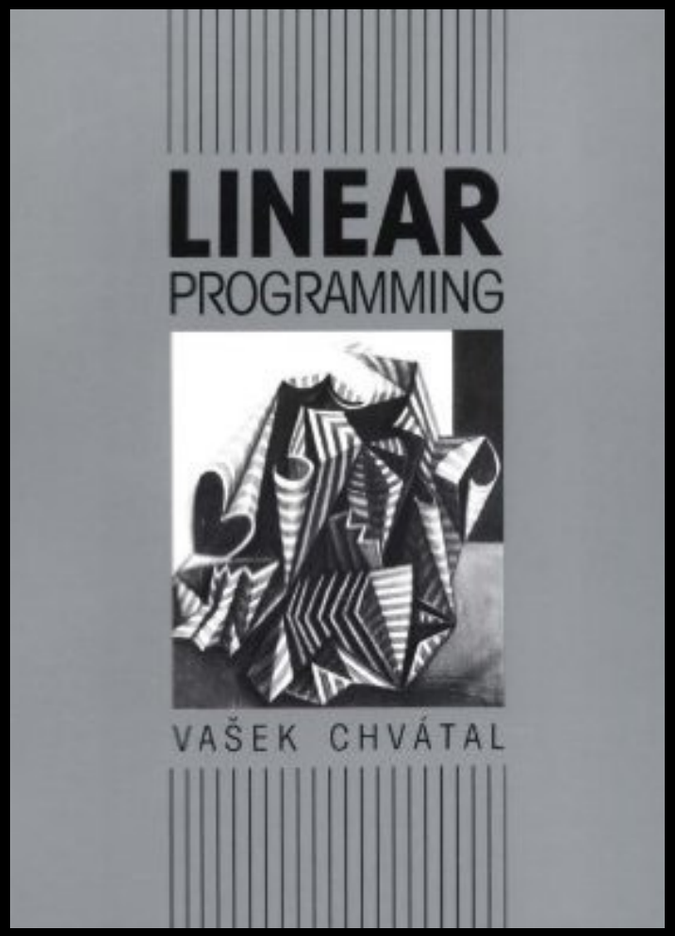
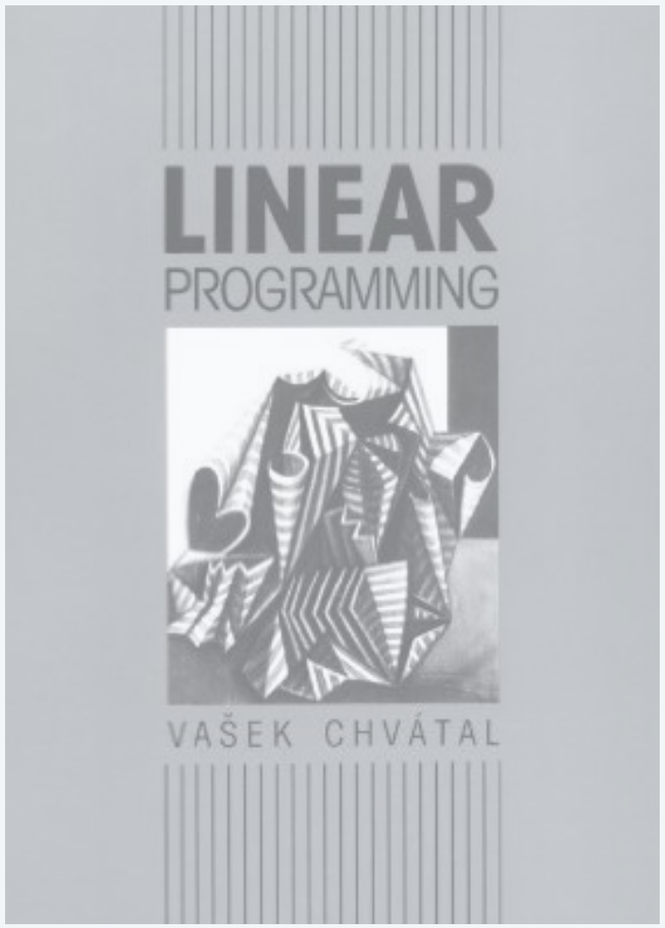


Linear Programming III

- *ellipsoid algorithm*
- *combinatorial optimization*
- *matrix games*
- *open problems*

Lecture slides by Kevin Wayne

Last updated on 7/25/17 11:09 AM



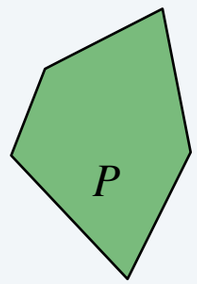
LINEAR PROGRAMMING III

- *ellipsoid algorithm*
- *combinatorial optimization*
- *matrix games*
- *open problems*

Lecture notes on the ellipsoid algorithm by Michel X. Goemans
`http://math.mit.edu/~goemans/18433S07/ellipsoid.pdf`

Geometric divide-and-conquer

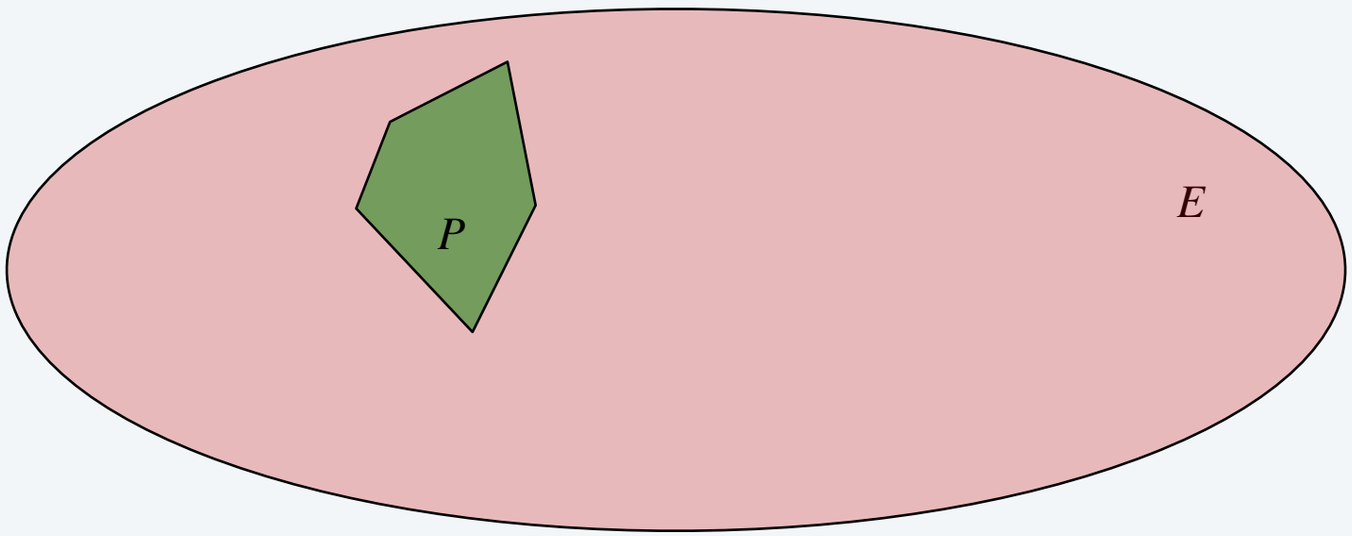
To find a point in P:



Geometric divide-and-conquer

To find a point in P:

- Maintain ellipsoid E containing P.

