Linear Programming: The Ellipsoid Algorithm

Michiel Smid*

November 29, 1999

1 Introduction

This chapter is based on the books [1, 2, 4].

Definition 1 (LP) Linear Programming is the following problem: Given an $m \times n$ matrix $A = (a_{ij})$, an m-vector $b = (b_1, \ldots, b_m)^T$, and an n-vector $c = (c_1, \ldots, c_n)^T$, compute an n-vector $x = (x_1, \ldots, x_n)^T$ for which

1. $Ax \leq b$ (component-wise), and
2. that maximizes $c^T x$.

Equivalently, given A, b, and c, compute $\max\{c^T x : Ax \leq b\}$.

There are two special cases. The first case is when there is no x such that $Ax \leq b$. Then

$$\max\{c^T x : Ax \leq b\} = \max \emptyset = -\infty.$$

The second case is when there is no upper bound on $c^T x$ for x satisfying $Ax \leq b$. Then, we have

$$\max\{c^T x : Ax \leq b\} = \infty.$$

The *simplex algorithm*, due to Dantzig (1949), is the best known algorithm for solving LP's. Its performance is very good in practice. In 1972, however, Klee and Minty constructed an input on which the algorithm takes exponential time. More precisely, the number of iterations on this input is exponential in $n + m$. For almost all variants of the simplex algorithm, examples are known that take exponential time. It should be noted that these

*Fakultät für Informatik, Otto-von-Guericke-Universität Magdeburg. E-mail: michiel@latrappe.cs.uni-magdeburg.de.

"expensive" inputs are artificial. On random inputs, the simplex algorithm has (expected) polynomial running time. It is an open problem if there exists a variant of the simplex algorithm that has polynomial running time for all inputs.

After Klee and Minty had shown that the simplex algorithm is not efficient in a theoretical sense, mathematicians and computer scientists started to search for other algorithms solving LP's. For a long time, it was an open problem if a polynomial time algorithm exists. The problem was solved in 1979 by Khachiyan, who designed the *ellipsoid algorithm*. This result was a major breakthrough in mathematical programming, although it is mainly of theoretical interest. The running time of the ellipsoid algorithm, although polynomial, is very high in practice. In fact, for practical instances, it cannot compete with the simplex algorithm.

Khachiyan's algorithm caused great excitement, not only in the world of mathematicians and computer scientists. Newspapers and magazines such as The Guardian, Der Spiegel, and the Nieuwe Rotterdamsche Courant wrote about it. On November 7, 1979, the New York Times had on its front page: *A Soviet Discovery Rocks the World of Mathematics*. For an account of this excitement, see [3].

In 1984, Karmarkar designed a new polynomial time algorithm for linear programming. This algorithm is much faster than the ellipsoid algorithm, and it seems that in practice, it can even compete with the simplex algorithm.

In this chapter, we will consider the ellipsoid algorithm. As mentioned already, its running time is polynomial. What do we mean by this? Consider an LP-instance consisting of an $m \times n$ matrix A, an m-vector b and an n-vector c. The length of this input, counted in bits, is proportional to

$$L := mn + \sum_{i,j} \lceil \log a_{ij} \rceil + \sum_{i} \lceil \log b_i \rceil + \sum_{j} \lceil \log c_j \rceil.$$

The ellipsoid algorithm has a running time that is polynomial in L.

Since we take the length of the input into account, we may assume w.l.o.g. that:

Assumption 1 *A, b, and c have integral entries.*

Moreover, we assume that:

Assumption 2 *Exact arithmetic is available.*

This assumption can be removed, but then the details become much more complicated. Moreover, these details do not give more insight into the nature of the ellipsoid algorithm. Finally, we assume that:

Assumption 3 *Each arithmetic operation takes one time unit.*

The running time of the ellipsoid algorithm depends on the size of the input elements. It is not known if there exists an algorithm for LP that makes a number of arithmetic operations that is polynomial in mn.

2 Ellipsoids

In this section, we recall some results from matrix theory, and introduce the notion of an ellipsoid.

A real symmetric $n \times n$ matrix B is called *positive definite*, if

$$x^T B x > 0 \text{ for all } x \in \mathbb{R}^n,\ x \neq 0.$$

Lemma 1 *Let B be a real symmetric $n \times n$ matrix. The following are equivalent.*

1. *B is positive definite.*
2. *All eigenvalues of B are positive real numbers.*
3. *There is a non-singular real $n \times n$ matrix Q such that $B = QQ^T$.*

A function $T : \mathbb{R}^n \to \mathbb{R}^n$ is called an *affine transformation*, if there is a vector $t \in \mathbb{R}^n$, and a non-singular $n \times n$ matrix Q such that

$$T(x) = t + Qx, \text{ for all } x \in \mathbb{R}^n.$$

The *length* $\|x\|$ of a vector $x \in \mathbb{R}^n$ is defined as

$$\|x\| := \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2} = \sqrt{x^T x}.$$

If $v \in \mathbb{R}^n$ and r is a positive real number, then $S(v, r)$ denotes the *ball* centered at v with radius r, i.e.,

$$S(v, r) := \{x \in \mathbb{R}^n : \|x - v\| \leq r\}.$$

Definition 2 An *ellipsoid* is the image of the unit-ball $S(0, 1)$ under an affine transformation. That is, if $T(x) = t + Qx$ is an affine transformation, then

$$T(S(0, 1)) = \{t + Qx : x \in \mathbb{R}^n \text{ and } x^T x \leq 1\}$$

is an ellipsoid. The vector t is called the *center* of the ellipsoid.

