Solving Large LPs via the Ellipsoid Method[1]

- In this lecture, I will introduce an algorithm for solving linear programs. Indeed, this method called the *ellipsoid method*, or more generally the *cutting plane* method, was used to produce the first polynomial time algorithm for solving LPs. An exciting feature of the ellipsoid method is that it *doesn't* need the set of constraints *explicitly* given, and this power can be used to solve LPs exactly, and even at times used in approximation algorithms even when one cannot solve the LPs exactly.

- *A Feasibility Problem.* Recall that a general LP is of the form

$$\mathsf{lp} := \text{minimize} \quad \mathbf{c}^\top \mathbf{x} = \sum_{j=1}^{n} c_j x_j \qquad \text{(A General Linear Program)}$$
$$A\mathbf{x} \geq \mathbf{b}, \qquad A \in \mathbb{R}^{m\times n}, \mathbf{b} \in \mathbb{R}^m$$

  It will be convenient to consider the following *feasibility* problem is given a linear system and asked whether it is empty or not.

$$\mathcal{P} := \{A\mathbf{x} \geq \mathbf{b} \;\; : \;\; A \in \mathbb{R}^{m\times n}, \mathbf{b} \in \mathbb{R}^m\} \quad =? \; \emptyset \qquad \text{(Feasibility)}$$

  Why is (Feasibility) helpful in solving (A General Linear Program)? Consider making a guess lp_g on the value of (A General Linear Program). Then, $\mathsf{lp}_\mathsf{g} \geq \mathsf{lp}$ if and only if $\mathcal{P}_\mathsf{g} := \{A\mathbf{x} \geq \mathbf{b}, \mathbf{c}^\top\mathbf{x} \leq \mathsf{lp}_\mathsf{g}\}$ is non-empty. In other words, lp is the smallest lp_g for which (Feasibility) returns non-empty for $\mathcal{P}_\mathsf{g}$. Therefore, if we have the ability to solve (Feasibility), then using *binary search* one can solve (A General Linear Program) as well. Indeed, when A, b consists of rational numbers, the number of iterations of the binary search is at most a polynomial in the length of the input. Henceforth, we only consider ourselves with (Feasibility).

- *Assumptions.* To describe the main idea, we make the following two assumptions. Both these assumptions can be removed with extra work to solve LPs exactly, but we won't discuss them.

  - $\mathcal{P} \subseteq B_n(\mathbf{0}, R)$, that is, $\mathcal{P}$ is contained in an n-dimensional ball of radius R where R is "not too big". For most of our applications in approximation algorithms this holds since most often $\mathcal{P} \subseteq [0,1]^n \subseteq B(\mathbf{0}, \sqrt{n})$.
  - If $\mathcal{P}$ is non-empty, then it is *full-dimensional* and furthermore we assume $\mathsf{vol}_n(\mathcal{P})$, the n-dimensional volume of $\mathcal{P}$, is not too small. In particular, $\mathsf{vol}_n(\mathcal{P}) > \gamma^n$ where $\log(\gamma^{-1})$ is at most a polynomial in the length of the input. The fact below shows why this is the case for rational full dimensional polytopes.

Fact 1. If $\mathcal{P} := \{A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \geq 0\}$ where $A, \mathbf{b}$ have rational entries, then if $\mathcal{P}$ is full-dimensional and non-empty we have $\mathsf{vol}_n(\mathcal{P}) \geq \rho^{-\mathrm{poly}(n)}$ where $\rho := \max_{i,j}\left(|A_{ij}|, |\mathbf{b}_j|\right)$ and $|\cdot|$ indicates

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 16th Feb, 2022*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

the maximum value of numerator or denominator of these rational numbers.