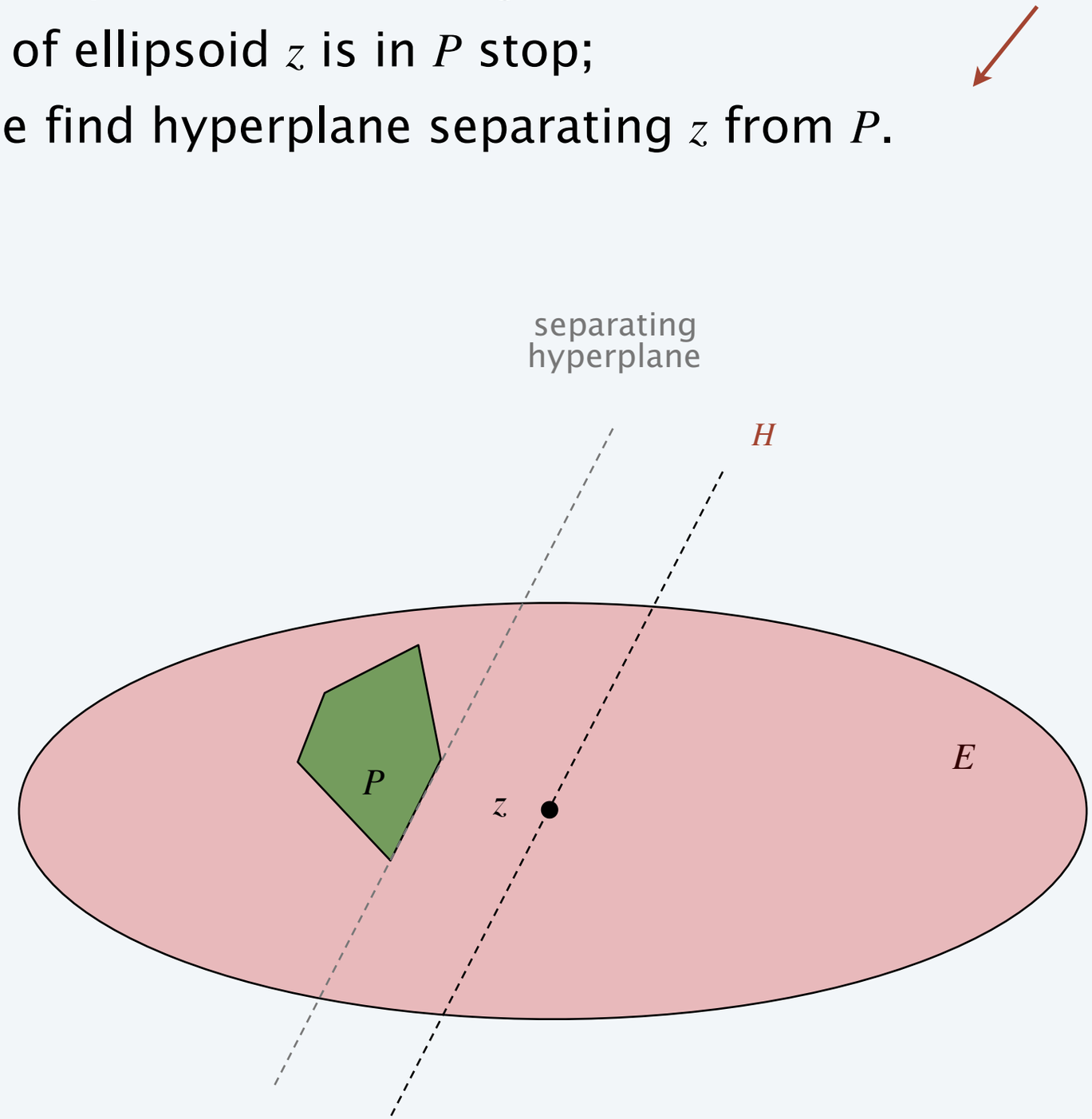


Geometric divide-and-conquer

To find a point in P:

- Maintain ellipsoid E containing P.
- If center of ellipsoid z is in P stop;
  otherwise find hyperplane separating z from P.

