Lecture Y: Linear programming

Previously: Solve linear equations

$$A\vec{x} = \vec{b}$$

Today: Linear programs

Solve linear inequalities $A\vec{x} \le \vec{b}$

coordinatewise $\le$

and optimize a linear objective over the solution set:

"primal"

$$\underset{\vec{x}\in\mathbb{R}^n}{\text{maximize}} \quad \vec{c}\cdot\vec{x}$$
$$\text{subject to} \quad A\vec{x} \le \vec{b}, \quad \vec{x} \ge 0$$

coordinatewise

"dual"

$$\underset{\vec{y}\in\mathbb{R}^m}{\text{minimize}} \quad \vec{y}\cdot\vec{b}$$
$$\text{s.t.} \quad A^T\vec{y} \ge \vec{c}, \quad \vec{y} \ge 0$$

coordinatewise

Applications:

Linear programming can be applied to various fields of study. It is widely used in business and economics, and is also utilized for some engineering problems. Industries that use linear programming models include transportation, energy, telecommunications, and manufacturing. It has proven useful in modeling diverse types of problems in planning, routing, scheduling, assignment, and design. https://en.wikipedia.org/wiki/Linear_programming

$\sim 10^5$ variables

Next: Semidefinite programs

Optimize with linear constraints on positive semidef. matrices

$$\text{minimize} \quad \langle C, X\rangle$$
$$\text{s.t.} \quad \langle A_i, X\rangle = b_i, \quad i=1,\dots,m$$
$$X \succeq 0$$

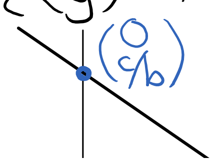
pos. semidef.

• Polytope geometry • Linear programs • Duality • Examples

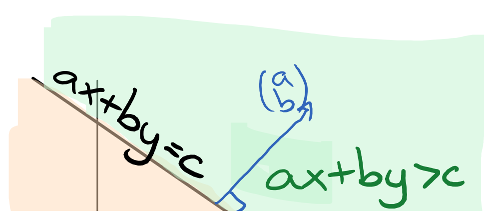
Geometry: Planes, halfspaces, polytopes

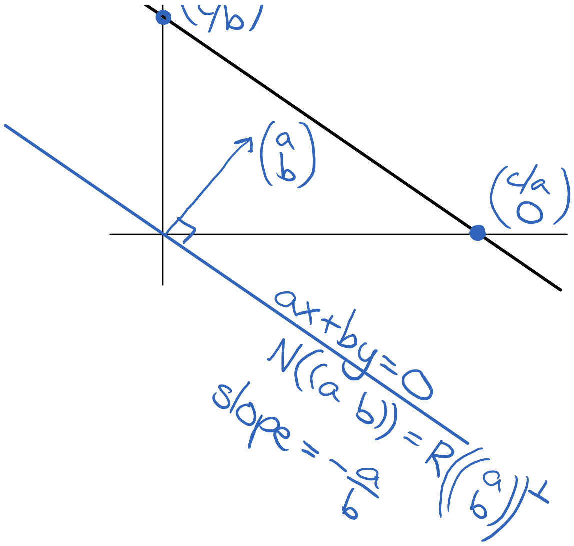
Lines:

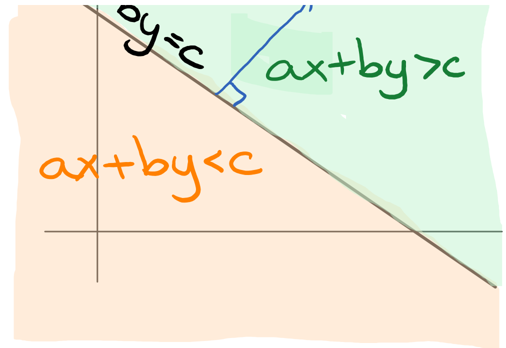
$$\left\{ \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2 \;\middle|\; ax+by=c \right\}$$



