1 Overview

In the last lecture, we introduced the concept of linear programming as well as some of its basic properties which include duality, Farkas' lemma and complementary slackness. In this lecture, after a short discussion of the basic geometry of LPs, we will briefly introduce several algorithms to solve LPs. These algorithms include simplex algorithm, ellipsoid algorithm and interior point algorithm. Moreover, we will discuss a phenomenon called separation oracles.

2 Geometry of LPs

First recall a linear programming problem has the following form

\[
\begin{align*}
\text{minimize} & \quad \sum_{j=1}^{n} c_j x_j \\
\text{subject to} & \quad \sum_{j=1}^{n} a_{ij} x_j = b(i) \quad \text{for all } i = 1, \ldots, m, \\
& \quad x_j \geq 0 \quad \text{for all } j = 1, \ldots, n.
\end{align*}
\]

Or more concisely in matrix form

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax = b, \\
& \quad x \geq 0.
\end{align*}
\]

Definition 1. A polyhedral set associated with the above LP is

\[ P = \{x : Ax = b, x \geq 0\}. \]

x is a vertex of P if there is no y ≠ 0, such that x + y ∈ P or x − y ∈ P.

x is a basic feasible solution to the LP if x ∈ P and there is B ⊂ \{1, \ldots, n\} such that |B| = m and

1. \( x_N = 0 \) for \( N = \{1, \ldots, n\} - B \)
2. \( A_B \) is non-singular
3. \( x_B = A_B^{-1} b \geq 0. \)

Lemma 1. x is a vertex if and only if it is a basic feasible solution in the following sense: the columns \( \{A_j : x_j > 0\} \) of the constraint matrix corresponding to the strictly positive variables are linearly independent.
**Proof.** We interpret the proof in [1] here. Suppose that \( x_1 > 0, x_2 > 0, \ldots, x_k > 0 \) and \( x_{k+1} = x_{k+2} = \cdots = x_n = 0 \). Suppose the columns \( A_1, A_2, \ldots, A_k \) are linearly dependent. Then there exist constants \( \omega_1, \omega_2, \ldots, \omega_k \) (not all zero), satisfying the condition

\[
A_1 \omega_1 + A_2 \omega_2 + \cdots + A_k \omega_k = 0.
\]

Define \( \omega_{k+1} = \omega_{k+2} = \cdots = \omega_n = 0 \) and let \( \omega \) denote the vector \( (\omega_1, \omega_2, \ldots, \omega_n) \). Then since each of the first \( k \) components of the solution \( x \) are positive, for some sufficiently small value of the scalar \( \theta \),

\[
x + \theta \omega \geq 0 \text{ and } x - \theta \omega \geq 0.
\]

Also, since \( A \omega = 0 \), \( A (x + \theta \omega) = A (x - \theta \omega) = A x = b \). Therefore, both \( x + \theta \omega \) and \( x - \theta \omega \) are feasible for the linear program. But then \( x = \frac{1}{2}(x + \theta \omega) + \frac{1}{2}(x - \theta \omega) \), which implies that \( x \) is not a vertex. Therefore, we have shown that if \( x \) is a vertex, the columns \( A_1, A_2, \ldots, A_k \) must be linearly independent.

To establish the converse, suppose that \( x = \theta x^1 + (1 - \theta) x^2 \) for some scalar \( 0 < \theta < 1 \) and two feasible points \( x^1 \) and \( x^2 \) to the linear program. Since \( x_j = \theta x_j^1 + (1 - \theta) x_j^2 \) for any \( j \geq k + 1 \), and both \( x_j^1 \) and \( x_j^2 \) are nonnegative, both \( x_j^1 \) and \( x_j^2 \) must have value zero. Therefore, since both these points are feasible solutions to the linear program, theory satisfy

\[
A_1 x_1^1 + A_2 x_1^2 + \cdots + A_k x_k^1 = b,
\]
and

\[
A_1 x_1^2 + A_2 x_2^2 + \cdots + A_k x_k^2 = b.
\]

Subtracting these equations from each other shows that

\[
A_1 (x_1^1 - x_1^2) + A_2 (x_2^1 - x_2^2) + \cdots + A_k (x_k^1 - x_k^2) = b.
\]

But since the columns \( A_1, A_2, \ldots, A_k \) are linearly independent, each component of \( x^1 \) and \( x^2 \) must be the same. But then we cannot represent \( x = \theta x^1 + (1 - \theta) x^2 \) in terms of two distinct feasible points of the linear program and therefore \( x \) is a vertex.

