PRINCETON UNIVERSITY
ORFE
Instructor:
Amir Ali Ahmadi

This lecture:

- Linear Programming (LP)
  - Applications of linear programming
  - History of linear programming
- Geometry of linear programming
  - Geometric and algebraic definition of a vertex and their equivalence
  - Optimality of vertices
- The simplex method
  - The idea behind the simplex algorithm
  - An example of the simplex algorithm in use

- LP is an important and beautiful topic covered in much more depth in ORF 307 [Van14]. We'll only be scratching the surface here.
- Our presentation in this lecture is mostly based on [DPV08], with some elements from [Ber09], [BT97], [CZ13], [Van14].

Linear Programming

Linear programming is a subclass of convex optimization problems in which both the constraints and the objective function are linear (or affine) functions.

- A linear program is an optimization problem of the form:

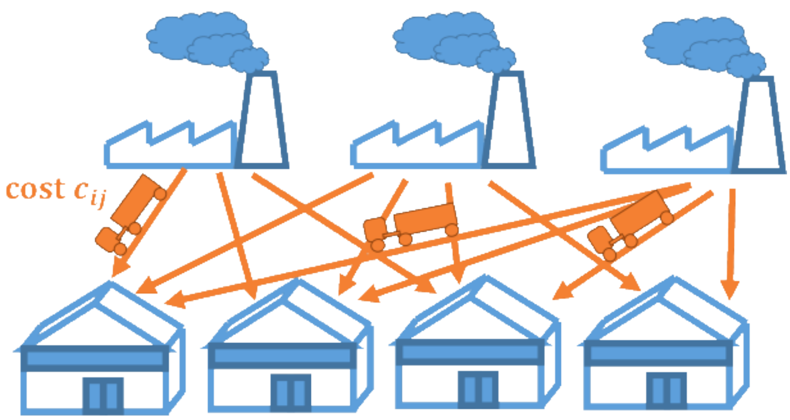
$$\begin{aligned} &\text{minimize } c^T x \\ &\text{subject to } Ax = b \\ &\qquad\qquad\quad x \geq 0 \end{aligned}$$

  where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$. This is called a linear program in *standard form*.
- Not all linear programs appear in this form but we will see later that they can all be rewritten in this form using simple transformations.

- In essence, linear programming is about solving *systems of linear inequalities*. The subject naturally follows the topic of the end of our last lecture, namely, that of solving *systems of linear equations*.

Applications of linear programming

Example 1: Transportation



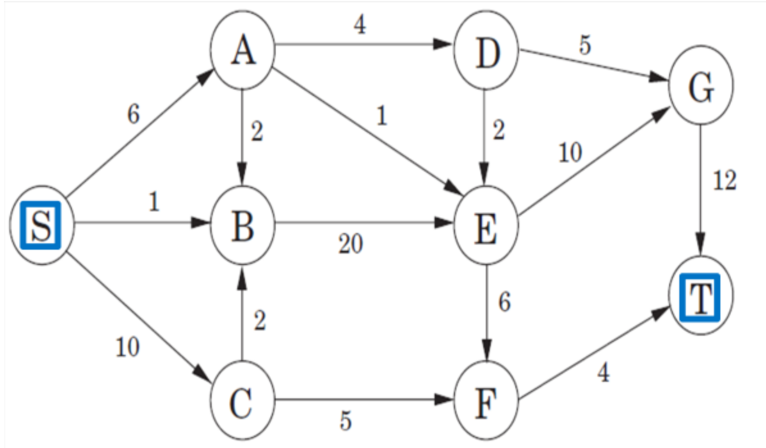
- All plants produce product A (in different quantities) and all warehouses need product A (also in different quantities).
- The cost of transporting one unit of product A from i to j is c_{ij}.

We want to minimize the total cost of transporting product A while still fulfilling the demand from the warehouses and without exceeding the supply produced by the plants.

- Decision variables: x_{ij}, quantity transported from i to j
- The objective function to minimize: $\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$
- The constraints are:
  - Not exceed the supply in any factory: $\sum_{j=1}^{n} x_{ij} \le s_i, \forall i = 1, \dots, m$
  - Fulfill needs of the warehouses: $\sum_{i=1}^{n} x_{ij} \ge d_j,\ \forall j = 1, \dots, n$
  - Quantity transported must be nonnegative: $x_{ij} \ge 0, \forall i, j$

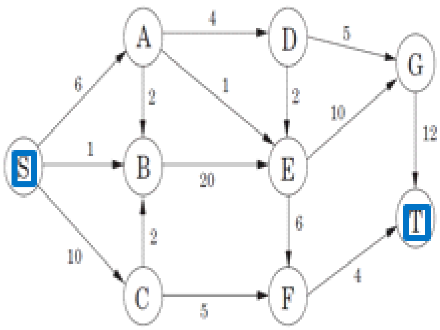
Lec11p3, ORF363/COS323

Example 2: The maximum flow problem



Recall from Lecture 1:

- The goal is to ship as much oil as possible from S to T.
- We cannot exceed the capacities on the edges.
- No storage at the nodes: for every node (except S and T), flow in=flow out.



$x_{SA}, x_{AD}, x_{BE}, \dots, x_{GT}$ ← Decision variables

max. $x_{SA} + x_{SB} + x_{SC}$ ← Objective function

s.t.

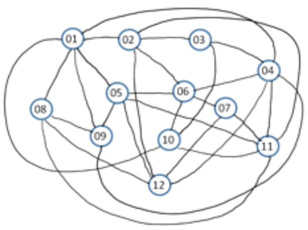
Constraints

- $x_{SA}, x_{AD}, x_{BE}, \dots, x_{GT} \geq 0$
- $x_{SA} \leq 6,\ x_{AB} \leq 2,\ x_{EG} \leq 10, \dots, x_{GT} \leq 12$
- $$\begin{cases} x_{SA} = x_{AD} + x_{AB} + x_{AE} \\ x_{SC} = x_{CB} + x_{CF} \\ \vdots \\ x_{CF} + x_{EF} = x_{FT}. \end{cases}$$

Example 3: LP relaxation for the largest independent set problem

- Given an undirected graph, the goal is to find the largest collection of nodes among which no two share an edge. (Recall that we saw some applications of this problem in scheduling in Lecture 1.)