Halfspaces:

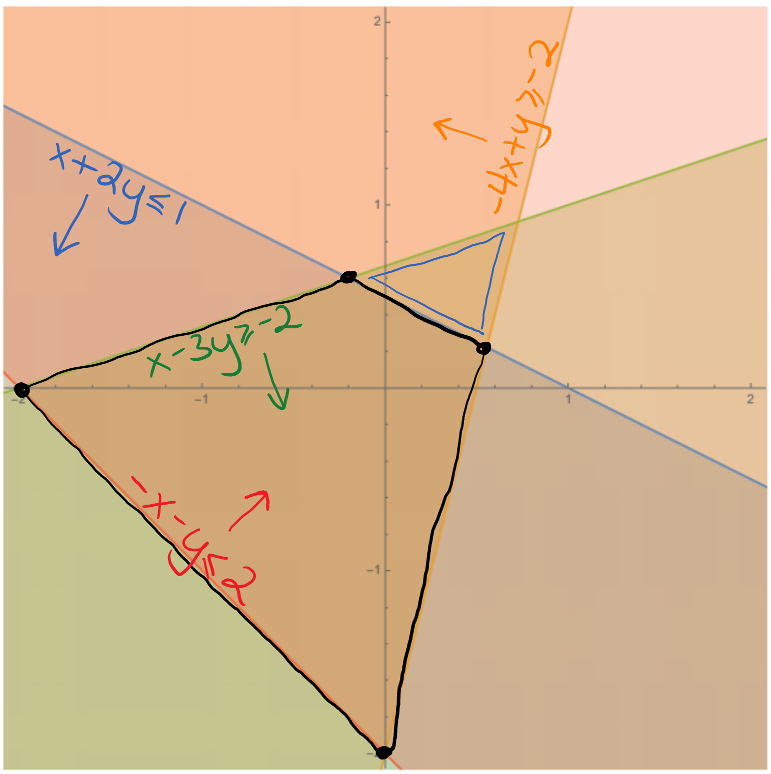






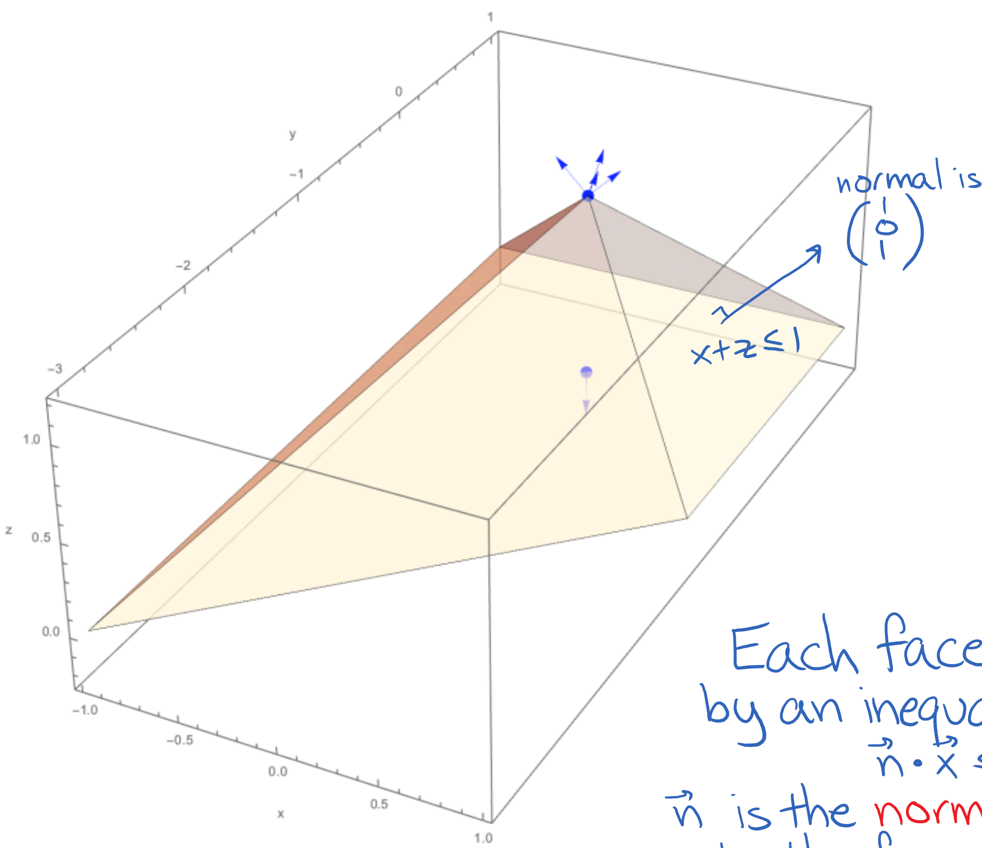
c controls translation from $\vec{0}$

Convex polygon = intersection of halfspaces




```
RegionPlot[{
    x + 2 y < 1,
    -4 x + y > -2,
    x - 3 y > -2,
    -x - y < 2
  }, {x, -2.1, 2.1}, {y, -2.1, 2.1}, Axes → True, Frame → False];
Show[%, PlotRange → {{-2, 2}, {-2, 2}}]
```

Polytopes: this all works in higher dimensions



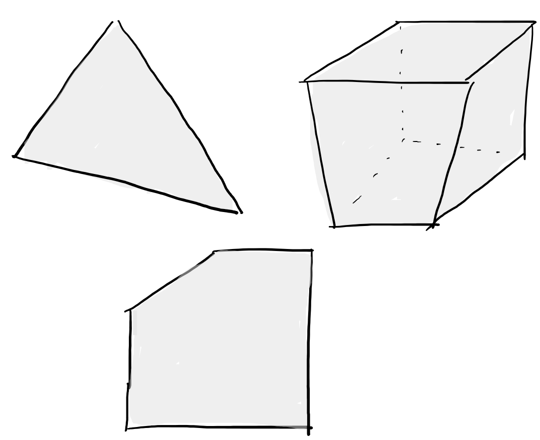
Each face is specified by an inequality

$$\vec{n} \cdot \vec{x} \le b$$