We derive an equivalent characterization. Let $y = t + Qx$. Then $x = Q^{-1}(y - t)$, and the ellipsoid can be rewritten as

$$\begin{aligned}
&\{y \in \mathbb{R}^n : \left(Q^{-1}(y-t)\right)^T \left(Q^{-1}(y-t)\right) \leq 1\} \\
&= \{y \in \mathbb{R}^n : (y-t)^T \left(Q^{-1}\right)^T Q^{-1}(y-t) \leq 1\} \\
&= \{y \in \mathbb{R}^n : (y-t)^T B^{-1}(y-t) \leq 1\},
\end{aligned}$$

where $B := QQ^T$. Since Q is non-singular, B is positive definite.

The converse is also true: If B is positive definite, then there is a non-singular real matrix Q such that $B = QQ^T$. This proves the following lemma.

Lemma 2 *For any $t \in \mathbb{R}^n$, and any positive definite $n \times n$ matrix B, the set*

$$\{x \in \mathbb{R}^n : (x-t)^T B^{-1}(x-t) \leq 1\}$$

is an ellipsoid. Moreover, every ellipsoid can be written in this way.

Let us look at an example. Let $n = 2$ and $B = \begin{pmatrix} 5 & 2 \\ 2 & 8 \end{pmatrix}$. Then B is positive definite, and $B^{-1} = \begin{pmatrix} 2/9 & -1/18 \\ -1/18 & 5/36 \end{pmatrix}$. The corresponding ellipsoid centered at the origin is the set

$$\begin{aligned}
E &= \{x \in \mathbb{R}^2 : x^T B^{-1} x \leq 1\} \\
&= \{x \in \mathbb{R}^2 : \frac{2}{9}x_1^2 - \frac{1}{9}x_1 x_2 + \frac{5}{36}x_2^2 \leq 1\}.
\end{aligned}$$

The definition of ellipsoid shows us more clearly how this set looks like. For that, we have to find the matrix Q for which $B = QQ^T$. Since B is positive definite, we can write $B = U^{-1}\Lambda U$, where Λ is a diagonal matrix containing the (positive!) eigenvalues of B, and U is an *orthogonal* matrix (i.e., $UU^T = I$) containing the normalized eigenvectors. In our case,

$$\begin{aligned}
B &= \begin{pmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \begin{pmatrix} 4 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} 2/\sqrt{5} & -1/\sqrt{5} \\ 1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \\
&= \begin{pmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix} \begin{pmatrix} 2/\sqrt{5} & -1/\sqrt{5} \\ 1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \\
&= QQ^T,
\end{aligned}$$

where

$$Q := \begin{pmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}.$$

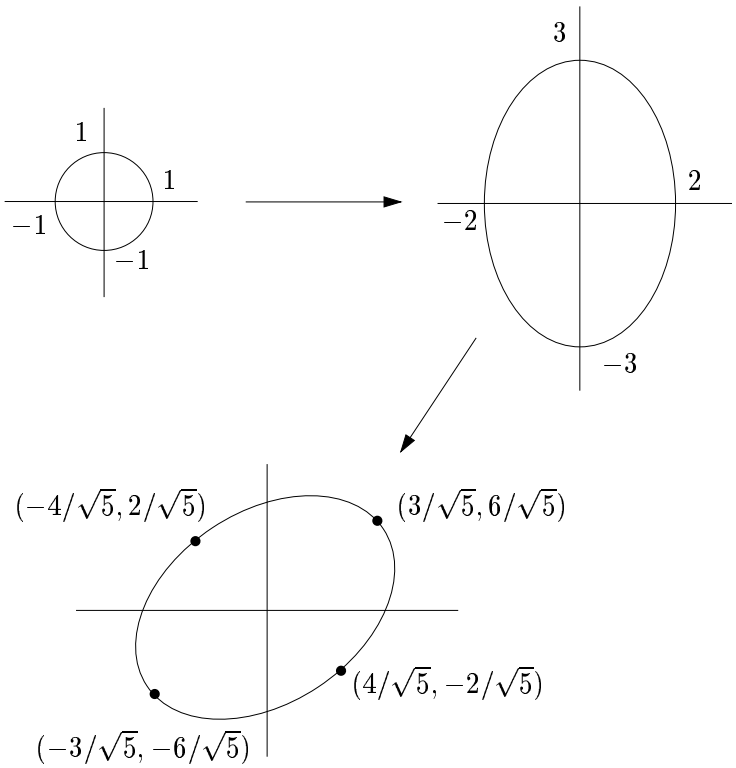

Figure 1: *Applying an affine transformation to the unit-ball gives an ellipsoid.*

Hence, we can write $E = \{Qx : x^T x \leq 1\}$. This ellipsoid is obtained from the unit-ball $S(0,1)$ in $\mathbb{R}^2$, as follows. (See Figure 1.) For each $x \in S(0,1)$, double the first coordinate, and triple the second coordinate. Then rotate the resulting figure using the matrix

$$\begin{pmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix}.$$

The ellipsoid $E' = \{x \in \mathbb{R}^2 : (x-t)^T B^{-1}(x-t) \leq 1\}$ is obtained from E by a translation over t.

