Linear Programming

סיכומי
הרצאות

Lecture notes: http://www.cs.cmu.edu/afs/cs/user/glmiller/public/Scientific-Computing/F-11/RelatedWork/Goemans-LP-notes.pdf

Linear Programming

- Linear Programming is the task or **maximizing (or minimizing a linear function** subject to **linear constraints**
- It is a very general problem with many applications
- For many years, it was open whether there's an efficient algorithm for this problem – this was settled in the late 1970's
- (Although, this problem was solvable in practice. We will see an algorithm which has worst case exponential complexity but works well in practice)

Linear Optimization

- Variables: $x_1, \ldots, x_n \in \mathbb{R}$
- Given **linear** constraints (equalities and inequalities) on these variables:
  - E.g., $x_1 \geq 0, 3x_2 + 4x_3 - x_7 \geq 6, x_4 + x_1 = 6$ ...
- We want to maximize/minimize an objective **linear function**
  - $3x_1 + 5x_2 + 2x_3 - 4x_4 + \cdots$

Examples

- Diet Problem:

Grams per kg:	**Rice**	**Corn**	**Wheat**
Starch	5	10	4
Proteins	2	5	3
Vitamins	3	7	1
Cost ($/kg)	9	8	7

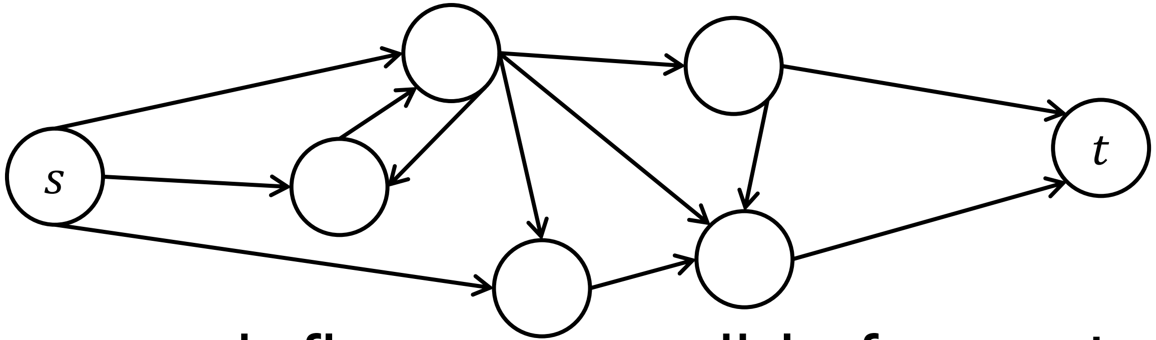
- We want our daily diet to contain at least 16, 20 and 3 grams of starch, proteins and vitamins, respectively
- While minimizing the cost…

Linear Program Formulation

- Our variables: x_R, x_C, x_W (each corresponding to how many kgs of rice, corn, and wheat we eat, respectively)
- Objective function to minimize: $9x_R + 8x_C + 7x_W$ למזער את מחיר הארוחה
- Constraints:
  - $5x_R + 10x_C + 4x_W \geq 16$ (Starch constraint) לפחות 16 גרם עמילן
  - Similarly for proteins and vitamins
  - Also, $x_R \geq 0, x_C \geq 0, x_W \geq 0$ (we can't eat a negative amount)

Another example: Max flow

- Given a directed graph G with a source s, sink t and edge capacities c_{uv}:



- Transfer as much flow as possible from s to t
- Subject to capacities and flow conservation constraints

אילוצי קיבולת אילוצי שימור זרימה

Linear Program Formulation

- Variables: for each edge (u, v) a variable f_{uv} for the amount of flow passed from u to v
- **Maximize** $\sum_{v:s\to v} f_{sv}$
- **Capacity** constraints: $0 \leq f_{uv} \leq c_{uv}$
- **Flow conservation** constraints: $\sum_{u:u\to v} f_{uv} = \sum_{w:v\to w} f_{vw}$ for every $v \neq s, t$

  סכום הקיבולות שיוצאות מ-v לשכנים w = סכום הקיבולות שנכנסים ל-v משכנים u
- Optimal solution = optimal flow

Representing Linear Programs

- As we know from linear algebra, it's convenient to represent a set of linear equations using matrices as $Ax = b$
- Similarly a set of inequalities can be represented as "$Ax \leq b$"
- If the objective is to minimize $c_1x_1 + \ \ldots + c_nx_n$, we can instead maximize $-c_1x_1 - \cdots - c_nx_n$ and vice versa

$\min \sum x$
$= \max - \sum x$

- Similarly we can reverse direction of inequalities
- We can transform an inequality $a_1x_1 + \cdots + a_nx_n \leq b$ to an equality $a_1x_1 + \cdots + a_nx_n + s = b$ and inequality $s \geq 0$
  - s is called a **slack variable**

slack variable (מש עודף)

- We can restrict every variable x to be non-negative by replacing x with $x^+ - x^-$ where $x^+, x^- \geq 0$

$X = X^+ - X^-$

מטיבציה!
אם רק מותר "≥" או "="

Standard Forms

- We say a linear program is in **standard form** if it is of the form:

  Minimize $c^T x$

  Subject to $Ax = b$, $x \geq 0$

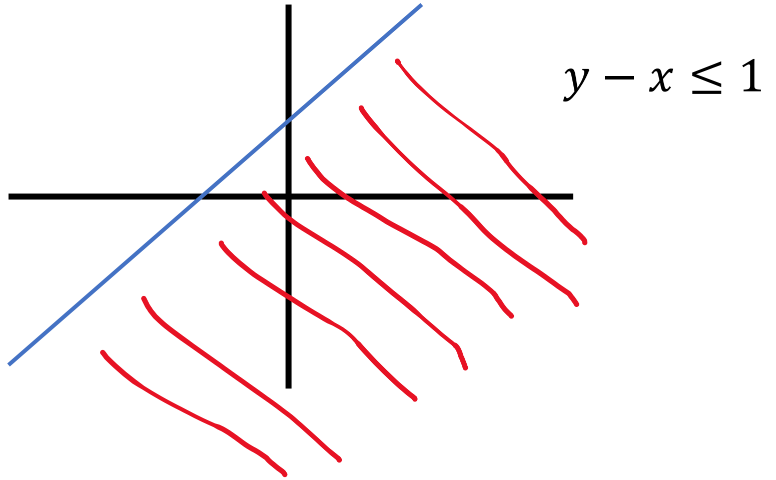
- As seen earlier, it's easy to transform any LP to standard form
- We can also transform equalities into inequalities and consider programs of the form

  Minimize $c^T x$

  Subject to $Ax \leq b$

Geometry of Linear Programs

- A set defined by a single linear inequality is called an **hyperplane**

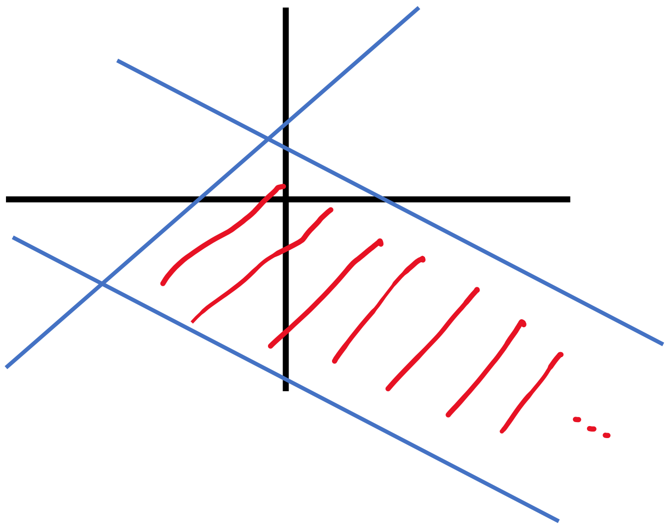


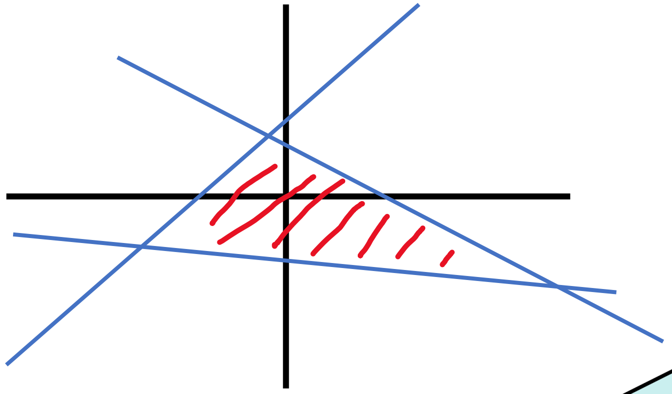
- A set defined by a system of linear inequalities is called a **polyhedron**

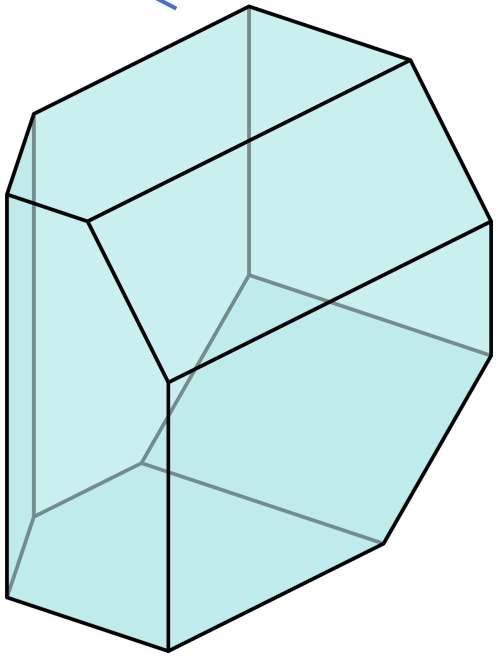
הערה!