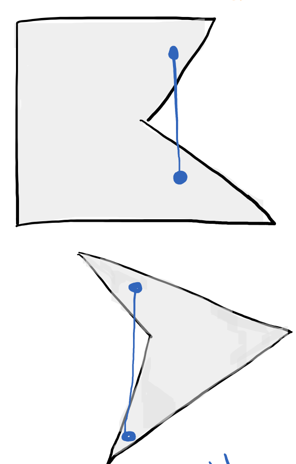
$\vec{n}$ is the normal, perpendicular to the face

Remarks:

- Set must be convex



not convex:

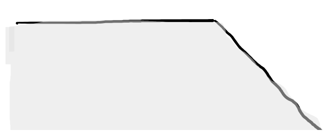


convex = between two points you stay in the set

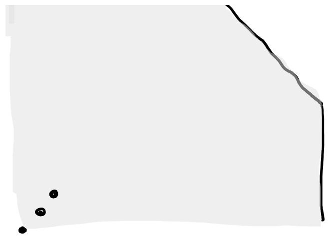
$$\vec{x}, \vec{y} \in S \quad \Rightarrow (1-p)\vec{x} + p\vec{y} \in S$$

for $0 \le p \le 1$

- Polytope may be unbounded



or empty (no solution)

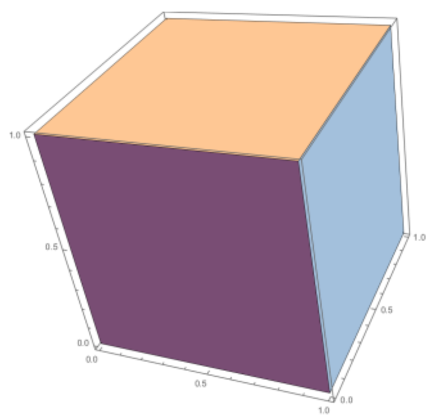


or empty (no solution)

- High-dim. polytopes can be very complex!

