

## Lecture 5

Lecturer: Michel X. Goemans

Scribe: Dennis Quan

## 1 The Ellipsoid Algorithm

**Definition 1** Let  $a$  be a point in  $\mathbb{R}^n$  and  $A$  be an  $n \times n$  positive definite matrix (i.e.,  $A$  has positive eigenvalues). The ellipsoid  $E(a, A)$  with center  $a$  is the set of points  $\{x : (x - a)^T A^{-1} (x - a) \leq 1\}$ . Therefore, the unit sphere is  $E(0, I)$ , where  $I$  is the identity matrix.

An ellipsoid can be seen as the result of applying a linear transformation on a unit sphere. In other words, there is a linear transformation  $T$  that maps  $E(a, A)$  to the unit sphere  $E(0, I)$ . It is known that for every positive definite matrix  $A$ , there is a  $n \times n$  matrix  $B$  such that:

$$A = B^T B. \quad (1)$$

Therefore,

$$A^{-1} = B^{-1} (B^{-1})^T. \quad (2)$$

Using  $B$ , the transformation  $T$  can be seen as mapping points  $x$  to  $(B^{-1})^T (x - a)$ .

The Ellipsoid Algorithm solves the problem of finding an  $x$  subject to  $Cx \leq d$  by looking at successively smaller ellipsoids  $E_k$  that contain the polyhedron  $P := \{x : Cx \leq d\}$ . Starting with an initial ellipsoid that contains  $P$ , we check to see if its center  $a$  is in  $P$ . If it is, we are done. If not, we look at the inequalities defining  $P$ , and choose one that is violated by  $a$ . This gives us a hyperplane through  $a$  such that  $P$  is completely on one side of this hyperplane. Then, we try to find an ellipsoid  $E_{k+1}$  that contains the *half-ellipsoid* defined by  $E_k$  and  $h$ .

The general step of finding the next ellipsoid  $E_{k+1}$  from  $E_k$  is given below. First we assume that  $E_k$  is a unit sphere centered at the origin, and the hyperplane  $h$  defines the half space  $-e_1^T x \leq 0$  that contains  $P$ . Here, by  $e_i$  we mean the vector whose  $i$ th component is 1 and whose other components are 0. We will show later that it is easy to translate the general case to this case.

Therefore, we need an ellipsoid that contains

$$E(0, I) \cap \{x : -e_1^T x \leq 0\} \quad (3)$$

To find an ellipsoid that contains  $E_k$ , we showed last time that:

$$\underbrace{\left\{ x : \left( \frac{n-1}{n} \right)^2 \left( x_1 - \frac{1}{n+1} \right)^2 + \frac{n^2-1}{n^2} \sum_{i=2}^n x_i^2 \leq 1 \right\}}_{E_{k+1}} \subseteq E(0, I) \cap \{x : x_1 \geq 0\} \quad (4)$$

Therefore, we can define

$$E_{k+1} = E \left( \frac{1}{n+1} e_1, \frac{n^2}{n^2-1} \left( I - \frac{2}{n+1} e_1 e_1^T \right) \right). \quad (5)$$

( $e_1 e_1^T$  = matrix with 1 in its top left cell, 0 elsewhere.) We also showed that

$$\text{Vol}(E_{k+1})/\text{Vol}(E_k) \leq \frac{n^2}{n^2-1} \frac{n}{n+1} \leq \exp\left(-\frac{1}{2n}\right) \quad (6)$$

For the more general case that we want to find an ellipsoid that contains  $E(0, I) \cap \{x : d^T x \leq 0\}$  (we let  $\|d\| = 1$ ; this can be done because the other side of the inequality is 0), it is easy to verify that we can take  $E_{k+1} = E(-\frac{1}{n+1}d, F)$ , where  $F = \frac{n^2}{n^2-1}(I - \frac{2}{n+1}dd^T)$ , and the ratio of the volumes is  $\leq \exp(-\frac{1}{2n})$ .

Now we deal with the case where  $E_k$  is not the unit sphere. We take advantage of the fact that linear transformations preserve ratios of volumes.

$$\begin{array}{ccc} E_k & \xrightarrow{T} & E(0, 1) \\ & & \downarrow \\ E_{k+1} & \xleftarrow{T^{-1}} & E' \end{array} \quad (7)$$