קבוצה סופית של אש"ל לינארים

Polyhedra







- A bounded polyhedron is called a **polytope**

Feasible solutions

- A vector x which satisfies all the constrains of an LP is called a **feasible solution**

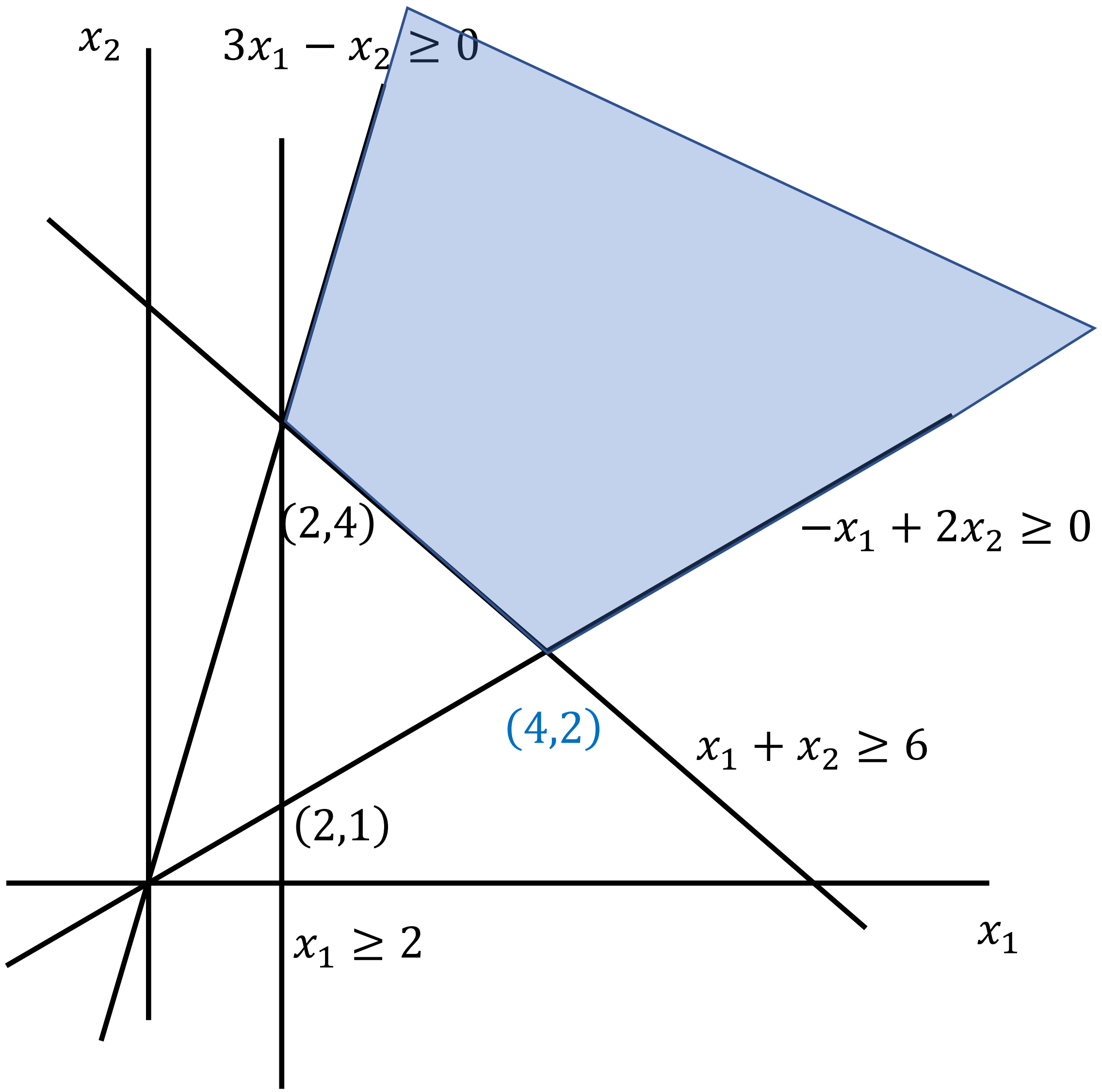
הערה: פיזיבלי לאו דווקא אופטימלי

- If there's a feasible solution the LP is called **feasible**
- Not every LP is feasible!
  - E.g., a trivial example are the constraints $x_1 \geq 1$ and $x_1 \leq 0$.
- A minimizing (or maximizing) feasible solution is called an **optimal solution**
- The set of feasible points of an LP is a polyhedron

Example

Minimize x_2

Subject to $\begin{cases} x_1 \geq 2 \\ 3x_1 - x_2 \geq 0 \\ x_1 + x_2 \geq 6 \\ -x_1 + 2x_2 \geq 0 \end{cases}$



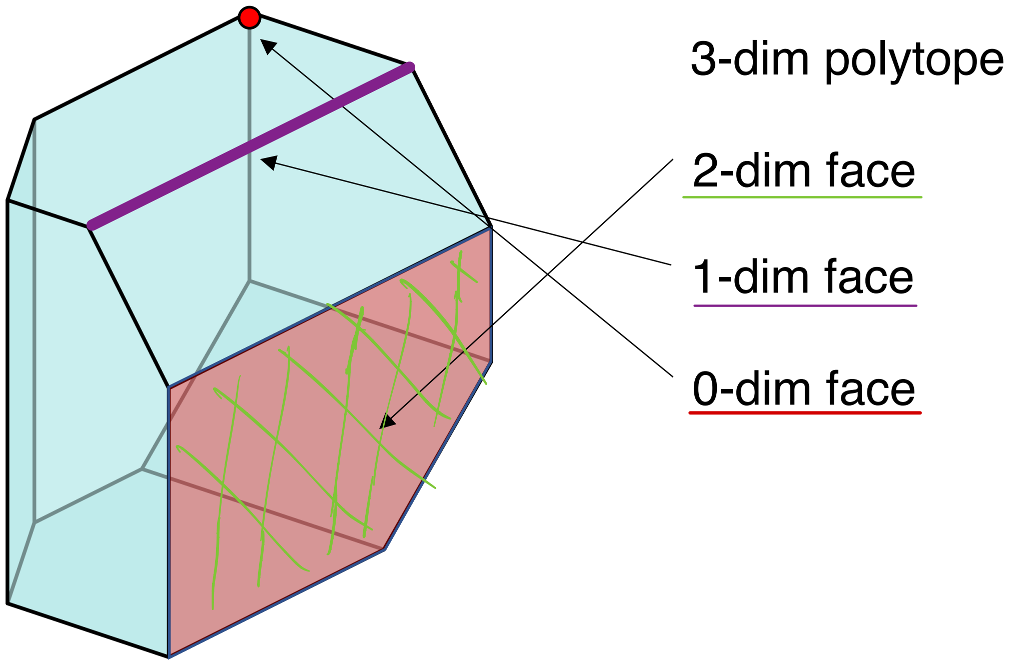
The optimal solution is $(4,2)$ with cost of 2
If we were to maximize x_2, the optimal solution is ∞ (we then say the LP is **unbounded**)

פאה

Faces of polytopes

A **face** of a polytope is a set of points obtained by replacing some of the inequalities with equalities

⇐מחליפים גם בשוויון

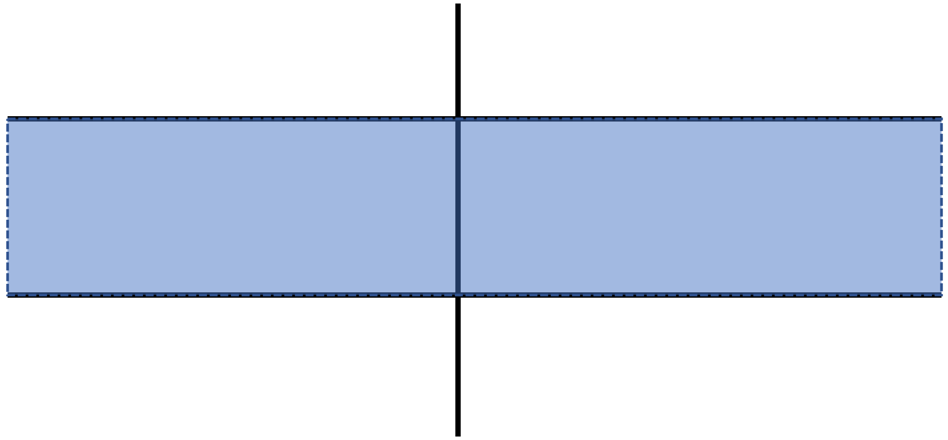


Faces of polytopes

- The inequalities that are satisfied with equality by a face are called the **tight constraints** for the face
- A zero-dimensional face is called a **vertex** or a **corner**
- A vertex satisfies n linearly-independent constraints with equality
- Equivalently: x is a vertex of P is there does **not** exist $y \neq 0$ such that $x + y, x - y \in P$

Vertices

- Not every polyhedron has a vertex. E.g., $\{x: 0 \leq x_1 \leq 1\}$



- However, we will prove that if an LP in standard form has an optimal solution, then it is obtained by a vertex

Optimal solution is a vertex

Theorem:

Let $P = \{x : Ax = b, x \geq 0\} \subseteq \mathbb{R}^n$.

Suppose $\min\{c^T x : x \in P\}$ is finite. Then for all $x \in P$ there's a vertex $x' \in P$ such that $c^T x' \leq c^T x$.

צומת

x סתם

Proof: If x is a vertex we are done. $\left(c^T x' = c^T x \quad \text{בפרט} \right)$

Otherwise, there's $y \neq 0$ such that $x + y, x - y \in P$. Since $A(x + y) = b$ and $A(x - y) = b$, $Ay = 0$.

$w = x + y \in P$
ולכן מקיים בפרט
אילוצים $Aw = b$

Optimal solution is a vertex

צועדים בכיוון y

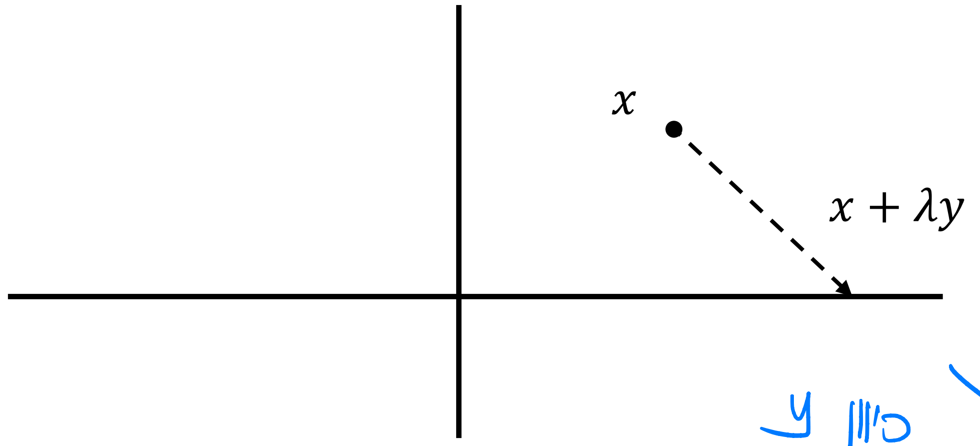
Suppose wlog $c^T y \leq 0$ (if necessary, replace y with $-y$).
Consider $x + \lambda y$ for $\lambda > 0$.

$$c^T(x + \lambda y) = c^T x + \overbrace{\lambda c^T y}^{<0} \leq c^T x.$$

Case 1: there's $1 \leq j \leq n$ such that $y_j < 0$. As λ increases, the j-th coordinate decreases until $x + \lambda y$ is no longer feasible:

לפי הגדרת ה-LP
בצורה הסטנדרטית
$x + \lambda y \geq 0$.