and consider corresponding
half-ellipsoid $\frac{1}{2}E = E \cap H$



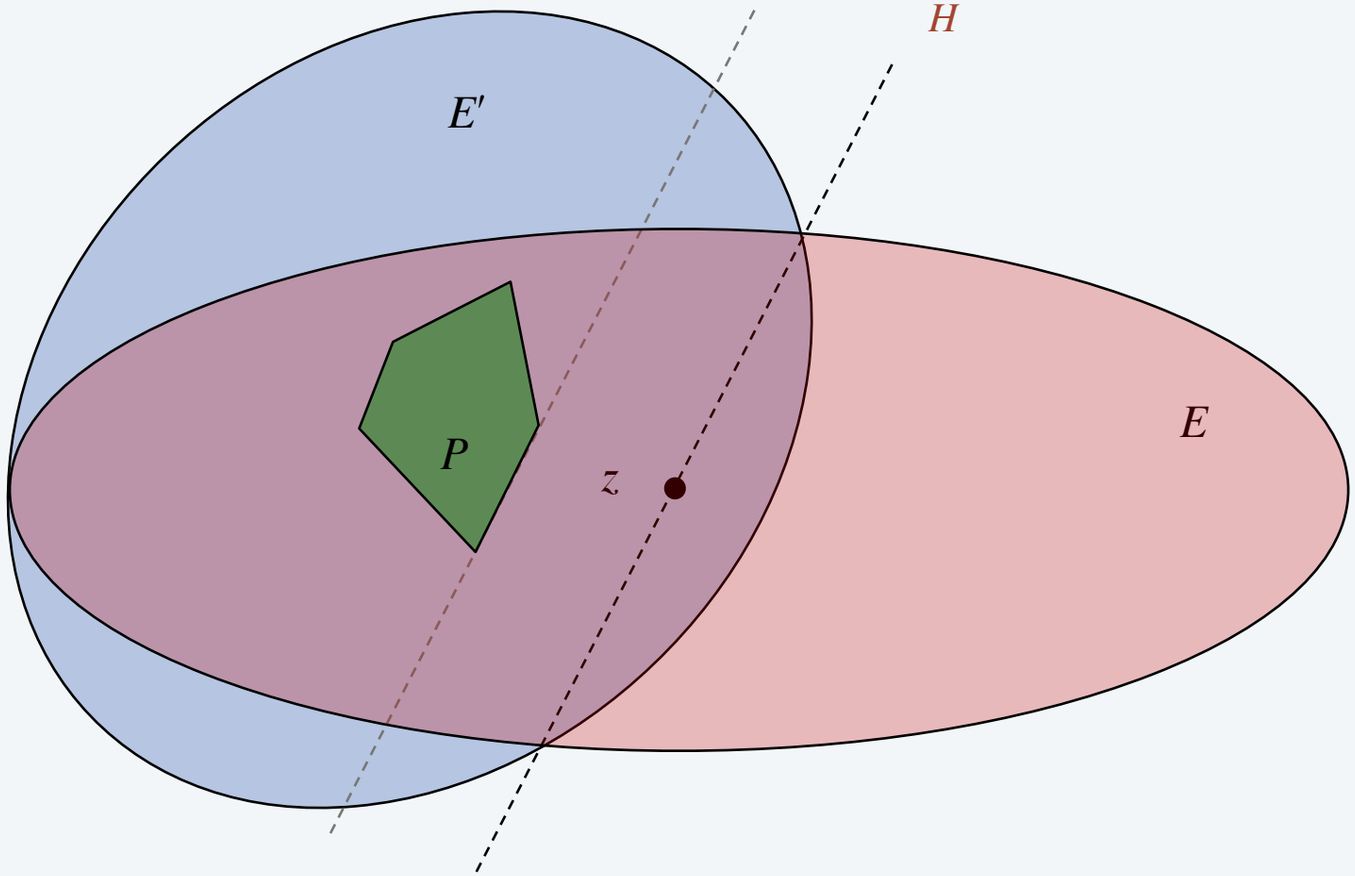
Geometric divide-and-conquer

To find a point in P:

- Maintain ellipsoid E containing P.
- If center of ellipsoid z is in P stop;
  otherwise find hyperplane separating z from P.
- Find smallest ellipsoid E' containing half-ellipsoid.

Lowner–John ellipsoid

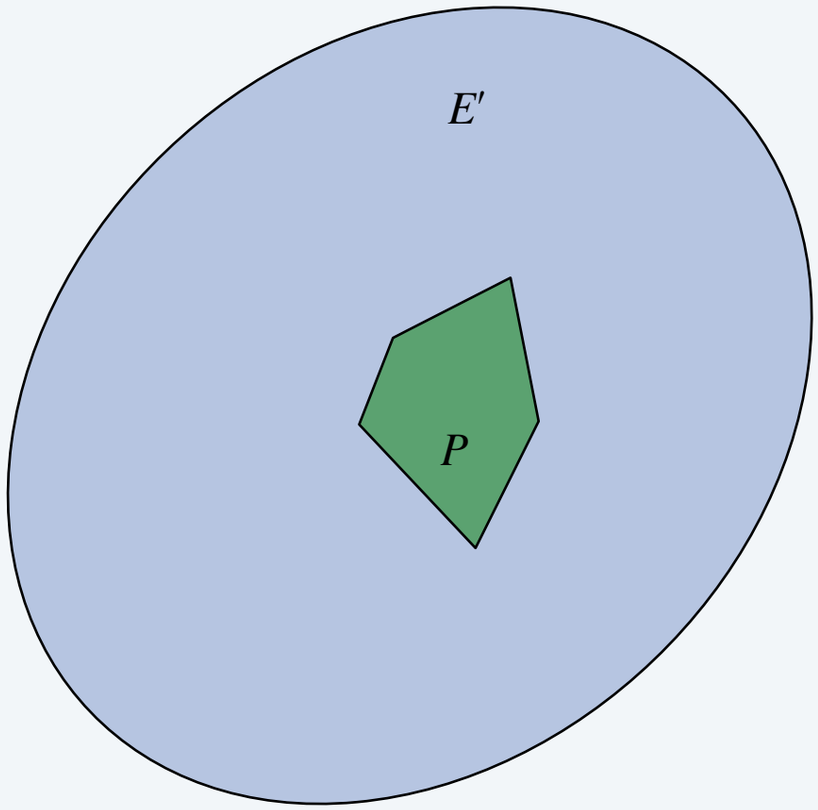
separating hyperplane



Geometric divide-and-conquer

To find a point in P:

- Maintain ellipsoid E containing P.
- If center of ellipsoid z is in P stop;
  otherwise find hyperplane separating z from P.
- Find smallest ellipsoid E' containing half-ellipsoid.
- Repeat.



Optimization to feasibility

Standard form.

$$\begin{aligned} \max\ & c^T x \\ \text{s.t.}\quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$Ax \leq b$ form.

$$\begin{aligned} \exists\ & x, y \\ \text{s.t.}\quad & Ax \leq b \\ & -Ax \leq -b \\ & -x \leq 0 \\ & A^T y \leq c \\ & c^T x - b^T y \leq 0 \end{aligned}$$

$Ax \leq b$

$x \geq 0$

dual feasible

optimal

Ellipsoid algorithm

Goal. Given $A \in \Re^{m \times n}$ and $b \in \Re^m$, find $x \in \Re^n$ such that $\underbrace{Ax \le b}_{P}$.

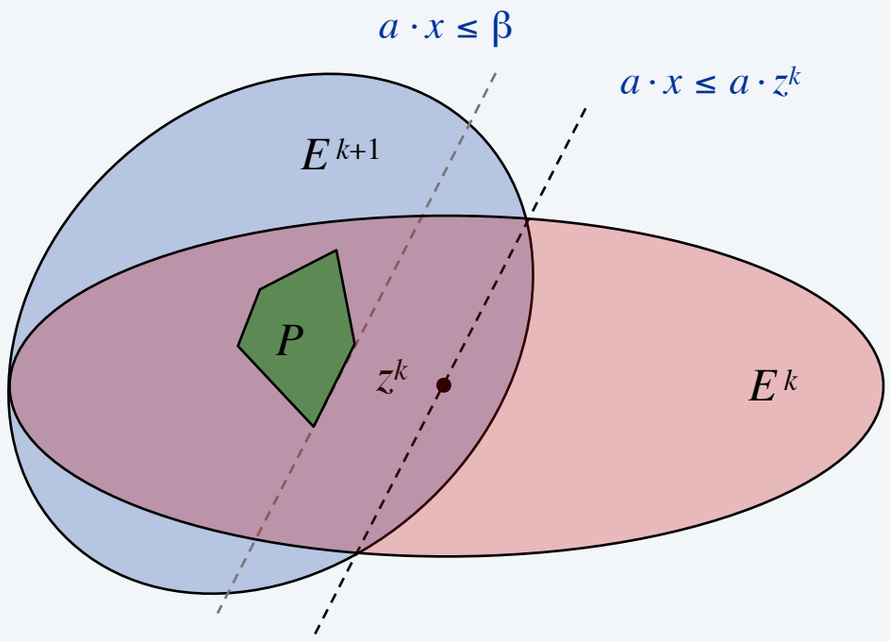
Ellipsoid algorithm.

