteq \mathbb{R}^n$ is convex if and only if the whole segment between any two point x, y in S belong to S:

$$x, y \in S \to [x, y] \subseteq S. \tag{2}$$

When f is differentiable, an equivalent definition will be that the first order approximation at any point x gives a lower bound on the function f:

$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle . \tag{3}$$

Convex optimization problems are optimization problems with convex objective function f and convex feasible domain S. We present several applications of convex optimization.

Example 1.1 (Routing with minimum congestion). Given a graph $G = (V, E)$ and k pairs of demand vertices (s_i, t_i), find a single path p_i from s_i to t_i for every i, trying to minimize the congestion C. The congestion is defined as the maximum number of paths going through the any single edge in a directed graph $C = \max_{e \in E} |\{i | e \in p_i|$. This problem can be formulated as a convex optimization problem (which can be turned into a linear optimization problem by setting C as a new variable):

$$\begin{aligned}
&\max C \\
&\sum_{e\in\delta^+(v)} x_{ie} = \sum_{e\in\delta^-(v)} x_{ie} && \forall i, v \in V, v \neq s_i, t_i \\
&\sum_{e\in\delta^-(s_i)} x_{ie} = \sum_{e\in\delta^+(t_i)} x_{ie} = 1 && \forall i \in V \\
&\sum_i x_{ie} \leq C && \forall e \in V \\
&x_{ie} \geq 0 && \forall i, e \in V
\end{aligned}$$

The sets $\delta^+(v)$ and $\delta^-(v)$ correspond to the incoming and outcoming vertices, respectively, for vertex v. The summation constraints enforce flow conservation for all vertices. ⧫

Example 1.2 (Learning with errors). Suppose we have a set of points $\{a_1, a_2, ..., a_N\} \in \mathbb{R}^n$, with each point marked with label either $+$ or $-$. We want to find a hyperplane (represented by norm vector w, where w_1 is set to 1 since $w = 0$ is not desirable) that separates the $+$'s and $-$'s. The problem can be formulated as:

$$\begin{aligned}
&\text{find } w, t \in \mathbb{R}^n \\
&\text{s.t. } w^T a_i \geq t, \text{ if } l(a_i) = + \\
&\quad\;\; w^T a_i < t, \text{ if } l(a_i) = - \\
&\quad\;\; w_1 = 1
\end{aligned}$$

The perfect separation is not always possible and minimizing the number of misclassified points is $\mathcal{NP}$-hard. Rather than counting the number of misclassified points, we can quantify the errors we made by a given hyperplane. Introduce an error y_i for each point a_i, vector w and number t to denote the position of the hyperplane, and using the squared sum of total errors as objective function, we have the following convex optimization problem:

$$\begin{aligned}
&\min \sum_i y_i^2 \\
&\text{s.t. } y_i \geq t - w^T a_i, \text{ if } l(a_i) = + \\
&\quad\;\; y_i \geq w^T a_i - t, \text{ if } l(a_i) = - \\
&\quad\;\; y_i \geq 0, \forall i
\end{aligned}$$

Here the objective function can be any convex function, instead of the quadratic function in the above. This problem can be solved in polynomial time. ⧫

Example 1.3 (Max Cut). Let a graph be $G = (V, E)$, and the edge weight be $w_{ij} \in \mathbb{R}$ for all $(i, j) \in E$. The goal is to find a cut (partition of vertices) such that the total weight across the cut is maximized. The max cut problem can be formulated as

$$\begin{aligned}
&\max \frac{1}{4} \sum_{ij\in E} w_{ij}(x_i - x_j)^2 \\
&\text{s.t. } x_i \in \{-1, 1\}, \forall i \in V(G).
\end{aligned}$$

This is not a convex optimization problem, and it is $\mathcal{NP}$-hard. Max cut is hard because the discreteness of the decision variables. A natural approach in approximating max cut is to relax the decision variables (which can be viewed as randomizing the choices). Let $k \leq n = |V|$ be an integer. Consider the relaxation:

$$\begin{aligned}&\max \frac{1}{4}\sum_{ij\in E} w_{ij}||x_i - x_j||^2\\ &\text{s.t. } x_i \in \mathbb{R}^k \text{ and } ||x_i||^2 = 1\end{aligned}$$

This relaxation can be written as a semidefinite programming (SDP) problem. As a side note, $X \succeq 0$ is convex since:

$$\begin{aligned}A, B \succeq 0 &\to x^T A x \geq 0, x^T B x \geq 0, \forall x \in \mathbb{R}^n\\ &\to x^T(\lambda A + (1-\lambda)B)x \geq 0, \forall \lambda \in [0,1]\\ &\to \lambda A + (1-\lambda)B \succeq 0,\end{aligned} \tag{4}$$

which satisfy the definition of convexity.

