CS7800: Advanced Algorithms

Class 11: Linear Programming III

- Minmax Theorem
- Algorithms for LPs

Jonathan Ullman

October 14, 2025

Exam Recap

- Median score: $74/90 \approx 82\%$
- Wide range of grades
- Will rescale so that $\begin{cases} [74, 90] = \text{A-/A} \\ [45, 74) = \text{B/B+} \end{cases}$
- Feel free to ask about your course grade any time

Strong LP Duality

Theorem: If the primal and dual are both feasible then they have the same optimal value

(P) $\max_x \; c^T x$
s.t. $Ax \leq b$
$x \geq 0$

(D) $\min_y \; b^T y$
s.t. $y^T A \geq c$
$y \geq 0$

Special cases:

1. If the dual is infeasible, the primal is unbounded
2. If the dual is unbounded, the primal is infeasible

Application: The Minimax Theorem

Zero-Sum Games:

- Two players Rowena and Colin
- Rowena chooses an action in $[m]$ Colin chooses in $[n]$
- Payoffs $A \in \mathbb{R}^{m \times n}$

Ex. Rock Paper Scissors

$$\begin{array}{c|ccc} & R & P & S \\ R & 0 & -1 & +1 \\ P & +1 & 0 & -1 \\ S & -1 & +1 & 0 \end{array}$$

$$\left.\begin{array}{l}\text{Rowena plays } i \\ \text{Colin plays } j\end{array}\right\} \Longrightarrow \begin{array}{l}\text{Rowena gets } A_{ij} \\ \text{Colin gets } -A_{ij}\end{array}$$

"zero-sum"

- Players can play randomly

$$\left.\begin{array}{lll}\text{Rowena: } r = (r_1, \dots, r_m) & \sum_i r_i = 1 & r_i \geq 0 \\ \text{Colin: } c = (c_1, \dots, c_n) & \sum_j c_j = 1 & c_j \geq 0\end{array}\right\} \Longrightarrow$$

Rowena's expected payoff is

$$\sum_{ij} r_i c_j A_{ij} = r^T A c$$

Application: Minimax Thm

Zero-Sum Games:

- Two players Rowena and Colin
- Rowena chooses an action in [m] Colin chooses in [n]
- Payoffs $A \in \mathbb{R}^{m \times n}$

$$\left.\begin{array}{l}\text{Rowena plays } i \\ \text{Colin plays } j\end{array}\right\} \Longrightarrow \underbrace{\begin{array}{l}\text{Rowena gets } A_{ij} \\ \text{Colin gets } -A_{ij}\end{array}}_{\text{"zero-sum"}}$$

- Players can play randomly

$$\left.\begin{array}{ll}\text{Rowena: } r = (r_1, \dots, r_m) & \sum_i r_i = 1 \quad r_i \geq 0 \\ \text{Colin: } c = (c_1, \dots, c_n) & \sum_j c_j = 1 \quad c_j \geq 0\end{array}\right\} \Longrightarrow$$

Rowena's expected payoff is

$$\sum_{ij} r_i c_j A_{ij} = r^T A c$$

How would Rowena play if she went first?

$$\max_r \left(\min_c r^T A c \right)$$

"A max-min strategy"

How would Colin play if he went first?

$$\min_c \left(\max_r r^T A c \right)$$

"A min-max strategy"

$$\min_c \max_r r^T A c$$

$$\geq \max_r \min_c r^T A c$$

Minimax Theorem: For every two-player sum game with payoffs $A \in \mathbb{R}^{m \times n}$

$$\min_c \max_r r^T A c = \max_r \min_c r^T A c$$

von Neumann '35

Application: Minimax Thm Proof

Minimax Theorem: For every two-player sum game with payoffs $A \in \mathbb{R}^{m \times n}$

$$\min_{c} \max_{r} r^T A c = \max_{r} \min_{c} r^T A c = \text{value}(A)$$

How can Rowena find an optimal strategy?