- We can write this problem as a linear program with integer constraints. Such a problem is called an *integer program* (more precisely a *linear integer program*). Many integer programs of interest in practice are in fact *binary programs*; i.e., they only have 0/1 constraints. The maximum independent set problem is such an example.



$x_1, x_2, \ldots, x_{12}$ Decision variables

max. $x_1 + x_2 + \cdots + x_{12}$ Objective function

s.t.

$x_i(1-x_i) = 0, \quad i = 1, \ldots, 12$

Integer constraint (equivalent to $x_i = 1$ or $x_i = 0$)

$x_1 + x_2 \leq 1$

$x_1 + x_8 \leq 1$ (one per edge)

$x_4 + x_6 \leq 1$

$\vdots$

$x_{12} + x_8 \leq 1$

Constraints

Integer programs (IPs) are in general difficult to solve (we will formalize this statement in a few lecture). However, we can easily obtain the so-called **"LP relaxation"** of this problem by replacing the constraint $x_i \in \{0,1\}$ with $x_i \in [0,1]$:

$$
\text{(LP)} \qquad
\begin{aligned}
&\text{max. } x_1 + x_2 + \cdots + x_{12} \\
&\text{s.t. } x_1 + x_2 \leq 1 \\
&\quad x_1 + x_8 \leq 1 \\
&\quad x_4 + x_6 \leq 1 \\
&\quad \vdots \\
&\quad x_{12} + x_8 \leq 1 \\
&\quad 0 \leq x_i \leq 1, \qquad i = 1, \ldots, 12
\end{aligned}
$$

Observe that the optimal solution to the LP (denoted by OPT_{LP}), is an upperbound to the optimal solution to the IP (denoted by OPT_{IP},); i.e.,

$$OPT_{IP} \leq OPT_{LP}.$$

(Why?)

Example 4: Scheduling nurses [Ber09]

This is another example of an LP relaxation for an integer program.

A hospital wants to start weekly nightshifts for its nurses. The goal is to hire the fewest number of nurses possible.

- There is demand d_j for nurses on days j=1,...7.
- Each nurse wants to work 5 consecutive days.

How many nurses should we hire?

- The decision variables here will be $x_1, \dots, x_7$, where x_j is the number of nurses hired for day j.

- The objective is to minimize the total number of nurses:

$$\sum_{j=1}^{7} x_j$$

- The constraints take into account the demand for each day but also the fact that the nurses want to work 5 consecutive days. This means that if the nurses work on day 1, they will work all the way through day 5.

$$\begin{aligned}
x_1 + \quad\quad\quad\quad\;\; x_4 + x_5 + x_6 + x_7 &\geq d_1 \\
x_1 + x_2 + \quad\quad\quad\quad\;\; x_5 + x_6 + x_7 &\geq d_2 \\
x_1 + x_2 + x_3 + \quad\quad\quad\quad + x_6 + x_7 &\geq d_3 \\
x_1 + x_2 + x_3 + x_4 \quad\quad\quad\quad\quad + x_7 &\geq d_4 \\
x_1 + x_2 + x_3 + x_4 + x_5 \quad\quad\quad\quad\quad &\geq d_5 \\
x_2 + x_3 + x_4 + x_5 + x_6 \quad\quad &\geq d_6 \\
x_3 + x_4 + x_5 + x_6 + x_7 &\geq d_7
\end{aligned}$$

- Naturally, we would want $x_1, \dots, x_7$ to be positive integers as it won't make sense to get fractions of nurses! Such a constraint results in an IP. The LP relaxation of this IP is the following: impose only nonnegativity constraints $x_1, \dots, x_7 \geq 0$.
- The optimal value of the LP will give a lower bound on the optimal value of the IP; i.e., it will say that it is not possible to meet the demand with less than LP_{OPT} number of nurses (why?).
- If it just so happens that the optimal solution to the LP is an integer vector, the same solution must also be optimal to the IP (why?).

History of Linear Programming

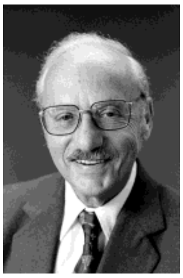
- Solving systems of linear inequalities goes at least as far back as the late 1700s, when Fourier invented a (pretty inefficient) solution technique, known today as the "Fourier-Motzkin" elimination method.

- In 1930s, Kantorovich and Koopmans brought new life to linear programming by showing its widespread applicability in resource allocation problems. They jointly received the Nobel Prize in Economics in 1975.

- Von Neumann is often credited with the theory of "LP duality" (the topic of our next lecture). He was a member of the IAS here at Princeton.

- In 1947, Dantzig invented the first practical algorithm for solving LPs: the *simplex method*. This essentially revolutionized the use of linear programming in practice. (Interesting side story: Dantzig is known for solving two open problems in statistics, mistaking them for homework after arriving late to a lecture at Berkeley. This inspired the first scene in the movie Good Will Hunting.)

John von Neumann (1903-1957)



George Dantzig (1914-2005)

- In 1979, Khachiyan showed that LPs were solvable in polynomial time using the "ellipsoid method". This was a theoretical breakthrough more than a practical one, as in practice the algorithm was quite slow.

- In 1984, Karmarkar developed the "interior point method", another polynomial time algorithm for LPs, which was also efficient in practice. Along with the simplex method, this is the method of choice today for solving LPs.