Define a matrix $X \in \mathbb{R}^{k\times k}$ by $X_{ij} = \langle x_i, x_j\rangle$, then $||x_i||^2 = X_{ii}$ and $||x_i - x_j||^2 = X_{ii} + X_{jj} - 2X_{ij}$, we get the following SDP problem:

$$\begin{aligned}&\max \frac{1}{4}\sum_{ij\in E} w_{ij}\left(X_{ii} + X_{jj} - 2X_{ij}\right)\\ &\text{s.t. } X_{ii} = 1, \forall i = 1, 2, ..., k\\ &\quad X \succeq 0\end{aligned}$$

We can derive a randomized algorithm from the solution of this convex optimization for the max-cut problem [1]. The solution of this problem gives a set of vectors $\{x_1, x_2, ..., x_n\}$. To obtain a partition of $V(G)$, pick a random hyperplane in $\mathbb{R}^k$ to partition all the vectors into two groups, in other words, pick a random vector $r = (r_1, ..., r_k)^T \in \mathbb{R}^k$ where each r_i is independently picked from $N(0,1)$. And look at $\langle x_i, r\rangle$, if $\langle x_i, r\rangle \geq 0$, let $y_i = 1$, otherwise let $y_i = -1$. Now we compare $\mathbb{E}\left(\frac{1}{4}\sum_{ij\in E} w_{ij}(y_i - y_j)^2\right)$ with $\frac{1}{4}\sum_{ij\in E} w_{ij}||x_i - x_j||^2$. Let the angle between x_i and x_j be θ_{ij}, we have

$$\frac{1}{4}\mathbb{E}(y_i - y_j)^2 = \frac{\theta_{ij}}{\pi}, \tag{5}$$

and

$$\frac{1}{4}||x_i - x_j||^2 = \frac{1}{2}(1 - \cos\theta_{ij}). \tag{6}$$

Therefore, term by term,

$$\frac{\mathbb{E}\left(\frac{1}{4}\sum_{ij\in E} w_{ij}(y_i - y_j)^2\right)}{\frac{1}{4}\sum_{ij\in E} w_{ij}||x_i - x_j||^2} \geq \frac{2\theta_{ij}}{\pi(1 - \cos\theta_{ij})} \geq \min_{\theta\in[0,\pi]} \frac{2\theta}{\pi(1 - cos(\theta))} \approx 0.87857. \tag{7}$$

It is shown that the best possible approximation performance is 0.941 [2]. ♦

Example 1.4 (Max weight perfect matching). A matching in a graph $G(V, E)$ is a set of edges without common vertices. The maximum weight perfect matching problem can be written as:

$$\begin{aligned} \max & \sum_{(i,j)\in E} w_{ij}x_{ij} \\ \text{s.t.} & \sum_j x_{ij} = 1, \forall i \\ & x_{ij} \in \{0,1\} \end{aligned}$$

Relax the constraints to a convex set (although there are exponentially many of them):

$$\begin{aligned} \max & \sum_{(i,j)\in E} w_{ij}x_{ij} \\ \text{s.t.} & \sum_j x_{ij} = 1, \forall i \\ & 0 \le x_{ij} \le 1 \\ & \sum_{i\in S, j\notin S} x_{ij} \ge 1, \text{ for any odd size subset } S \subset V \end{aligned}$$

There may be exponentially many odd subset constraints, so we cannot use them directly for solving the optimization problem. However, there exists a polynomial-time separation oracle for the perfect matching polytopes [3]. ♦

Going back to the general convex optimization problem: find $\min f(x)$ subject to $x \in K$, where f is a convex function and $K \in \mathbb{R}^n$ is a convex set. Denote the optimal value of f by OPT. For a fixed $\epsilon > 0$, we want an algorithm for the following problem: find $x \in K$, such that $f(x) \le \text{OPT} + \epsilon$. In order to find such an algorithm, we need oracle access to f and K. The oracle for f is simply given x, output $f(x)$.