We need one more result, whose proof can be found in [5].

Lemma 3 *The volume of the ellipsoid*

$$\{x \in \mathbb{R}^n : (x-t)^T B^{-1}(x-t) \leq 1\}$$

is equal to

$$\sqrt{det(B)} \cdot V_n,$$

where V_n is the volume of the n-dimensional unit-ball $S(0,1)$.

It is known that

$$V_n = \frac{\pi^{n/2}}{\Gamma\left(\frac{n}{2}+1\right)} \sim \frac{1}{\sqrt{\pi n}}\left(\frac{2e\pi}{n}\right)^{n/2},$$

where

$$\Gamma(x) := \int_0^\infty e^{-t}\, t^{\,x-1}\, dt,$$

for $x > 0$, is Euler's *gamma function*. Note that $V_n \to 0$ if $n \to \infty$.

3 The ellipsoid algorithm

The ellipsoid algorithm solves the following problem, called the *linear-strict-inequalities* problem.

Definition 3 (LSI) Given an $m \times n$ matrix A, and an m-vector b, both having integer entries, decide if the set $\{x \in \mathbb{R}^n : Ax < b\}$ is empty or not. Moreover, if this set is non-empty, find any vector x such that $Ax < b$.

This problem looks much easier than the LP problem. However, in Section 4, we will see that the polynomial solvability of LSI implies that the general LP problem can also be solved in polynomial time. In this section, we show how the LSI problem can be solved in polynomial time.

3.1 The idea

Let $F := \{x \in \mathbb{R}^n : Ax < b\}$. Assume for the moment that F is bounded or empty. The ellipsoid algorithm starts by computing a ball—which is an ellipsoid—that is centered at the origin and that contains F. Given this ball, a sequence of ellipsoids is constructed, each containing F, and whose volumes decrease.

Consider one iteration of the construction of this sequence, and let $E = \{x : (x-t)^T B^{-1}(x-t) \leq 1\}$ be the ellipsoid at the start of this iteration. Note that E contains F.

First, we check if the center t of E is contained in F. That is, we check if $At < b$. If so, we output t, and the algorithm terminates. Assume that $t \notin F$. (See Figure 2.) Then we have found an i, $1 \leq i \leq m$, such that $A^i t \geq b_i$, where A^i denotes the i-th row of A. Consider the hyperplane

$$h := \{x \in \mathbb{R}^n : A^i x = A^i t\}.$$

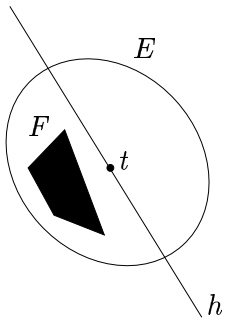

Figure 2: *The center t of the ellipsoid E is not contained in F. F is contained in the part of E that is below the hyperplane h.*

We claim that F is contained in the half-ellipsoid "below" h. Indeed, every vector x on or "above" h satisfies $A^i x \geq A^i t \geq b_i$ and, therefore, does not belong to F.

The next ellipsoid E' in our sequence contains the "lower" half of E and has a volume that is smaller than that of E. Note that E' contains F.

All ellipsoids obtained in this way contain F, and they shrink in volume. If the center t of any of these ellipsoids satisfies $At < b$, then we output t, and the algorithm terminates.

Of course, a problem arises if F is empty. Then none of the centers t satisfies $At < b$, and we would iterate forever. It turns out, however, that there is a real number $v > 0$, which depends on the entries of A and b, such that either F is empty, or the volume of F is at least equal to v. Therefore, if the current ellipsoid has volume less than v, we can conclude that F is empty, and terminate the algorithm.

Until now, we have assumed that F is bounded. If this is not the case, the algorithm still works: We require that a "large" part of F is contained in each ellipsoid of the sequence.

This was a brief sketch of the ellipsoid algorithm. To fill in the details, at least three issues have to be resolved.

1. How to construct the initial ellipsoid? This question will be answered in Section 3.3.

2. How to construct the ellipsoid E' from E? Recall that E' must contain a "large" part of F, and its volume must be smaller than that of E. This question will be answered in Section 3.2.

3. How many iterations do we have to make? That is, after how many iterations can we conclude that F is empty? Note that in order to

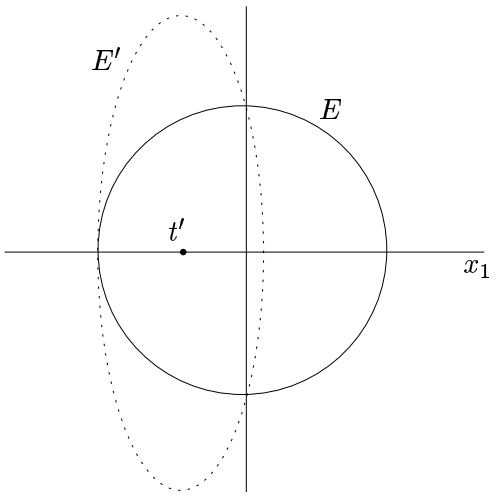

Figure 3: *E' is the smallest ellipsoid that contains the left part of E.*

---

have a "small" number of iterations, each next ellipsoid E' should be "much" smaller than E. This question will be answered in Section 3.3.