Let  $a_k$  be the center of  $E_k$ , and  $c^T x \leq c^T a_k$  be the halfspace through  $a_k$  that contains  $P$ . Therefore, the half-ellipsoid that we are trying to contain is  $E(a_k, A) \cap \{x : c^T x \leq c^T a_k\}$ . Let's see what happens to this half-ellipsoid after the transformation  $T$  defined by  $T(x) = (B^{-1})^T(x - a)$ . This transformation transforms  $E_k = E(a_k, A)$  to  $E(0, I)$ . Also,

$$\{x : c^T x \leq c^T a_k\} \xrightarrow{T} \{x : c^T(a_k + B^T y) \leq c^T a_k\} = \{x : c^T B^T y \leq 0\} = \{x : d^T x \leq 0\}, \quad (8)$$

where  $d$  is given by the following equation.

$$d = \frac{BC}{\sqrt{c^T B^T B c}} = \frac{BC}{\sqrt{c^T A c}} \quad (9)$$

Let  $b = B^T d = \frac{Ac}{\sqrt{c^T A c}}$ . This implies:

$$E_{k+1} = E\left(a_k - \frac{1}{n+1}b, \frac{n^2}{n^2-1}B^T\left(I - \frac{2}{n+1}dd^T\right)B\right) \quad (10)$$

$$= E\left(a_k - \frac{1}{n+1}b, \frac{n^2}{n^2-1}\left(A - \frac{2}{n+1}bb^T\right)\right) \quad (11)$$

To summarize, here is the Ellipsoid Algorithm:

1. Start with  $k = 0$ ,  $E_0 = E(a_0, A_0) \supseteq P$ ,  $P = \{x : Cx \leq d\}$ .
2. While  $a_k \notin P$  do:
  - Let  $c^T x \leq d$  be an inequality that is valid for all  $x \in P$  but  $c^T a_k > d$ .
  - Let  $b = \frac{A_k c}{\sqrt{c^T A_k c}}$ .
  - Let  $a_{k+1} = a_k - \frac{1}{n+1}b$ .
  - Let  $A_{k+1} = \frac{n^2}{n^2-1}(A_k - \frac{2}{n+1}bb^T)$ .

**Claim 1**  $\frac{\text{Vol}(E_{k+1})}{\text{Vol}(E_k)} \leq \exp\left(-\frac{1}{2n}\right)$

After  $k$  iterations,  $Vol(E_k) \leq Vol(E_0) \exp\left(-\frac{k}{2n}\right)$ . If  $P$  is nonempty then the Ellipsoid Algorithm should find  $x \in P$  in at most  $2n \ln \frac{Vol(E_0)}{Vol(P)}$  steps.

What if  $P$  has volume 0 but is nonempty? In this case, we create an inflated polytope around  $P$  such that this new polytope is empty iff  $P$  is empty.

**Theorem 2** *Let  $P := \{x : Ax \leq b\}$  and  $e$  be the vector of all ones. Assume that  $A$  has full column rank (certainly true if  $Ax \leq b$  contains the inequalities  $-Ix \leq 0$ ). Then  $P$  is nonempty iff  $P' = \{x : Ax \leq b + \frac{1}{2^L}e, -2^L \leq x_j \leq 2^L \text{ for all } j\}$  is nonempty. ( $L$  is the size of the LP  $P$ , as we defined in the previous lecture, but here we can remove the  $c_{max}$  term.)*

This theorem allows us to choose  $E_0$  to be a ball centered at the origin containing the cube  $[-2^L, 2^L]^n$ . In this way, if there exists a  $\hat{x}$  such that  $A\hat{x} \leq b$  then

$$\hat{x} + \left[-\frac{1}{2^{2L}}, \frac{1}{2^{2L}}\right]^n \in P' \quad (12)$$

Indeed, for a  $x$  in this little cube, we have  $(Ax)_j \leq (A\hat{x})_j + (\max_{i,j} a_{ij})n\frac{1}{2^{2L}} \leq b_j + \frac{1}{2^L}$ .

The time for finding an  $x$  in  $P'$  is in  $O(n \cdot nL)$ , because the ratio of the volumes of  $[-2^L, 2^L]^n$  to  $[-\frac{1}{4^L}, \frac{1}{4^L}]^n$  is  $8^{Ln}$ , and previously we showed that finding  $x$  in  $P$  was  $O(n \ln \frac{Vol(E_0)}{Vol(P)})$ . Thus, this process is polynomial in  $L$ .

**Proof of Theorem 2:** We first prove the forward implication. If  $Ax \leq b$  is nonempty then we can consider a vertex  $x$  in  $P$  (and there exists a vertex since  $A$  has full column rank). This implies that  $x$  will be defined by  $A_S x = b_S$ , where  $A_S$  is a submatrix of  $A$  (by problem 1 in Problem Set 1). Therefore, by a theorem from the previous lecture,

$$x = \left(\frac{p_1}{q}, \frac{p_2}{q}, \dots, \frac{p_n}{q}\right) \quad (13)$$

with  $|p_i| < 2^L$  and  $1 \leq q < 2^L$ . Therefore,

$$|x_j| \leq |p_j| < 2^L. \quad (14)$$

This proves the forward implication.