Definition 1.2 (Well-guaranteed separation oracle). *A convex body $K \in \mathbb{R}^n$ is sepcified by $r, R > 0$ and an oracle that takes as input any point $z \in \mathbb{R}^n$ and returns YES if $z \in K$ and NO otherwise, along with a separating halfspace, i.e., a vector s.t. $a^T x \le a^T z$ for all $x \in K$. If K is non empty, there exisits a point x_0 s.t. $x_0 + rB^n \subset K \subset RB^n$.*

A membership oracle only returns YES if $z \in K$ and NO otherwise.

Example 1.5. If K is a polyhedron, then a separation oracle for K is given by r and R such that $x_0 + rB^n \subset K \subset RB^n$, and upon a query for z. It checks all the defining linear inequalities, and outputs $z \in K$ if all the inequalities are satisfied, otherwise it outputs $x \notin K$ and the separating hyperplane is given by the violated linear inequality. ♦

Example 1.6. In a SDP problem, upon a query for Y, the separation oracle computes the eigenvalues of Y using SVD, and outputs $Y \succeq 0$ if all the eigenvalues are nonnegative, otherwise, SVD will output a vector v such that $v^T Y v < 0$, i.e. $\text{tr}(vv^T Y) < 0$, and for any $X \succeq 0$, $\text{tr}(vv^T X) \ge 0$, therefore providing a separating hyperplane. ♦

Solving a convex optimization problem is equivalent to solving a feasibility problem:

1. Feasibility $\Rightarrow$ OPT. If f is a convex function, then

   (a) Fact 1: the sublevel set $S = \{x : f(x) \leq t\}$ is convex. Take two points $a, b \in S$ and $\lambda \in [0, 1]$. Since f is convex,

   $$f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda)f(b) \leq \lambda t + (a - \lambda t) = t. \tag{8}$$

   (b) Fact 2: if S and K are convex, then $S \cap K$ is also convex.

   Therefore, a convex optimization problem

   $$\begin{aligned} &\min f(x) \\ \text{s.t. } & x \in K \end{aligned}$$

   can be solved by solving the feasibility problem

   $$\text{find } x \in \{x : f(x) \leq t\} \cap K,$$

   and a binary search for t.

2. OPT $\Rightarrow$ Feasibility: just make the objective function a constant, and the optimization problem will return a solution that is feasible.

Note it is possible to get a separation oracle for the convex set $K \cap \{x : f(x) \leq t\}$ using the separation oracle for K and the values of f. If the exact form of f or its gradient is known, it is also easy to get a separation oracle.

1.2 Ellipsoid Algorithm

An ellipsoid $Ell(z, D)$ is the set $\{x : (x-z)^T D^{-1}(x-z) \leq 1\}$, where $D \succeq 0$. It can also be written as

$$Ell(z, A) = z + D^{1/2}B^n = \{y : y = z + D^{1/2}x, x \in B^n\}$$

and the volume of the ellipsoid is $|\det(A)|^{1/2}\, Vol(B^n)$.

Algorithm 1.7. (Ellipsoid Algorithm)

1. Let $i = 0, E_i = Ell(0, R^2 I) = RB^n$, $z_i = 0$;

2. Query the oracle for $z_i \in K$?, if the answer is YES, output z_i and terminate; if the answer is NO, the oracle outputs a separating hyperplane with norm a;

3. $E_{i+1} \leftarrow$ the minimal volume ellipsoid containing $E_i \cap \{x : a^T x \leq a^T z\}$ via the following updating formula:

   $$z_{i+1} = z_i - \frac{1}{n+1}\frac{D_i a}{\sqrt{a^T D_i a}} \tag{9}$$

   $$D_{i+1} = \frac{n^2}{n^2-1}\left(D_i - \frac{2}{n+1}\frac{D_i a a^T D_i}{a^T D_i a}\right) \tag{10}$$

4. Repeat step 2 and 3 for T times. If the algorithm does not find a feasible z, output "K is empty".

To show the algorithm is a polynomial-time algorithm, we need to show the minimum volume ellipsoid in step 3 can be found efficiently and only polynomial size T is needed. We prove the following lemma and theorem.