Leonid Khachiyan (1952 - 2005)

Narendra Karmarkar (b. 1957)

LP in alternate forms

As mentioned before, not all linear programs appear in the standard form:

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

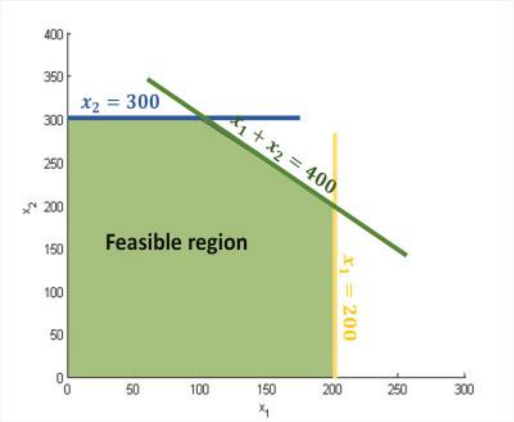
Essentially, there are 3 ways in which a linear program can differ from the standard form:

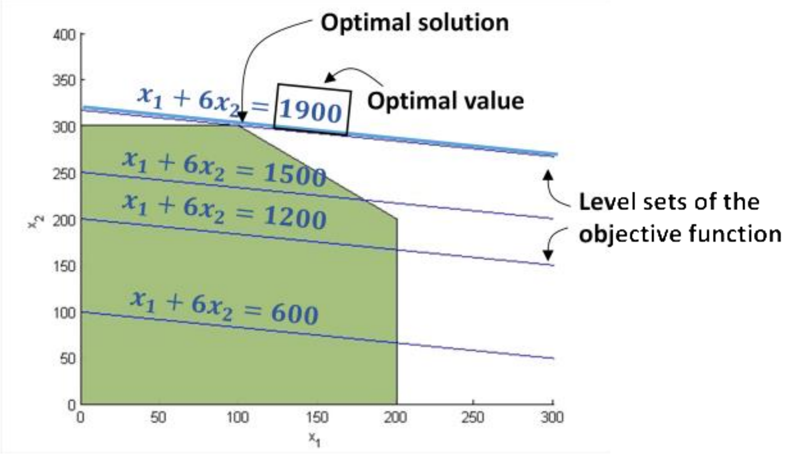
- it is a maximization problem instead of a minimization problem,
- some constraints are inequalities instead of equalities: $a_i^T x \geq b_i$,
- some variables are unrestricted in sign.

There are simple transformations that reduce these alternative forms to standard form. We briefly showed how this is done in class. The reader can find these simple transformations in section 7.1.4 of [DPV08].

Solving an LP in two variables geometrically

$$\begin{aligned} &\max.\ x_1 + 6x_2 \\ &\text{s.t. } x_1 \leq 200 \\ &\quad\ x_2 \leq 300 \\ &\quad\ x_1 + x_2 \leq 400 \\ &\quad\ x_1, x_2 \geq 0 \end{aligned}$$



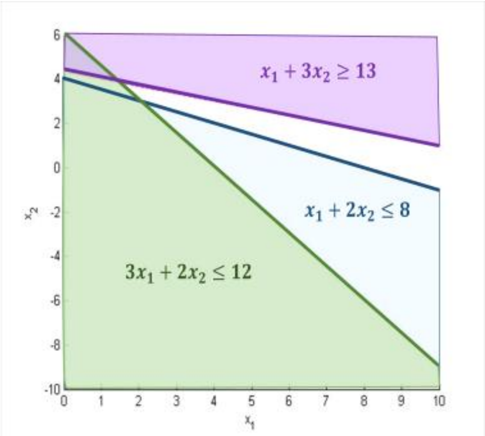


We are trying to find the "largest" level set of the objective function that still intersects the feasible region.

All possibilities for an LP

Infeasible case

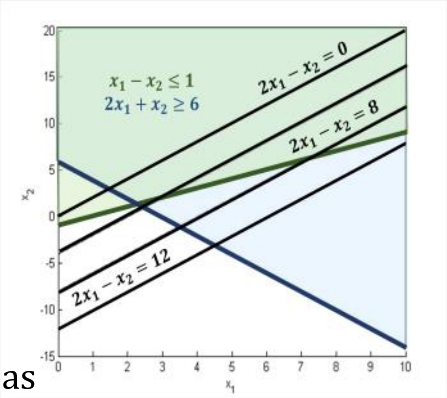
$$\begin{aligned} &\min.\ x_1 + x_2 \\ &\text{s.t. } x_1 + 2x_2 \leq 8 \\ &\quad 3x_1 + 2x_2 \leq 12 \\ &\quad x_1 + 3x_2 \geq 13 \end{aligned}$$



The three regions in the drawing do not intersect : there are no feasible solutions.

Unbounded case

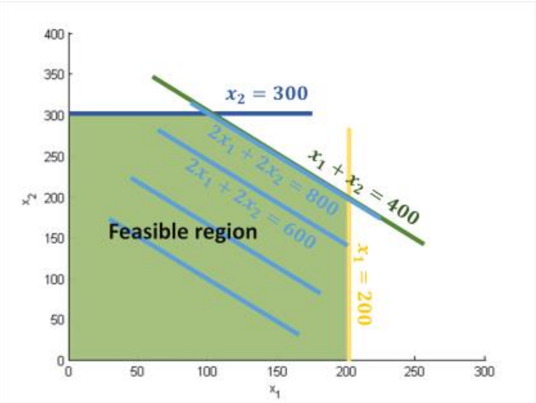
$$\begin{aligned} &\min.\ 2x_1 - x_2 \\ &\text{s.t. } x_1 - x_2 \leq 1 \\ &\quad 2x_1 + x_2 \geq 6 \end{aligned}$$



The intersection of the green and the blue regions is unbounded; we can push the objective function as high up as we want.

Infinite number of optimal solutions

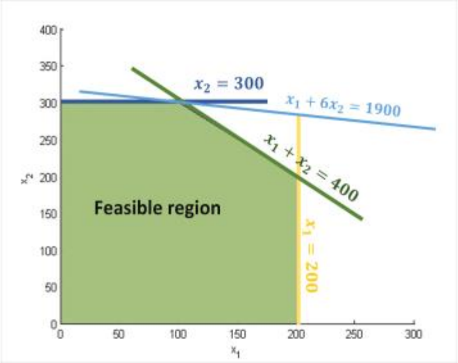
$$\begin{aligned} &\min.\ 2x_1 + 2x_2 \\ &\text{s.t. } x_1 \leq 200 \\ &\quad x_2 \leq 300 \\ &\quad x_1 + x_2 \leq 400 \\ &\quad x_1, x_2 \geq 0 \end{aligned}$$



There is an entire "face" of the feasible region that is optimal. Notice that the normal to this face is parallel to the objective vector.