אם נגדיל את λ מספיק,
אז אחד הx׳ שלילית יעבור



כיוון y

Optimal solution is a vertex

$X + \lambda y_j = X - \frac{x_j \cdot y_j}{y_j} = 0$

Choose $\lambda = \min_{\{j: y_j < 0\}} \left\{ \frac{x_j}{-y_j} \right\}$.

This is the largest λ such that $x + \lambda y \geq 0$, and

$$A(x + \lambda y) = Ax + \lambda Ay = b + 0 = b,$$

so $x + \lambda y \in P$. $x + \lambda y$ has one more zero coordinate than x.

This case can happen at most n times until reaching a vertex.

Optimal solution is a vertex

Case 2: $y_j \geq 0$ for all $1 \leq j \leq n$.

We may further assume $c^T y < 0$ (if $c^T y = 0$, replace y with $-y$ and we're back at case 1).

Also note that $x + \lambda y$ is feasible for all $\lambda > 0$ since $x + \lambda y \geq 0$ and $A(x + \lambda y) = b$.

But now:

$$c^T(x + \lambda y) = c^T x + \lambda c^T y \to -\infty$$

implying the LP is unbounded, a contradiction.

Remarks

- The theorem was phrased and proved for LPs in standard form $\{Ax = b, x \geq 0\}$
- As we observed – it's not necessarily true for general polytopes since they don't even always have a vertex
- Nevertheless, one can similarly show that the set of optimal solution is always a face of P. The proof is similar but more technical
- For our purposes right now it's enough to consider LPs in standard form

x קודקוד $\iff$ A_x מדרגה מלאה

Linear algebraic criterion for vertex

כיוון ראשון:

A_x דרגה מלאה $\Leftarrow$ x קודקוד

$\Updownarrow$

x לא קודקוד $\Leftarrow$ A_x לא מדרגה מלאה

Lemma: Let $P = \{x : Ax = b, x \geq 0\}$. For $x \in P$, let A_x be the submatrix of A corresponding to j s.t. $x_j > 0$. Then x is a vertex iff A_x has full column rank.

נניח בשלילה ש-x לא קודקוד, ו-A_x מדרגה מלאה.

Proof: If x isn't a vertex, $\exists$ non-zero y such that $x + y, x - y \in P$. Let A_y be the submatrix of A corresponding to cols s.t. $y_j \neq 0$.

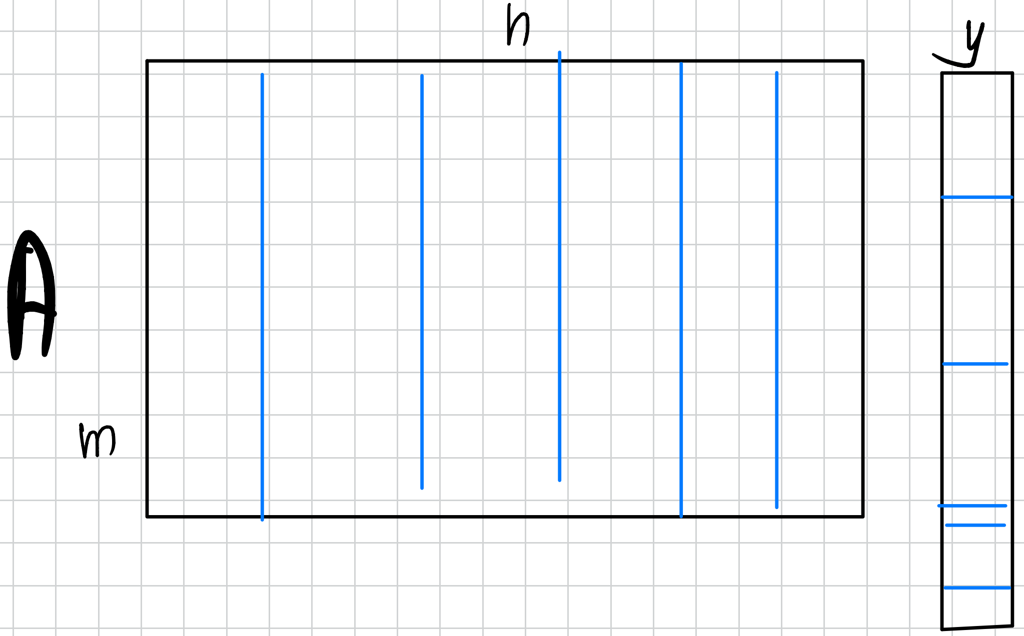
As before, $A(x + y) = A(x - y) = b \Rightarrow Ay = 0$ and thus A_y has dependent columns.

צירוף לינארי של עמודות A לפי מקדמים של y

Further $x + y \geq 0, x - y \geq 0$ so $y_j = 0$ if $x_j = 0$. Therefore, A_y is a submatrix of A_x.

סתירה.

דוגמה: x לא קודקוד $\Leftarrow$ A_x עמודות ת"ל.



לפי הגדרה $\exists y\neq 0$ כך ש-

$$x+y\in P$$
$$x-y\in P$$

$$A(x+y)=A(x-y)=b$$
$$\Downarrow$$
$$Ay=0$$
$$\Downarrow$$

עמודות A_y ת"ל

הבחנה: A_y תת מטריצה של A_x !

אם קיים j כך ש- $y_j\neq 0$ $\Leftarrow$ $x_j>0$ ($x+y\geq 0$, $x-y\geq 0$ לכל קואורדינטה)

$x_j=0$ $\Leftarrow$ $y_j=0$ בהכרח · $y_j\geq 0 \cap y_j\leq 0$

Linear algebraic criterion for vertex

כיוון שני

x קודקוד $\Leftarrow$ A_x דרגה מלאה

A_x עמודות ת"ל $\Leftarrow$ x לא קודקוד

נק' כלשהי

Lemma: Let $P = \{x : Ax = b, x \geq 0\}$. For $x \in P$, let A_x be the submatrix of A corresponding to j s.t. $x_j > 0$. Then x is a vertex iff A_x has full column rank.

ממש, לא $\geq$

חלק מהאילוצים מתקיימים עם שוויון, וחלק ממש

הערה: חישוב דרגה של מטריצות נעשית בזמן פולינומי

כיוון שני

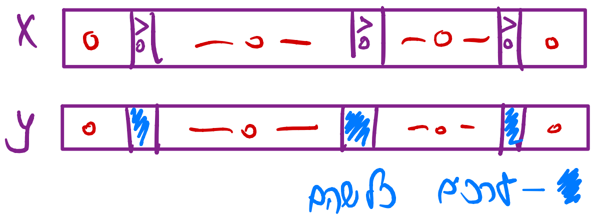
Conversely, suppose A_x has lin. dep. cols. Then $\exists y \neq 0$ such that $A_x y = 0$. Extend y to $\mathbb{R}^n$ by adding 0's. Then $\exists y \in \mathbb{R}^n$ such that $Ay = 0, y \neq 0$ and $y_j = 0$ whenever $x_j = 0$.

y עם

ס'ם

For small $\varepsilon > 0$, $x + \varepsilon y, x - \varepsilon y \in P$. Hence x isn't a vertex.

כפי שטענו

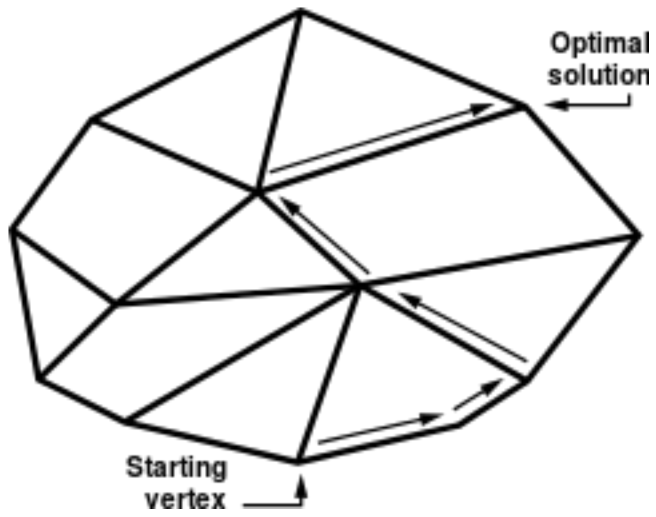


אפשר לבחור ε קטן מספיק כך ש-

$x \pm \varepsilon y \geq 0$, $\in P$ $\Longrightarrow$ x לא קודקוד

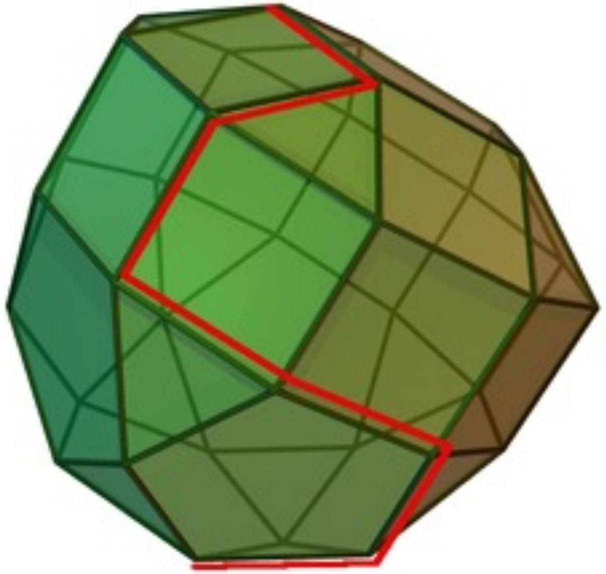
The simplex method

- We will describe an algorithm called **the simplex algorithm** for solving linear programs
- Intuitively, the algorithm starts from a vertex and walks along the edges until it finds an optimal solution



The simplex method

- We can’t show that the algorithm runs in polynomial time but in practice it’s very efficient
- It is considered one of the most important algorithms in history
- We need a few more technical definitions before introducing it



לפי הגדרה A של העמודות למרחב בסיס -B

מס' הקורדינטות החיוביות של z הוא לכל היותר m. נניח $=m$.