Proof. (Sketch) If $\mathcal{P}$ is full-dimensional, then it contains an $(n+1)$-simplex with corners $\{\mathbf{v}_0, \ldots, \mathbf{v}_n\}$ such that each $\mathbf{v}_i$ is a basic feasible solution of $\mathcal{P}$ and the vectors $(\mathbf{v}_i - \mathbf{v}_0)$ form a basis of $\mathbb{R}^n$. One can lower bound $\mathsf{vol}_n(\mathcal{P})$ by the volume of the simplex which is precisely

$$\mathsf{vol}_n(\mathcal{P}) \geq \frac{1}{n!}\Big| \det \underbrace{(\mathbf{v}_0 - \mathbf{v}_1, \mathbf{v}_0 - \mathbf{v}_2, \ldots, \mathbf{v}_0 - \mathbf{v}_n)}_{\text{call this matrix } M} \Big|$$

where the vectors are written as colmumn vectors.

Each entry of M is a rational number if $A, \mathbf{b}$ are rational. Furthermore, each $\mathbf{v}_i := B^{-1}\mathbf{b}_B$ for some $n \times n$ non-singular sub-matrix of A. By standard linear algebra (Cramer's Rule), every entry of $\mathbf{v}_i$, and therefore also M, is a rational number p/q where $|p|, |q| \leq \rho^{\text{poly}(n)}$. If M is non-singlar, applying Cramer's rule again, we get each entry of M^{-1} is also bounded by $\gamma \leq \rho^{\text{poly}(n)}$ By Hadamard's inequality, which states that the determinant of any matrix is at most the product of the ℓ_2 lengths of the rows, we get $|\det(M^{-1})| \leq \gamma^{\text{poly}(n)}$ implying $|\det(M)| \geq \gamma^{-\text{poly}(n)} = \rho^{-\text{poly}(n)}$. $\square$

- ***Access to the matrix : the separation oracle.*** It is important to understand how $\mathcal{P}$ is given to us. One could describe $\mathcal{P}$ by describing all the rows of the matrix A. However, if A has exponentially many constraints in n, for instance such an LP was the Steiner Forest LP we worked with, then this is not feasible.

  Instead, the ellipsoid algorithm works with the following *weaker* "oracle" called the separation oracle.

  Definition 1 (Separation Oracle)**.** Given any $\mathbf{z} \in \mathbb{R}^n$ to $\mathcal{O}_{\mathsf{sep}}(\mathbf{z})$ either asserts $\mathbf{z} \in \mathcal{P}$, or returns a non-zero $\mathbf{c} \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$ such that (i) $\mathbf{c}^\top \mathbf{x} \geq \beta$ for all $\mathbf{x} \in \mathcal{P}$, but (ii) $\mathbf{c}^\top \mathbf{z} < \beta$.

  In other words, $\mathcal{O}_{\mathsf{sep}}(\mathbf{z})$ either asserts $A\mathbf{z} \geq \mathbf{b}$, or returns a constraint $\mathbf{c}$ in the non-negative span of the rows of A such that $\mathbf{c}^\top \mathbf{z} < \beta$, where β is the same non-negative combination of $\mathbf{b}$. Later on, we will see some examples where separation oracles exist. For now, let's assume this oracle exists. Note that if A was explicitly given, we can clearly simulate this oracle.

- ***Geometric Preliminary I : Ellipsoids.*** Before we dive into the ellipsoid algorithm, it is important to brush up on what ellipsoids are.

  Let's begin with two dimensions where the ellipsoids are simply called ellipses, and what is covered in high-school coordinate geometry. The simplest ellipse is a ball, and the equation of a radius 1 ball centered at the origin is precisely $B_2(\mathbf{0}, 1) := \{(x, y) : x^2 + y^2 \leq 1\}$. An *axis-aligned* ellipse has a major axis of length $2a$ and minor axis of length $2b$, with $a \geq b$. The region bounded by this two-dimensional ellipse is

$$E_2^{||}(a, b) := \left\{(x, y) : \frac{x^2}{a^2} + \frac{y^2}{b^2} \leq 1\right\} = \left\{\mathbf{x} := \begin{pmatrix} x \\ y \end{pmatrix} : \mathbf{x}^\top \underbrace{\begin{bmatrix} \frac{1}{a^2} & 0 \\ 0 & \frac{1}{b^2} \end{bmatrix}}_{\text{call this } D} \mathbf{x} \leq 1\right\}$$

  where the second description, right now, seems to be a fancy way of writing the same thing.