Example: Hypercube

$0 \le x_1 \le 1, \ 0 \le x_2 \le 1, \ \ldots, \ 0 \le x_n \le 1$

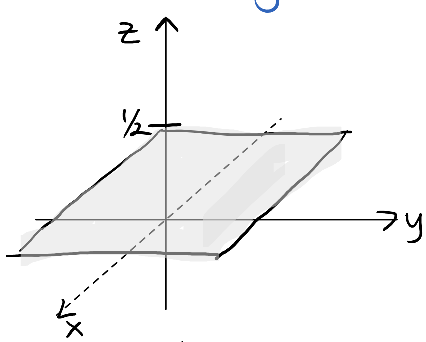


$2n$ inequalities ($2n$ faces)

but 2^n vertices!

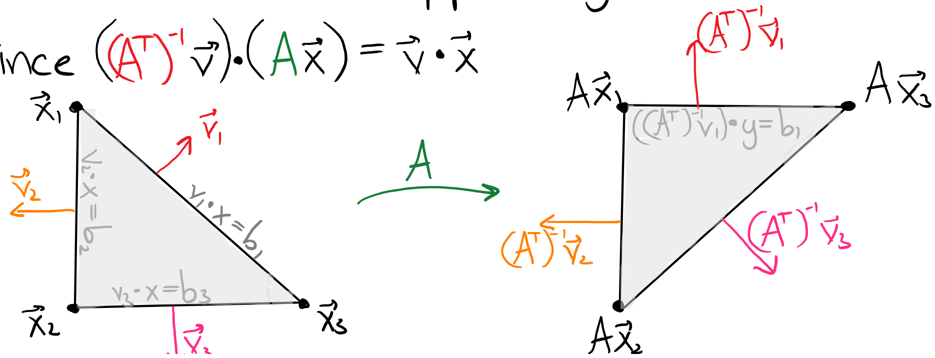
- Equalities are also okay

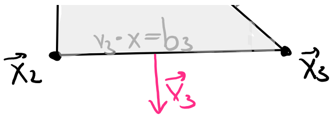
$0 \le x \le 1, \ 0 \le y \le 1, \ z = 1/2$ in $\mathbb{R}^3$

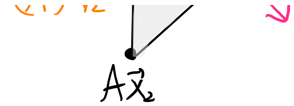


- Mapping a polytope forward by A, the normals are mapped by $(A^T)^{-1}$.

since $((A^T)^{-1}\vec{v})\cdot(A\vec{x}) = \vec{v}\cdot\vec{x}$







in this example, $A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$, $(A^T)^{-1} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}$

Linear programs

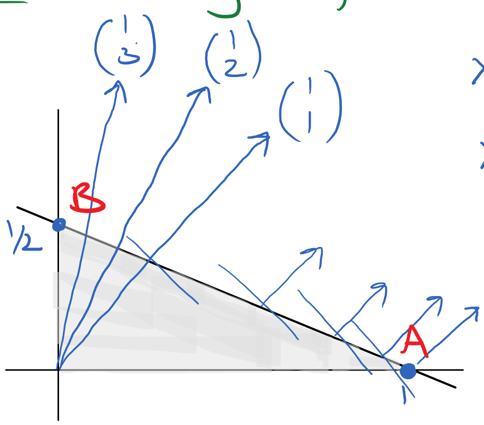
A linear program maximizes (or minimizes) a linear function over a polytope (given by inequalities).

e.g., $f(x,y,z) = ax+by+cz$

$$= \begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Observe: The maximum is at a vertex (or possibly all along a facet).