If we let \( k = m \) in the above argument, the columns \( A_1, A_2, \ldots, A_k \) form a basis of the linear program, so \( x \) is a basic feasible solution. On the other hand, if \( k < m \), we can add \( m - k \) other linearly independent columns to \( A_1, A_2, \ldots, A_k \) to form a basis \( B \). In this case \( x \) is still a basic feasible solution, but with some basic variables at value zero. Therefore, basic feasible solutions and vertices are identical. 

Next we give a characterization of an optimal solution based on the previous observation.

**Theorem 2.** For a bounded LP so that \( \min \{ c^T x : x \in P \} \) is finite, then for any \( x \in P \), there exists a vertex \( x' \) such that \( c^T x' \leq c^T x \), which implies the LP has an optimal solution at a vertex or basic feasible solution.

**Proof.** We interpret the proof in [3] here. If \( x \) is a vertex, then \( x' = x \). If \( x \) is not a vertex, then by definition, there exists \( y \neq 0 \) such that \( x + y \in P \) and \( x - y \in P \). Since \( A (x + y) = b \) and \( A (x - y) = b \), we have \( A y = 0 \). Without loss of generality, assume \( c^T y \leq 0 \). If \( c^T y = 0 \), choose \( y \) such that there exists \( y_j < 0 \). Since \( y \neq 0 \) and \( c^T y = c^T (-y) = 0 \), this must be true for either \( y \) or \(-y \). Consider \( x + \lambda y \), \( \lambda > 0 \). \( c^T (x + \lambda y) = c^T x + \lambda c^T y \leq c^T x \), since \( c^T y \) is assumed non-positive.

Case 1 There exists \( y_j < 0 \). As \( \lambda \) increases, component \( j \) decreases until \( x + \lambda y \) is no longer feasible. Choose \( \lambda = \min_{y_j < 0} \{ |y_j| / -y_k \} = x_k / -y_k \) which is the largest \( \lambda \) such that \( x + \lambda y \geq 0 \). Since \( A y = 0 \), \( A (x + \lambda y) = A x + \lambda A y = A x = b \). So \( x + \lambda y \in P \), and moreover \( x + \lambda y \) has more zero component than \( x \). So we can replace \( x \) by \( x + \lambda y \).
Case 2 $y_j \geq 0$ for any index $j$. Then by assumption, $c^T y < 0$ and $x + \lambda y$ is feasible for all $\lambda \geq 0$, since 
$s(x + \lambda y) = s x + \lambda s y = s x = b$, and $x + \lambda y \geq x \geq 0$. But $c^T (x + \lambda y) = c^T x + \lambda c^T y \to -\infty$ as $\lambda \to \infty$, implying the LP is unbounded which is a contradiction.

3  Brief Introduction to Simplex Algorithm

The simplex algorithm maintains a basic feasible solution at every step. Given a basic feasible solution, the algorithm first applies the optimality condition to test the optimality of the current solution. If the current solution does not fulfill this condition, the algorithm performs an operation, known as a pivot operation, to obtain another basis structure with a lower or identical cost. The simplex algorithm repeats this process until the current basic feasible solution satisfies the optimality condition.

4  Ellipsoid Algorithm and Separation Oracles

We follow the interpretation in [4].

Ellipsoid algorithm is the first polynomial-time algorithm discovered for linear programming. It works on the polyhedral set $P$ associated with a linear program problem which is a bounded convex set. We start with a big ellipsoid $E$ that is guaranteed to contain $P$. We then check if the center of the ellipsoid is in $P$. If it is, we are done, we found a point in $P$. Otherwise, we find an inequality $c^T x \leq d_i$ which is satisfied by all points in $P$ which is not satisfied by our center. More formally, the ellipsoid algorithm is the following.