$$\max_{r \in \Delta(m)} \left(\min_{c \in \Delta(n)} r^T A c \right) = \max_{r \in \Delta(m)} \left(\min_{j \in [n]} (r^T A)_j \right)$$

$\Delta(m)$ = probability dist over $[m]$

$$= \max_{r \in \Delta(m)} \left(\min_{j \in [n]} \sum_{i=1}^{m} r_i A_{ij} \right)$$

Write Rowena's problem as an LP

Write Colin's problem as an LP

Note that these LPs are dual!

$$\max_{v, r} \; v$$

$$\text{s.t.} \quad \sum_{i=1}^{m} r_i A_{ij} \geq v \quad \text{for all } j \in [n]$$

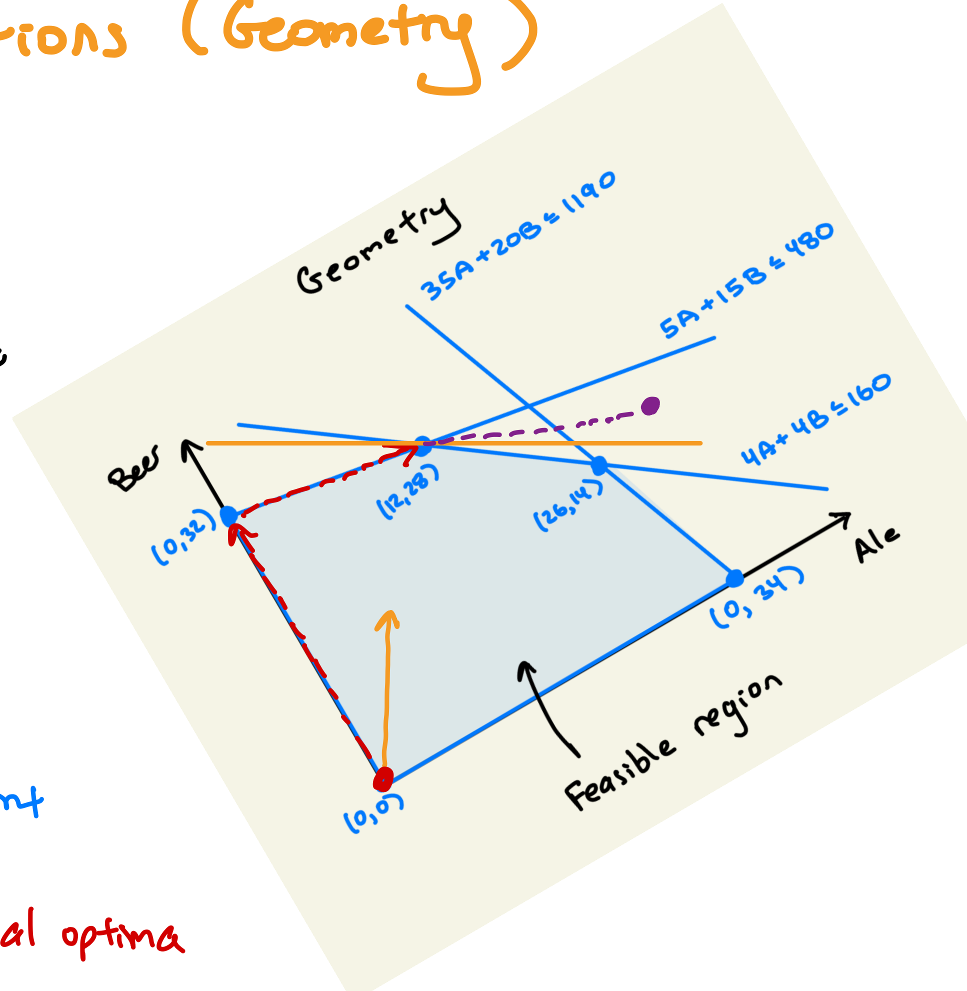
$$\sum_{i=1}^{m} r_i = 1 \quad \text{and} \quad r_i \geq 0 \text{ for all } i \in [m]$$

Solving Linear Programs: Simplex

Basic Feasible Solutions (Geometry)

Fact: There exists an optimal solution at a vertex of the feasible region.

