CS-E400204: Approximation Algorithms **Spring (Fifth Period) 2022**

Lecture 5 (May 4): Linear Programming and The Ellipsoid Algorithm

Lecturer: Kamyar Khodamoradi *Scribe: Michał Osadnik*

5.1 Introduction

5.1.1 Complexity of LP

- Feasibility problem (decision) – emptiness test.
  Given $P = \{Ax \geq B\}$ decide if $P \neq \emptyset$
- Search version
  Given $P = \{Ax \geq B\}$ decide if $P \neq \emptyset$ and give $x \in P$
- Optimization version
  Given $P = \{Ax \geq B\}$, $C \in \mathbb{R}^n$ decide if $P \neq \emptyset$, find $\arg\min_{c \in P} cx$

Say, Π is our *feasibility problem*

5.1.2 $\Pi \in$ NP?

Yes! We can give a feasible solution and verify this by evaluating each equation

5.1.3 $\Pi \in$ co $-$ NP?

Yes! The non-feasibility witness has been presented in 1898 by Julius Farkas:

Lemma 1 *If $P \neq \emptyset$ then:*

$$\exists_{y \in \mathbb{R}^n, y \geq \mathbf{0}} (y^T A = \mathbf{0}) \wedge (y^T b = 1)$$

Thereby,

Corollary 1

$$\Pi \in \mathbf{NP} \cap \mathbf{co} - \mathbf{NP}$$

Also,

Lemma 2

$$\Pi \in \mathbf{P}$$

Proof: **(Khachyan, 1979)** The ellipsoid algorithm solves Π in poly-time

5.2 Methods to solve LP

- Simplex method **(Dantzig, 1947)**
  - Can cycle forever looking for OPT
  - Exponential time (with pivoting)

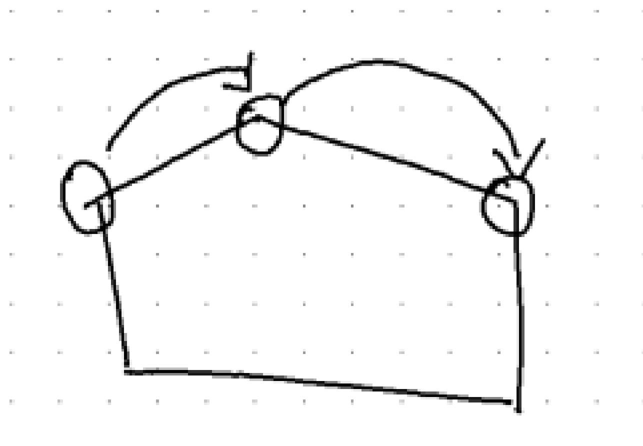

Figure 5.1: The simplex algorithm iterating over vertices of the polytop hyperplane

- Ellipsoid method **(Khachyan, 1979)**
  - P oly time
  - Not very practical due to high (although poly) runtime complexity.
- Interior Points Method (IPM) **(Karmarkar, 1984)**
  - Poly time
  - Optimal
  - Practical

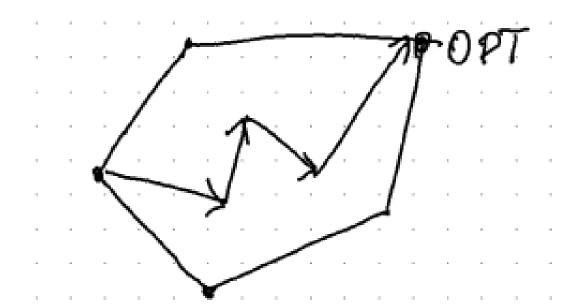

Figure 5.2: The Interior Points Method algorithm iterating for the optimal solution not necessarily visiting vertices of the polytop

5.3 *Optimization* version of the problem reduces to *feasibility* version

Theorem 1 *Optimization version of the problem reduces to feasibility version*

Assume we have an access to an feasibility oracle.

Definition 1 *for $P = \{Ax \geq b\}$ basic feasible solution is inside of the polytop defined by inequalities*

For any point of P, e.g., v:

$$< v >= poly(< P >)$$

if $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$ and $r \in \mathbb{Q}^n$

Proof (sketch):

We use Gaussian elimination in polynomial time **(Edmond, 1967)**.

v is produced by the Gaussian elimination:

$$< v >= poly(< v >), v \in \mathbb{Q}^n$$

Assume polytop is bounded. Then, we know M st.

$$P \cap \{C \cdot n \geq M\}$$

is not feasible and

$$P \cap \{C \cdot n \leq -M\}$$

is not feasible and we have a lower and upper bound.

