Equivalence of separation and optimization

$G \subseteq \mathbb{R}^n$, convex body: $B(0, r) \subseteq G \subseteq B(0, R)$ (“well-rounded”).

(i) **Strong separation $\Rightarrow$ weak optimization** (proved last time).

**Application** Network synthesis problem.

We want to build capacity on the edges of an undirected graph $G = (V, E)$ to satisfy flow requirements: we need to sustain a flow of $r_{ij}$ from $i$ to $j$ for all $i, j$. There are costs $d_{ij}$ to build each unit of capacity on edge $ij$.

**Formulation**

$$\min \sum_{ij \in E} d_{ij}x_{ij},$$

$$\sum_{k \in I, l \in V \setminus I} x_{kl} \geq r_{ij} \quad \text{for all } I \subseteq V, i \in I, j \notin I,$$

$$x_{ij} \geq 0 \quad \text{for all } ij \in E.$$

Here each $x_{ij}$ is the amount of capacity installed on edge $ij$. The constraints assure that the requirements can be met by making sure each minimum cut has the required capacity.

We have $|E|$ variables, but (at least) $2^n - 2$ constraints.

However, given $x \in \mathbb{R}^n$, we can either find $x$ is feasible or obtain a violated constraint (= strong separation) by solving about $\frac{n^2}{2}$ max-flow problems ($n = |V|$). If the flow is not sufficient for a particular $i, j$, the algorithm will give a min-cut with capacity smaller than $r_{ij}$, hence a violated constraint. Therefore we have a polynomial-time algorithm for weak optimization.

[Another LP formulation has polynomial size, by using extra variables $f_{kl}^{ij} =$ flow on edge $kl \in E$ in a flow from $i$ to $j$ of size $r_{ij}$, but then we have $O(n^4)$ variables.]

(ii) **Strong optimization $\Rightarrow$ weak separation.**

**Strong optimization:** Given $c \in \mathbb{R}^n$, solve $\max\{c^T x : x \in G\}$.  


**Weak Separation:** Given \( x \in \mathbb{R}^n \) and \( 0 < \eta \leq 1/2 \), either determine \( x \in G \) or find \( v \in \mathbb{R}^n \) with \( v^T z \leq 1 \) for all \( z \in G \) but \( v^T x \geq 1 - \eta \).

We use the polar \( G^* := \{ y : x^T y \leq 1 \text{ for all } x \in G \} \). Note that if \( G \subseteq H \), then \( H^* \subseteq G^* \). Hence, if \( B(0,r) \subseteq G \subseteq B(0,R) \), \( B(0,\frac{1}{R}) \subseteq G^* \subseteq B(0,\frac{1}{r}) \). Also, if \( G \) is closed and convex and contains 0, then \( G^{**} = G \). This can be proved using separating hyperplanes.

**Theorem 1** If there is an algorithm for strong optimization polynomial in \( n \), \( \ln \frac{1}{r} \), and \( \ln R \), then there is an algorithm for weak separation, polynomial in \( n \), \( \ln \frac{1}{r} \), \( \ln R \), and \( \ln \frac{1}{\eta} \).

**Proof:** We use the previous theorem for \( G^* \). Given \( x \in \mathbb{R}^n \), if \( x = 0 \), declare \( x \in G \), and if \( \|x\| > R \), return \( v = \frac{x}{\|x\|} \). So assume \( 0 < \|x\| \leq R \).

Now we solve the weak optimization problem for \( G^* \), with \( c = \frac{x}{\|x\|}, \varepsilon = \frac{\eta}{R} \).

If we can do this, we get \( \max \{ \frac{x}{\|x\|}^T v : v \in G^* \} \) within \( \varepsilon = \frac{\eta}{R} \).

If the maximum is at most \( \frac{1}{\|x\|} \), then \( \max \{ x^T v : v \in G^* \} \leq 1 \), so \( x \in G^{**} \). But this is \( G \) if \( G \) is closed and convex and contains 0.