3.2 Constructing the next ellipsoid

We first show how to construct the next ellipsoid in our sequence for a special case.

Let E be the n-dimensional unit-ball, i.e., $E = S(0,1)$. Note that E is indeed an ellipsoid, because it can be written as $E = \{x \in \mathbb{R}^n : x^T I^{-1} x \leq 1\}$. Assume that the matrix A has $A^1 = (1, 0, 0, \ldots, 0)$ as its first row, and that b has the form $b = (-1, b_2, b_3, \ldots, b_n)^T$.

The center $t = 0$ of E satisfies $A^1 t = 0 \geq -1 = b_1$. Therefore, $t \notin F$. Let h^- be the halfspace $\{x \in \mathbb{R}^n : A^1 x \leq A^1 t\}$, i.e., $h^- = \{x \in \mathbb{R}^n : x_1 \leq 0\}$. Then we have to find a small ellipsoid E' that contains the "left" part of E. Let us denote this left part by HS:

$$HS := E \cap h^- = \{x \in \mathbb{R}^n : x^T x \leq 1 \text{ and } x_1 \leq 0\}.$$

We use our geometric intuition to derive the smallest ellipsoid E' that contains HS. Afterwards, we will prove that the ellipsoid we get is indeed the right one.

Intuitively, the center t' of E' should be on the negative x_1-axis. (See Figure 3.) Moreover, E' should be symmetric w.r.t. the x_1-axis. Hence, we

expect that E' has the form

$$E' = \{x \in \mathbb{R}^n : a^2(x_1 - t'_1)^2 + \sum_{i=2}^{n} b^2 x_i^2 \leq 1\}, \tag{1}$$

where a, b, and t'_1 are to be determined by us. We may assume without loss of generality that $a > 0$ and $b > 0$. Moreover, our intuition says that $-1 < t'_1 < 0$. We can write E' as

$$E' = \{x \in \mathbb{R}^n : (x - t')^T (B')^{-1}(x - t') \leq 1\},$$

where

$$t' = (t'_1, 0, 0, \ldots, 0)^T$$

and

$$B' = diag(1/a^2, 1/b^2, 1/b^2, \ldots, 1/b^2).$$

Again intuitively, if E' is the smallest ellipsoid containing HS, then the vector $(-1, 0, 0, \ldots, 0)$ should be on the boundary of E'. Substituting this vector into (1) gives

$$a^2(1 + t'_1)^2 = 1. \tag{2}$$

Similarly, vectors $(0, x_2, x_3, \ldots, x_n)$ with $\sum_{i=2}^{n} x_i^2 = 1$ should be on the boundary of E'. This yields

$$a^2(t'_1)^2 + b^2 = 1. \tag{3}$$

Equations (2) and (3) give

$$a^2 = \frac{1}{(1 + t'_1)^2} \quad \text{and} \quad b^2 = \frac{1 + 2t'_1}{(1 + t'_1)^2}. \tag{4}$$

Hence, we need one more equation to determine E'. We know from Lemma 3 that

$$\begin{aligned}
\frac{vol(E')}{vol(S(0,1))} &= \sqrt{det(B')} \\
&= \sqrt{\frac{1}{a^2}\left(\frac{1}{b^2}\right)^{n-1}} \\
&= \frac{1}{a \cdot b^{n-1}} \\
&= (1 + t'_1)\left(\frac{1 + t'_1}{\sqrt{1 + 2t'_1}}\right)^{n-1} \\
&= \frac{(1 + t'_1)^n}{(1 + 2t'_1)^{(n-1)/2}}.
\end{aligned} \tag{5}$$

Since we want E' to be as small as possible, we choose t'_1, $-1 < t'_1 < 0$, such that the quotient (5) is minimum. Elementary calculus shows that this is the case when $t'_1 = -1/(n+1)$. Then (4) implies that $a = (n+1)/n$ and $b = \sqrt{n^2-1}/n$.

What have we learned? We have derived a *candidate* ellipsoid

$$E' = \{x \in \mathbb{R}^n : (x - t')^T (B')^{-1} (x - t') \leq 1\}, \tag{6}$$

where

$$t' = \left(\frac{-1}{n+1}, 0, 0, \ldots, 0\right)^T$$

and

$$B' = diag\left(\frac{n^2}{(n+1)^2}, \frac{n^2}{n^2-1}, \frac{n^2}{n^2-1}, \ldots, \frac{n^2}{n^2-1}\right).$$

It remains to show that this ellipsoid is a good one.