Bases

Polytope

Let z be a vertex of $P = \{x : Ax = b, x \geq 0\}$.

Suppose: A is $m\times n$ of rank m, and z has m positive coordinates.

Let $B = \{j : z_j > 0\}$ and $A_B = A_z$.

z קודקוד $\Leftarrow$ A_B מטריצה מלאה $\Leftarrow$ הפיכה

לא סינגולרית $\Leftarrow$ ריבועית עם דטרמיננטה $\neq 0$ $\Leftarrow$ הפיכה

The matrix A_B is $m\times m$ and non-singular and $A_B z_B = b$.

B now fully describes z:

$$\text{“}z_j = 0 \text{ if } j \notin B \text{ and } z_B = (A_B)^{-1}b\text{”}$$

The variables corresponding to B are called **basic**

The variables corresponding to $N = \{1, \ldots, n\} \setminus B$ are **nonbasic.**

יכולים להיות קודקודים עם פחות מ-m קורדינטות ב"ת

נשלים לבסיס עם עמודות ב"ת של A.

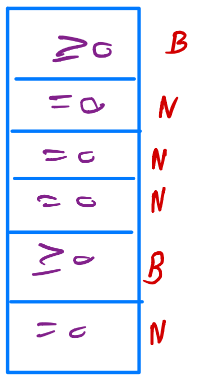
$|B| = m$

Bases

If z has $< m$ non-zero coords., we can augment the set B by adding lin. indep. cols. of A.

Summary: z is a vertex iff there's $B \subseteq \{1, \dots, n\}$ of size m and:

1. $z_N = 0$ for $N = \{1, \dots, n\} \setminus B$ מתאפס למשתנים לא בסיסיים
2. A_B is non-singular הפיך A_B
3. $z_B = (A_B)^{-1} b \geq 0.$ אי שלילי בכל הקורדינטות



We call z a **basic feasible solution.**

(Total number of vertices $\leq \binom{n}{m}$) אקספוננציאלי

The simplex method

Let z be a basic feasible solution.

Some constraints in z are tight:

אילוצים הדוקים מתקיימים עם "="

$$(z_1 = 0, z_7 = 0, z_{11} = 0, \dots)$$

Some are not: loose

אילוצים לא הדוקים מתקיימים עם ">"

$$(z_2 > 0, z_3 > 0, \dots)$$

Relax tight constraint by increasing its variable until a non-tight constraint becomes **tight**. When this happens, we moved to a different basic feasible solution.

צעד אחד באלגוריתם הסימפלקס

סה"כ יש 3 שלבים:

1. loosening
2. tightening
3. swapping

ברגע שמרככים אילוץ אחד,
מקשיחים אילוץ אחר עד שהוא הופך להיות הדוק.
ברגע שהוא הופך להיות הדוק, הוא נכנס לבסיס, וזה שריככנו יוצא. לכן קוראים החלפה.

The simplex method

Recall: we have a basic feasible solution z with a basis B
(we'll later speak about how to get an initial solution)

Rewrite the program as:

Minimize $(c_B)^T x_B + (c_N)^T x_N$

מפרידים בין המשתנים x_B , x_N

Subject to $A_B x_B + A_N x_N = b$

$$x_B, x_N \geq 0$$

תזכורת:
z בקורדינטות N שווה ל-0,
וגם $c_N^T z = 0$
$A_N z = 0$

The simplex method

Minimize $(c_B)^T x_B + (c_N)^T x_N$

Subject to $A_B x_B + A_N x_N = b$

$x_B, x_N \geq 0$

$A_B x_B = b - A_N x_N$

הפיכה

Recall that A_B is non-singular: $x_B = A_B^{-1} b - A_B^{-1} A_N x_N$, and

פונקציה לינארית רק במשתנים של N

פונקציית מטרה. נציב x_B.

$$c^T x = (c_B)^T x_B + (c_N)^T x_N$$
$$= c_B^T (A_B^{-1} b - A_B^{-1} A_N x_N) + c_N^T x_N$$

קבוע, לא תלוי במשתנים

$$= c_B^T A_B^{-1} b + (c_N^T - c_B^T A_B^{-1} A_N) x_N$$

$c^T x = \bullet + \bullet$

קבוע

Reduced Cost Vector

Call $\tilde{c}_N^T = c_N^T - c_B^T A_B^{-1} A_N$ the **reduced cost vector**

הבענו: $C^TX = \boxed{\text{מספר}} +$ פונקציה לינארית במשתנים של X_N

דוגמא:

מספר קבוע

$$C^TX = \boxed{50} + 5X_1 + 7X_5 + 13X_9$$

תזכורת: $X_j \geq 0$ $\forall j \in N$:

נשים לב: אם כל המקדמים חיוביים $\Leftarrow$ פתרון אופטימלי

אם קיים מקדם שלילי $\Leftarrow$ שווה להגדיל את ה-X_j המתאים לו (כמה שאפשר) כדי להקטין את פונקצית המטרה

הלכה:

כשמשנים קורדינטה של X ששייך ל-N, גם קורדינטות ב-B משתנות

(loosening $\Updownarrow$ tightening)

הבחירה של ה-X-ים שמשנים נקראת כלל pivot

The simplex method

$$c^T x = c_B^T A_B^{-1} b + \tilde{c}_N^T x_N$$

אם מציבים $x_N = z_N$

Where $\tilde{c}_N^T = c_N^T - c_B A_B^{-1} A_N$. Since $z_N = 0$,

$$c^T z = c_B^T A_B^{-1} b.$$

If $\tilde{c}_N \geq 0$: z is an optimal solution! For every feasible y,

$$c^T y = c_B^T A_B^{-1} b + \tilde{c}_N^T y_N \geq c^T z$$

(since $Ay = b$ and $y_N \geq 0$)

If there's $j \in N$ such that $(\tilde{c}_N)_j < 0$, we can reduce the cost by increasing x_j.

loosening

We can only increase x_j as long as x_B (which depends on x_N) remains positive. It's no longer possible when a variable in B becomes 0.

tightening

tight

The simplex method

One step of the simplex method:

1. Pick j such that $\tilde{c}_j < 0$ and increase x_j as much as possible while keeping $x_B \geq 0$
2. When x_i becomes 0 for $i \in B$, replace j with i in the basis.

This step is called **pivoting**.

$A_B^{-1} A_N = \bar{A}$

סימונים:

$A_B^{-1} b = \bar{b} = z_B$

The simplex method

- How much can we increase x_j?
- We need to keep the solution feasible: $A_B x_B + A_N x_N = b$ $\Big\backslash \cdot A_B^{-1}$
- Write this as $x_B + \bar{A} x_N = \bar{b}$ for $\bar{A} = A_B^{-1} A_N$ and $\bar{b} = (A_B)^{-1} b$

פה רואים ששינוי של x_N משפיע על x_B

- Since $x_B = \bar{b} - \bar{A} x_N$, we can change x_j as long as $x_B \geq 0$
- If $\bar{A}_{i,j} \leq 0$ for all i, we can increase x_j as much as we want
    - x remains feasible and the optimum goes to $-\infty$ so LP is unbounded
- If there's i such that $\bar{A}_{i,j} > 0$, x_j can't be larger than $\dfrac{\bar{b}_i}{\bar{A}_{i,j}}$

$0 \leq x_B = \bar{b} - \bar{A} \cdot \frac{\bar{b}}{\bar{A}} = 0$

The simplex method

Summary: $\bar{A} = A_B^{-1} A_N$ and $\bar{b} = (A_B)^{-1} b$. Let $\varepsilon = \min_{i:\bar{A}_{ij}>0} \frac{\bar{b}_i}{\bar{A}_{ij}}$.

If $x_i > 0$ for all $i \in B$, $\varepsilon > 0$.

Suppose $x_i > 0$ for all $i \in B$. We say x is **non-degenerate**.

Let i^* be the index which achieves the minimum.

The value ε is the maximum value we can increase x_j until some x_i for $i \in B$ becomes 0: $x_B \leftarrow \bar{b} - \bar{A}\varepsilon e_j$ so $x_{i^*} = 0$

We get a new vertex $\hat{x}$ and a new basis $\hat{B} = B \setminus \{i^*\} \cup \{j\}$.

It turns out that $\hat{x}$ is the vertex which corresponds to $\hat{B}$.

Recap:

Assuming x is non-degenerate, we make some progress.

We need to show: $\hat{B}$ is a basis, and $\hat{x}$ is the vertex that corresponds to $\hat{B}$:

"$\hat{x}_{\hat{B}} = A_{\hat{B}}^{-1} b$ and $x_{\hat{N}} = 0$ for $\hat{N} = \{1, \ldots, n\} \setminus \hat{B}$"

$\hat{B}$ is a basis

צריך להוכיח שאחרי ה-pivot B עדיין בסיס!
הוצאנו את העמודה ה-i^* מ-A_B והחלפנו באחרת.
נוודא ש-$A_{\hat{B}}$ עדיין הפיכה

We need to show: $A_{\hat{B}}$ is non-singular.

$A_{\hat{B}}$ is obtained from A_B by putting A_j (j-th col of A) in the i^*-th col:

$$A_{\hat{B}} = \underbrace{A_B}_{\text{הפיכה}} \cdot \underbrace{\begin{array}{c} \\ \left[\begin{array}{ccc|c|ccc} 1 & 0 & \cdots & & \cdots & 0 & 0 \\ 0 & 1 & & & & 0 & 0 \\ \vdots & \vdots & \ddots & \underbrace{(A_B)^{-1}A_j}_{\bar{A}_j \text{ עמודה}} & \ddots & \vdots & \ddots \\ 0 & 0 & & & & 1 & 0 \\ 0 & 0 & & & & 0 & 1 \end{array}\right] \end{array}}_{\text{הפיכה}}$$

i^* (above the middle column)

כפל של מטריצות
הפיכות $\Rightarrow$ מטריצה הפיכה!

$(A_B)^{-1}A_j$ is the j-th col of $\bar{A}$ on which the i^*-entry is non-zero (x is non-degenerate) so the right big matrix has full rank and thus also $A_{\hat{B}}$.

$\hat{x}$ corresponds to $\hat{B}$

Need to show: $\hat{x}_{\hat{N}} = 0$ and $\hat{x}_{\hat{B}} = (A_{\hat{B}})^{-1}b$.

First part is true by construction: i^* became zero and we added it to $\hat{N}$.