We can assume the $M \leq 2^{poly(<M>)}$

The algorithm is bisecting with hyperplane c_i perpendicular to an objective value vector.

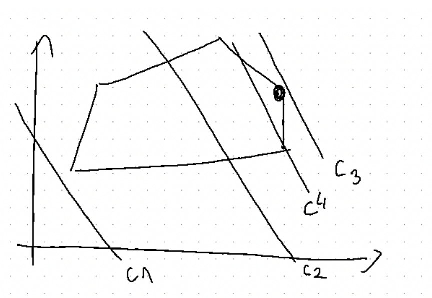

Figure 5.3: The algorithm is bisecting with hyperplane c_i.

5.4 Ellipsoid algorithm

The algorithm takes a system of inequalities as an input. We assume that we have a bounding box of size R such that the polytop is fully included in this box.

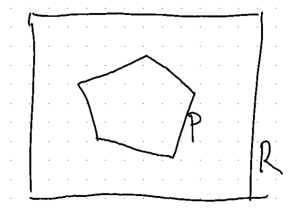

Figure 5.4: Polytop P fully included in a box of size R.

The algorithm is starting with a preparation phrase that we convert all our inequalities and transform them all into equalities with non-negative factors. Them we displace them slightly to obtain *robustness* property. This being said, we expect our polytop P to not only be fully contained into box of size R, but also to contain itself box of the size r.

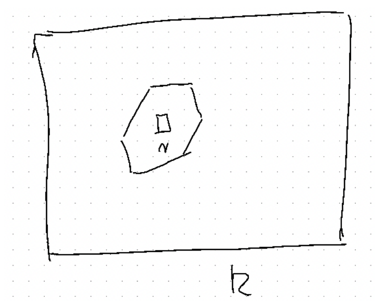

Figure 5.5: Polytop P fully included in a box of size R containing itself box of the size r.

Then we design an algorithm that is running in $poly(< P >, \log(R), \log(\frac{1}{r}))$.

Firstly, we draw an ellipsoid over the set R.

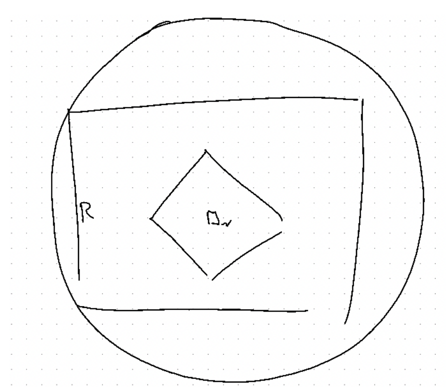

Figure 5.6: The initial phrase of the ellipsoid algorithm.

Then we observe if the of the ellipsoid is a feasible solution. If no, we cut the ellipsoid with a hyperplane (representing not satisfied equation) obtaining a smaller instance.

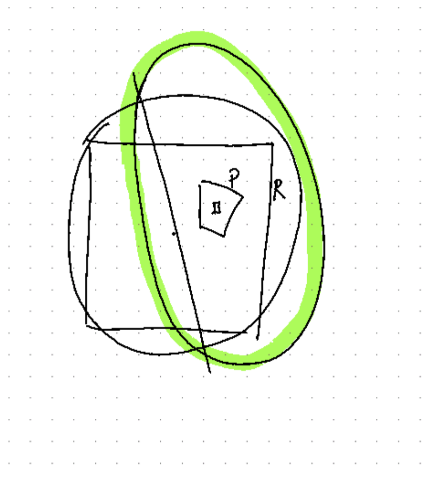

Figure 5.7: The second phrase of the ellipsoid algorithm with a new ellipsoid (green).

On each step we're decreasing the volume of ellipsoid E_i

If eventually the volume of a new ellipsoid $V(E_i)$ will be smaller that r^d, the instance is not feasible.

$$V(E_{i+1}) \leq (1 - \frac{1}{3n})V(E_i)$$

Hence, after $3n$ repetition:

$$V(E_{i+3n}) \leq (1 - \frac{1}{3n})^{3n}V(E_i) \leq e^{-1}V(E_i)$$

.

The overall complexity is from decreasing instance from the original ellipsoid ($O(R^d)$) to r^d is:

$$poly(n)(\log(R) + \log(\frac{1}{r}))$$

Such an algorithm is called a separation oracle.