Lemma 4 *The set E' in (6) satisfies:*

1. *B' is positive definite, so that E' is an ellipsoid.*
2. *E' contains the half-ball*

$$HS = \{x \in \mathbb{R}^n : x^T x \leq 1 \text{ and } x_1 \leq 0\}.$$

3. *$vol(E')/vol(S(0,1)) < e^{-1/(2(n+1))}$.*

Proof. To prove the first claim, let

$$Q := diag\left(\frac{n}{n+1}, \frac{n}{\sqrt{n^2-1}}, \frac{n}{\sqrt{n^2-1}}, \ldots, \frac{n}{\sqrt{n^2-1}}\right).$$

Then $B' = QQ^T$ and therefore, by Lemma 1, B' is positive definite.

To prove the second claim, let $x \in HS$. We have to show that $x \in E'$, i.e., $(x - t')^T (B')^{-1} (x - t') \leq 1$. This follows from a straightforward calculation:

$$\begin{aligned}
&(x - t')^T (B')^{-1} (x - t') \\
&= \left(x_1 + \frac{1}{n+1}, x_2, \ldots, x_n\right) \cdot \\
&\qquad diag\left(\frac{(n+1)^2}{n^2}, \frac{n^2-1}{n^2}, \ldots, \frac{n^2-1}{n^2}\right) \begin{pmatrix} x_1 + \frac{1}{n+1} \\ x_2 \\ \vdots \\ x_n \end{pmatrix}
\end{aligned}$$

$$
\begin{aligned}
&= \left(x_1 + \frac{1}{n+1}, x_2, \ldots, x_n\right) \cdot \begin{pmatrix} \frac{(n+1)^2}{n^2}\left(x_1 + \frac{1}{n+1}\right) \\ \frac{n^2-1}{n^2}\, x_2 \\ \vdots \\ \frac{n^2-1}{n^2}\, x_n \end{pmatrix} \\
&= \frac{(n+1)^2}{n^2}\left(x_1 + \frac{1}{n+1}\right)^2 + \sum_{i=2}^{n} \frac{n^2-1}{n^2}\, x_i^2 \\
&= \frac{(n+1)^2}{n^2}\left(x_1 + \frac{1}{n+1}\right)^2 - \frac{n^2-1}{n^2}\, x_1^2 + \sum_{i=1}^{n} \frac{n^2-1}{n^2}\, x_i^2 \\
&= \frac{2n+2}{n^2}\, x_1^2 + \frac{2n+2}{n^2}\, x_1 + \frac{1}{n^2} + \frac{n^2-1}{n^2}\, x^T x.
\end{aligned}
$$

Since $x^T x \leq 1$, it follows that

$$
\begin{aligned}
(x - t')^T (B')^{-1} (x - t') &\leq \frac{2n+2}{n^2}\left(x_1^2 + x_1\right) + \frac{1}{n^2} + \frac{n^2-1}{n^2} \\
&= \frac{2n+2}{n^2}\left(x_1^2 + x_1\right) + 1.
\end{aligned}
$$

The inequality $x^T x \leq 1$ also implies that $x_1 \geq -1$. This, together with the fact that $x_1 \leq 0$, yields

$$
(x - t')^T (B')^{-1} (x - t') \leq \frac{2n+2}{n^2}\, x_1(x_1 + 1) + 1 \leq 1,
$$

which is exactly what we wanted to show.

It remains to prove the third claim. We know from Lemma 3 that

$$
\frac{vol(E')}{vol(S(0,1))} = \sqrt{det(B')} = \frac{n}{n+1}\left(\frac{n^2}{n^2-1}\right)^{(n-1)/2}.
$$

We know from calculus that $1 + y < e^y$ for all $y \neq 0$. Therefore,

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

and

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

Hence,

$$
\begin{aligned}
\frac{vol(E')}{vol(S(0,1))} &< e^{-1/(n+1)}\left(e^{1/(n^2-1)}\right)^{(n-1)/2} \\
&= e^{-1/(n+1)} \cdot e^{1/(2(n+1))} \\
&= e^{-1/(2(n+1))}.
\end{aligned}
$$

This completes the proof. ∎

We have shown how to construct the ellipsoid E' from E, for the special case when (i) E is the unit-ball, (ii) the matrix A has $A^1 = (1, 0, 0, \ldots, 0)$ as its first row, and (iii) b has the form $b = (-1, b_2, b_3, \ldots, b_n)^T$. The general case can be reduced to this special case: Let

$$E = \{x \in \mathbb{R}^n : (x - t)^T B^{-1}(x - t) \leq 1\}$$

be the ellipsoid, and let i, $1 \leq i \leq m$, be such that $A^i t \geq b_i$. Hence, we want to find a small ellipsoid E' that contains the "lower" half of E. That is, we want a small ellipsoid E' such that

$$E \cap \{x \in \mathbb{R}^n : A^i x \leq A^i t\} \subseteq E'.$$

We obtain E' as follows.

1. Using an affine transformation T, map E onto $S(0, 1)$ and $\{x \in \mathbb{R}^n : A^i x \leq A^i t\}$ onto $\{x \in \mathbb{R}^n : x_1 \leq 0\}$.

2. We are now in the special case that was treated above. So let E'_0 be the corresponding ellipsoid. (Above, this ellipsoid was denoted by E'.)

3. Apply the inverse of the affine transformation T. This maps E'_0 onto the ellipsoid E' we are looking for.

This gives the following result. For a proof, which is technical, see [2].