So we need to show that $A\hat{x} = b$

By construction: $\hat{x}_N = \varepsilon e_j$, $\hat{x}_B = (A_B)^{-1}b - \left((A_B)^{-1}A_N\right)\varepsilon e_j$.

$$A\hat{x} = A_N\hat{x}_N + A_B\hat{x}_B = A_N\varepsilon e_j + b - A_N\varepsilon e_j = b$$

Also $\hat{x} \geq 0$, so $\hat{x}$ is a basic feasible solution corresponding to $\hat{B}$.

Issues with the Simplex Method

Many questions arise:

1. How to pick the initial basic feasible solution?
    - An even more basic question: given an LP, is it feasible?
2. What if there's x_i with $i \in B$ that equals 0?
    - In this case we can't increase x_j. We call such solutions **degenerate**
3. Which $j \in N$ should we pick? The first one? The one that we can increase the most? The one that gives the largest improvement in the objective function?
4. What is the running time of this algorithm?

Solutions to some of the issues

Running time:

1. The running time of each pivoting step is polynomial
2. There's no known rule for pivoting that guarantees a polynomial running time (the total number of bases is exponential)
3. Nevertheless, the algorithm works well in practice.
4. If x is degenerate, we can still try to swap elements in the basis but it's not clear that we're making progress and we may end up being stuck in a cycle.

Degenerate solutions

? [illegible]

- One way to deal with them: randomly perturb b by a small amount different for every b_i.
- This will ensure (whp) every vertex will have $x_i > 0$ for all $i \in B$ and doesn't affect objective by much
- There are also deterministic rules that guarantee the algorithm always terminates:
  - Bland's Rule: "Pick j such that x_j is smallest index with $\hat{c}_j < 0$, and if there are multiple possible choices for i^*, choose smallest"
- So this is not a major issue

Finding initial basic feasible solution

Idea: we'll modify the LP such that there's an easy initial basic feasible solution.

A solution to the modified LP will give a (not necessarily optimal) basic feasible solution to the original LP.

Modified LP: (suppose wlog $b \geq 0$ by possibly multiplying by -1)

Minimize $\sum_{i=1}^{m} z_i$

Subject to $Ax + Iz = b$ ($I = m \times m$ identity matrix)

$x \geq 0, z \geq 0$

Modified LP

Minimize $\sum_{i=1}^{m} z_i$

Subject to $Ax + Iz = b$ ($I = m \times m$ identity matrix)

$x \geq 0, z \geq 0$

The z's are called **artificial variables**, the x's **real variables**

Let $A' = [A|I]$, $x' = [x|z]$

We assume $b \geq 0$: $x = 0, z = b$ is a **basic feasible solution**

(A'_B is the identity matrix)

We can run the simplex algorithm on the modified LP.

Modified LP

Minimize $\sum_{i=1}^{m} z_i$

Subject to $Ax + Iz = b$ ($I = m \times m$ identity matrix)

$x \geq 0, z \geq 0$

This LP is never unbounded: $\sum_{i=1}^{m} z_i$ is ≥ 0.

Case 1: the optimal value of the modified LP is > 0.

⇒ there's no feasible solution to the original LP. If there was such a solution x, take $z = 0$ and obtain a feasible solution to the new LP with value 0, a contradiction.

Modified LP

Case 2: the value of the modified LP is zero.

Let B be the basis corresponding to the optimal solution x'.

Good case: all the z's in the solution are non-basic. Then the x's are a basic feasible solution to the original LP and B is a basis.

$A'_B = A_B,\, x_B = (A_B)^{-1}b,\, x_N = 0.$

We can run the simplex algorithm from the original LP, starting from B.

Bad case: some z's are in the basis.

Handling the bad case

Note that we assumed that $\sum_{i=1}^{m} z_i = 0\,, z \geq 0$ so $z_i = 0$ for all i.

Idea: Pivot by swapping the z vars in the basis with real vars.

Let $i \in B$ that corresponds to z_i.

Recall: $\bar{A}' = (A'_B)^{-1} A'_N$.

Case B1: If there's a real var x_j with $j \in N$ such that $\bar{A}'_{ij} \neq 0$:

Pivot $\hat{B} \leftarrow B \setminus \{i\} \cup \{j\}$

Claim: $\hat{B}$ is a basis and x' is also a solution associated with $\hat{B}$.

Proof of Claim

$\hat{B}$ is a basis: we need to show $A'_{\hat{B}}$ is non-singular.

$A'_{\hat{B}}$ obtained from A'_B by swapping the i-th col by j-th col of A'_N.

$$A'_{\hat{B}} = A'_B \cdot \begin{bmatrix} 1 & 0 & \cdots & & \cdots & 0 & 0 \\ 0 & 1 & & & & 0 & 0 \\ \vdots & \vdots & \ddots & (A'_B)^{-1}(A'_N)_j & \ddots & \vdots & \ddots \\ 0 & 0 & & & & 1 & 0 \\ 0 & 0 & & & & 0 & 1 \end{bmatrix}$$

The i-th entry in the i-th col is non-zero. As before, this shows $A'_{\hat{B}}$ is non-singular.

Proof of Claim

x' associated with $\hat{B}$: we need to show that $A'x' = b$ and $x'_k = 0$ outside of $\hat{B}$.

We didn't change x' so $A'x' = b$ as before.

For $k \notin \hat{B}$: either $k \notin B$ or $k = i$. If $k \notin B$, $x'_k = 0$ as before.
For $k = i$, $x'_i = z_i = 0$ since i is an artificial variable.

We proved the claim and ended the case B1.

Case B2: For all $j \in N$, $\overline{A'}_{ij} = 0$.

Case B2

Case B2: For all $j \in N$, $\bar{A'}_{ij} = 0$.

We'll show that this implies that A has linearly dependent rows.

(A is from the original LP)

Linearly dependent constraints are useless – we can eliminate them and continue as before.

Recall $\bar{A}' = (A'_B)^{-1} A'_N$.

We assume that the i-th row of $\bar{A}'$ is zero.

Case B2

Let w^T denote the i-th row of $(A'_B)^{-1}$ ($w^T \neq 0$)

For every real variable $j \in N$:

$$(w^T A')_j = \bar{A}'_{i,j} = 0$$

For every real variable $j \in B$:

$$(w^T A')_j = \left((A'_B)^{-1}(A'_B)\right)_{i,j} = 0$$

since $i \neq j$ (as i is artificial and j is real).

So w^T is orthogonal to all the cols of A' corresponding to real variables: in other words, $w^T A = 0$ so the rows of A are linearly dependent.

Summary of Simplex Algorithm

- **Phase 1:** find a basic feasible solution
- **Phase 2:** try to improve the solution while walking along the vertices of the polytope
- No guarantee for polynomial upper bound, but works well in practice
- **Smoothed analysis**: for every input, with high probability simplex will run in polynomial time on a random perturbation
- Many packages and implementations

When is an LP feasible?

When does $Ax = b$ have a solution?

If it doesn't, b isn't in the image of A, which is the linear span of the columns of A.

This implies that there's a vector y which is orthogonal to all the columns of A but not orthogonal to b:

$$y^T A = 0, \qquad y^T b = 1$$

Also clearly if there's such y, the system is unsolvable.

Such a vector y is a "proof" for unsolvability.

Unsolvability of linear equations

Theorem: for any system of linear equations $Ax = b$ exactly one of the following is true:

1. There's a solution, i.e., x' such that $Ax' = b$.
2. There's y such that $y^T A = 0$ and $y^T b = 1$.

זה כדי ניגמול

The above fact isn't sufficient: system $Ax = b$ can be solvable but we're interested in **non-negative** solutions. i.e $x \geq 0$

Fortunately, there's an analogy.

Farkas' Lemma

Farkas' Lemma: for any system $Ax = b, x \geq 0$, exactly one of the following is true:

1. There's a solution, i.e., $x' \geq 0$ such that $Ax' = b$.
2. There's y such that $y^T A \geq 0$ and $y^T b < 0$.

Obvious part: if there's such a y, the system is unsolvable.

Non-obvious part: the second direction.

Proof of Farkas' Lemma

It's more convenient the following variant:

Farkas' Lemma II: for any system $Cx \leq d$, if it is not solvable then there's $z \geq 0$ such that $z^T C = 0$ and $z^T d < 0$.

(You'll prove the original using the variant in HW)

Proof using **Fourier Motzkin Elimination**.

We'll give a proof by example.

Fourier Motzkin Elimination

Suppose we have the following set of inequalities:

$$\begin{cases} x - 5y + 2z \geq 7 \\ 3x - 2y - 6z \geq -12 \\ -2x + 5y - 4z \geq -10 \\ -3x + 6y - 3z \geq -9 \\ -10y + z \geq -15 \end{cases}$$

First step: eliminate x. Multiply every constraint by a positive constant so that the coefficient of x is $1, -1$ or 0.

Fourier Motzkin Elimination

$$\begin{cases} x - 5y + 2z \geq 7 \\ x - \dfrac{2}{3}y - 2z \geq -4 \\ -x + \dfrac{5}{2}y - 2z \geq -5 \\ -x + 2y - z \geq -3 \\ -10y + z \geq -15 \end{cases}$$

Now put x alone in the left hand size of each inequality.

Fourier Motzkin Elimination

$$\begin{cases} x \geq 5y - 2z + 7 \\ x \geq \dfrac{2}{3}y + 2z - 4 \\ x \leq \dfrac{5}{2}y - 2z + 5 \\ x \leq 2y - z + 3 \\ -10y + z \geq -15 \end{cases}$$

For each pair $x \geq \ell$ and $x \leq u$, we must have $\ell \leq u$.

Fourier Motzkin Elimination

$$\begin{cases} \frac{5}{2}y - 2z + 5 \geq 5y - 2z + 7 \\ \frac{5}{2}y - 2z + 5 \geq \frac{2}{3}y + 2z - 4 \\ 2y - z + 3 \geq 5y - 2z + 7 \\ 2y - z + 3 \geq \frac{2}{3}y + 2z - 4 \\ -10y + z \geq -15 \end{cases}$$

We eliminated x and the new system is feasible iff the original one was.

Fourier Motzkin Elimination

At the end we get a sequence of inequalities in 1 variable:

$$x_n \geq \alpha_1, x_n \geq \alpha_2, \ldots, x_n \leq \beta_1, x_n \leq \beta_2, \ldots, 0 \geq -2, 0 \geq -10, \ldots$$

The original system is satisfiable iff the minimum of the β_i's is greater than the maximum of the α_i's.

Every inequality we derive is of the form $z^T C \leq z^T d$ for $z \geq 0$:

- Either taking a non-negative multiple of an inequality to get the coefficient to be ± 1, or adding to inequalities to eliminate x_i.