Next, we consider a general ellipse centered at the origin. This is simply a *rotation* of an axis-aligned ellipse. So, it can be written as $\left\{(x', y') \ : \ \frac{x'^2}{a^2} + \frac{y'^2}{b^2} \leq 1\right\}$, where (x', y') is found by applying a *rotation linear transform* R on the original coordinates (x, y). Here, the rotation linear transform R is a 2×2 matrix satifying $R^\top R = RR^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Such matrices are also called *orthogonal/orthonormal* matrices.

And therefore, the equation of such ellipses look like

$$E_2(a,b) := \left\{\mathbf{x} \ : \ (R\mathbf{x})^\top D(R\mathbf{x}) \leq 1\right\} = \{\mathbf{x} : \mathbf{x}^\top \left(R^T DR\right) \mathbf{x} \leq 1\}$$

The matrix $R^\top DR = R^{-1}DR$ is a very special kind of matrix when R is a rotation matrix and D is a diagonal matrix with positive entries. It is a symmetric matrix and all its eigenvalues are positive and are indeed the diagonal entries of D. Furthermore, any symmetric matrix with positive eigenvalues looks like this. Such matrices have a name : they are called ***positive definite matrices*** with the notation $A \succ 0$ to denote them. And this leads to the general definition of an ellipsoid in high-dimensions.

Definition 2. An ellipsoid $\mathcal{E}$ in n-dimensions is characterized by an $n \times n$ symmetric *positive definite* matrix $Q \succ 0$ and a center $\mathbf{v} \in \mathbb{R}^n$, and is defined to be

$$\mathcal{E} := \mathcal{E}(Q, \mathbf{v}) = \{\mathbf{x} \in \mathbb{R}^n \ : \ (\mathbf{x} - \mathbf{v})^\top A^{-1} (\mathbf{x} - \mathbf{v}) \leq 1\}$$

The volume of the ellipsoid is precisely $\det(Q)$.

Note that the $R^\top DR$ matrix is replaced by Q^{-1} instead of Q; this is simply the preferred notation.

- ***Geometric Preliminary II : Enclosing Ellipsoids.*** Let $K \subseteq \mathbb{R}^n$ be a *convex* set in n-dimensions. The *minimum volume enclosing* (MVE) ellipsoid of K, as the name suggests, is the ellipsoid $\mathcal{E}$ with the smallest volume such that $K \subseteq \mathcal{E}$. This ellipsoid is also called the Löwner-John ellipsoid and arises in *many* applications, has many beautiful properties, and something which is worth knowing. Unfortunately, it is out of the scope of these notes.

  What we would be interested in are MVEs of a very special type of convex set K: *"hemillipsoids"*.

Definition 3. Given an ellipsoid $\mathcal{E} := \mathcal{E}(Q, \mathbf{v}) \subseteq \mathbb{R}^n$ and a non-zero vector $\mathbf{c} \in \mathbb{R}^n$, the hemillipsoid $\mathcal{H} := \mathcal{E}(Q, \mathbf{v}, \mathbf{c})$ is defined as

$$\mathcal{H} := \mathcal{E}(Q, \mathbf{v}, \mathbf{c}) = \{\mathbf{x} \in \mathbb{R}^n \ : \ (\mathbf{x} - \mathbf{v})^\top A^{-1} (\mathbf{x} - \mathbf{v}) \leq 1\} \cap \{\mathbf{x} \in \mathbb{R}^n \ : \ \mathbf{c}^\top \mathbf{x} \leq \mathbf{c}^\top \mathbf{v}\}$$

That is, it is the half of the ellipsoid cut by a hyperplane passing through its center.

One can precisely figure out the formula for the minimum volume ellipsoid for a hemillipsoid. I will simply state this without proof. What is important really is that there exists a formula which is easy to calculate.