Unique optimal solution

$$\begin{aligned} &\min.\ x_1 + 6x_2 \\ &\text{s.t. } x_1 \leq 200 \\ &\quad x_2 \leq 300 \\ &\quad x_1 + x_2 \leq 400 \\ &\quad x_1, x_2 \geq 0 \end{aligned}$$



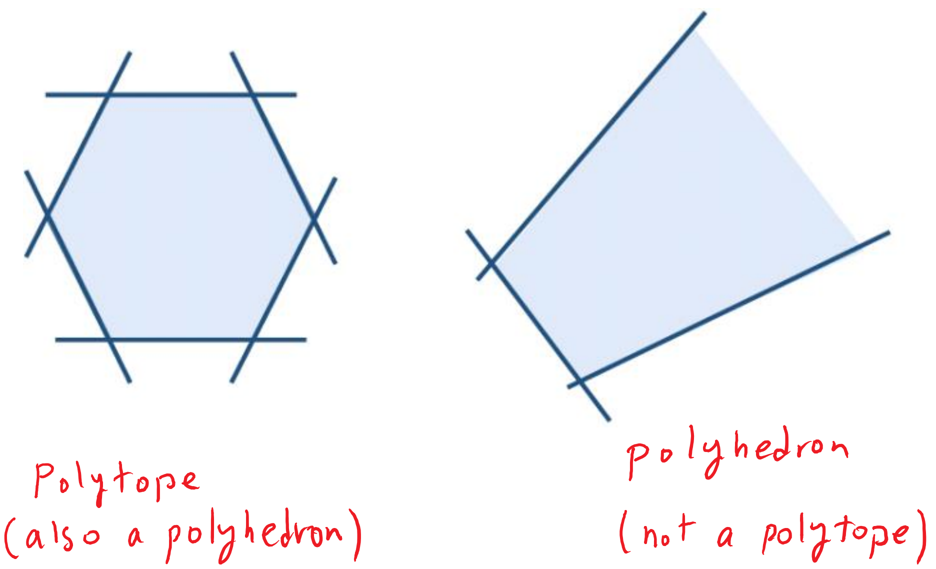
Geometry of LP

The geometry of linear programming is very beautiful. The simplex algorithm exploits this geometry in a very fundamental way. We'll prove some basic geometric results here that are essential to this algorithm.

Definition of a polyhedron:

- The set $\{x \ |a^T x = b\}$ where a is a nonzero vector in $\mathbb{R}^n$ is called a *hyperplane*.
- The set $\{x \ |a^T x \geq b\}$ where a is a nonzero vector in $\mathbb{R}^n$ is called a *halfspace*.
- The intersection of finitely many half spaces is called a *polyhedron*.
  - This is always a convex set:
  - Halfspaces are convex (why?) and intersections of convex sets are convex (why?).

- A set $S \subset \mathbb{R}^n$ is *bounded* if $\exists K \in \mathbb{R}_+$ such that $||x|| \leq K, \forall\, x \in S$.
- A bounded polyhedron is called a *polytope*.

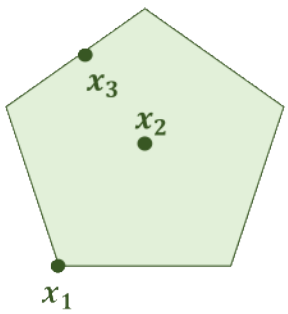


Extreme points

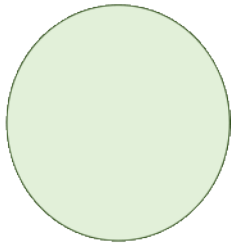
A point x is an *extreme point* of a convex set P if it cannot be written as a convex combination of two other points in P. In other words, there does not exist $y, z \in P$, $y \neq x, z \neq x$, and $\lambda \in [0,1]$ such that
$x = \lambda y + (1-\lambda)z.$

Alternatively, $x \in P$ is an extreme point if $x = \lambda y + (1-\lambda)z,\ \ y, z \in P$, $\lambda \in [0,1] \Rightarrow x = y$ or $x = z$.

Which is extreme?



What are the extreme points here?



Extreme points are always on the boundary, but not every point on the boundary is extreme.

Definition. Consider a set of constraints

$$\begin{aligned} a_i^T x &\geq b_i, \quad && i \in M_1 \\ a_i^T x &\leq b_i, \quad && i \in M_2 \\ a_i^T x &= b_i, \quad && i \in M_3. \end{aligned}$$

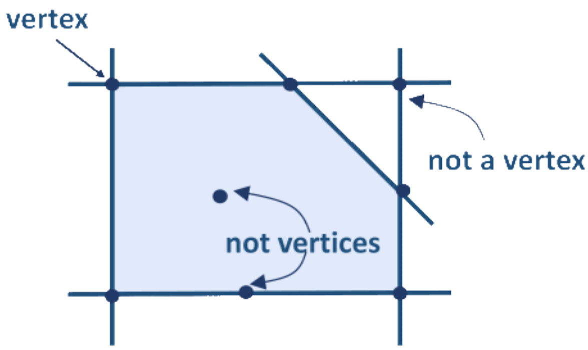
Given a point $\bar{x}$, we say that a constraint i is *tight* (or *active* or *binding*) at $\bar{x}$ if $a_i^T \bar{x} = b_i$.
Equality constraints are tight by definition.

Definition. Two constraints are *linearly independent* if the corresponding $a_i's$ are independent.

- With these two definitions, we can now define the notion of a *vertex* of a polyhedron.