Vertex = n linearly independent constraints are tight



Simplex Algorithm:

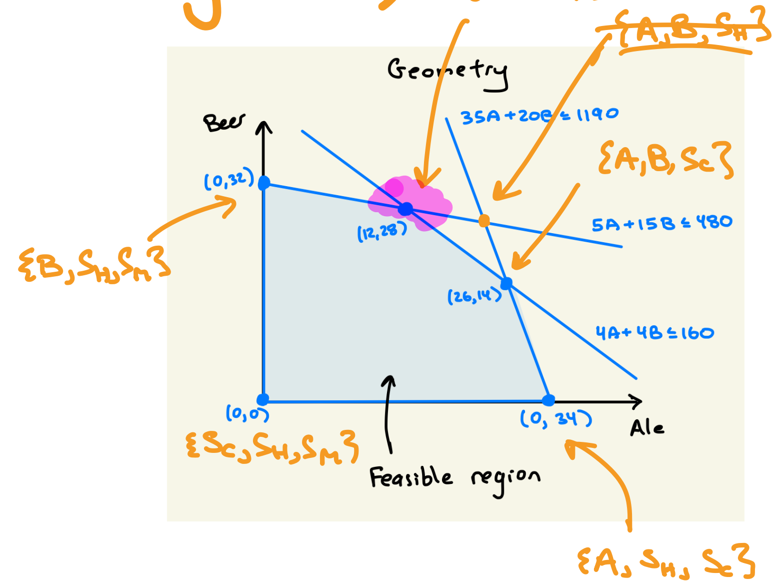
Iteratively walk around vertices of the feasible region until you get to an optimal point

Convexity ⇒ Local optima are global optima

Basic Feasible Solutions (Algebra)

slack form LP

$$
\begin{aligned}
\max_{A,B,S_C,S_H,S_M} \quad & 13A + 23B \\
\text{s.t.} \quad & 5A + 15B + S_C = 480 \\
& 4A + 4B + S_H = 160 \\
& 35A + 20B + S_M = 1190 \\
& A, B, S_C, S_H, S_M \geq 0
\end{aligned}
$$



constraint matrix

$$
\begin{array}{ccccc}
(A) & (B) & (S_C) & (S_H) & (S_M)
\end{array}
$$

$$
\begin{bmatrix} 5 & 15 & 1 & 0 & 0 \\ 4 & 4 & 0 & 1 & 0 \\ 35 & 20 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} A \\ B \\ 0 \\ S_H \\ 0 \end{bmatrix}
=
\begin{bmatrix} 480 \\ 160 \\ 1190 \end{bmatrix}
$$