Example: $x+2y \le 1$, $x \ge 0$, $y \ge 0$

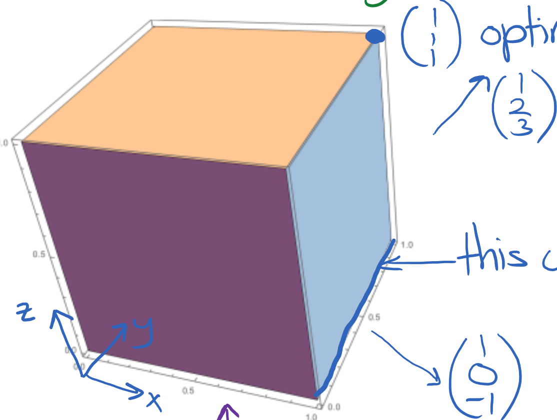


$x+y$ is maximized at A

$x+3y$ maximized at B

$x+2y$ maximized along $\overline{AB}$

Example: Cube $0 \le x,y,z \le 1$



$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ optimizes $x+2y+3z$

this whole edge maximize $x-z$

this whole face maximizes $-x$

$\swarrow \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$

Examples: LPs in Microeconomics

Example:

LP:
$$\begin{aligned} &\underset{\vec{x} \in \mathbb{R}^4}{\text{minimize}} \quad 3x_1 + x_2 + x_3 + 2x_4 \\ &\text{s.t.} \quad \begin{pmatrix} 1 & 1 & -2 & 0 \\ 0 & -1 & 1 & 1 \end{pmatrix} \vec{x} \geq \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ &\qquad \vec{x} \geq 0 \end{aligned}$$

in each coordinate

Story:

Your firm has four production processes, that can be used to produce two outputs.
Each process uses some amount of labor, and produces one of the two outputs.
(Some processes use the other output as an intermediate input.)
You want to minimize the labor cost to produce given amounts of each output.

Process 1:
costs \$3 / needs 3 workers
produces 1 unit of first output

Process 2:
costs \$1
produces 1 of 1^{st} output
consumes 1 of 2^{nd} output

⋱

Matlab:

```
>> c = [3 1 1 2]';
A = [1 1 -2 0; 0 -1 1 1];
b = [1 1]';

cvx_begin
  variable x(4,1)
  minimize (c' * x)
  A * x >= b
  x >= 0
cvx_end
```

$\Longrightarrow$

```
Status: Solved
Optimal value (cvx_optval): +5

>> x

x =

    0.5849
    0.4151
    0.0000
    1.4151
```

```
>> cvx_begin
  variable x(4,1)
  dual variables y1 y2
  minimize (c' * x)
  A(1,:) * x >= b(1) : y1
  A(2,:) * x >= b(2) : y2
  x >= 0
cvx_end
```

$\Longrightarrow$

```
>> [y1 y2]

ans =

    3.0000    2.0000
```

The dual variables can be interpreted as shadow costs.

Example:

The Silverado Mining Company's contracts with uranium-processing plants require it to produce a certain amount of each grade of uranium ore every week.

Output (tons/hour)