1. Let $E_0$ be an ellipsoid containing $P$
2. while center $a_k$ of $E_k$ is no in $P$ do:
   (a) Let $c^T x \leq c^T a_k$ be such that $P \subset \{x : c^T x \leq c^T a_k\}$
   (b) Let $E_{k+1}$ be the minimum volume ellipsoid containing $E_k \cap \{x : c^T x \leq c^T a_k\}$
   (c) $k \leftarrow k + 1$

If the set $P$ has positive volume, we will eventually find a point in $P$. This is because the ellipsoids constructed shrink in volume as the algorithm proceeds.

Lemma 3.

$$\frac{\text{Vol}(E_{k+1})}{\text{Vol}(E_k)} < e^{-\frac{1}{2(n+1)}}.$$ 

Proof. The volume of an ellipsoid is proportional to the product of its side lengths. Hence the ratio between
the unit ellipsoid $E_k$ and $E_{k+1}$ is

$$\frac{\Vol(E_{k+1})}{\Vol(E_k)} = \left(\frac{n}{n+1}\right)^{\frac{n+1}{2}}$$

$$< e^{-\frac{1}{n+1}} e^{\frac{n-1}{2(n-1)}}$$

$$= e^{-\frac{1}{n+1}} e^{\frac{1}{2(n+1)}}$$

$$= e^{-\frac{1}{2(n+1)}}.$$ 

\[\square\]

4.1 Separation Oracles

To run the ellipsoid algorithm, we need to be able to decide, given a point $x$, whether $x \in P$ or find a violated inequality. It is lucky that we don’t need a complete and explicit description of $P$ in terms of linear inequalities. A separation oracle for $P$ is that: given $x^* \in \mathbb{R}^n$, either decide that $x^* \in P$ or find an inequality $a^T x \leq b$ valid for $P$ such that $a^T x^* > b$. If this separation oracle runs in polynomial-time, we have succeeded in finding the optimal value $d$ when optimizing $c^T x$ over $P$.

**Theorem 4.** Let $S = \{0,1\}^n$ and $P = \text{conv}(S)$. Assume that $P$ is full-dimensional and we are given a separation oracle for $P$. Then, given $c \in \mathbb{Z}^n$, one can find $\min\{c^T x : x \in S\}$ by the ellipsoid algorithm by using a polynomial number of operations and calls to the separation oracle.

5 Brief Introduction to Interior Point Algorithm

Interior point algorithm is a more practical algorithm for LPs based on primal-dual analysis which means that it simultaneously solves both the primal and dual problems. The basic idea of this algorithm is to stay away from the boundaries of the polyhedron while approaching optimality. It uses a potential function to measure how small the duality gap is and how far the current iterate is away from the boundaries.

6 Minimum Cost Arborescence Problem

The LP for minimum cost arborescence problem is given as

$$\min \sum_{a \in A} c_a x_a$$

subject to

$$\sum_{a \in \partial^+(S)} x_a \geq 1, \quad \forall S \subset V \setminus \{r\}$$

$$x_a \geq 0, \quad a \in A.$$

We can find a separation oracle for the above problem. Consider a point $x^*$. If $X^*_a < 0$ for some $a \in A$, return the inequality $x_a \geq 0$. Otherwise, for every $t \in V \setminus \{r\}$, consider the minimum $r-t$ cut problem in which the capacity on arc $a$ is given by $x^*_a$ which can be solved by maximum flow computations. If for some
$t \in V \setminus \{r\}$, the minimum $r - t$ cut has value less than 1 then we have found a violated inequality by $x^\ast$. Otherwise, we have that $x^\ast \in P$. Thus the separation oracle can be implemented by doing $|V| - 1$ maximum flow computations, and hence is polynomial.

## 7 Summary

In this lecture, we discussed basic geometry of LPs including vertex and basic feasible solution which are two equivalent concepts. Based on these concepts, we briefly introduced simplex algorithm. Next, we discussed ellipsoid algorithm in detail as well as the separation oracles. Moreover, we also briefly introduced another algorithm called the interior point algorithm. At the end, we analyzed a concrete example in which we have a separation oracle.

### References.


