15.081J/6.251J Introduction to Mathematical Programming

Lecture 18: The Ellipsoid method
1 Outline

- Efficient algorithms and computational complexity
- The key geometric result behind the ellipsoid method
- The ellipsoid method for the feasibility problem
- The ellipsoid method for optimization

2 Efficient algorithms

- The LO problem
  \[
  \begin{align*}
  \min & \quad c'x \\
  \text{s.t.} & \quad Ax = b \\
  & \quad x \geq 0
  \end{align*}
  \]

- A LO instance
  \[
  \begin{align*}
  \min & \quad 2x + 3y \\
  \text{s.t.} & \quad x + y \leq 1 \\
  & \quad x, y \geq 0
  \end{align*}
  \]

- A problem is a collection of instances

2.1 Size

- The size of an instance is the number of bits used to describe the instance, according to a prespecified format
- A number \( r \leq U \)
  \[
  r = a_k 2^k + a_{k-1} 2^{k-1} + \cdots + a_1 2^1 + a_0
  \]
  is represented by \( (a_0, a_1, \ldots, a_k) \) with \( k \leq \lceil \log_2 U \rceil \)
- Size of \( r \) is \( \lceil \log_2 U \rceil + 2 \)
- Instance of LO: \( (c, A, b) \)
- Size is
  \[
  (mn + m + n) \left( \lceil \log_2 U \rceil + 2 \right)
  \]

2.2 Running Time

Let \( A \) be an algorithm which solves the optimization problem \( \Pi \).
If there exists a constant \( \alpha > 0 \) such that \( A \) terminates its computation after at most
\( \alpha f(I) \) elementary steps for each instance \( I \), then \( A \) runs in \( O(f) \) time.

Elementary operations are
- variable assignments
- random access to variables
- conditional jumps
- comparison of numbers
- arithmetic operations
- \( \ldots \)
A “brute force” algorithm for solving the min-cost flow problem:
Consider all spanning trees and pick the best tree solution among the feasible ones.

Suppose we had a computer to check $10^{15}$ trees in a second. It would need more than $10^9$ years to find the best tree for a 25-node min-cost flow problem. It would need $10^{59}$ years for a 50-node instance.

**That’s not efficient!**

Ideally, we would like to call an algorithm “efficient” when it is sufficiently fast to be usable in practice, but this is a rather vague and slippery notion.

The following notion has gained wide acceptance:
An algorithm is considered efficient if the number of steps it performs for any input is bounded by a polynomial function of the input size.

Polynomials are, e.g., $n$, $n^3$, or $10^6 n^8$.

### 2.3 The Tyranny of Exponential Growth

<table>
<thead>
<tr>
<th></th>
<th>100 $n \log n$</th>
<th>$10 n^2$</th>
<th>$n^{1.5}$</th>
<th>$2^n$</th>
<th>$n!$</th>
<th>$n^{1.22}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^9$/sec</td>
<td>1.19 · 10³</td>
<td>600,000</td>
<td>3,868</td>
<td>41</td>
<td>15</td>
<td>13</td>
</tr>
<tr>
<td>$10^{10}$/sec</td>
<td>1.08 · 10⁸</td>
<td>1,897,370</td>
<td>7,468</td>
<td>45</td>
<td>16</td>
<td>13</td>
</tr>
</tbody>
</table>

Maximum input sizes solvable within one hour.

### 2.4 Punch line

The equation

\[
\text{efficient} = \text{polynomial}
\]

has been accepted as the best available way of tying the empirical notion of a “practical algorithm” to a precisely formalized mathematical concept.

### 2.5 Definition

An algorithm runs in *polynomial time* if its running time is $O(|I|^k)$, where $|I|$ is the input size, and all numbers in intermediate computations can be stored with $O(|I|^k)$ bits.

### 3 The Ellipsoid method

- $D$ is an $n \times n$ positive definite symmetric matrix
- A set $E$ of vectors in $\mathbb{R}^n$ of the form
  \[
  E = E(z, D) = \{ x \in \mathbb{R}^n \mid (x - z)' D^{-1} (x - z) \leq 1 \}
  \]
  is called an ellipsoid with center $z \in \mathbb{R}^n$
3.1 The algorithm intuitively

- Problem: Decide whether a given polyhedron

\[ P = \{ x \in \mathbb{R}^n \mid Ax \geq b \} \]

is nonempty

- Key property: We can find a new ellipsoid \( E_{t+1} \) that covers the half-ellipsoid and whose volume is only a fraction of the volume of the previous ellipsoid \( E_t \)

3.2 Key Theorem

- \( E = E(z, D) \) be an ellipsoid in \( \mathbb{R}^n \); \( a \) nonzero \( n \)-vector.
- \( H = \{ x \in \mathbb{R}^n \mid a' x \geq a' z \} \)

\[ \Psi = z + \frac{1}{n+1} \frac{Da}{\sqrt{a'Da}} \]

\[ \mathcal{D} = \frac{n^2}{n^2 - 1} \left( D - \frac{2}{n+1} \frac{Da'aD}{a'Da} \right) \]