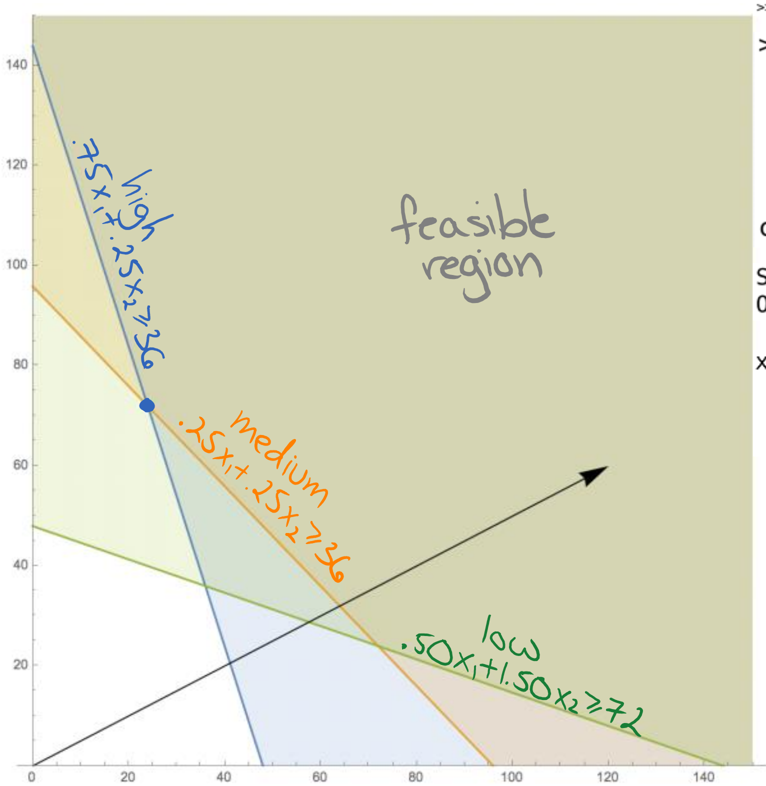
Type of ore	Mine A	Mine B	Requirements (tons/week)
High grade	.75	.25	36
Medium grade	.25	.25	24
Low grade	.50	1.50	72
Operating cost ($/hour)	$60	$30	

Variables: x_1 = Mine A hours operated per week
x_2 = Mine B " " "

Objective: minimize $60x_1 + 30x_2$

Constraints:

$$.75x_1 + .25x_2 \geq 36$$
$$.25x_1 + .25x_2 \geq 24$$
$$.50x_1 + 1.50x_2 \geq 72$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$




```
>> c = [60 30]';
>> A = [.75 .25; .25 .25; .50 1.5];
>> b = [36 24 72]';

>> cvx_begin
   variable x(2)
   dual variable y
   minimize (c' * x)
   A * x >= b : y
   x >= 0
cvx_end

Status: Solved
Optimal value (cvx_optval): +3600

x =            >> A * x      y =

   24.0000     ans =            60.0000
   72.0000                      60.0000
                  36.0000        0.0000
                  24.0000
                 120.0000
```

surplus low-grade ore is produced

Interpretation of $\vec{y}$:

Requiring one extra ton of high-grade ore will increase cost by y_1.

" " medium " " & y_2.

Requiring one extra ton of low-grade ore will not change costs, since there is a surplus ($y_3 = 0$).

Dual linear programs

"primal"

$$\underset{\vec{x} \in \mathbb{R}^n}{\text{maximize}} \quad \vec{c} \cdot \vec{x}$$

$$\text{subject to} \quad A\vec{x} \le \vec{b}, \quad \vec{x} \geq 0$$

"dual"

$$\underset{\vec{y} \in \mathbb{R}^m}{\text{minimize}} \quad \vec{y} \cdot \vec{b}$$

$$\text{s.t.} \quad A^T\vec{y} \geq \vec{c}, \quad y \geq 0$$

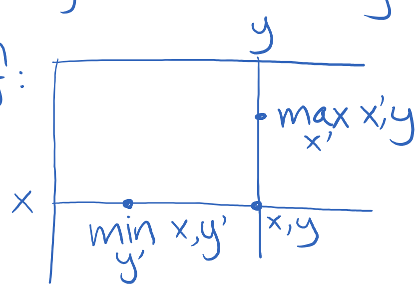
$$= \max_{\vec{x} \geq 0} \min_{\vec{y} \geq 0} \underbrace{\vec{c} \cdot \vec{x} + \vec{y} \cdot (\vec{b} - A\vec{x})}$$

$$\le \min_{\vec{y} \geq 0} \max_{\vec{x} \geq 0} \vec{y} \cdot \vec{b} + \vec{x} \cdot (\vec{c} - A^T \vec{y})$$

(For any function $f(x,y)$,

$$\max_x \min_y f(x,y) \le \min_y \max_x f(x,y)$$

Proof:



)

Terminology note: The primal and dual are both LPs. Which you call "primal" and which "dual" just depends on where you started. The dual of the dual is the primal.