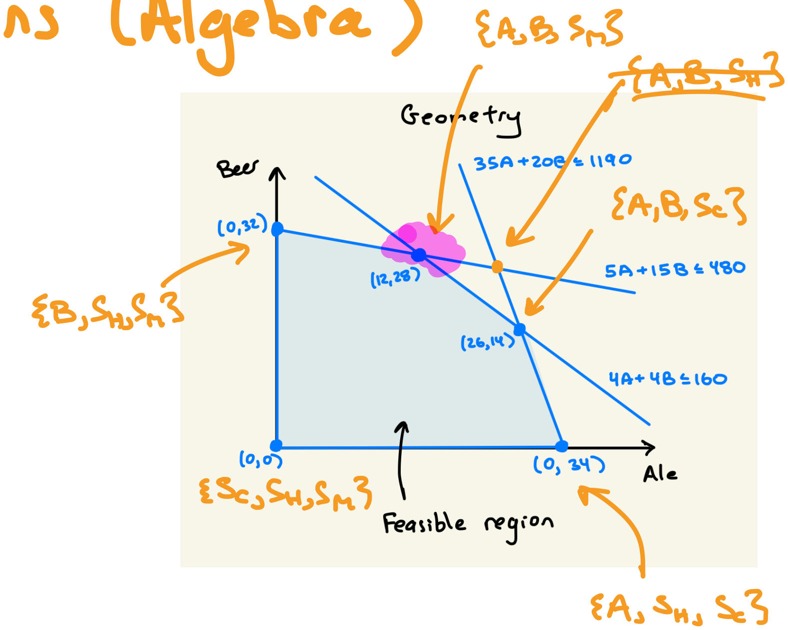
$$
\underbrace{\begin{bmatrix} 5 & 15 & 0 \\ 4 & 4 & 1 \\ 35 & 20 & 0 \end{bmatrix}}
\begin{bmatrix} A \\ B \\ S_H \end{bmatrix}
=
\begin{bmatrix} 480 \\ 160 \\ 1190 \end{bmatrix}
$$

unique solution if these columns are linearly independent

Basic Feasible Solutions (Algebra)

slack form LP

$$
\begin{aligned}
\max_{A,B,S_C,S_H,S_M} \quad & 13A + 23B \\
\text{s.t.} \quad & 5A + 15B + S_C = 480 \\
& 4A + 4B + S_H = 160 \\
& 35A + 20B + S_M = 1190 \\
& A, B, S_C, S_H, S_M \geq 0
\end{aligned}
$$



constraint matrix

$$
\begin{array}{ccccc}
(A) & (B) & (S_C) & (S_H) & (S_M)
\end{array}
$$

$$
\begin{bmatrix} 5 & 15 & 1 & 0 & 0 \\ 4 & 4 & 0 & 1 & 0 \\ 35 & 20 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} A \\ B \\ 0 \\ S_H \\ 0 \end{bmatrix}
=
\begin{bmatrix} 480 \\ 160 \\ 1190 \end{bmatrix}
$$

T

A BFS is defined by a set of $n-m$ strictly positive vars T

$A_T^{-1} b$ is the BFS

① A_T is invertible

② $A_T^{-1} b \geq 0$

The Simplex Algorithm (30,000' view)

Given an LP in standard form

$$\max_x \; c^T x \quad Ax = b, \; x \geq 0$$

Simplex algorithm

- Start with a BFS x_0 corresponding to constraint set S_0

How?

- Repeat until optimality:

How?

  - Find an adjacent BFS x_i corresponding to constraint set S with

$$c^T x_i \geq c^T x_{i-1}$$

Thm: Only terminates at an optimal solution

The Simplex Al orithm (Pivot)

program

max z s.t.

$$\begin{array}{rcl} 13A + 23B & -z & = 0 \\ \hline 5A + 15B + S_c & & = 480 \\ 4A + 4B + S_H & & = 160 \\ 35A + 20B + S_M & & = 1190 \end{array}$$

matrix

basis: $\{S_c, S_H, S_M\}$

$$\begin{bmatrix} 5 & 15 & 1 & 0 & 0 \\ 4 & 4 & 0 & 1 & 0 \\ 35 & 20 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ S_c \\ S_H \\ S_M \end{bmatrix} = \begin{bmatrix} 480 \\ 160 \\ 1190 \end{bmatrix}$$

sol

$z = 0$

$A = 0$

$B = 0$

$S_c = 480$

$S_H = 160$

$S_M = 1190$

---

program

max z s.t.