Lemma 5 *Let a be the n-vector $(A^i)^T$, let t' be the n-vector*

$$t' := t - \frac{1}{n+1} \frac{Ba}{\sqrt{a^T Ba}},$$

let B' be the $n \times n$ matrix

$$B' := \frac{n^2}{n^2 - 1} \left[B - \frac{2}{n+1} \frac{(Ba)(Ba)^T}{a^T Ba} \right],$$

and let

$$E' := \{x \in \mathbb{R}^n : (x - t')^T (B')^{-1} (x - t') \leq 1\}.$$

Then the following holds.

1. *B' is positive definite, so that E' is an ellipsoid. In particular, since $a \neq 0$[1], we have $a^T Ba > 0$.*

[1] it does not make sense to have a row in A all whose entries are zero

2. *E' contains the half-ellipsoid*

$$E \cap \{x \in \mathbb{R}^n : A^i x \leq A^i t\}.$$

3. *$vol(E')/vol(E) < e^{-1/(2(n+1))}$.*

Exercise 1 Let E be the unit-ball, let the matrix A have $A^1 = (1, 0, 0, \ldots, 0)$ as its first row, and let b have the form $b = (-1, b_2, b_3, \ldots, b_n)^T$. Check that the ellipsoid E' of Lemma 5 is indeed the same as that of Lemma 4.

We apply Lemma 5 to a specific example. Let $n = 2$, $B = \begin{pmatrix} 5 & 2 \\ 2 & 8 \end{pmatrix}$, $t = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $b_1 = -1$, and let A have $(1, -1)$ as its first row. Then

$$A^1 t = (1, -1) \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0 \geq b_1.$$

Hence, the center t is not contained in the set $\{x \in \mathbb{R}^2 : Ax < b\}$. In this case,

$$Ba = B\left(A^1\right)^T = \begin{pmatrix} 5 & 2 \\ 2 & 8 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 3 \\ -6 \end{pmatrix},$$

and

$$a^T Ba = (1, -1) \begin{pmatrix} 3 \\ -6 \end{pmatrix} = 9.$$

Hence,

$$t' = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - \frac{1}{3} \cdot \frac{1}{3} \begin{pmatrix} 3 \\ -6 \end{pmatrix} = \begin{pmatrix} -1/3 \\ 2/3 \end{pmatrix},$$

and

$$\begin{aligned} B' &= \frac{4}{3}\left[\begin{pmatrix} 5 & 2 \\ 2 & 8 \end{pmatrix} - \frac{2}{3} \cdot \frac{\begin{pmatrix} 3 \\ -6 \end{pmatrix}(3, -6)}{9}\right] \\ &= \begin{pmatrix} 52/9 & 40/9 \\ 40/9 & 64/9 \end{pmatrix}. \end{aligned}$$

Consider the hyperplane (which is in fact a line)

$$h = \{x \in \mathbb{R}^2 : A^1 x = A^1 t\} = \{x \in \mathbb{R}^2 : x_1 - x_2 = 0\}.$$

The ellipsoid E' defined by B' and t' contains the part of the ellipsoid defined by B and t that is "below" h, i.e.,

$$E \cap \{x \in \mathbb{R}^2 : A^1 x \leq A^1 t\} = E \cap \{x \in \mathbb{R}^2 : x_1 \leq x_2\} \subseteq E'.$$

Note that in this case, "below" does not mean below in the usual sense.

Ellipsoid Algorithm
$l := 0$; $K := 16n(n+1)L$;
$t_l := 0$; (* the zero-vector of length n *)
$B_l := n^2 \cdot 2^{2L} I_n$; (* I_n is the $n \times n$ identity matrix *)
for $l := 0$ **to** $K - 1$
do (* B_l is a positive definite $n \times n$ matrix; the l-th ellipsoid is
$E_l = \{x \in \mathbb{R}^n : (x - t_l)^T B_l^{-1}(x - t_l) \leq 1\}$ *)
if $At_l < b$
then output t_l, and terminate
else (* construct a new ellipsoid *)
let i be such that $A^i t_l \geq b_i$;
$a := (A^i)^T$;
$t_{l+1} := t_l - \frac{1}{n+1} \frac{B_l a}{\sqrt{a^T B_l a}}$;
$B_{l+1} := \frac{n^2}{n^2-1} \left[B_l - \frac{2}{n+1} \frac{(B_l a)(B_l a)^T}{a^T B_l a} \right]$
endif
endfor;
output "F is empty"

Figure 4: *The ellipsoid algorithm.*

3.3 The algorithm

Let A be an $m \times n$ matrix, and b an m-vector, both having integer entries. We want to decide if the set $F = \{x \in \mathbb{R}^n : Ax < b\}$ is empty or not. Let

$$L := mn + \sum_{i,j} \lceil \log a_{ij} \rceil + \sum_i \lceil \log b_i \rceil.$$

Then L is proportional to the length of the input.

Before we give the complete algorithm, let us look at the first ellipsoid in our sequence of ellipsoids. Recall that we require that a large part of the region F is contained in each ellipsoid. Let E_0 be the ball centered at the origin and having radius $n \cdot 2^L$. The proof of the following lemma can be found in [2, page 97].