Lemma 1.8 (Half-ellipsoid). *Let E_i be the ellipsoid found in the i^{th} iteration of the ellipsoid algorithm, then*

$$Vol(E_{i+1}) \leq e^{-1/(2n+2)}\, Vol(E_i). \tag{11}$$

Proof. By a linear transform $y \mapsto A^{1/2}(y-z)$, the ellipsoid E_i is transformed to a unit ball. Using a further rotation transform, the separating hyperplane can be transformed the coordinate plane that is perpendicular to the x_1 axis and K is contained in $\{x : x_1 \geq 0\}$. Note this transformation is explicitly given by the A matrix and the separating hyperplane. After obtaining the transformed ellipsoid E_{i+1}, we can perform the inverse transform to get E_{i+1} in the original coordinates, therefore the minimum volume ellipsoid in step 3 can be found in polynomial-time.

By symmetry, the center of the E_{i+1} is on the x_1 axis. The feasible points inside E_{i+1} can be denoted by

$$\frac{(x_1-z)^2}{a^2} + \frac{x_2^2}{b^2} + ... + \frac{x_n^2}{b^2} = 1. \tag{12}$$

E_{i+1} goes through $(1,0,...,0)$ and $(0,1,0,...,0)$, so $a = 1-z$, and

$$\frac{z^2}{(1-z)^2} + \frac{1}{b^2} = 1 \Rightarrow b = \frac{1-z}{\sqrt{1-2z}}. \tag{13}$$

Therefore the ratio between the volumes of the two consecutive ellipsoids is

$$f(z) = \frac{Vol(E_{i+1})}{Vol(E_i)} = \frac{ab^{n-1}}{1} = \frac{(1-z)^n}{(1-2z)^{(n-1)/2}}. \tag{14}$$

To minimize this ratio, we set $\frac{\partial f(z)}{\partial z} = 0$ and get $z = 1/(n+1)$. And the ratio becomes:

$$f(z) = \left(1 - \frac{1}{n+1}\right)\left(1 + \frac{1}{n^2-1}\right)^{(n-1)/2} \leq e^{-\frac{1}{n+1} + \frac{1}{n^2-1}\cdot\frac{n-1}{2}} = e^{-1/(2n+2)}.$$

□

Theorem 1.9 (Ellipsoid Algorithm). *Given a convex body specified by a well-guaranteed separation oracle with parameters r, R, the Ellipsoid Algorithm finds a feasible point in K or returns "K is empty" using $O(n^2 \log(R/r))$ oracle calls and poly$(n, log(R/r))$ additional arithmetic operations.*

Proof. By the lemma, after T iterations,

$$Vol(E_T) \leq e^{-T/(2n+2)} Vol(E_0). \tag{15}$$

Note upon termination, the ellipsoid contains a ball of radius r, so $Vol(E_T) \geq r^n Vol(B^n)$. The initial ellipsoid has volume $Vol(E_0) = R^n Vol(B^n)$. Therefore we get

$$e^{T/(2n+2)} \leq \frac{R^n}{r^n} \Rightarrow T \leq (2n+2)n \ln\frac{R}{r} \Rightarrow T = O(n^2 \log(R/r)). \tag{16}$$

□

We also have the following theorem about the best possible time complexity of an algorithm that utilize the oracle model.

Theorem 1.10 (Optimality). *The complexity of the convex feasibility problem with a separation oracle is* $\Omega(n \log(R/r))$.

Proof. Suppose $rQ^n \subset K \subset RQ^n$ with real numbers $r < R$. For any query $z \in K$?, if the answer is YES, then stop and output this point; otherwise, the oracle gives a separating hyperplane that passes through the geometric center of the current cube, and the remaining set is still a cube. This step reduces the volume of the cube by at most 1/2. Therefore, at least

$$\log\left(\frac{R}{r}\right)^n = n \log \frac{R}{r} \tag{17}$$

oracle calls are needed. □ There exists a shallow-cut ellipsoid algorithm that attains this bound.

1.3 Rounding

Another application of the ellipsoid algorithm is the problem of "rounding" or "sandwiching" a convex body. A set $K \in \mathbb{R}^n$ is called a convex body if it is convex, bounded and fully dimensional. The "rounding" problem is:

Problem 1.11. Given a convex body K, find an ellipsoid E, such that $E \subset K \subset \alpha E$ for anα as small as possible.