Otherwise, we have a near optimal solution \( v \in G^* \) with \( \frac{x}{\|x\|}^T v \geq 1 - \frac{\eta}{R} \), so \( v^T x \geq 1 - \eta \frac{\|x\|}{R} \geq 1 - \eta \), and since \( v \in G^* \), \( v^T z \leq 1 \) for all \( z \in \mathbb{Z} \). So we have solved the weak separation problem.

So we are done if we can solve the strong separation problem for \( G^* \) in time polynomial in \( n \), \( \ln \frac{1}{r} \), and \( \ln R \).

To do this, we solve the strong optimization problem for \( G \).

Given \( x \in \mathbb{R}^n \), find \( z \) with \( x^T z = \max \{ x^T y : y \in G \} \).

If the maximum is at most 1, then \( x \in G^* \). If the maximum is more than 1, then we have \( z \in G \) with \( x^T z > 1 \geq y^T z \) for all \( y \in G^{**} \) since \( z \in G^* \). Thus we have solved the strong separation problem for \( G^* \), completing the proof.

We still don’t have symmetry between separation and optimization: we need to show that weak separation allows us to do weak optimization.

**Theorem 2** If there is an algorithm for the weak separation problem for \( G \), polynomial in \( n \), \( \ln \frac{1}{r} \), \( \ln R \), and \( \ln \frac{1}{\eta} \), then there is an algorithm for the weak optimization problem for \( G \), polynomial in \( n \), \( \ln \frac{1}{r} \), \( \ln R \) and \( \ln \frac{1}{\varepsilon} \).
Proof: We use the ellipsoid method, now with shallow cuts.

At each iteration, we have $x_k$. If $x_k = 0$, declare $x_k \in G$ and use $a_k = -c$ as usual. If $\|x_k\| > R$, we can easily find a suitable $v_k$. Otherwise we solve the weak separation problem for $x_k$, with $\eta = \frac{\varepsilon r}{6(n+1)R} < \frac{1}{2}$. This either states $x_k \in G$ and then we use $a_k = -c$ as usual, or gives $v$ with $v^T x \leq 1$ for all $x \in G$ and $v^T x_k \geq 1 - \eta$.

If the algorithm hasn’t terminated, $E_k$ contains a ball of radius $\varepsilon r$, so $\max\{v^T x : x \in E_k\} \geq v^T x_k + \varepsilon r \|v\|$. But $\|v\| \|x_k\| \geq v^T x_k \geq 1 - \eta \geq \frac{1}{2}$, so $\|v\| \geq \frac{1}{2R}$.

Hence, $\max\{v^T x : x \in E_k\} \geq v^T x_k + \frac{\varepsilon r}{2R}$. Thus, $\eta \leq \frac{v^T B_k v}{3(n+1)}$.

So we can find a new ellipsoid $E_{k+1}$ using a cut with $\alpha \geq -\frac{1}{3(n+1)}$. (See Figure 1.)

Figure 1: Illustration of Ellipsoid method with a shallow cut.
Then the volume reduction is at least
\[
\left( \frac{n^2}{n^2 - 1} \right)^{\frac{n-1}{2}} \left( \frac{n}{n+1} \right) \left( 1 + \frac{1}{3(n+1)} \right)
\leq \exp \left( \frac{1}{2(n+1)} \right) \exp \left( -\frac{1}{n+1} \right) \exp \left( \frac{1}{3(n+1)} \right).
\]

The RHS is equal to \( \exp \left( -\frac{1}{6(n+1)} \right) \).

Hence, even though the volume reduction is smaller, as before, we get a polynomial-time algorithm for weak optimization. \( \square \)

We work towards algorithms for large \( n \) (\( n \approx 10^4 \)) and moderate \( \varepsilon \) (\( \varepsilon \approx 10^{-2} \)).

Subgradient algorithms: at each step, move in the direction of the negative of a subgradient. Unfortunately, the negative of a subgradient may not be a descent direction! For example, consider
\[
f(x) = \max\{2x^{(1)} - x^{(2)}, -x^{(1)} + 2x^{(2)}, 0\}.
\]
See Figure 2.

![Figure 2: Example: the negative of a subgradient is not a descent direction.](image)