- Let E_0 be an ellipsoid containing P.
- $k = 0$.
- While center z^k of ellipsoid E^k is not in P :
  - find a constraint, say $a \cdot x \le \beta$, that is violated by z^k
  - let E^{k+1} be min volume ellipsoid containing $E^k \cap \{ x : a \cdot x \le a \cdot z^k \}$
  - $k = k + 1$

enumerate constraints

easy to compute

half-ellipsoid ½ E



Shrinking lemma

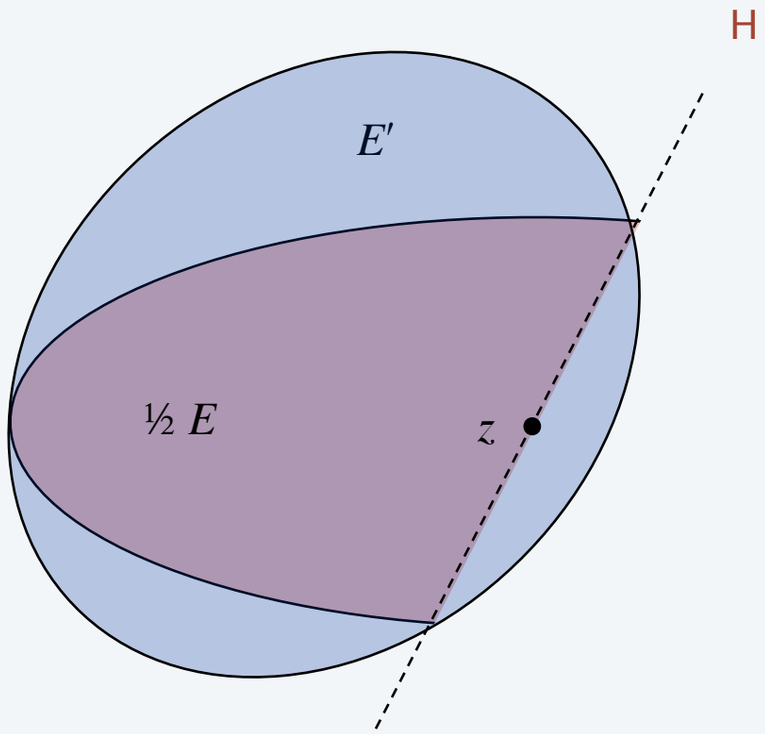
Ellipsoid. Given $D \in \Re^{n \times n}$ positive definite and $z \in \Re^n$, then

$$E = \{ x \in \Re^n : (x - z)^T D^{-1} (x - z) \leq 1 \}$$

is an ellipsoid centered on z with $\text{vol}(E) = \sqrt{\det(D)} \times vol(B(0, 1))$

unit sphere

Key lemma. Every half-ellipsoid $\frac{1}{2} E$ is contained in an ellipsoid E' with $\text{vol}(E') / \text{vol}(E) \leq e^{-1/(2n+1)}$.



Shrinking lemma: unit sphere

Special case. E = unit sphere, $H = \{\, x : x_1 \geq 0 \,\}$.

$$E = \{\, x : \sum_{i=1}^{n} (x_i)^2 \leq 1 \,\} \qquad E' = \{\, x : \left(\tfrac{n+1}{n}\right)^2 (x_1 - \tfrac{1}{n+1})^2 + \tfrac{n^2-1}{n^2} \sum_{i=2}^{n} (x_i)^2 \leq 1 \,\}$$

Claim. E' is an ellipsoid containing $\tfrac{1}{2}\, E = E \cap H$.

Pf. If $x \in \tfrac{1}{2}\, E$:

$$\left(\frac{n+1}{n}\right)^2 \left(x_1 - \frac{1}{n+1}\right)^2 + \frac{n^2-1}{n^2} \sum_{i=2}^{n} x_i^2$$

$$= \frac{n^2+2n+1}{n^2} x_1^2 - \left(\frac{n+1}{n}\right)^2 \frac{2x_1}{n+1} + \frac{1}{n^2} + \frac{n^2-1}{n^2} \sum_{i=2}^{n} x_i^2$$

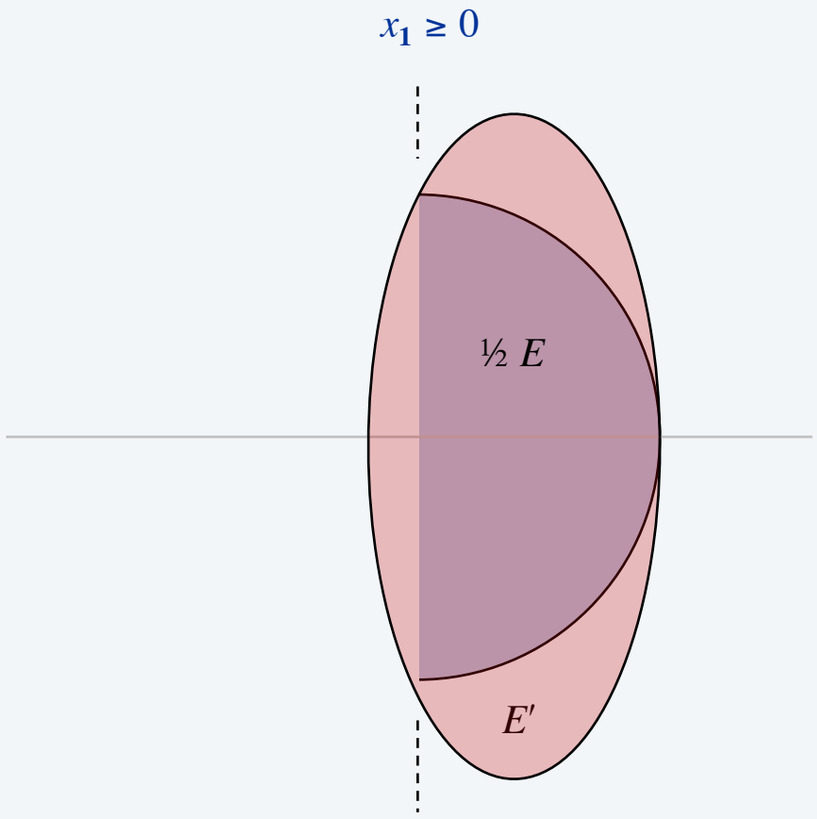
$$= \frac{2n+2}{n^2} x_1^2 - \frac{2n+2}{n^2} x_1 + \frac{1}{n^2} + \frac{n^2-1}{n^2} \sum_{i=1}^{n} x_i^2$$

$$= \frac{2n+2}{n^2} x_1 (x_1 - 1) + \frac{1}{n^2} + \frac{n^2-1}{n^2} \sum_{i=1}^{n} x_i^2$$

$$\leq 0 + \frac{1}{n^2} + \frac{n^2-1}{n^2}$$

$$= 1$$

$0 \leq x_1 \leq 1$ $\qquad$ $\sum x_i^2 \leq 1$



Shrinking lemma: unit sphere

Special case. E = unit sphere, $H = \{\, x : x_1 \geq 0 \,\}$.

$$E = \{\, x : \sum_{i=1}^{n} (x_i)^2 \leq 1 \,\} \qquad E' = \{\, x : \left(\tfrac{n+1}{n}\right)^2 (x_1 - \tfrac{1}{n+1})^2 + \tfrac{n^2-1}{n^2} \sum_{i=2}^{n} (x_i)^2 \leq 1 \,\}$$

Claim. E' is an ellipsoid containing $\tfrac{1}{2}\, E = E \cap H$.