Vertex

A point $x \in \mathbb{R}^n$ is a vertex of a polyhedron P, if
(i) it is feasible ($x \in P$),
(ii) $\exists\, n$ linearly independent constraints that are tight at x.



- You may be wondering if extreme points and vertices are the same thing. The theorem below establishes that this is indeed the case.
- Note that the notion of an extreme point is defined *geometrically* while the notion of a vertex is defined *algebraically*.
- The algebraic definition is more useful for algorithmic purposes and is crucial to the simplex algorithm. Yet, the geometric definition is used to prove the fundamental fact that an optimal solution to an LP can always be found at a vertex. This is crucial to correctness of the simplex algorithm.

Theorem 1: Equivalence of extreme point and vertex

Let $P = \{x \in \mathbb{R}^n \mid Ax \le b\}$ be a non-empty polyhedron with $A \in R^{m\times n}$.
Let $\bar{x} \in P$. Then,

$$\bar{x} \text{ is an extreme point} \Leftrightarrow \bar{x} \text{ is a vertex.}$$

Proof:

($\Leftarrow$) Let $\bar{x} \in P$ be a vertex. This implies that n linearly independent constraints are tight at $\bar{x}$. Denote by $\tilde{A}$ an $n \times n$ matrix whose rows are that of A associated with the tight constraints. Similarly let $\tilde{b}$ be a vector of size n collecting entries of b corresponding to the tight constraints. So $\tilde{A}\bar{x} = \tilde{b}$.

Suppose we could write $\bar{x} = \lambda y + (1-\lambda)z$ for some $y, z \in P$ and $\lambda \in [0,1], y \neq \bar{x}, z \neq \bar{x}$. Then $Ay \le b, Az \le b \Rightarrow \tilde{A}y \le \tilde{b}, \tilde{A}z \le \tilde{b}$. We have:

$$\tilde{A}\bar{x} = \tilde{b} = \lambda \cdot \tilde{A}y + (1-\lambda) \cdot \tilde{A}z.$$

If $\lambda = 0$ then $x = z$ as $\tilde{A}$ is invertible. If $\lambda = 1$ then $x = y$. If $\lambda \in (0,1)$, then the previous equality forces $\tilde{A}y = \tilde{A}z = \tilde{A}\bar{x}$ which means that $\bar{x} = y = z$ as $\tilde{A}$ is invertible.

($\Rightarrow$) Suppose $\bar{x} \in P$ is not a vertex. Let $I = \{i \mid a_i^T \bar{x} = b_i\}$. Since $\bar{x}$ is not a vertex, there does not exist n linearly independent vectors $a_i, i \in I$.

We claim that there exists a vector $d \neq 0\ s.t.\ d^T a_i = 0, \forall i \in I$.
Indeed, take at most $(n-1)$ linearly independent a_i, $i \in I$. We want to argue that the linear system

$$\begin{bmatrix} - & a_1^T & - \\ & \vdots & \\ - & a_{n-1}^T & - \end{bmatrix} \begin{bmatrix} | \\ d \\ | \end{bmatrix} = 0$$

has nontrivial solution. But recall that each a_i^T is of length n. So this is an underconstrained linear system. Hence, it has infinitely many solutions, among which there is at least one nonzero solution, which we take to be d.

Let $y = \bar{x} + \epsilon d$ and $z = \bar{x} - \epsilon d$ where ϵ is some positive scalar.
We claim that for ϵ small enough we have $y, z \in P$:

- for $i \in I$: $a_i^T y = a_i^T z = b_i$, because $a_i^T d = 0$.
- for $i \notin I$: the claim follows from continuity of the function $w \to b_i - a_i^T w$ and the fact that $b_i - a_i^T x > 0$ when $i \notin I$.

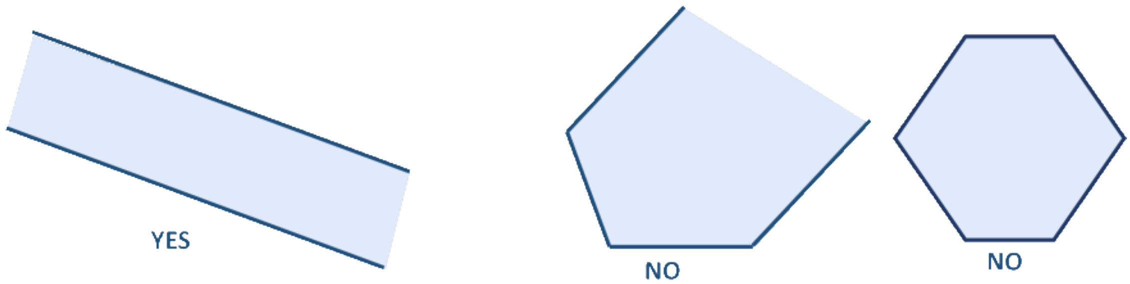
As $\bar{x} = \frac{y}{2} + \frac{z}{2}$, this implies that x is not an extreme point of P.

Corollary. Given a finite set of linear inequalities, there can only be a finite number of extreme points.

Proof:
We have shown that extreme points and vertices are the same, so we prove the result for vertices. Suppose we are given a total of m constraints. To obtain a vertex, we need to pick n linearly independent constraints that are tight. There are at most $\binom{m}{n}$ ways of doing this and each subset of n linearly independent constraints can give a unique vertex (recall the vertex x satisfies $\tilde{A}x = \tilde{b}$ where $\tilde{A}$ is invertible). We conclude that there are at most $\binom{m}{n}$ vertices. (Note that this may be a loose upper bound as the solution to some of these $\binom{m}{n}$ linear systems may not be feasible and hence won't qualify as a vertex.)

Definition. A polyhedron *contains a line* if $\exists x \in P$ and $d \in \mathbb{R}^n, d \neq 0$, such that

$$x + \lambda d \in P, \forall\, \lambda \in \mathbb{R}.$$



Theorem 2: Existence of extreme points

Consider a nonempty polyhedron P. The following are equivalent:
1. P does not contain a line.
2. P has at least one extreme point.