Fact 2. The minimum volume ellipsoid containing the hemillipsoid $\mathcal{E}(Q, \mathbf{v}, \mathbf{c})$ is the ellipsoid $\mathcal{E}' := \mathcal{E}(Q', \mathbf{v}')$ where

- $\mathbf{v}' := \mathbf{v} - \mathbf{d}$
- $Q := \frac{n^2}{n^2-1} \cdot \left(Q - \frac{2}{n+1}\mathbf{d}\mathbf{d}^T\right)$

where $\mathbf{d} := \frac{1}{\sqrt{\mathbf{c}^\top Q\mathbf{c}}} \cdot Q\mathbf{c}$.

The main theorem that will be used by the ellipsoid algorithm qualitatively states that for any ellipse $\mathcal{E}$ and any hemillipsoid $\mathcal{H} \subseteq \mathcal{E}$ defined by any hyperplane $\mathbf{c}$, the minimum volume ellipsoid $\mathcal{E}'$ containing $\mathcal{H}$ has volume a *multiplicative factor* smaller than that of $\mathcal{E}$. It is not too difficult to prove the theorem below using the formula above.

Theorem 1 (Considerable Shrinkage)**.** Let $\mathcal{E} := \mathcal{E}(Q, \mathbf{v})$ be any n-dimensional ellipsoid, and let $\mathbf{c} \in \mathbb{R}^n$ be any non-zero vector defining the hemillipsoid $\mathcal{H} := \mathcal{E}(Q, \mathbf{v}, \mathbf{c})$. Let $\mathcal{E}'$ be the minimum volume ellipsoid containing $\mathcal{H}$ defined in Fact 2. Then

$$\mathsf{vol}_n(\mathcal{E}') \leq \mathsf{vol}_n(\mathcal{E}) \cdot e^{-1/2n}$$

- ***The Basic Ellipsoid Algorithm.*** Now we have all the tools to state and semi-analyze the ellipsoid algorithm.

```
1:  procedure BASIC ELLIPSOID($\mathcal{O}_{\mathsf{sep}}$, $R$, $\gamma$): ▷ $R$ and $\gamma$ as in the assumptions.
2:      $\mathcal{E}_0 \leftarrow B(\mathbf{0}, R)$; $\mathbf{v}_0 \leftarrow \mathbf{0}$. ▷ Start with a ball, and by assumption $\mathcal{P} \subseteq \mathcal{E}_0$
3:      $i \leftarrow 0$; $Q_i \leftarrow R \cdot I_n$; $\mathbf{v}_i \leftarrow \mathbf{0}$.
4:      while true do:
5:          Query $\mathcal{O}_{\mathsf{sep}}(\mathbf{v}_i)$.
6:          If $\mathcal{O}_{\mathsf{sep}}$ returns $\mathbf{v}_i \in \mathcal{P}$, return FEASIBLE.
7:          Else, $\mathcal{O}_{\mathsf{sep}}$ returns $\mathbf{c}, \beta$ with $\mathbf{c}^\top\mathbf{v}_i < \beta \leq \mathbf{c}^\top\mathbf{x}$ for all $\mathbf{x} \in \mathcal{P}$.
8:          ▷ In that case, note that $\mathcal{P} \subseteq \mathcal{E}_i \cap \{\mathbf{x} : (-\mathbf{c})^\top\mathbf{x} \leq (-\mathbf{c})^\top\mathbf{v}_i\}$.
9:          Find MVE $\mathcal{E}_{i+1} := \mathcal{E}(Q_{i+1}, \mathbf{v}_{i+1})$ of $\mathcal{E}(Q_i, \mathbf{v}_i, -\mathbf{c})$ using Fact 2. ▷ Note $\mathcal{P} \subseteq \mathcal{E}_{i+1}$
10:         if $\mathsf{vol}_n(\mathcal{E}_{i+1}) \leq \gamma^n$ then:
11:             return INFEASIBLE.
12:         $i \leftarrow i + 1$.
```

Theorem 2. If $\mathcal{P}$ satisfies the assumptions mentioned in Page 1, then BASIC ELLIPSOID algorithm returns the correct answer in $2n^2 \cdot \ln(2R/\gamma)$ iterations.