Weak duality theorem:

If $\vec{x}$ is primal feasible, $\vec{y}$ dual feasible, then

$$\vec{c}\cdot\vec{x} \le \vec{b}\cdot\vec{y}$$

Proof:

$$\vec{c}\cdot\vec{x} \le (A^T\vec{y})\cdot\vec{x} = \vec{y}\cdot(A\vec{x}) \le \vec{y}\cdot\vec{b} \qquad \boxed{\checkmark}$$

Note: Solvers can optimize primal and dual together, and stop when $b\cdot y - c\cdot x \approx 0$.

Strong duality theorem:

$$\boxed{\text{Primal value} = \text{Dual value}}$$ if either is feasible

primal $\max_{x\ge 0} c\cdot x$, $Ax \le b$ dual $\min_{y\ge 0} b\cdot y$, $A^Ty \ge c$

Four cases:

- Both feasible: $\vec{c}\cdot\vec{x}^* = \vec{b}\cdot\vec{y}^*$
- Both infeasible
- Primal $= +\infty$, Dual infeasible
- Primal infeasible, Dual $= -\infty$

For a nondegenerate program (meaning optimum achieved at intersection of exactly n halfspace),

Optimal dual variables give "prices":

Relaxing ith constraint to $b_i + \varepsilon$, objective increases by εy_i (for suff. small $\varepsilon > 0$)

Formally, letting $P(\vec{b}) = \max_{x\ge 0}\{\vec{c}\cdot\vec{x} \mid A\vec{x} \le \vec{b}\}$,

$$y_i^* = \frac{\partial}{\partial b_i} P(\vec{b}).$$ the marginal value of resource i

"Complementary slackness": "No leftovers"

If some input is left over $(Ax^*)_i < b_i$, then it is worthless $y_i^* = 0$.

(So if $y_i^* > 0$, then $(Ax^*)_i = b_i$.)

Example: Degenerate program

```
>> cvx_begin
>>   variable x
>>   dual variables y1 y2
>>   maximize ( x )
>>   x <= 1 : y1
>>   x <= 1 : y2
>> cvx_end
```

b_1

$\Longrightarrow$

```
>> [y1 y2]

ans =

     0.5000    0.5000
```

but in this case increasing b_1 does not increase the optimum at all

Example: LPs in Macroeconomics

n goods you can export

prices $p_1, \dots, p_n$

inputs $v_1, \dots, v_m$

$$A = \begin{array}{c} j \\ \left(\begin{array}{ccc} \; & \vert & \; \\ & \vert & \\ & \vert & \end{array} \right)_{m \times n} \end{array}$$

inputs needed to make 1 unit of j

primal LP:
$$\underset{\vec{x} \in \mathbb{R}^n}{\text{maximize}} \quad \vec{p} \cdot \vec{x}$$
$$\text{s.t.} \quad A\vec{x} \le \vec{v}, \quad \vec{x} \ge 0$$

$$= \max_{x \ge 0} \min_{y \ge 0} \; \vec{p} \cdot \vec{x} + \vec{y} \cdot (\vec{v} - A\vec{x})$$

$$= \min_{y \ge 0} \max_{x \ge 0} \; \vec{v} \cdot \vec{y} + \vec{x} \cdot (\vec{p} - A^T \vec{y})$$

dual LP =
$$\underset{y \in \mathbb{R}^n}{\text{minimize}} \quad \vec{v} \cdot \vec{y}$$
$$\text{s.t.} \quad A^T \vec{y} \ge \vec{p}, \quad \vec{y} \ge 0$$

Interpretation: $\vec{y}$ is a vector of primary input prices / resource costs. If you add ε of input i, earnings will grow by εy_i.