If no solution: we got $x_1 \leq U$ and $x_1 \geq L$ for $L > U$.
Add $x_1 \leq U$ and $-x_1 \leq -L$ to get $0 \leq U - L$ which gives the z we wanted.

Farkas' Lemma: Summary

Farkas' Lemma: for any system $Ax = b, x \geq 0$, if it is not solvable there's y such that $y^T A \geq 0$ and $y^T b < 0$.

Farkas' Lemma II: for any system $Cx \leq d$, if it is not solvable then there's $z \geq 0$ such that $z^T C = 0$ and $z^T d < 0$.

Duality

Suppose we have an LP in standard form:

Minimize $c^T x$

Subject to $Ax = b, x \geq 0$

We'd like to find a lower bound on the optimum.

Suppose we have y such that $y^T A \leq c^T$:

Then we claim that $y^T b$ must be a lower bound on the optimum of the LP.

Indeed: for every $x \geq 0$ such that $Ax = b$: $c^T x \geq y^T Ax = y^T b$

Duality

Minimize $c^T x$

Subject to $Ax = b, x \geq 0$

y such that $y^T A \leq c^T$ "certifies" a lower bound on the optimum

What is the best lower bound achievable using this method?

We'd like to

Maximize $b^T y$

Subject to $A^T y \leq c$

This is another LP which is called the **dual** of the original LP.

Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

Let α be the optimum of (P) and β be the optimum of (D)

We proved the **weak duality theorem**: $\alpha \geq \beta$

For every (P) feasible x and (D) feasible y: $c^T x \geq y^T A x = y^T b$

The **strong duality theorem**: $\alpha = \beta$.

Strong Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

Strong duality theorem:

Let α be the optimum of (P) and β the optimum of (D).
If (P) and (D) are feasible, then $\alpha = \beta$.

Easier cases:

1. (P) and (D) infeasible
2. (P) infeasible, (D) unbounded
3. (D) infeasible, (P) unbounded

Proof:

Weak duality says $\alpha \geq \beta$, so we only need to show $\alpha \leq \beta$.

Strong Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

Let α be the optimum of (P):

Claim: $\exists y$ such that $A^T y \leq c$ and $b^T y \geq \alpha$.

If so, y is feasible for (D) and $\beta \geq b^T y \geq \alpha$ and we are done.

Suppose towards contradiction there's no such y:

The system $A^T y \leq c$ and $-b^T y \leq -\alpha$ has no solution.

Recall **Farkas' lemma II:**

For any system $Cx \leq d$, either it is solvable, or there's $y' \geq 0$ such that $(y')^T C = 0$ and $(y')^T d < 0$.

Strong Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

- The system $A^T y \leq c$ and $-b^T y \leq -\alpha$ has no solution.
- For any unsolvable system $Cx \leq d$, there's $y' \geq 0$ such that $(y')^T C = 0$ and $(y')^T d < 0$.

We use it with $C = \begin{pmatrix} A^T \\ -b^T \end{pmatrix}, d = \begin{pmatrix} c \\ -\alpha \end{pmatrix}$ to match our assumption.

We get: there's $y' = \begin{pmatrix} x \\ \lambda \end{pmatrix} \geq 0$ such that

$x^T A^T - \lambda b^T = 0$ $\quad | \cdot ()^T$

$$(y')^T C = (\, x^T \mid \lambda \,) \begin{pmatrix} A^T \\ -b^T \end{pmatrix} = 0 \Rightarrow Ax = \lambda b$$

and $(y')^T d = (\, x^T \mid \lambda \,) \begin{pmatrix} c \\ -\alpha \end{pmatrix} < 0 \Rightarrow c^T x < \lambda\alpha.$

$x^T c - \lambda\alpha$

Strong Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

Cor: There's $x \geq 0, \lambda \geq 0$ such that $Ax = \lambda b$ and $c^T x < \lambda\alpha$

Case 1: $\lambda > 0$. By replacing y' with y'/λ we can assume $\lambda = 1$.

We get: there's $x \geq 0$ such that $Ax = b, c^T x < \alpha$. // $\lambda\alpha = \alpha$

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

opt $= \alpha$!

This contradicts α being the optimum for (P).

Case 2: $\lambda = 0$. There's $x \geq 0$ such that $Ax = 0$ and $c^T x < 0$.

The easy direction of **Farkas**: no y such that $A^T y \leq c$ $\leftarrow 0 \leq x^T c < 0$
(multiply by x^T on both sides to get inconsistency)

Dual is infeasible, a contradiction.

min-cut /cNd 19

max flow

Duality

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

- The dual of the dual is the primal
- Duality is the most important concept in linear programming
- When you have a linear program, it's usually a good idea to take its dual and to try to interpret it.
- Example: recall the max-flow LP
  - **Maximize** $\sum_{v:s\to v} f_{sv}$
  - **Subject to** capacity constraints: $0 \leq f_{uv} \leq c_{uv}$
  - and flow conservation constraints: $\sum_{v:u\to v} f_{uv} = \sum_{w:v\to w} f_{vw}$ for every $v \neq s, t$

Min-cut Max-flow

Maximize $\sum_{v:s\to v} f_{sv}$
Subject to $0 \le f_{uv} \le c_{uv}$
$\sum_{v:u\to v} f_{uv} = \sum_{w:v\to w} f_{vw}$

After taking the dual and cleaning it up a little bit, we get:

Minimize $\sum_{i\to j} c_{i,j} y_{i,j}$

Subject to $x_s = 1, x_t = 0 \quad y_{i,j} \ge 0 \quad y_{i,j} \ge x_i - x_j$

The variables are $y_{i,j}$ for every edge $i \to j$ and x_u for every vertex u

Suppose we have a solution where $x_u \in \{0,1\}$ for every u:

Let $S = \{u: x_u = 1\}$ and $T = \{v: x_v = 0\}$

The objective is to minimize the sum of c_{ij}'s crossing the cut

Min-cut Max-flow

- By **strong duality**, the optimum of the primal and the dual are equal
- One can also prove that the optimum of the dual is such that $x_u \in \{0,1\}$ for every u
    - We'll discuss a bit later the concept of LPs with integer constraints
- This proves (using LP duality) the **min cut max flow** theorem
- "Rule of life": if you have a linear program, try to take its dual and interpret it!

Polynomial time Algorithms

- The simplex algorithm doesn't run in polynomial time
- There are, however, algorithms for linear programming that run in **polynomial time**
  - In many cases, they are less efficient in practice than simplex
- We will sketch one of them: the **Ellipsoid Algorithm**
- Our first question will be: What is the size of the input?

הכוונה לגודל המספרים.

שונה "$x \le 1{,}000{,}000$..." מ- "$x \le 2^{20}$".

LP Input Size

נניח לשם הפשטות שהמספרים הם שלמים (אפשר גם לייצג ממשיים אבל מוסיף מורכבות...)

- To represent an integer of size $\leq K$ we need $k = \log K$ bits
- Suppose all coefficients in our LP are integers of size $\leq K$:
  - It takes nk bits to describe the length n vectors c^T and b
  - It takes nmk bits to describe the $m{\times}n$ matrix A
  - Total input size $L = O(mnk)$
- We want an algorithm that in time polynomial in L returns a point x which is an optimal solution to the LP
- Question: Is there even such a solution?

LP Output Size

- Answer: Yes.

Proof: Recall that an optimal solution is a vertex x which we associate with a basis B of size m.

$$\text{“}x_B = (A_B)^{-1}b \text{ and } x_N = 0\text{”}$$

We just need to show an upper bound on each entry in x_B.

$A_B x_B = b$

By **Cramer's rule**, $x_j = \det\left(A_B^j\right) / \det(A_B)$ where A_B^j is A_B with the j-th column replaced by b.

מסקנה: ניתן למצוא חסם עליון לכל
קורדינטה ב-x_B

LP Output Size

$x_j = \det\left(A_B^j\right) / \det(A_B)$ where A_B^j is A_B with the j-th column replaced by b.

- $\det\left(A_B^j\right)$ is a sum of $m!$ products $\prod_{i=1}^{m} (A_B)_{i,\sigma(i)}$
- Each entry is $\leq 2^k$ so the product is $\leq 2^{km}$ in abs value
- Therefore $\left|\det\left(A_B^j\right)\right| \leq m!\,2^{km} \leq m^m \cdot 2^{km}$ $\quad m! \leq m^m$
- $|\det(A_B)| \geq 1$ since it's a non-zero integer ← הנתון שכל המספרים במטריצה טבעיים
- It follows that $|x_j| \leq m^m \cdot 2^{km}$ ← חסם עליון
- Thus, $\log|x_j| \leq m \log m + km$ ← חסם עליון על כמות הביטים

Ellipsoid Method

- Now that we know that there exists a polynomially bounded solution, we can start describing an algorithm that finds it.
- Claim: it's enough to solve the **feasibility problem**
  - Feasibility: given a polyhedron $P = \{x : \ Cx \leq d\}$, either find $x \in P$ or say "$P = \emptyset$"
- We'll show: if we can solve the feasibility problem, we can solve the optimization problem.

נמצא פתרון פיזיבילי, ובעזרתו ובעזרת
הבעיה הדואלית נמצא פתרון אופטימלי.