Pf. Volume of ellipsoid is proportional to side lengths:

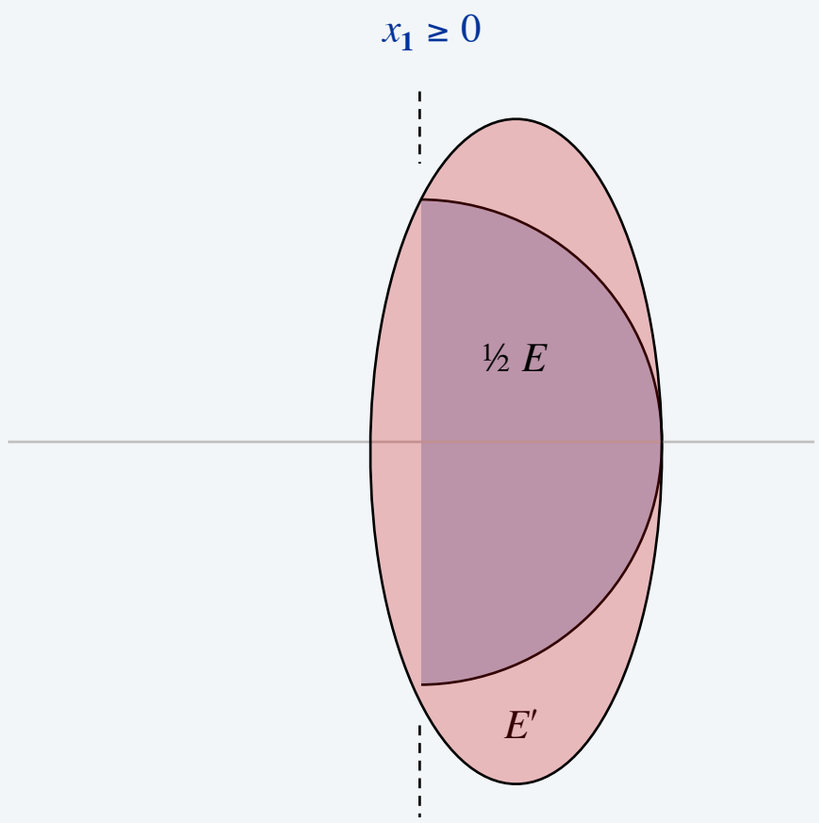
$$\frac{vol(E')}{vol(E)} = \left(\frac{n^2}{n^2-1}\right)^{\frac{n-1}{2}} \left(\frac{n}{n+1}\right)$$

$$= \left(1 + \frac{1}{n^2-1}\right)^{\frac{n-1}{2}} \left(1 - \frac{1}{n+1}\right)$$

$$\leq e^{\frac{1}{n^2-1}\frac{n-1}{2}} \; e^{\frac{-1}{n+1}}$$

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

$1 + x \leq e^x$



Shrinking lemma

Shrinking lemma. The min volume ellipsoid containing the half-ellipsoid $\frac{1}{2}E = E \cap \{ x : a \cdot x \le a \cdot z \}$ is defined by:

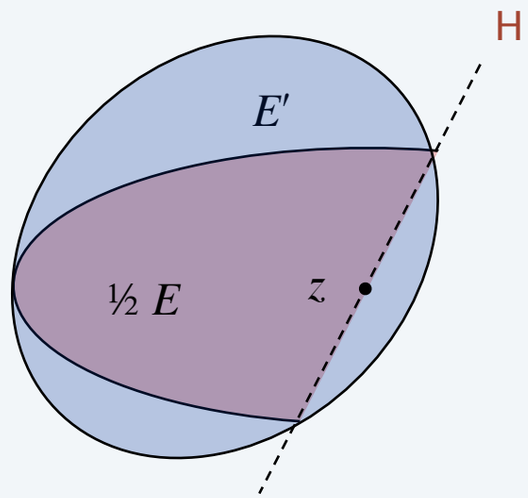
$$z' = z - \frac{1}{n+1}\frac{Da}{\sqrt{a^T Da}}, \quad D' = \frac{n^2}{n^2-1}\left(D - \frac{2}{n+1}\frac{Daa^T D}{a^T Da}\right)$$

$$E' = \{ x \in \Re^n : (x - z')^T (D')^{-1} (x - z') \le 1 \}$$

Moreover, $\mathrm{vol}(E') / \mathrm{vol}(E) < e^{-1/(2n+1)}$.

Pf sketch.

- We proved E = unit sphere, $H = \{ x : x_1 \ge 0 \}$
- Ellipsoids are affine transformations of unit spheres.
- Volume ratios are preserved under affine transformations.



Shrinking lemma

Shrinking lemma. The min volume ellipsoid containing the half-ellipsoid $\frac{1}{2} E = E \cap \{ x : \ a \cdot x \leq a \cdot z \}$ is defined by:

$$z' \ = \ z \ - \ \frac{1}{n+1} \ \frac{Da}{\sqrt{a^T Da}} \ , \quad D' \ = \ \frac{n^2}{n^2-1} \left(D \ - \ \frac{2}{n+1} \frac{Daa^T D}{a^T Da} \right)$$

$$E' \ = \ \{ \ x \in \Re^n : \ (x - z')^T \, (D')^{-1} (x - z') \ \leq \ 1 \ \}$$

Moreover, $\mathrm{vol}(E') \ / \ \mathrm{vol}(E) < e^{\ -1/(2n+1)}$.

Corollary. Ellipsoid algorithm terminates after at most $2(n+1) \ln (\mathrm{vol}(E_0) \ / \ \mathrm{vol}(P))$ steps.

Ellipsoid algorithm

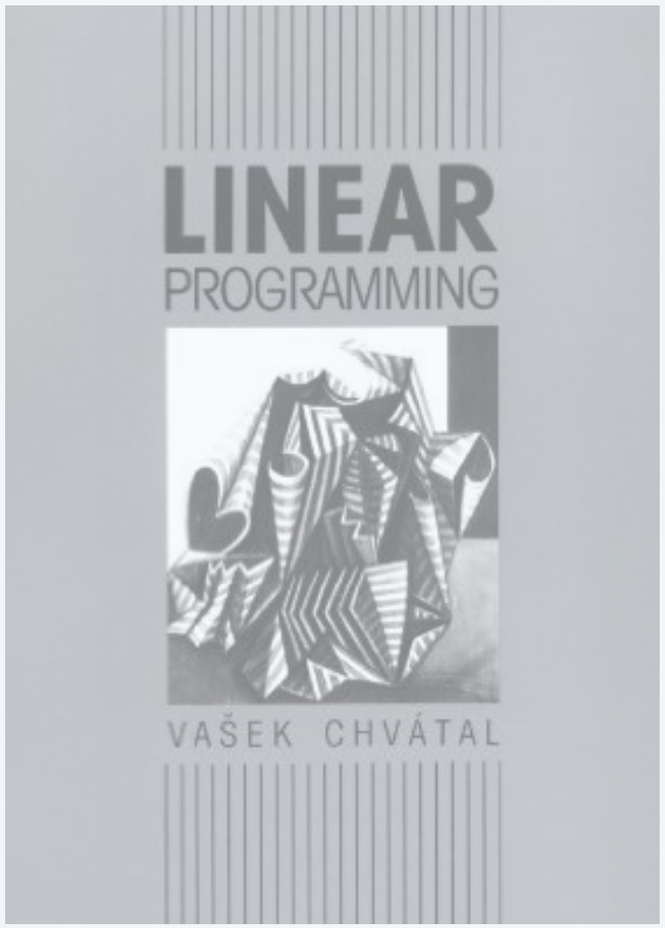
Theorem. LP is in **P**.