Corollary.
Every bounded polyhedron (i.e., every polytope) has an extreme point.

Theorem 3: Optimality of extreme points

Consider the LP: $\min c^T x$
s.t. $Ax \geq b$ (P)

Suppose P has at least one extreme point. Then, if there exists an optimal solution, there also exists an optimal solution which is at a vertex.

Proof.
Let Q be the set of optimal solutions (assumed to be nonempty). In other words, if v is the optimal value of the LP, then
$Q := \{x \mid Ax \geq b, c^T x = v\}$.
Observe the following implications:
P has an extreme point $\Rightarrow$ (by Thm. 2) P has no lines $\Rightarrow$(since Q⊆P) Q has no lines $\Rightarrow$ (by Thm.2) Q has an extreme point. Let x^* be an extreme point of Q. We will show that x^* is also an extreme point of P. Once this is proved, since we know $c^T x^* = v$, we would be done.

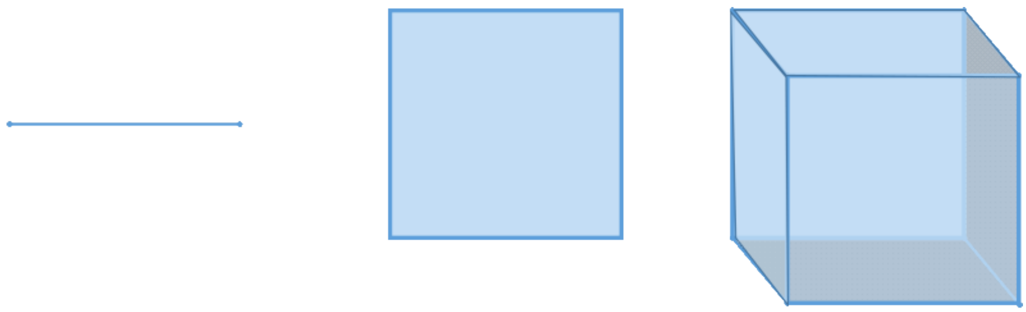
Suppose that x^* is not an extreme point of P. Then $\exists y \neq x^*, z \neq x^*$,
$$\lambda \in [0,1], \text{ s.t. } x^* = \lambda y + (1-\lambda)z.$$
Multiplying by c^T on both sides, we obtain:
$$v = c^T x^* = \lambda c^T y + (1-\lambda) c^T z$$
Since v is optimal, $c^T y \geq v, c^T z \geq v$. Combined with the previous equality, this implies: $c^T y = v, c^T z = v \Rightarrow y \in Q, z \in Q \Rightarrow x^*$ is not an extreme point of Q. Contradiction.

An implication of these theorems

- These theorems show that when looking for an optimal solution, it is enough to examine only the extreme points (=vertices).

- This leads to an algorithm for solving an LP: if there are m constraints in $\mathbb{R}^n$ (the polytope is $\{Ax \leq b\}$), then pick all possible subsets of n linearly independent constraints out of the m. Solve (in worst case) $\binom{m}{n}$ systems of equations of the type $\tilde{A}x = \tilde{b}$ where $\tilde{A}, \tilde{b}$ are the restrictions of A and b to the subset of n constraints. This can be done, e.g., by the conjugate gradien method from the previous lecture or by Gaussian elimination. Check feasibility of the solution, evaluate the objective function at each solutuion, and pick the best.

- Unfortunately, this algorithm, even though correct, is very inefficient. The reason is that there are too many vertices to explore as the number $\binom{m}{n}$ is exponential in n. For example, consider the constraints $\{-1 \leq x_i \leq 1\}$, $i = 1, \dots, n$. You see that we have only $2n$ inequalities, but 2^n extreme points:



n=1	n=2	n=3
2 inequalities	4 inequalities	6 inequalities
2 extreme points	4 extreme points	8 extreme points

- The simplex method is an intelligent algorithm for reducing the number of vertices that we have to visit.

The Simplex algorithm

Main idea

Consider some generic LP: $max.\, c^T x$

$$\text{s.t. } Ax \leq b$$
$$x \geq 0$$

In a nutshell, this is all the simplex algorithm does:
- start at a vertex,
- while there is a better neighboring vertex, move to it.

Definition. Two vertices are neighbors if they share $n-1$ tight constraints.

Example:

$$\begin{aligned} \max.\ & 20x_1 + 10x_2 + 15x_3 & \\ \text{s.t. } & 3x_1 + 2x_2 + 5x_3 \leq 55 & (1) \\ & 2x_1 + x_2 + x_3 \leq 26 & (2) \\ & x_1 + x_2 + 3x_3 \leq 30 & (3) \\ & 5x_1 + 2x_2 + 4x_3 \leq 57 & (4) \\ & x_1, x_2, x_3 \geq 0 & (5), (6), (7) \end{aligned}$$

Here is the feasible set:

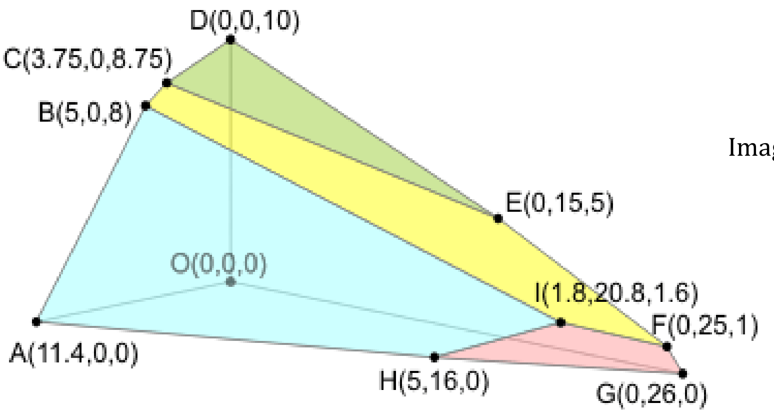


Image credit: [Jon06]

Points A,..., I are vertices of the polyhedron. Consider B=(5,0,8): the tight constraints here are (6) as $x_2 = 0$, (1) as $3 \cdot 5 + 2 \cdot 0 + 5 \cdot 8 = 55$, and (4) as $25 + 32 = 57$. These are three linearly independent constraints so B is indeed a vertex.