- The matrix \( \mathcal{D} \) is symmetric and positive definite and thus \( E' = E(\Psi, \mathcal{D}) \) is an ellipsoid
- \( E \cap H \subset E' \)
- \( \text{Vol}(E') < e^{1/(2(n+1))} \text{Vol}(E) \)
3.3 Illustration

3.4 Assumptions

- A polyhedron $P$ is **full-dimensional** if it has positive volume
- The polyhedron $P$ is bounded: there exists a ball $E_0 = E(x_0, r^2 I)$, with volume $V$, that contains $P$
- Either $P$ is empty, or $P$ has positive volume, i.e., $\text{Vol}(P) > v$ for some $v > 0$
- $E_0$, $v$, $V$, are a priori known
- We can make our calculations in infinite precision; square roots can be computed exactly in unit time

3.5 Input-Output

**Input:**

- A matrix $A$ and a vector $b$ that define the polyhedron $P = \{ x \in \mathbb{R}^n \mid a_i'x \geq b_i, \; i = 1, \ldots, m \}$
- A number $v$, such that either $P$ is empty or $\text{Vol}(P) > v$
A ball $E_0 = E(x_0, r^2 I)$ with volume at most $V$, such that $P \subseteq E_0$

Output: A feasible point $x^* \in P$ if $P$ is nonempty, or a statement that $P$ is empty

### 3.6 The algorithm

1. **(Initialization)**
   
   Let $t^* = \left[2(n+1) \log(V/v)\right]$; $E_0 = E(x_0, r^2 I)$; $D_0 = r^2 I$; $t = 0$.

2. **(Main iteration)**
   
   - If $t = t^*$ stop; $P$ is empty.
   - If $x_t \in P$ stop; $P$ is nonempty.
   - If $x_t \notin P$ find a violated constraint, that is, find an $i$ such that $a_i'x_t < b_i$.
   - Let $H_t = \{x \in \mathbb{R}^n \mid a_i'x \geq a_i'x_t\}$. Find an ellipsoid $E_{t+1}$ containing $E_t \cap H_t$:
     
     $x_{t+1} = x_t + \frac{1}{n+1} \frac{D_t a_i}{\sqrt{a_i' D_t a_i}}$
     
     $D_{t+1} = \frac{n^2}{n^2-1} \left(D_t - \frac{2}{n+1} \frac{D_t a_i a_i' D_t}{a_i' D_t a_i}\right)$.

   - $t := t + 1$.

### 3.7 Correctness

**Theorem:** Let $P$ be a bounded polyhedron that is either empty or full-dimensional and for which the prior information $x_0, r, v, V$ is available. Then, the ellipsoid method decides correctly whether $P$ is nonempty or not, i.e., if $x_{t^*-1} \notin P$, then $P$ is empty

### 3.8 Proof

- If $x_t \in P$ for $t < t^*$, then the algorithm correctly decides that $P$ is nonempty

- Suppose $x_0, \ldots, x_{t^*-1} \notin P$. We will show that $P$ is empty.

- We prove by induction on $k$ that $P \subseteq E_k$ for $k = 0, 1, \ldots, t^*$. Note that $P \subseteq E_0$, by the assumptions of the algorithm, and this starts the induction.

- Suppose $P \subseteq E_k$ for some $k < t^*$. Since $x_k \notin P$, there exists a violated inequality: $a_{(i(k))}'x \geq b_{i(k)}$ be a violated inequality, i.e., $a_{(i(k))}'x_k < b_{i(k)}$, where $x_k$ is the center of the ellipsoid $E_k$.
• For any \( x \in P \), we have
\[
\mathbf{a}'_i(k)x \geq \mathbf{b}_i(k) > \mathbf{a}'_i(k)x_k
\]
• Hence, \( P \subset H_k = \{x \in \mathbb{R}^n \mid \mathbf{a}'_i(k)x \geq \mathbf{a}'_i(k)x_k\} \)
• Therefore, \( P \subset E_k \cap H_k \)

By key geometric property, \( E_k \cap H_k \subset E_k + 1 \); hence \( P \subset E_k + 1 \) and the induction is complete

\[
\frac{\text{Vol}(E_{i+1})}{\text{Vol}(E_i)} < e^{-1/(2(n+1))} \\
\frac{\text{Vol}(E_{i+1})}{\text{Vol}(E_i)} < e^{-c^*/(2(n+1))}
\]

\[
\text{Vol}(E_{i+1}) < V e^{-\,(2(n+1) \log V)/(2(n+1))} \leq V e^{-\log V} = v
\]

If the ellipsoid method has not terminated after \( t^* \) iterations, then \( \text{Vol}(P) \leq \text{Vol}(E_{t^*}) \leq v \). This implies that \( P \) is empty

3.9 Binary Search

• \( P = \{x \in \mathbb{R} \mid x \geq 0, x \geq 1, x \leq 2, x \leq 3\} \)
• \( E_0 = [0,5] \), centered at \( x_0 = 2.5 \)
• Since \( x_0 \notin P \), the algorithm chooses the violated inequality \( x \leq 2 \) and constructs \( E_1 \) that contains the interval \( E_0 \cap \{x \mid x \leq 2.5\} = [0,2.5] \)
• The ellipsoid \( E_1 \) is the interval \([0,2.5]\) itself
• Its center \( x_1 = 1.25 \) belongs to \( P \)
• This is binary search

3.10 Boundedness of \( P \)

Let \( A \) be an \( m \times n \) integer matrix and let \( \mathbf{b} \) a vector in \( \mathbb{R}^n \). Let \( U \) be the largest absolute value of the entries in \( A \) and \( \mathbf{b} \).