Pf sketch.

- Shrinking lemma.
- Set initial ellipsoid E_0 so that $\mathrm{vol}(E_0) \le 2^{cnL}$.
- Perturb $Ax \le b$ to $Ax \le b + \varepsilon \;\Rightarrow$ either P is empty or $\mathrm{vol}(P) \ge 2^{-cnL}$.
- Bit complexity (to deal with square roots).
- Purify to vertex solution.

Caveat. This is a theoretical result. Do not implement.

$O(mn^3 L)$ arithmetic ops on numbers of size $O(L)$,
where L = number of bits to encode input



Linear Programming III

- *ellipsoid algorithm*
- *combinatorial optimization*
- *matrix games*
- *open problems*

Lecture notes on the ellipsoid algorithm by Michel X. Goemans
`http://math.mit.edu/~goemans/18433S07/ellipsoid.pdf`

Separation oracle

Separation oracle. Given $x \in \Re^n$, assert x is in P or return a separating hyperplane.

Theorem. Let $S \subseteq \{0, 1\}^n$, $P = \text{conv}(S)$, and $c \in Z^n$. Assume that P is full-dimensional. There exists an algorithm that finds $\min \{ c^T x : x \in P \}$ using a poly number of ops and calls to separation oracle for P.

Remark. Don't need a polynomial representation of P.

Min s-t cut problem

Min s-t cut. Given digraph $G = (V, E)$, distinguished vertices s and t, and edge costs $c_e > 0$, find a min weight set of edges that intersects every s-t path.

$$\begin{aligned} \min\ & c^T x \\ \text{s. t.}\quad & \sum_{e \in P} x_e \geq 1 \quad \forall\ s-t \text{ paths } P \\ & x_e \geq 0 \end{aligned}$$

exponentially many constraints

Separation oracle. Shortest s-t path with weights x_e.

Min cost arborescence problem

Min cost arborescene. Given digraph $G = (V, E)$, distinguished vertex r, and edge costs $c_e > 0$, find a subgraph of G that contains a directed path from r to all other vertices.

$$\begin{aligned} \min\ & c^T x \\ \text{s. t.}\ & \sum_{\substack{e=(i,j)\in E \\ i\in S,\ j\notin S}} x_e \geq 1 \quad \forall S \subseteq V \text{ where } r \in S \\ & x_e \geq 0 \end{aligned}$$

exponentially many constraints

Separation oracle. Perform at most $n - 1$ min cut procedures with r as the source and edge weights x_e.

Note. Faster combinatorial algorithm exist.

Ellipsoid and combinatorial optimization

Grötschel–Lovász–Schrijver. Poly-time algorithms for:

- Network synthesis.
- Matroid intersection.
- Chinese postman problem.
- Min weight perfect matching.
- Minimize submodular set function.
- Stability number of a perfect graph.
- Covering of directed cuts of a digraph.
- ...

Totally unimodular matrices

Def. A matrix $A \in \Re^{m \times n}$ is totally unimodular if the determinant of each square submatrix is $0, +1$, or -1.

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

no

$$\begin{bmatrix} 1 & 0 & 1 & -1 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 \end{bmatrix}$$

yes

Totally unimodular matrices

Theorem. If A is totally unimodular and b is integral, then every vertex of $\{ Ax = b, x \geq 0 \}$ is integral.

Pf. Each vertex is a solution to $A_B x = b$ for invertible A_B.

Apply Cramer's rule.

Cramer's rule. For $B \in \Re^{n \times n}$ invertible, $b \in \Re^n$,

the solution to $Bx = b$ is given by:

$$x_i = \frac{\det(B_i)}{\det(B)}$$

replace ith column of B with b

Totally unimodular matrices

Theorem. A $(0, +1, -1)$ matrix is totally unimodular if it contains at most one $+1$ and at most one -1 in each column.

$$\begin{bmatrix} 1 & 0 & 1 & -1 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 \end{bmatrix}$$

Pf. [induction on size of a square submatrix B]

- Base case: 1-by-1 matrix.
- Case 1: a column of B is all 0s.
- Case 2: a column of B has exactly one 1 (or one −1).
- Case 3: all columns of B have exactly one 1 and one −1.

Ex. Network flow matrices.

Assignment problem

Assignment problem. Assign n jobs to n machines to minimize total cost, where c_{ij} = cost of assignment job j to machine i.

	1'	2'	3'	4'	5'
1	3	8	9	15	10
2	4	10	7	16	14
3	9	13	11	19	10
4	8	13	12	20	13
5	1	7	5	11	9

cost = 3 + 10 + 11 + 20 + 9 = 53

	1'	2'	3'	4'	5'
1	3	8	9	15	10
2	4	10	7	16	14
3	9	13	11	19	10
4	8	13	12	20	13
5	1	7	5	11	9