Points A and B are neighbors as they both have constraints (6) and (4) tight, but A has (5) tight whereas B has (1) tight. The point C is another neighbor of B.

Section 7.6 of [DPV08] gives a very neat presentation of the simplex algorithm in surprisingly few pages. We won't repeat this here since it's done beautifully in the book and we followed the book closely in lecture. Instead, we just give an outline of the steps involved and an example. The example is different from the one in the book, so you can read one and do the other for practice.

At every iteration of the simplex algorithm, we complete two tasks:

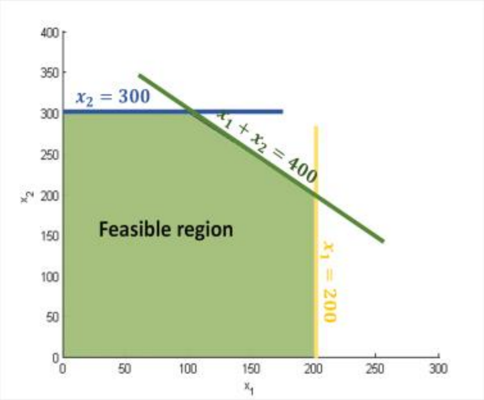
1. Check whether the current vertex is optimal (if yes, we're done),
2. If not, determine the vertex to move to next.

- We start the algorithm at the origin. We are assuming for now that the origin is feasible (i.e., $b \geq 0$). Note that if $x = 0$ is feasible, then it is a vertex (why?)
- Both tasks listed above are easy if the vertex is at the origin (we are assuming throughout that there are no "degenerate" vertices; see [DPV]):
  - The origin is optimal if and only if $c_i \leq 0, \forall i$ (why?).
  - If $c_i > 0$, for some i, we can increase x_i until a new constraint becomes tight. Now we are at a new vertex (why?).
- Once at a new vertex, we move it to the origin by a "change of coordinates". Then we simply repeat. See Section 7.6 of [DPV08] for details.
- Finally, if the origin is not feasible, to get the algorithm started we first solve an "auxiliary LP"; see Section 7.6.3 of [DPV].

An example in two dimensions

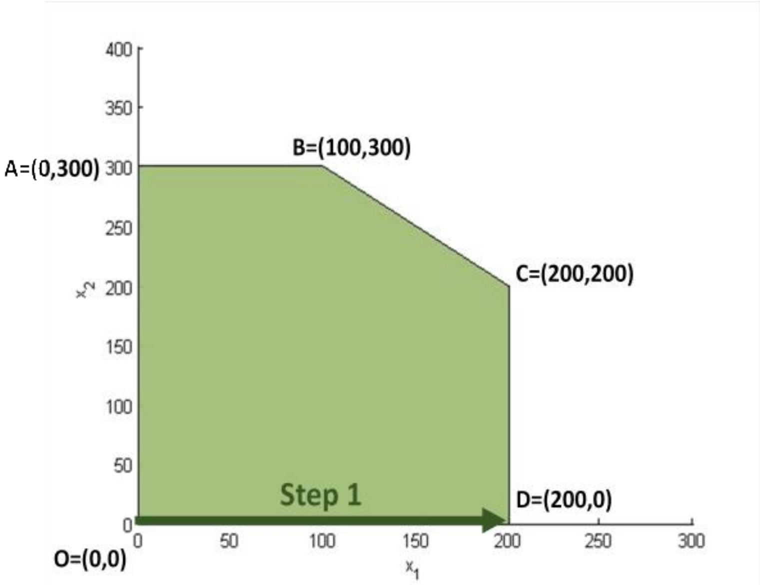
$$
\begin{array}{lll}
\text{max.} & x_1 + 6x_2 & \\
\text{s.t.} & x_1 \leq 200 & (1) \\
& x_2 \leq 300 & (2) \\
& x_1 + x_2 \leq 400 & (3) \\
& x_1 \geq 0 & (4) \\
& x_2 \geq 0 & (5)
\end{array}
$$

Here is the feasible set:



Iteration 1: The origin is feasible and all c_i are positive. Hence we can pick either x_1 or x_2 as the variable we want to increase. We pick x_1 and keep $x_2 = 0$.

The origin corresponds to the intersection of (4) and (5). By increasing x_1, we are releasing (4). As we increase x_1, we must make sure we still satisfy the constraints. In particular, we must have $x_1 \leq 200$ (1) and $x_1 + x_2 \leq 400$ (3) which means $x_1 \leq 400$. Note that constraint (1) becomes tight before constraint (3) (recall that x_2 remains at zero as we increase x_1). Hence, the new vertex is at the intersection of (1) and (5), i.e., D=(200,0).



We now need to bring D to the origin via a change of coordinates from x to y: $y_1 = 200 - x_1$ and $y_2 = x_2$.