$$\begin{array}{rcl} \frac{16}{3}A - \frac{23}{15}S_c & -z & = -736 \\ \hline \frac{1}{3}A + B + \frac{1}{15}S_c & & = 32 \\ \frac{8}{3}A + -\frac{4}{15}S_c + S_H & & = 32 \\ \frac{85}{3}A + -\frac{4}{3}S_c + S_M & & = 550 \end{array}$$

matrix

basis: $\{B, S_H, S_M\}$

$$\begin{bmatrix} 1/3 & 1 & 1/15 & 0 & 0 \\ 8/3 & 0 & -4/15 & 1 & 0 \\ 85/3 & 0 & -4/3 & 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ S_c \\ S_H \\ S_M \end{bmatrix} = \begin{bmatrix} 32 \\ 32 \\ 550 \end{bmatrix}$$

sol

$z = 736$

$A = 0$

$B = 32$

$S_c = 0$

$S_H = 32$

$S_M = 550$

The Simplex Algorithm (30,000' view)

Given an LP in standard form

$$\max_{x} \; c^T x$$
$$Ax = b$$
$$x \geq 0$$

Simplex algorithm

- Start with a BFS x_0 corresponding to constraint set S_0

How?

- Repeat until optimality:

How?

  - Find an adjacent BFS x_i corresponding to constraint set S with

$$c^T x_i \geq c^T x_{i-1}$$

Thm: Only terminates at an optimal solution

Simplex in Practice

Theory: Might need exponentially many pivots to terminate

Practice: Can solve LPs with millions of variables/constraints (usually $\le 2(n+m)$ pivots)

Many Issues to Resolve:

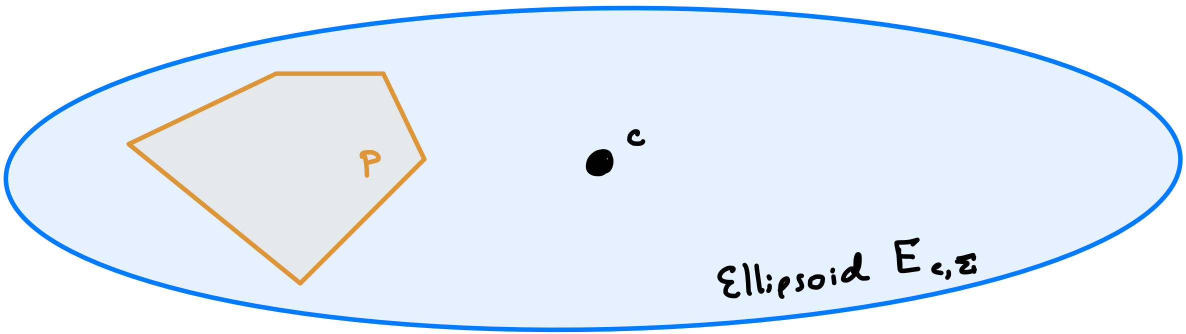
1. What if the LP in infeasible/unbounded?
2. How to choose a good pivot rule?
3. How to avoid cycling?
4. How to maintain sparsity?
5. How to be numerically stable?
6. How to preprocess the LP to be smaller?

Solving Linear Programs: Ellipsoid

Solving linear programs in worst-case polynomial time

⓪ Enough to find a feasible point. (Why?)

① Find an ellipsoid containing P. (How?)