Reducing Feasibility to Optimization

Primal (P):

Minimize $c^T x$
Subject to $Ax = b, x \geq 0$

Dual (D):

Maximize $b^T y$
Subject to $A^T y \leq c$

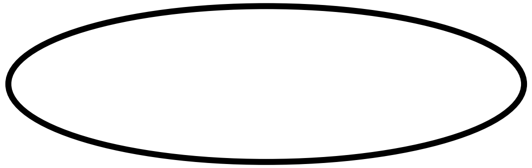
- Step 1: check for primal feasibility
  - Is there x such that $Ax = b, x \geq 0$?
  - If not, done. Primal is infeasible.
- Step 2: check for dual feasibility
  - Is there y such that $A^T y \leq c$?
  - If not, done. Primal is unbounded.
- Step 3: Find optimum by running the ellipsoid algorithm on

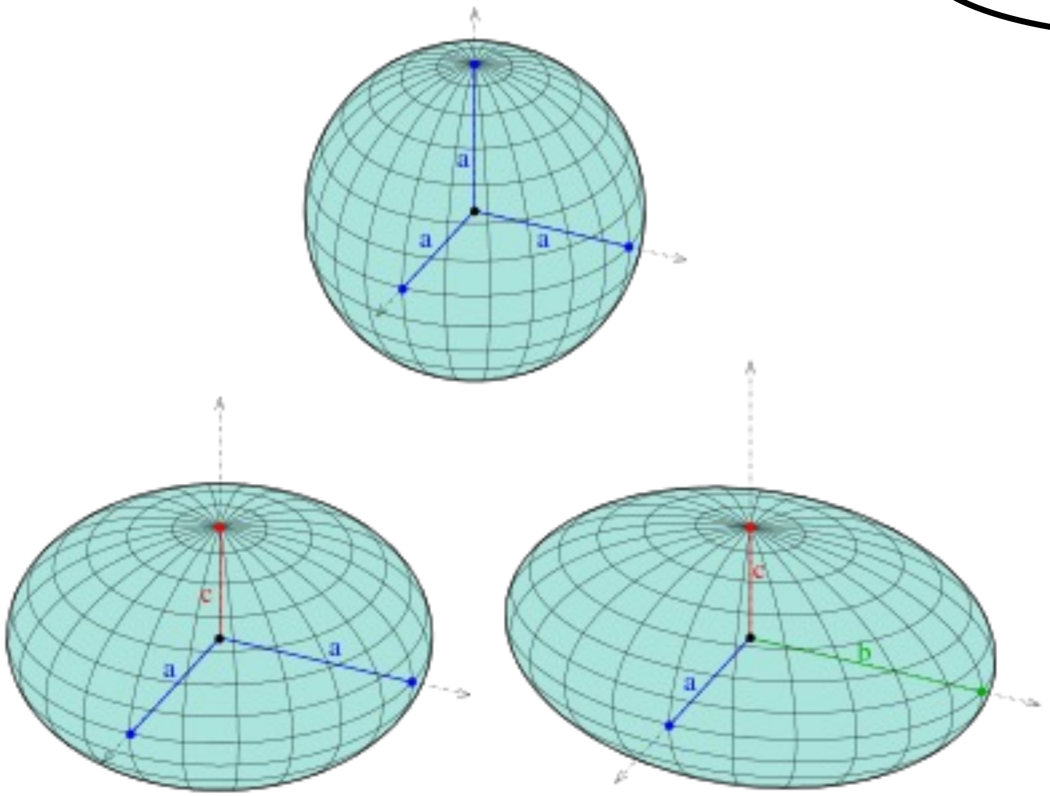
$$P' = \{(x, y): \underbrace{c^T x \leq b^T y}_{P \,\leq\, D}, \underbrace{Ax = b, x \geq 0}_{\text{P תנאי פיזיביליות}}, \underbrace{A^T y \leq c}_{\text{D תנאי פיזיביליות}}\}$$

- By strong duality, if (P) and (D) are feasible, there exist feasible solutions whose objective are equal. This step will return them.

Ellipsoid Method

- What is even an ellipsoid?
- Higher-dimensional analog of ellipse





Ellipsoids

An **ellipsoid** can be obtained from a **sphere** by:

- Stretching the axes
- Rotation
- Shifting the center (translation)

In 3 dimensions: unit sphere=$\{(x, y, z): x^2 + y^2 + z^2 \leq 1\}$

Ellipsoid centered at the origin: $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} \leq 1$

מחפשים נקודה בתוך P

Ellipsoid Method Overview

- We're given a polyhedron $P = \{x : Cx \leq d\}$ and want to find a point in P
- We start with a big sphere Q_0 that contains P

  ball
- At each step we maintain the **invariant** the Q_k contains P
- We check if the center q_k of Q_k is in P. If yes, we're done
- Otherwise, we get a **hyperplane** separating q_k from P:

  $q_k \to q_{k+1}$
  - Say: $a^T x \leq b$ for all $x \in P$ but $a^T q_k > b$
- We find a smaller ellipsoid containing $Q \cap \{x : a^T x \leq b\}$ and continue

Ellipsoid Method Efficiency

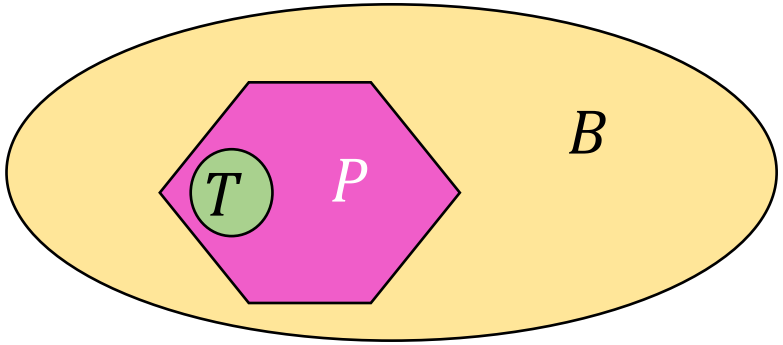
- The progress measure of the algorithm is the volume of Q_k
- We need to show: at every step, the volume of Q_{k+1} is significantly smaller than the volume of Q_k
- We also need:
  - An upper bound on the volume of P and thus Q_0 (we sort of proved that)
  - A volume lower bound on P, showing that if Q_k becomes small enough then this implies that $P = \emptyset$
  - To be able to find Q_{k+1} efficiently given Q_k and the separating hyperplane
- We'll address all these issues

Ellipsoid Method Overview

- Let $P = \{x : Cx \leq d\}$
- We start with a big ~~sphere~~ ball Q_0 centered at $q_0 = 0$ which is large enough to contain a feasible point
  - Recall that we have some bound on how large a solution can be
- At step k: if $q_k \in P$ (i.e., $Cq_k \leq d$), return q_k.
- If not, $C_j q_k > d_j$ for some j (where C_j=j-th row of C)
  - P is contained in $S = Q_k \cap \{C_j x \leq d\}$.
  - Compute an ellipsoid Q_{k+1} containing S and continue

Lower and Upper Bounds on Volume

- Recall that we proved that there's some U (polynomial in the input size) such that all vertices have norm at most 2^U
- This implies that a sphere B of radius $R = 2^U$ contains P
- The volume of B is $2^{O(nU)}$
- Similarly, using similar arguments we can assume that $\overset{P}{\cancel{Q}}$ contains a sphere T of radius 2^{-U}
- Whose volume is $2^{-\Omega(nU)}$

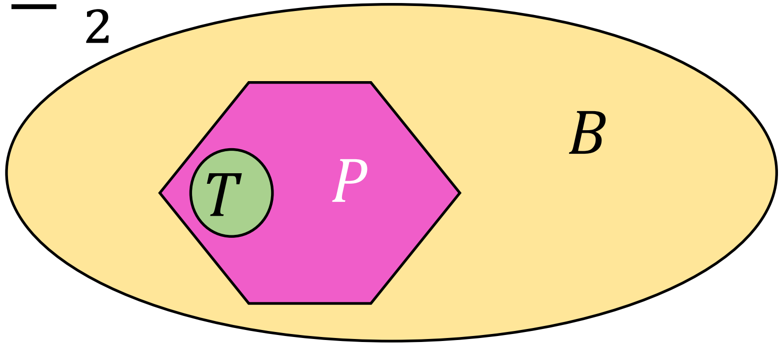


אם כל פעם מקטינים פי 2 את הנפח, צריך לעשות צעדים מ- nU ל- $-nU$

אם הקטנו מספיק את האליפסואיד מתחת לרדיוס 2^{-U} נפסיק את האלגוריתם

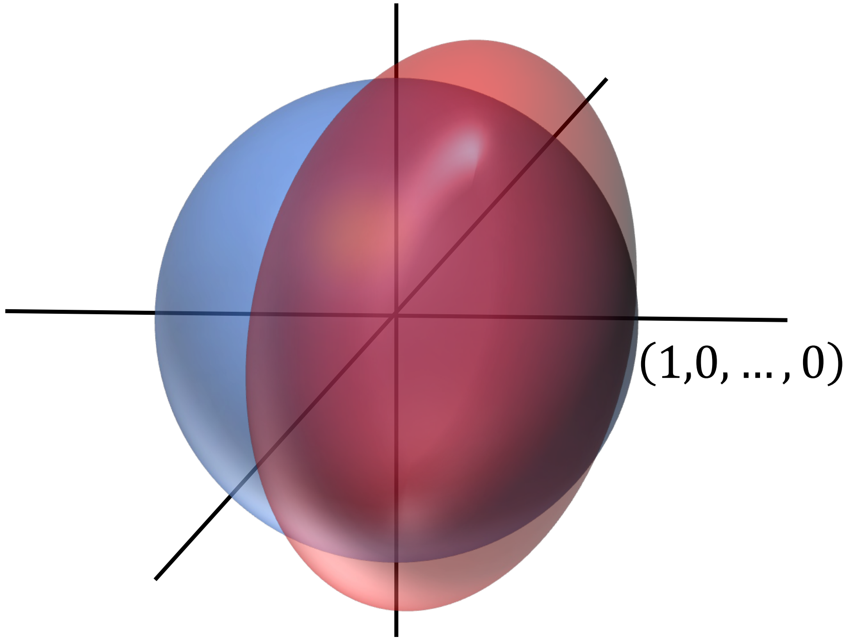
Lower and Upper Bounds on Volume

- Volume of $Q_0 = B$ is $2^{O(nU)}$
- If $P \neq \emptyset$, it contains an ellipsoid of volume $2^{\Omega(-nU)}$
- We'll later show: $\dfrac{\mathrm{volume}(Q_{k+1})}{\mathrm{volume}(Q_k)} \leq e^{\frac{-1}{2n+1}} < 1$ היינו רוצים $\frac{1}{2}$
- After k steps: volume decreased by $e^{\frac{-k}{2n+1}}$
- After $2n+1$ steps: volume decreased by $\frac{1}{e} \leq \frac{1}{2}$ אחרי $k = 2n+1$ צעדים $\Leftarrow \frac{1}{2}$!
- Total number of steps: polynomial in n, U



Finding a smaller ellipsoid

- Recall: we want to intersect Q_k with a halfspace $C_j x \leq d$ and find a small ellipsoid containing the intersection
- Suppose $Q_0 = \left\{x : \sum_{i=1}^{n} x_i^2 \leq 1\right\}$ is the unit sphere
- And we want to intersect it with the halfspace $x_1 \geq 0$



Finding a smaller ellipsoid

- "Half sphere": $\left\{x : \sum_{i=1}^{n} x_i^2 \leq 1\right\} \cap \{x_1 \geq 0\}$
- The center of the new ellipsoid will be at $(\alpha, 0, \ldots, 0)$ (we'll pick α later)
- $Q_1 = \left\{x : \frac{(x_1-\alpha)^2}{\beta^2} + \frac{x_2^2}{\gamma^2} + \cdots + \frac{x_n^2}{\gamma^2} \leq 1\right\}$
- The boundary of Q_1 should pass through $(1,0, \ldots, 0)$ and through $\left\{x : \sum_{i=2}^{n} x_i^2 = 1\right\}$
- $\frac{(1-\alpha)^2}{\beta^2} = 1$ and $\frac{\alpha^2}{\beta^2} + \frac{1}{\gamma^2} = 1$