Lemma 6 *If F is non-empty, then $F \cap E_0$ has volume at least $v := 2^{-(n+2)L}$.*

The algorithm that decides if F is empty or not is given in Figure 4.

Exercise 2 Convince yourself that in the algorithm of Figure 4, the initial ellipsoid E_0 is indeed equal to $S(0, n \cdot 2^L)$.

Let us analyze the running time of our algorithm. The initialization can be carried out in $O(L + n^2)$ time. Since we assume that exact arithmetic is available, each iteration of the for-loop takes $O(mn + n^2)$ time. Hence, the total running time of the algorithm is bounded by

$$O\left(L + n^2 + K(mn + n^2)\right) = O\left(n^2 L(mn + n^2)\right) = O\left(L^5\right).$$

Hence, the running time of the ellipsoid algorithm is polynomial in the length of the input. Of course, we still have to prove that this algorithm is correct.

Theorem 1 *The ellipsoid algorithm correctly solves the LSI problem.*

Proof. If the algorithm reports that F is non-empty, then it also gives an element $t_l \in F$. Therefore, in this case, F really is non-empty. It remains to prove the converse. That is, we have to show that if F is non-empty, then the algorithm reports this.

Assume that F is non-empty, but the algorithm reports that it is empty. We will derive a contradiction. First note that the for-loop makes K iterations. We know from Lemma 6 that

$$vol(F \cap E_0) \geq 2^{-(n+2)L}. \tag{7}$$

We claim that $F \cap E_0 \subseteq E_l$ for all l, $0 \leq l \leq K$. This is clearly true for $l = 0$. Let $0 \leq l < K$, and assume that $F \cap E_0 \subseteq E_l$. Consider the index i in the l-th iteration. By Lemma 5, we have

$$E_l \cap \{x \in \mathbb{R}^n : A^i x \leq A^i t_l\} \subseteq E_{l+1}.$$

Our construction of E_{l+1} guarantees that (see the discussion in Section 3.1)

$$F \cap E_0 \subseteq \{x \in \mathbb{R}^n : A^i x \leq A^i t_l\},$$

which implies that

$$F \cap E_0 \subseteq E_l \cap \{x \in \mathbb{R}^n : A^i x \leq A^i t_l\}.$$

Therefore, we have $F \cap E_0 \subseteq E_{l+1}$, proving the claim.

In particular, since $F \cap E_0 \subseteq E_K$, we get

$$vol(F \cap E_0) \leq vol(E_K). \tag{8}$$

By Lemma 5, we have

$$\begin{aligned} vol(E_K) &< e^{-1/(2(n+1))}\, vol(E_{K-1}) \\ &< e^{-2/(2(n+1))}\, vol(E_{K-2}) \\ &< e^{-3/(2(n+1))}\, vol(E_{K-3}) \\ &\vdots \\ &< e^{-K/(2(n+1))}\, vol(E_0). \end{aligned}$$

Since

$$E_0 = S(0, n \cdot 2^L) \subseteq \{x \in \mathbb{R}^n : |x_i| \leq n \cdot 2^L, 1 \leq i \leq n\},$$

we have

$$vol(E_0) \leq \left(2n \cdot 2^L\right)^n.$$

It follows that

$$\begin{aligned} vol(E_K) &< e^{-K/(2(n+1))} \left(2n \cdot 2^L\right)^n \qquad (* 2n \leq 2^L *) \\ &\leq e^{-K/(2(n+1))}\, 2^{2nL} \qquad (* K = 16n(n+1)L *) \\ &= e^{-8nL}\, 2^{2nL} \\ &< 2^{-8nL}\, 2^{2nL} \\ &= 2^{-6nL} \\ &< 2^{-(n+2)L}. \end{aligned}$$

Hence, by (8), we get

$$vol(F \cap E_0) < 2^{-(n+2)L},$$

which contradicts (7). This completes the proof. ∎

4 Solving the general linear programming problem

At this moment, we know that any LSI problem can be solved in polynomial time. In this section, we show that this result can be used to solve general linear programs in polynomial time. First, we consider the *linear-inequalities* problem.

Definition 4 (LI) Given an $m \times n$ matrix A, and an m-vector b, both having integer entries, decide if the set $\{x \in \mathbb{R}^n : Ax \leq b\}$ is empty or not. Moreover, if this set is non-empty, find any vector x such that $Ax \leq b$.

Note that Lemma 6 does not necessarily hold for the set $F := \{x \in \mathbb{R}^n : Ax \leq b\}$. Take e.g. $m = 3$, $n = 2$, $b = (0, 0, 0)^T$, and

$$A = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{pmatrix}.$$

Then, $F = \{(0, 0)\}$, i.e., F is non-empty. The volume of F, however, is equal to zero.

As before, let

$$L := mn + \sum_{i,j} \lceil \log a_{ij} \rceil + \sum_{i} \lceil \log b_i \rceil,$$

which is the length of the input. A proof of the following lemma can be found in [2, page 76].