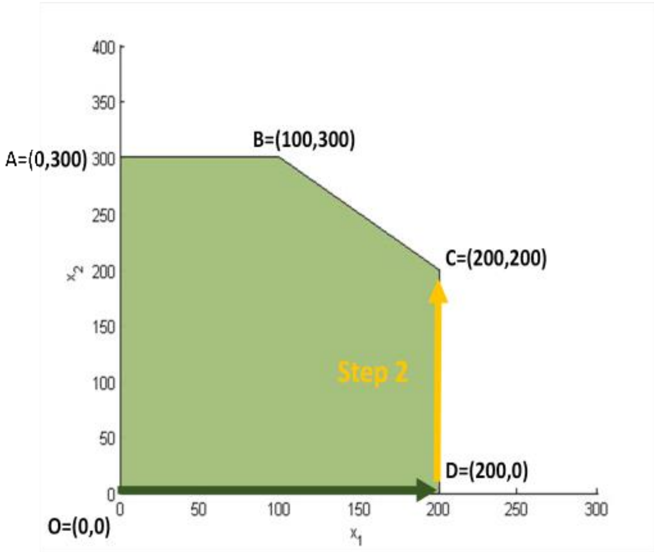
Rewriting x_1 and x_2 in terms of y_1, and y_2, we get a new LP:

$$
\begin{aligned}
\text{max. } & 200 - y_1 + 6y_2 \\
\text{s.t. } & y_1 \geq 0 \quad (1) \\
& y_2 \leq 300 \quad (2) \\
& -y_1 + y_2 \leq 200 \quad (3) \\
& y_1 \leq 200 \quad (4) \\
& y_2 \geq 0 \quad (5)
\end{aligned}
$$

Since the coefficient of y_2 is positive, we must continue.

Iteration 2: The current vertex corresponds to the intersection of (1), (5). As the coefficient of y_2 is positive, we pick y_2 as the variable to increase, i.e., we release (5). The other constraints have to be satisfied, namely (2) and (3). This requires $y_2 \leq 300$ and $y_2 \leq 200$. As $200 \leq 300$, constraint (3) is the one becoming tight next.

The new vertex is then at the intersection of (1) and (3). Reverting to the original LP, we can see that this is C.



We then change coordinates again to bring C to the origin. This is done by letting $z_1 = y_1$ and $z_2 = 200 + y_1 - y_2$; i.e., $y_1 = z_1$ and $y_2 = 200 + z_1 - z_2$.

The problem becomes:

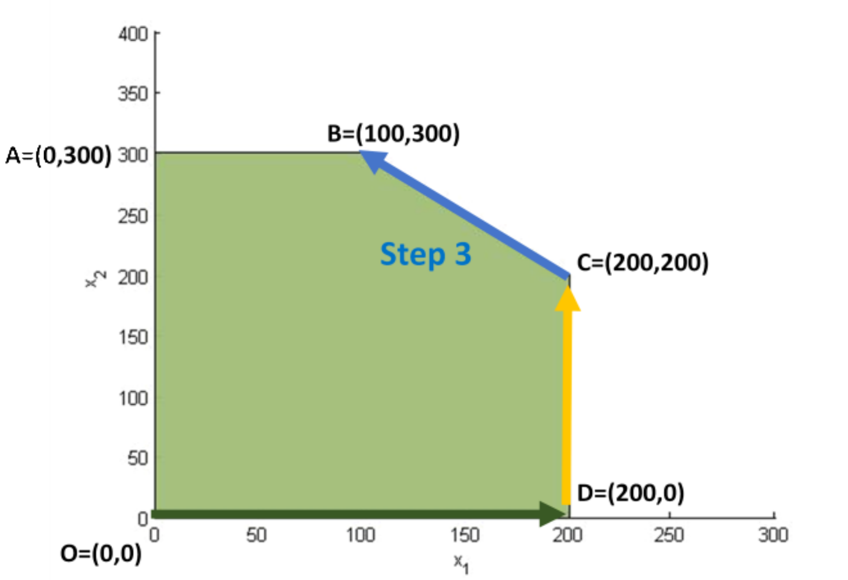
$$
\begin{aligned}
&\text{max. } 1400 + 5z_1 - 6z_2 \\
&\text{s.t.} \quad z_1 \geq 0 \quad (1) \\
&z_1 - z_2 \leq 100 \quad (2) \\
&z_2 \geq 0 \quad (3) \\
&z_1 \leq 200 \quad (4) \\
&z_1 - z_2 \geq -200 \quad (5)
\end{aligned}
$$

Since coefficient of z_1 is positive, we must continue.

Iteration 3: The current vertex is at the intersection (1) and (3). As the coefficient for z_1 is positive, we pick z_1 as the variable we will increase while keeping $z_2 = 0$. This means that we are releasing constraint (1).

We have to meet all the constraints, limiting how much we can increase z_1: $z_1 \leq 100, z_1 \leq 200$ and $z_1 \geq -200$. So (2) is the constrint becoming tight next.

The new vertex is the intersection of (3) and (2). This is the point B.



We change coordinates to make B the new origin. This is done by letting $w_1 = 100 - z_1 + z_2$ and $w_2 = z_2$, i.e., $z_1 = 100 - w_1 + w_2$ and $z_2 = w_2$.

The problem becomes:

$$
\begin{aligned}
&\text{max. } 1900 - 5w_1 - w_2 \\
&\text{s.t. } w_1 - w_2 \leq 100 && (1) \\
&\qquad w_1 \geq 0 && (2) \\
&\qquad w_2 \geq 0 && (3) \\
&\qquad w_1 - w_2 \geq -100 && (4) \\
&\qquad w_1 \leq 300 && (5)
\end{aligned}
$$

Both coefficients are negative. Hence we conclude that vertex B is optimal. The optimal value of our LP is 1900.

Notes:

The relevant chapter form [DPV08] for this lecture is Chapter 7. To minimize overlap with ORF 307, we skipped the sections on bimatrix games and the network algorithm for max-flow. Your [CZ13] book also has a chapter on linear programming, but reading that is optional.

References:

- [Bert09] D. Bertsimas. Lecture notes on optimization methods (6.255). MIT OpenCourseWare, 2009.

- [BT97] D. Bertsimas and J. Tsitsiklis, Introduction to Linear Optimization, Athena Scientific Series in Optimization and Neural Computation, 1997.

- [CZ13] E. K. P. Chong and S. H. Zak, An Introduction to Optimization, 4th Edition, Wiley Press, 2013.

- [DPV08] S. Dasgupta, C. Papadimitriou and U. Vazirani, Algorithms, McGraw Hills, 2008.

- [Jon06] J. Jones, Notes on linear programing, 2006.

- [Van14]: R. Vanderbei, Linear Programming: Foundations and Extensions, Fourth edition, Springer, 2014.