Example 1.12. The minimum α values for the n-dimensional ball B^n, cube Q^n and simplex Δ^n are $\alpha(B^n) = 1$, $\alpha(Q^n) = \sqrt{n}$, and $\alpha(\Delta^n) = n$, respectively. ♦

A convex body K is said to be centrally symmetric if for any $x \in K$, we have $-x \in K$.

Theorem 1.13 (The Löwner - John ellipsoid theorem). *In general,*

1. *for any n-dimensional convex body, $\alpha \le n$;*
2. *for any centrally symmetric convex body, $\alpha \le \sqrt{n}$.*

The ellipsoid E is the maximal volume inscribed ellipsoid of K. Finding the optimal rounding ellipsoid E is a hard problem [4].

However for polytopes, we have the following theorem:

Theorem 1.14. *Given a n-dimensional convex body K,*

1. *there exists a deterministic polynomial-time algorithm that finds an ellipsoid E, such that $E \subset K \subset (n+1)\sqrt{n}E$, and*
2. *for any $\epsilon > 0$, there exists a randomized polynomial-time algorithm that finds an ellipsoid E, such that $E \subset K \subset (1+\epsilon)\sqrt{n}E$.*

The deterministic algorithm is a modification of the ellipsoid algorithm

Algorithm 1.15. Given a convex body K.

1. Use the ellipsoid algorithm to find a $z \in K$, and an ellipsoid $Ell(z, A)$ that contains K.
2. Denote the half axis of $Ell(z, A)$ by $a_1, a_2, ..., a_n$. For each $i = 1, 2, ..., n$, query whether $z \pm (a_i)/(n+1) \in K$. If any of the answers is no, then the separating oracle provides a separating hyperplane at the same time. Using the same procedure as in the ellipsoid algorithm, we get a new ellipsoid $Ell(z', A')$ that still contains K. Repeat this step with the new ellipsoid. We refer to this step as a "shallow cut". If all the answers are YES, go to step 3.
3. Denote E the current ellipsoid that contains K. Since all the answers to queries are YES, the cross-polytope with half-axis $a_1/(n+1), a_2/(n+1), ..., a_n/(n+1)$ is contained in the set K, and it is easy to see the inscribed ellipsoid of this cross-polytope is $\frac{1}{(n+1)\sqrt{n}}E$. Therefore $\frac{1}{(n+1)\sqrt{n}}E \subset K \subset E$.

To see the algorithm terminates in polynomial time, we need every shallow cut to reduce the volume of ellipsoid by a sufficient amount. The proof of this claim is similar to the argument in the ellipsoid algorithm.

Exercise 1.1 (Shallow-cut ellipsoid). *Let E be an ellipsoid with axes $a_1, ..., a_n$ and center z. Thake any halfspace H containing $z + (a_1/(n+1))$ in its bounding hyperplane. Let E_1 be the minimum volume ellipsoid containing $E \cap H$. Show that $Vol(E_1) \leq e^{-c/n} Vol(E)$ for some absolute constant c.*

By the conclusion of the above exercise, the algorithm needs at most $O(n^2 \log(R/r))$ shallow cuts, and therefore is polynomial time.

We discussed the convex optimization algorithms using the separation oracle for the feasible domain. For the case where only membership oracle access is available, we also have similar results, that is

Theorem 1.16. *Given a convex optimization problem*

$$\begin{aligned} &\min f(x) \\ &s.t.\ x \in K \end{aligned}$$

where f is a convex function, K is a convex set and $\epsilon > 0$. Denote OPT the optimal value of f. If a membership oracle for K is available, then there exists an algorithm that finds a $x \in K$, such that $f(x) \leq OPT + \epsilon$ in poly$(n, \log(1/\epsilon), \log(R/r))$ time.

References

[1] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Acm*, 42(6):1115–1145, 1995.

[2] J. Hastad. Some optimal inapproximability results. *Journal of the Acm*, 48(4):798–859, 2001.

[3] M. W. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7(1):67–80, 1982.

[4] K. Ball. Ellipsoids of maximal volume in convex-bodies. *Geometriae Dedicata*, 41(2):241–250, 1992.