Lemma 7 *Let $\epsilon := 2^{-2L}$ and $b'_i := b_i + \epsilon$, $1 \leq i \leq m$. The set $F := \{x \in \mathbb{R}^n : Ax \leq b\}$ is non-empty if and only if the set $F' := \{x \in \mathbb{R}^n : Ax < b'\}$ is non-empty. Moreover, in polynomial time, any element of F' can be transformed into an element of F.*

This lemma of course implies that any LI problem can be solved in polynomial time.

The final step of our proof uses the *duality theorem*, due to von Neumann (1947). This theorem is one of the most important results in mathematical programming. For any LP $\max\{c^T x : Ax \leq b\}$, the *dual program* is defined as the linear programming problem $\min\{b^T y : A^T y = c, y \geq 0\}$. We denote the dual program by LD. A proof of the duality theorem can be found in [2].

Theorem 2 (Duality Theorem) *Let $F_p := \{x \in \mathbb{R}^n : Ax \leq b\}$, and $F_d := \{y \in \mathbb{R}^m : A^T y = c, y \geq 0\}$.*

1. *If F_p and F_d are both non-empty, then*

$$\max\{c^T x : Ax \leq b\} = \min\{b^T y : A^T y = c, y \geq 0\}.$$

2. *If $F_p = \emptyset$, then either $F_d = \emptyset$ or LD is unbounded.*

3. *If $F_d = \emptyset$, then either $F_p = \emptyset$ or LP is unbounded.*

4. *If LP is unbounded, then $F_d = \emptyset$.*

5. *If LD is unbounded, then $F_p = \emptyset$.*

For positive integers a and b, we denote by O_{ab} the $a \times b$ matrix having zero entries, and by I_a the $a \times a$ identity matrix.

The following algorithm solves a general LP.

Step 1: Test if the set $F := \{x \in \mathbb{R}^n : Ax \leq b\}$ is empty. If it is, output "F is empty", and terminate. Otherwise, go to Step 2.

Step 2: Let A' be the $(2m + 2n + 2) \times (m + n)$ matrix

$$A' := \begin{pmatrix} A & O_{mm} \\ O_{nn} & A^T \\ O_{nn} & -A^T \\ O_{mn} & -I_m \\ c^T & -b^T \\ -c^T & b^T \end{pmatrix},$$

and b' the $(2m + 2n + 2)$-vector

$$b' := (b^T, c^T, -c^T, 0, 0, \ldots, 0)^T.$$

Test if the set $F' := \{(x, y) \in \mathbb{R}^{m+n} : A'\binom{x}{y} \leq b'\}$ is empty.

If F' is empty, then output "$\max\{c^T x : Ax \leq b\} = \infty$". Otherwise, the algorithm has found an element (x_0, y_0) in F'. In this case, output "$\max\{c^T x : Ax \leq b\} = c^T x_0$".

Theorem 3 *This algorithm solves any LP problem in polynomial time.*

Proof. Since any LI problem can be solved in polynomial time, the given algorithm has polynomial running time. So it remains to prove that the algorithm is correct. It is clear that the algorithm reports that F is empty if and only if F really is empty.

Assume that F is non-empty. Also, assume that $\max\{c^T x : Ax \leq b\}$ is finite. We claim that then (i) F' is non-empty, and our (ii) algorithm computes this maximum.

To prove (i), let x_0 be such that

$$c^T x_0 = \max\{c^T x : Ax \leq b\}.$$

By the duality theorem, there is a vector y_0 such that

$$b^T y_0 = \min\{b^T y : A^T y = c, y \geq 0\}.$$

We have

$$Ax_0 \leq b, A^T y_0 = c, y_0 \geq 0, \text{ and } c^T x_0 = b^T y_0, \tag{9}$$

or equivalently,

$$Ax_0 \leq b, A^T y_0 \leq c, \left(-A^T\right) y_0 \leq c, -y_0 \leq 0, \tag{10}$$

$$c^T x_0 - b^T y_0 \leq 0, \text{ and } -c^T x_0 + b^T y_0 \leq 0. \tag{11}$$

Hence, $A'\binom{x_0}{y_0} \leq b'$ and, therefore, F' is non-empty, proving (i).

To prove (ii), let (x_0, y_0) be the element of F' that is found by the algorithm. Then (10) and (11) hold for x_0 and y_0. Hence, also (9) holds. We have

$$\begin{aligned} c^T x_0 &\leq \max\{c^T x : Ax \leq b\} \\ &= \min\{b^T y : A^T y = c, y \geq 0\} \\ &\leq b^T y_0 \\ &= c^T x_0. \end{aligned}$$

It follows that all inequalities in this chain are in fact equalities. That is, $c^T x_0 = \max\{c^T x : Ax \leq b\}$, proving (ii).

The proof can now easily be completed using the duality theorem. ∎

This concludes the chapter on linear programming. All results that were shown here remain valid for computation models that have finite precision arithmetic. The details, however, are technical. For details, refer to the books mentioned below.

References

[1] M. Grötschel, L. Lovász and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, 1988.

[2] H. Karloff. *Linear Programming.* Birkhäuser, 1991.

[3] E.L. Lawler. *The great mathematical Sputnik of 1979.* The Mathematical Intelligencer **2** (1980), pp. 191–198.

[4] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, 1982.

[5] W. Rudin. *Real and Complex Analysis, third edition.* McGraw-Hill, 1987, pp. 54–55.