cost = 8 + 7 + 20 + 8 + 11 = 44

Assignment problem: LP formulation

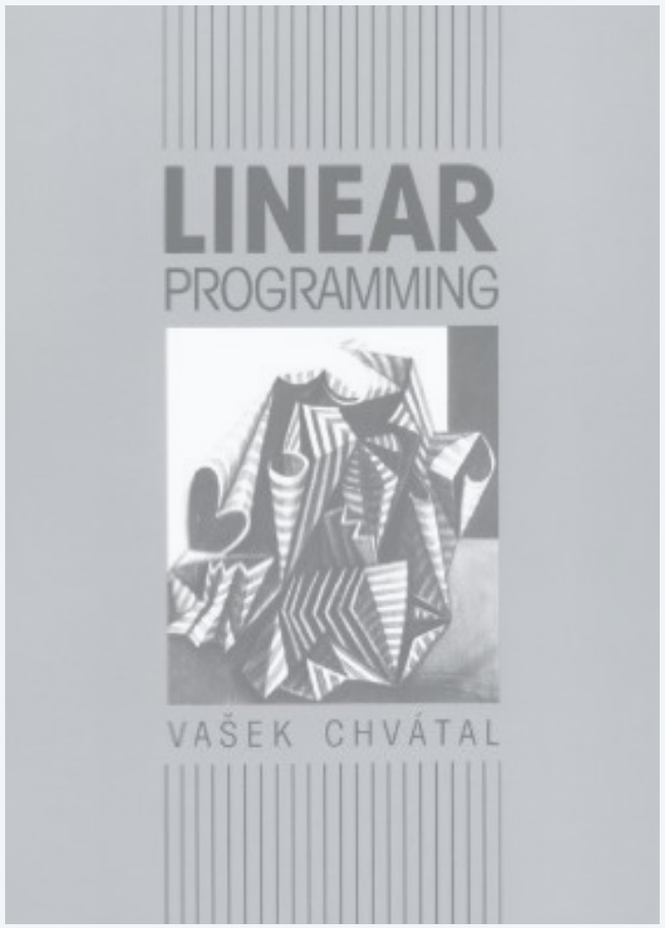
$$
\begin{aligned}
(P)\quad \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} & & & \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} & = & 1 & 1 \le i \le n \\
& \sum_{i=1}^{m} x_{ij} & = & 1 & 1 \le j \le n \\
& x_{ij} & \ge & 0 & 1 \le i, j \le n
\end{aligned}
$$

Interpretation: if $x_{ij} = 1$, then assign job j to machine i

Theorem. [Birkhoff 1946, von Neumann 1953] All vertices of the above polyhedron are $\{0\text{–}1\}$-valued.

Pf. Total unimodularity.

Corollary. There exist poly-time algorithm for assignment problem.



LINEAR PROGRAMMING III

- *ellipsoid algorithm*
- *combinatorial optimization*
- *matrix games*
- *open problems*

Matrix games

Matrix game. For $A \in \Re^{m \times n}$, define a game for two players.

- Row player A selects one of rows $i = 1, 2, \ldots, m$.
- Column player B selects one of columns $j = 1, 2, \ldots, n$.
- Payoff to row player is a_{ij}.

		column player B	
		1	2
row player A	1	-2	+3
	2	+3	-4

Ex. A plays row 2: B plays column 2; A loses $4.

Ex. A plays row 1: B plays column 1; A loses $2.

A can guarantee to lose at most $2.

Matrix games

Matrix game. For $A \in \Re^{m \times n}$, define a game for two players.

- Row player A selects one of rows $i = 1, 2, \ldots, m$.
- Column player B selects one of columns $j = 1, 2, \ldots, n$.
- Payoff to row player is a_{ij}.

		column player B	
		1	2
row player A	1	-2	+3
	2	+3	-4

Ex. A plays row 1 with prob 3/5 and row 2 with prob 2/5.

- B plays column 1: A wins = -2(3/5) + 3(2/5) = 0.
- B plays column 2: A wins = +3(3/5) - 4(2/5) = 1/5.

A can guarantee to lose at most $0 (in expectation).

Matrix games

Matrix game. For $A \in \Re^{m \times n}$, define a game for two players.

- Row player A selects one of rows $i = 1, 2, \ldots, m$.
- Column player B selects one of columns $j = 1, 2, \ldots, n$.
- Payoff to row player is a_{ij}.

		column player B	
		1	2
row player A	1	-2	+3
	2	+3	-4

Ex. A plays row 1 with prob 7/12 and row 2 with prob 5/12.

- B plays column 1: A wins = -2(7/12) + 3(5/12) = 1/12 on average.
- B plays column 2: A wins = +3(7/12) - 4(5/12) = 1/12 on average.

A can guarantee to win 1/12 in expectation.

Matrix games

Matrix game. For $A \in \Re^{m \times n}$, define a game for two players.

- Row player A selects one of rows $i = 1, 2, \ldots, m$.
- Column player B selects one of columns $j = 1, 2, \ldots, n$.
- Payoff to row player is a_{ij}.

		column player B	
		1	2
row player A	1	-2	+3
	2	+3	-4

Ex. B plays column 1 with prob 7/12 and column 2 with prob 5/12.

- A plays row 1: B loses = -2(7/12) + 3(5/12) = 1/12 on average.
- A plays row 2: B loses = +3(7/12) - 4(5/12) = 1/12 on average.

B can guarantee to lose at most 1/12.

Matrix games

Matrix game. For $A \in \Re^{m \times n}$, define a game for two players.

- Row player A selects one of rows $i = 1, 2, \ldots, m$.
- Column player B selects one of columns $j = 1, 2, \ldots, n$.
- Payoff to row player is a_{ij}.

Pure strategy. Player chooses a given row (or column).

Mixed strategy. Player chooses a row (or column) at random, according to some probability distribution $x \in \Delta_m$ (or $y \in \Delta_n$).

Expected payoff. $\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} x_i y_j = y^T A x$

$$\Delta_p = \left\{ z \in \Re^p \ : \ \sum_{k=1}^{p} z_k = 1, \ z_k \geq 0 \right\}$$

stochastic vector

Matrix games: LP formulation

Row player strategy. If row player uses strategy x, he guarantees an expected payoff of $\min_{y \in \Delta_n} y^T A x$ so, goal is to find

$$\max_{x \in \Delta_m} \min_{y \in \Delta_n} y^T A x$$

nested min max

not linear in x, y

Observation. If row player uses fixed strategy x, then column player wants to solve linear program:

fixed vector

$$\begin{aligned} \min \; & y^T A x \\ \text{s.t.} \; & \sum_{j=1}^{n} y_j = 1 \\ & y \geq 0 \end{aligned}$$

All vertices of this LP are unit vectors (pure strategies). Thus

$$\min_{y \in \Delta_n} y^T A x = \min_{j} \sum_{i=1}^{m} a_{ij} x_i$$

Matrix games: LP formulation

Optimal strategy for row player:

$$\begin{aligned} (\mathrm{P}) \quad \max_{x} \ \min_{j} \ & \sum_{i=1}^{m} a_{ij}\, x_i \\ \text{s.t.} \ & \sum_{i=1}^{m} x_i = 1 \\ & x \geq 0 \end{aligned}$$

every optimal solution (x^*, z^*) to (P) satisfies at least of one these constraints with equality, so $z^* = \sum a_{ij}\, x_j^*$

Equivalent to following linear program:

$$\begin{aligned} (\mathrm{P}') \quad \max_{x,\, z} \ & z \\ \text{s.t.} \ & \sum_{i=1}^{m} a_{ij}\, x_i \geq z \qquad (j = 1,2,\ldots,n) \\ & \sum_{i=1}^{m} x_i = 1 \\ & x_i \geq 0 \qquad (i = 1,2,\ldots,m) \end{aligned}$$

Matrix games

Optimal strategy for row player:

$$
\begin{aligned}
(\mathrm{P}') \quad \max_{x,\, z} \quad & z \\
\text{s.t.} \quad & \sum_{i=1}^{m} a_{ij}\, x_i \;\geq\; z \\
& \sum_{i=1}^{m} x_i \;=\; 1 \\
& x_i \;\geq\; 0
\end{aligned}
$$

Optimal strategy for column player:

$$
\begin{aligned}
(\mathrm{D}') \quad \min_{y,\, t} \quad & t \\
\text{s.t.} \quad & \sum_{j=1}^{n} a_{ij}\, y_j \;\leq\; t \\
& \sum_{j=1}^{n} y_j \;=\; 1 \\
& y_j \;\geq\; 0
\end{aligned}
$$

Observation. (P′) and (D′) are LP duals!