Every extreme point of the polyhedron \( P = \{x \in \mathbb{R}^n \mid Ax \geq b\} \) satisfies
\[
-(nU)^n \leq x_j \leq (nU)^n, \quad j = 1, \ldots, n
\]

• All extreme points of \( P \) are contained in
\[
P_B = \{x \in P \mid |x_j| \leq (nU)^n, \quad j = 1, \ldots, n\}
\]
• Since \( P_B \subset E(\mathbf{0}, n(nU)^n) \), we can start the ellipsoid method with \( E_0 = E(\mathbf{0}, n(nU)^n) \)
• \[
\text{Vol}(E_0) \leq V = (2n(nU)^n)^n = (2n)^n(nU)^n^2
\]
3.11 Full-dimensionality

Let \( P = \{ x \in \mathbb{R}^n \mid Ax \geq b \} \). We assume that \( A \) and \( b \) have integer entries, which are bounded in absolute value by \( U \). Let

\[
\epsilon = \frac{1}{2(n+1)}((n+1)U)^{-(n+1)}.
\]

Let

\[
P_\epsilon = \{ x \in \mathbb{R}^n \mid Ax \geq b - \epsilon e \},
\]

where \( e = (1, 1, \ldots, 1) \).

(a) If \( P \) is empty, then \( P_\epsilon \) is empty.

(b) If \( P \) is nonempty, then \( P_\epsilon \) is full-dimensional.

Let \( P = \{ x \in \mathbb{R}^n \mid Ax \geq b \} \) be a full-dimensional bounded polyhedron, where the entries of \( A \) and \( b \) are integer and have absolute value bounded by \( U \). Then,

\[
\text{Vol}(P) > v = n^{-n}(nU)^{-n^2(n+1)}
\]

3.12 Complexity

- \( P = \{ x \in \mathbb{R}^n \mid Ax \geq b \} \), where \( A, b \) have integer entries with magnitude bounded by some \( U \) and has full rank. If \( P \) is bounded and either empty or full-dimensional, the ellipsoid method decides if \( P \) is empty in \( O(n \log V/v) \) iterations
- \( v = n^{-n}(nU)^{-n^2(n+1)}, \quad V = (2n)^n(nU)^{n^2} \)
- Number of iterations \( O(n^4 \log(nU)) \)

- If \( P \) is arbitrary, we first form \( P_B \), then perturb \( P_B \) to form \( P_{B,\epsilon} \) and apply the ellipsoid method to \( P_{B,\epsilon} \)
- Number of iterations is \( O(n^6 \log(nU)) \).
- It has been shown that only \( O(n^3 \log U) \) binary digits of precision are needed, and the numbers computed during the algorithm have polynomially bounded size
- The linear programming feasibility problem with integer data can be solved in polynomial time

4 The ellipsoid method for optimization

\[
\begin{align*}
\min & \quad c'x \\
\text{s.t.} & \quad Ax \geq b,
\end{align*}
\]

\[
\begin{align*}
\max & \quad b'\pi \\
\text{s.t.} & \quad A'\pi = c \\
& \quad \pi \geq 0.
\end{align*}
\]

By strong duality, both problems have optimal solutions if and only if the following system of linear inequalities is feasible:

\[
b'p = c'x, \quad Ax \geq b, \quad A'p = c, \quad p \geq 0.
\]

LO with integer data can be solved in polynomial time.
4.1 Sliding objective

- We first run the ellipsoid method to find a feasible solution $x_0 \in P = \{ x \in \mathbb{R}^n \mid Ax \geq b \}$.
- We apply the ellipsoid method to decide whether the set
  $$P \cap \{ x \in \mathbb{R}^n \mid c'x < c'x_0 \}$$
  is empty.
- If it is empty, then $x_0$ is optimal. If it is nonempty, we find a new solution $x_1$ in $P$ with objective function value strictly smaller than $c'x_0$.
- More generally, every time a better feasible solution $x_t$ is found, we take $P \cap \{ x \in \mathbb{R}^n \mid c'x < c'x_t \}$ as the new set of inequalities and reapply the ellipsoid method.

4.2 Performance in practice

- Very slow convergence, close to the worst case
- Contrast with simplex method
- The ellipsoid method is a tool for classifying the complexity of linear programming problems