Proof. As noted in the algorithm, we maintain the invariant $\mathcal{P} \subseteq \mathcal{E}_i$, and thus, $\mathsf{vol}_n(\mathcal{P}) \leq \mathsf{vol}_n(\mathcal{E}_i)$. In each iteration where the algorithm doesn't return, the volume $\mathsf{vol}_n(\mathcal{E})$ shrinks by $e^{-1/2n}$. Therefore, if the algorithm runs for T iterations and doesn't return, we know that

$$\mathsf{vol}_n(\mathcal{P}) \leq \mathsf{vol}_n(\mathcal{E}_0) \cdot e^{-T/2n} \leq (2R)^n \cdot e^{-T/2n}$$

When $T > n^2 \ln(R/\gamma)$, the RHS is at most $(2R)^n \cdot \left(\frac{\gamma}{2R}\right)^n = \gamma^n$. This contradicts Line 10. □

- ***An Application to algorithms.*** I end this note by noting one application of the ellipsoid algorithm to so-called "design problems". A particular example is the *max-min spanning tree* problem. In this problem we are given an unweighted, undirected graph $G = (V, E)$. The objective is to find non-negative *weights* $w(e)$ for every edge such that $\sum_{e \in E} w(e) \leq 1$ *and* the weight of the *minimum weight* spannning tree in $G = (V, E)$ with respect to these weights, is *maximized*. This is a game between the algorithm, which is trying to spread mass on the edges, and is trying to compete with an adversary who will pick the minimum weight tree after the algorithm's play. Can we find the optimal way of putting weights?

  At first glance, it is not obvious that this is an easy problem. We now show, using the ellipsoid method, that *just* the fact that solving the minimum weight spanning tree is "easy", implies that the max-min spanning tree problem can be solved in polynomial time.

  First we observe that the max-min spanning tree problem is actually a linear program on exponentially many constraints. To that end, let $\mathcal{T}$ denote the set of *all* spanning trees in G. Then, the max-min spanning tree problem is asking us to solve

$$\begin{aligned} \mathsf{mmst} := \text{maximize} \quad & \lambda & \text{(Max-Min Spanning Tree)} \\ & \lambda - \sum_{e \in T} \mathbf{w}_e \leq 0, \qquad \forall T \in \mathcal{T} & (1) \\ & \sum_{e \in E} \mathbf{w}_e \leq 1, \qquad\qquad \mathbf{w}_e \geq 0, \qquad \forall e \in E \end{aligned}$$

  The above LP has $|E| + 1$ variables, but $|\mathcal{T}|$ many constraints, and the latter can be as large as n^n. However, we can solve the above problem using the ellipsoid algorithm. It's not too hard to see that the above region is full-dimensional (unless the number of trees is too small). More interestingly, what is $\mathcal{O}_{\mathsf{sep}}$ for this polytope? Well, given $(\lambda, \mathbf{z}_e)$ can we check if it satisfies all the constraints? The last two are trivial to check. And now observe that (1) can be checked by using the minimum spanning tree algorithm on $(G, \mathbf{z})$ and seeing if this minimum cost is $\geq \lambda$ or not. If yes, then it's feasible. If not, then (1) corresponding to the minimum weight spanning tree T is the separating hyperplane.

Notes

The book [2] is **the** book to read about the ellipsoid algorithm and much, much more. There are many details we have left out above, even for the basic ellipsoid method. Especially the fact how one deals with irrational numbers which definitely arises when one takes square roots (take a look again at the $\mathbf{d}$ vector in Fact 2). Tackling non-full dimensional polytopes also needs many new ideas. We refer the interested reader to [2] for all these things. The example on max-min spanning tree is taken from the paper [1] by Chakrabarty, Mehta and Vazirani.

References

[1] D. Chakrabarty, A. Mehta, and V. Vazirani. Design is as easy as Optimization. *Proceedings of* 33^{rd} *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 477–488, 2006.

[2] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Springer-Verlag, 1988.