Minimax theorem

Theorem. [von Neumann 1928] For every $A \in \Re^{m \times n}$,

$$\max_{x \in \Delta_m} \min_{y \in \Delta_n} y^T A x = \min_{y \in \Delta_n} \max_{x \in \Delta_m} y^T A x$$

Pf. LP duality.

Consequence. As long as your mixed strategy is optimal, you can reveal it to your opponent.

Theorem. Nash equilibrium exist for 2-person zero-sum games. Moreover, they are poly-time computable.

Application: poker

Kuhn's simplified poker.

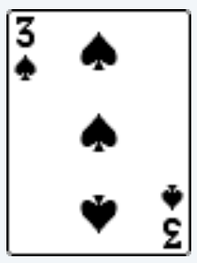
- Deck of 3 cards, numbered 1, 2, and 3.
- Each player antes $1.
- One round of betting ($1 bet).
- If pass-pass, pass-bet-bet, or bet-bet, player with higher card wins; otherwise player that bet wins.

Strategies for X.

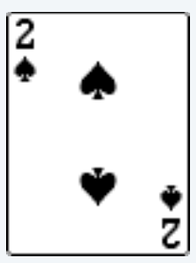
1. Pass; if Y bets; pass.
2. Pass; if Y bets, bet.
3. Bet.

Strategies for Y.

1. Pass no matter what X did.
2. If X passes, pass; if X bets, bet.
3. If X passes, bet; if X bets, pass.
4. Bet no matter what X did.



X



Y

Application: poker

Optimal strategy for X.

- When dealt 1, mix strategies 1 and 3 in ratio 5:1.
- When dealt 2, mix strategies 1 and 2 in ratio 1:1.
- When dealt 3, mix strategies 2 and 3 in ratio 1:1.

bluff 17% of time

trap 50% of time

Optimal strategy for Y.

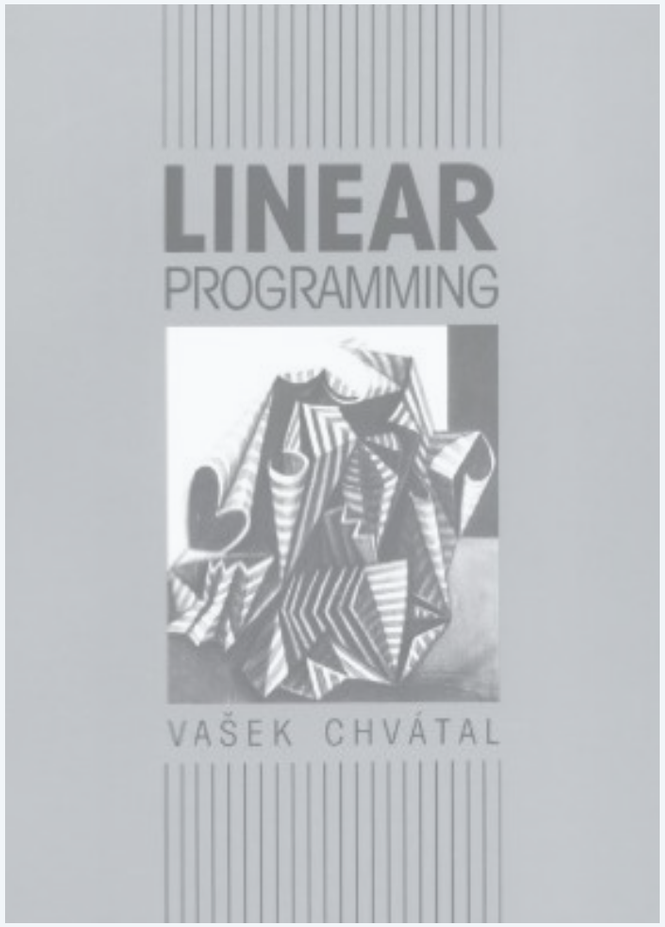
- When dealt 1, mix strategies 1 and 3 in ratio 2:1.
- When dealt 2, mix strategies 1 and 2 in ratio 2:1.
- When dealt 3, use strategy 4.

bluff 33% of time

Value of game. −1/18 for X.

Gambling lessons. Optimal strategies involve bluffing and trapping.

Player who acts last has advantage.



LINEAR PROGRAMMING III

- *ellipsoid algorithm*
- *combinatorial optimization*
- *matrix games*
- *open problems*

Strongly polynomial

An algorithm is strongly polynomial if:

- Elementary ops: +, −, *, /, comparison.
- # ops is polynomial in the dimension of input.
- Polynomial space on a classic TM.

Ex. Mergesort: $O(n \lg n)$.

Ex. Edmonds–Karp max-flow: $O(m n^2)$.

Ex. Gaussian elimination: $O(n^3)$ arithmetic ops.

weakly polynomial

Ex. Ellipsoid: $O(m n^3 L)$ arithmetic ops.

Ex. Ye's interior point method: $O(n^3 L)$ arithmetic ops.

Open problem. Strongly-polynomial algorithm for LP ?

Open problem. Is LP in $\mathbf{P}_{\mathfrak{R}}$?

New York Times article

An Approach to Difficult Problems

Mathematicians disagree as to the ultimate practical value of Leonid Khachiyan's new technique, but concur that in any case it is an important theoretical accomplishment.

Mr. Khachiyan's method is believed to offer an approach for the linear programming of computers to solve so-called "traveling salesman" problems. Such problems are among the most intractable in mathematics. They involve, for instance, finding the shortest route by which a salesman could visit a number of cities without his path touching the same city twice.

Each time a new city is added to the route, the problem becomes very much more complex. Very large numbers of variables must be calculated from large numbers of equations using a system of linear programming. At a certain point, the compexity becomes so great that a computer would require billions of years to find a solution.

In the past, "traveling salesmen" problems, including the efficient scheduling of airline crews or hospital nursing staffs, have been solved on computers using the "simplex method" invented by George B. Dantzig of Stanford University.

As a rule, the simplex method works well, but it offers no guarantee that after a certain number of computer steps it will always find an answer. Mr. Khachiyan's approach offers a way of telling right from the start whether or not a problem will be soluble in a given number of steps.

Two mathematicians conducting research at Stanford already have applied the Khachiyan method to develop a program for a pocket calculator, which has solved problems that would not have been possible with a pocket calculator using the simplex method.

Mathematically, the Khachiyan approach uses equations to create imaginary ellipsoids that encapsulate the answer, unlike the simplex method, in which the answer is represented by the intersections of the sides of polyhedrons. As the ellipsoids are made smaller and smaller, the answer is known with greater precision. MALCOLM W. BROWNE