Finding a smaller ellipsoid

- Summary: we want to pick α, β, γ such that

$$\beta^2 = (1-\alpha)^2 \quad \text{and} \quad \gamma^2 = \frac{1}{1-\frac{\alpha^2}{(1-\alpha)^2}} = \frac{(1-\alpha)^2}{1-2\alpha}$$

- Verify: any choice of α, β, γ satisfying these parameters will give an ellipsoid containing the half ~~sphere~~ ball

עדיין אין הבטחה
לגבי כמה הנפח קטן

- (note $0 \leq \alpha < \frac{1}{2}$)
- However, we have another goal: to reduce the volume by a significant amount

Volumes

- Area of circle: $\{(x, y) : x^2 + y^2 \leq r^2\} = \left\{(x, y) : \frac{x^2}{r^2} + \frac{y^2}{r^2} \leq 1\right\}$
- Area is πr^2
- Area of ellipse $\left\{(x, y) : \frac{x^2}{a^2} + \frac{y^2}{b^2} \leq 1\right\}$ is πab
- Volume of ellipsoid $\left\{(x, y, z) : \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} \leq 1\right\}$ is $\frac{4}{3}\pi abc$

  מקבילים
- In n dimensions: $\left\{x : \frac{x_1^2}{(a_1)^2} + \cdots + \frac{x_n^2}{(a_n)^2} \leq 1\right\}$: $c_n \cdot a_1 a_2 \cdots a_n$ where c_n is some constant that depends on n

Volumes

- The volume of Q_0, the unit ball, is c_n
- $Q_1 = \left\{x : \frac{(x_1-\alpha)^2}{\beta^2} + \frac{x_2^2}{\gamma^2} + \cdots + \frac{x_n^2}{\gamma^2} \le 1\right\}$: volume is $c_n \cdot \beta \cdot \gamma^{n-1}$
- $\beta^2 = (1-\alpha)^2$ and $\gamma^2 = \frac{1}{1-\frac{\alpha^2}{(1-\alpha)^2}} = \frac{(1-\alpha)^2}{1-2\alpha}$
- $\frac{\text{volume}(Q_1)}{\text{volume}(Q_0)} = \frac{(1-\alpha)^n}{(1-2\alpha)^{\frac{n-1}{2}}}$

רוצים לבחור α שתביא את
המנה למינימום $\Leftarrow$ גוזרים ומשווים
ל-0.

- We want to minimize $\frac{(1-\alpha)^n}{(1-2\alpha)^{\frac{n-1}{2}}}$ for $0 \le \alpha < \frac{1}{2}$

Minimizing volume

Input interpretation

minimize	function	$\dfrac{(1-x)^3}{(1-2x)^{(3-1)/2}}$
	domain	

Global minimum

$\min\left\{\dfrac{(1-x)^3}{(1-2x)^{(3-1)/2}} \,\middle|\, 0 \le x \le \right.$

Input interpretation

minimize	function	$\dfrac{(1-x)^4}{(1-2x)^{(4-1)/2}}$
		$0 \le x \le 1\,2^{-1}$

$x \le \left.\dfrac{1}{2}\right\} = \dfrac{256}{75\sqrt{15}}$ at $x = \dfrac{1}{5}$

Input interpretation

minimize	function	$\dfrac{(1-x)^5}{(1-2x)^{(5-1)/2}}$
	domain	$0 \le x \le 1\,2^{-1}$

Global minimum

$$\min\left\{\frac{(1-x)^5}{(1-2x)^{(5-1)/2}} \,\middle|\, 0 \le x \le \frac{1}{2}\right\} = \frac{3125}{3456} \text{ at } x = \frac{1}{6}$$

Minimizing Volume

- The minimum of $\frac{(1-\alpha)^n}{(1-2a)^{\frac{n-1}{2}}}$ is obtained at $\alpha = \frac{1}{n+1}$
  - Simple calculus exercise of calculating derivatives
- $\beta^2 = (1-\alpha)^2 = \left(\frac{n}{n+1}\right)^2, \gamma^2 = \frac{\beta^2}{1-2a} = \frac{n^2}{n^2-1}$
- $Q_1 = \left\{x : \frac{\left(x_1 - \frac{1}{n+1}\right)^2}{\left(\frac{n}{n+1}\right)^2} + \frac{x_2^2}{\frac{n^2}{n^2-1}} + \cdots + \frac{x_n^2}{\frac{n^2}{n^2-1}} \leq 1\right\}$
- Ratio of volumes is $\beta \cdot \gamma^{n-1}$

Minimizing Volume

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

Use $1 + y \leq e^y$ to get

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

- Phew! Deep breath…

General Ellipsoids and Hyperplanes

- **Claim**: what we proved is the worst case
- Easy to convince ourselves for hyperplanes parallel to $x_1 = 0$ (i.e. intersecting with $x_1 \geq \frac{1}{2}$ will only make volume smaller)
- For general hyperplanes: rotate to reduce to the previous case (Rotations don't change the volumes)
- For general ellipsoids: stretch and shift to reduce to previous case (stretching affect volumes of both ellipsoids similarly so doesn't affect ratio, shifting doesn't change anything)
- Every ellipsoid obtained from unit sphere by stretching and shifting

Summary: Ellipsoid Algorithm

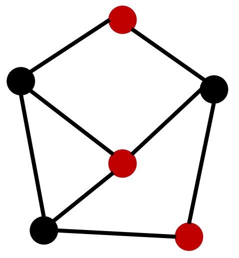
- We showed that the volume of the initial ellipsoid is $2^{O(nU)}$
- We showed that after a number of steps which is polynomial in n, U, we either find a point in P or obtain an ellipsoid of volume $2^{-\Omega(nU)}$ from which we deduce $P = \emptyset$
- We showed that in each step we can efficiently compute the next ellipsoid
- Unfortunately, running time is too slow to be practical
- There are faster polynomial time algorithms called **interior point methods**

Separation Oracles

- An important observation regarding the Ellipsoid algorithm: We don't really need to "see" the entire polyhedron
- All we need is:
  - Upper bound R on the volume
  - Lower bound r on the volume
  - A **separation oracle**: efficient procedure that given a point x not in P, produces a separating hyperplane: a vector c and number d such that $c^T x > d$ and for all $y \in P$, $c^T y \leq d$
- We could "imagine" P with exponentially many constraints that has these properties
- Or even more complicated set, as we shall later see…

Integer Programming

- Integer programming (IP): like LP except that we add the constraints that some (or all) variables are integers
- Example: **maximum independent set**: given a graph (V, E), find a maximum size subset $S \subseteq V$ such that no two vertices in S have an edge between them



Maximize $\sum_{v \in V} x_v$

Subject to $0 \leq x_v \leq 1$, $\boldsymbol{x_v}$ **integer**

$x_u + x_v \leq 1$ for every $(u, v) \in E$

Integer Programming

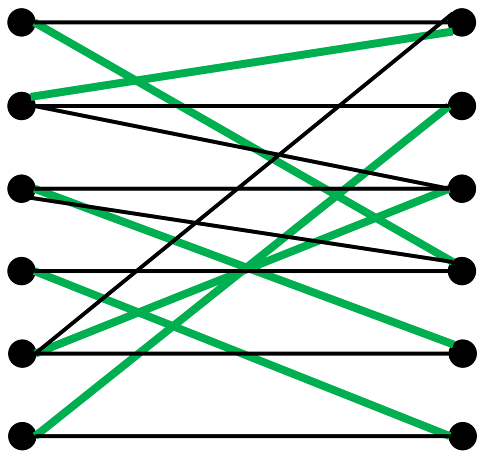
- This example already shows that IP is NP complete
  - We don't expect a polynomial time algorithm
- Another example: perfect matching

Maximize $\sum_{(u,v)\in E} x_{uv}$

Subject to $\sum_{v:(u,v)\in E} x_{u,v} \leq 1$ for every $u \in R$

$\sum_{u:(u,v)\in E} x_{u,v} \leq 1$ for every $v \in L$

$x_{u,v} \in \{0,1\}$



Relaxations

- We can **relax** integer programs to obtain linear programs:
    - Instead of $y \in \{0,1\}$, require $0 \leq y \leq 1$
- For example: here's a relaxation for the independent set IP

Maximize $\sum_{v \in V} x_v$

Subject to $0 \leq x_v \leq 1$

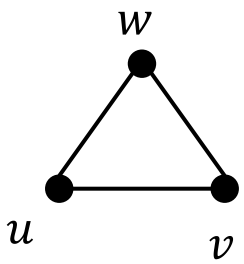
$x_u + x_v \leq 1$ for every $(u, v) \in E$

- The solutions we get now can be **fractional** and don't always correspond to solutions to the original problem

Relaxations for Independent Set

Maximize $\sum_{v\in V} x_v$
Subject to $0 \le x_v \le 1$
$x_u + x_v \le 1$ for every $(u, v) \in E$

- Consider the triangle graph:



- An optimal solution:
  $x_u = x_v = x_w = \frac{1}{2}$,   optimal value = $\frac{3}{2}$
- Maximal independent set has size 1
- We can efficiently optimize the relaxed LP, but it's not always clear what is the relation between the solution and the solutions for the original IP
- It might make sense to try to round the fractional solution to an integral solution: we'll revisit this idea later in the class

Relaxations for Perfect Matching

- Here something different happens
- One can show that even for the relaxed IP, the vertices are integral points
- This is a special property of the **bipartite matching polytope** (the polytope defined by these constraints)
- A similar thing happened earlier when we considered the dual of the max flow LP, which was the min cut LP

Maximize $\sum_{(u,v)\in E} x_{uv}$
Subject to $\sum_{v:(u,v)\in E} x_{u,v} \leq 1$
for every $u \in R$
$\sum_{u:(u,v)\in E} x_{u,v} \leq 1$
for every $v \in L$
$0 \leq x_{u,v} \leq 1$

Summary – Linear Programming

- A powerful algorithmic technique for optimization: we'll soon see more applications
- Corresponds to many real-world problem, and generalizes many well studied algorithmic problems
- We saw two algorithms:
  - **Simplex**: works well in practice
  - **Ellipsoid**: polynomial time algorithm
- We discussed the important concept of **duality**
- We discussed **integer programs** and **relaxations**