To show the converse,  $\{x : Ax \leq b\} = \emptyset$  implies, by Farkas' Lemma, there exists a  $y$  such that  $y \geq 0$ ,  $A^T y = 0$ , and  $b^T y = -1$ . We can choose a vertex of  $A^T y = 0$ ,  $b^T y = -1$ ,  $y \geq 0$ . We can also phrase this as:

$$\begin{pmatrix} A^T \\ b^T \end{pmatrix} y = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, y \geq 0 \quad (15)$$

By using Cramer's rule (like we did in the last lecture), we can bound the components of a basic feasible solution  $y$  in the following way:

$$y^T = \left(\frac{r_1}{s}, \dots, \frac{r_m}{s}\right), \quad (16)$$

with  $0 \leq s, r_i \leq \det_{max} \begin{pmatrix} A^T \\ b^T \end{pmatrix}$ , where  $\det_{max}(D)$  denotes the maximum subdeterminant in absolute value of any submatrix of  $D$ . By expanding the determinant along the last row, we see that  $\det_{max} \begin{pmatrix} A^T \\ b^T \end{pmatrix} \leq m b_{max} \det_{max}$  (where this last  $\det_{max}$  refers to the matrix  $A$ ). Using the fact that  $2^L > 2^m 2^n \det_{max} b_{max}$ , we get that  $0 \leq s, r_i < \frac{m}{2^m} 2^L \leq \frac{m}{2^{m+1}} 2^L$ .

Therefore,

$$\left(b + \frac{1}{2^L}e\right)^T y = \underbrace{b^T y}_{-1} + \frac{1}{2^L}e^T y = -1 + \frac{m^2}{2^{m+1}} < 0,$$

the last inequality following from the fact that  $m^2 < 2^{m+1}$  for any integer  $m \geq 1$ . Therefore, by Farkas' Lemma again, this  $y$  shows that there exists no  $x$  where  $Ax \leq b + \frac{1}{2^L}e$ , i.e.,  $P'$  is empty.  $\square$

There is also the problem of when  $x$  is found within  $P'$ ,  $x$  may not necessarily be in  $P$ . One solution is to round the coefficients of the inequalities to rational numbers and "repair" these inequalities to make  $x$  fit in  $P$ . This is called simultaneous Diophantine approximations, and will be discussed later on.

Here we solve this problem using another method: We give a general method for finding a feasible solution of a linear program, assuming that we have a procedure that checks whether or not the linear program is feasible.

Assume, we want to find a solution of  $Ax \leq b$ . The inequalities in this linear program can be written as  $a_i^T x \leq b_i$  for  $i = 1, \dots, m$ . We use the following algorithm:

1.  $I \leftarrow \emptyset$ .
2. For  $i \leftarrow 1$  to  $m$  do
  - If the set of solutions of

$$\left\{ \begin{array}{ll} a_j^T x \leq b_j & \forall j = i + 1, \dots, m \\ a_j^T x = b_j & \forall j \in I \cup \{i\} \end{array} \right\}$$

is nonempty, then  $I \leftarrow I \cup \{i\}$ .

3. Finally, solve  $x$  in  $a_i^T x = b_i$  for  $i \in I$  with Gaussian elimination.

The correctness follows from the fact that if, in step 2, the system of inequalities has no solution then the inequality  $i$  can be discarded since it is redundant (removing it does not affect the set of solutions).

## 2 Applying the Ellipsoid Algorithm to Linear Programming

The algorithm we described today checks whether a set of inequalities are feasible, and if they are, finds a feasible solution. However, our initial goal was to find a feasible solution that minimizes a given linear objective function. Here, we give a general method for solving linear program, given a procedure that finds a feasible solution to a set of inequalities.

**To solve the LP:**  $\min c^T x$  subject to  $Ax = b, x \geq 0$ :

**Step 1:** Check if  $\{x : Ax = b, x \geq 0\}$  is nonempty; if it is empty, then the LP is infeasible; stop.

**Step 2:** Consider the dual LP:  $\max b^T y$  subject to  $A^T y \leq c$ .

Check if there exists a  $y$  such that  $A^T y \leq c$ . If there does not exist such a  $y$ , then the original LP is unbounded by strong duality.

**Step 3:** If the dual LP is feasible, find a solution  $(x, y)$  where  $Ax = b, x \geq 0, A^T y \leq c, c^T x = b^T y$ . By strong duality,  $c^T x = b^T y$  will be the optimal solution.