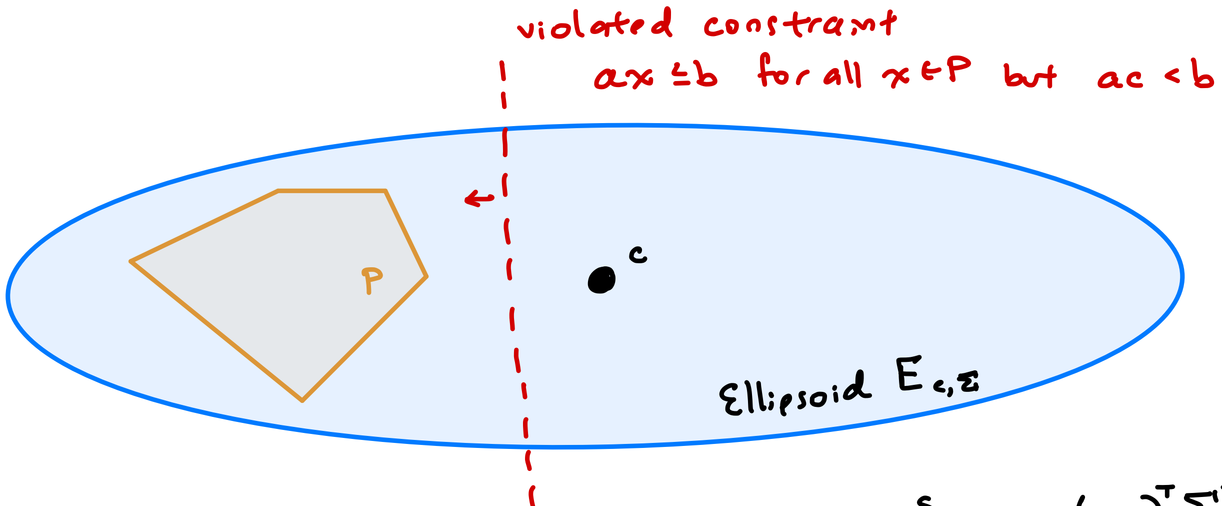


$$E = \{ x : (x-c)^T \Sigma^{-1} (x-c) \leq 1 \}$$

Solving Linear Programs: Ellipsoid

Solving linear programs in worst-case polynomial time

② Either $c \in P$ or there is a violated constraint

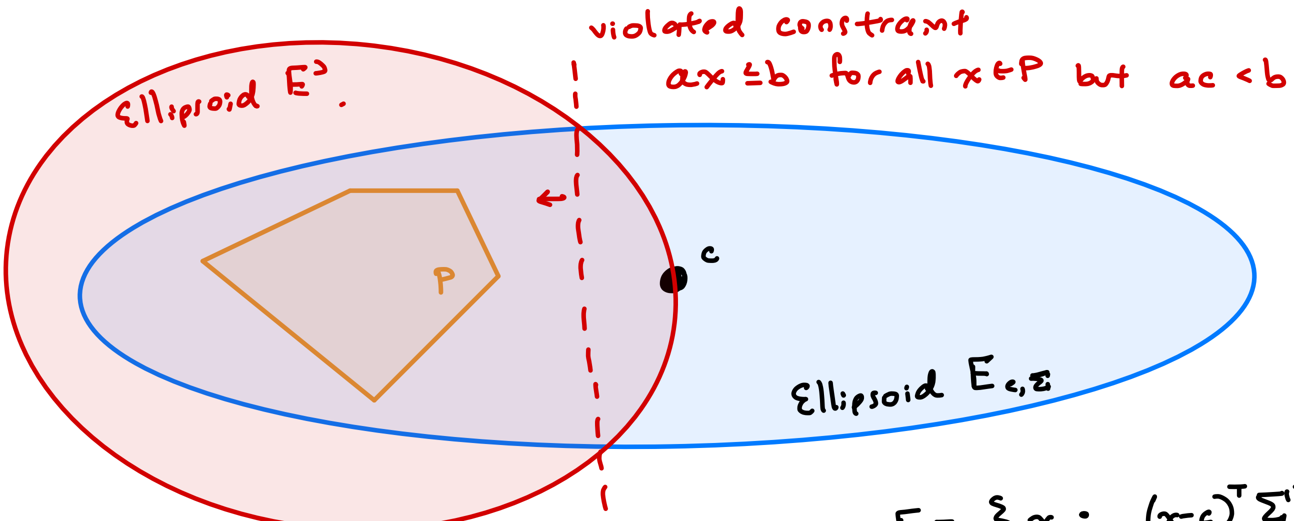


$$E = \{ x : (x-c)^T \Sigma^{-1} (x-c) \le 1 \}$$

Solving Linear Programs: Ellipsoid

Solving linear programs in worst-case polynomial time

③ Use the violated constraint to find a smaller ellipsoid containing P



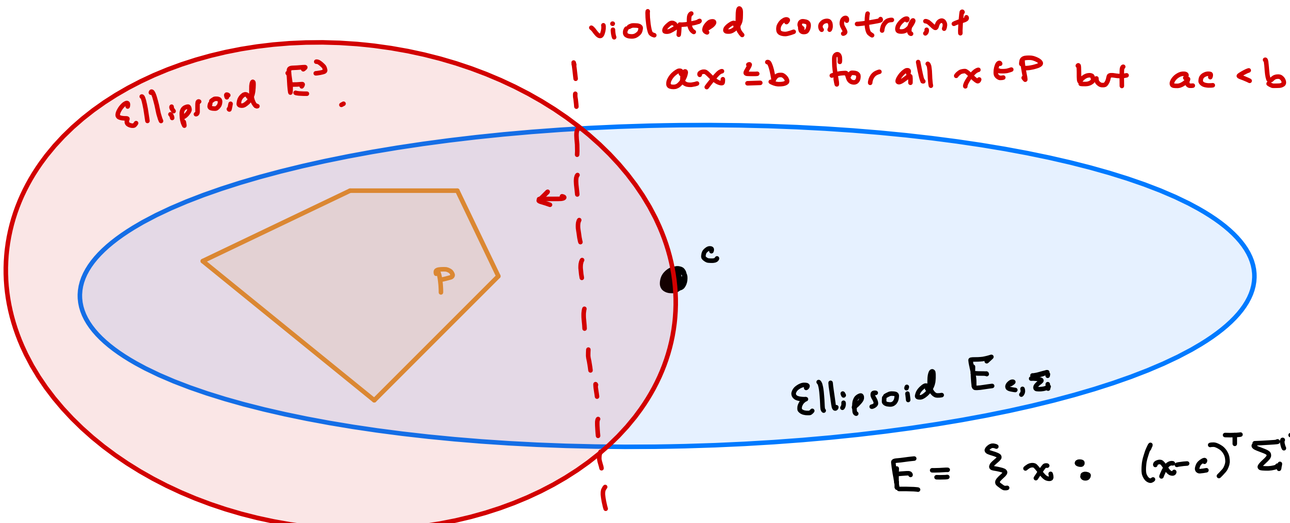
$$E = \{ x : (x-c)^T \Sigma^{-1} (x-c) \le 1 \}$$

Key Theorem: Can choose E' so that $\dfrac{\mathrm{vol}(E')}{\mathrm{vol}(E)} \le \left(1 - \dfrac{1}{2n+2}\right)$

Solving Linear Programs: Ellipsoid

Solving linear programs in worst-case polynomial time

③ Use the violated constraint to find a smaller ellipsoid containing P



$$E = \{ x : (x-c)^T \Sigma^{-1} (x-c) \leq 1 \}$$

Key Theorem: Can choose E' so that $\frac{\text{vol}(E')}{\text{vol}(E)} \leq \left(1 - \frac{1}{2n+2}\right)$

Key Fact: Works any time we can find a "separation oracle" for P!

Linear Programming : Summary

Summary (of Network Flow Algorithms)

- **Last Class:** Can solve maximum flow in time $O(m \cdot v^*)$
  - Can be very slow when capacities are large
  - Cannot be improved if we allow arbitrary augmenting paths

- **Today:** Improving running time by choosing better paths
  - **Widest Augmenting Path:** $O(m \cdot \log v^*)$
  - **Shortest Augmenting Path:** $O(m^2 n)$

- **Still actively studied!**
  - Can solve maximum flow in $O(mn)$ using augmenting path* algos
  - **Recent Breakthrough:** Can solve maximum flow in time* $m^{1+o(1)}$

- **Later On:** Using maximum-flow as a building block for solving many more problems