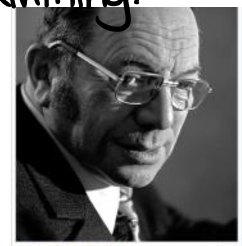
Dual: Find input costs to minimize production cost $(\vec{v} \cdot \vec{y})$

s.t. for every good j, cost to produce 1 unit $\geq p_j$.

Complementary slackness: if $x_j > 0$, cost $(A^T y)_j = p_j$
(0 profit made in equilibrium)

History of linear programming:

In 1939 a linear programming formulation of a problem that is equivalent to the general linear programming problem was given by the Soviet economist Leonid Kantorovich, who also proposed a method for solving it.[2] It is a way he developed, during World War II, to plan expenditures and returns in order to reduce costs of the army and to increase losses incurred to the enemy.[citation needed] Kantorovich's work was initially neglected in the USSR.[3] About the same time as Kantorovich, the Dutch-American economist T. C. Koopmans formulated classical economic problems as linear programs. Kantorovich and Koopmans later shared the 1975 Nobel prize in economics.[1] In 1941, Frank Lauren Hitchcock also formulated transportation problems as linear programs and gave a solution very similar to the later Simplex method;[2] Hitchcock had died in 1957 and the Nobel prize is not awarded posthumously.



Leonid Kantorovich

During 1946–1947, George B. Dantzig independently developed general linear programming formulation to use for planning problems in US Air Force. In 1947, Dantzig also invented the simplex method that for the first time efficiently tackled the linear programming problem in most cases. When Dantzig arranged a meeting with John von Neumann to discuss his Simplex method, Neumann immediately conjectured the theory of duality by realizing that the problem he had been working in game theory was equivalent. Dantzig provided formal proof in an unpublished report "A Theorem on Linear Inequalities" on January 5, 1948.[4] In the post-war years, many industries applied it in their daily planning.

John von Neumann

Dantzig's original example was to find the best assignment of 70 people to 70 jobs. The computing power required to test all the permutations to select the best assignment is vast; the number of possible configurations exceeds the number of particles in the observable universe. However, it takes only a moment to find the optimum solution by posing the problem as a linear program and applying the simplex algorithm. The theory behind linear programming drastically reduces the number of possible solutions that must be checked.

The linear programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979, but a larger theoretical and practical breakthrough in the field came in 1984 when Narendra Karmarkar introduced a new interior-point method for solving linear-programming problems.

Spufford

Irony: In math derived for communist central planning, prices arise naturally.

Caution!: LPs don't always look linear.

Example:

$$\min \|\vec{x}\|_\infty \quad = \max_i |x_i|$$
$$\text{s.t. } A\vec{x} = \vec{b}$$

$$= \min_{t, \vec{x}} t$$
$$\text{s.t. } A\vec{x} = \vec{b}$$
$$-t \leq x_i \leq t \text{ for } i = 1, \ldots, n$$

Other norms, like $\|\vec{x}\| = \sqrt{\sum_i x_i^2}$, $\|\vec{x}\|_1 = \sum_i |x_i|$, do not give LPs.

Example: Max flows and min cuts

$$\max \sum_u f_{s\to u} - \sum_v f_{v\to s} \qquad \text{Max flow}$$

$$\text{s.t. } \forall\, \omega \neq s,t \quad \sum_u f_{\omega\to u} = \sum_u f_{u\to\omega}$$

$$\forall \text{ edges} \quad f_{u\to v} \leq 1$$

$$f_{u\to v} \geq 0$$

$$= \max_{f\geq 0} \min_{\substack{s_\omega \\ x_{u\to v}\geq 0}} \sum_u (f_{s\to u} - f_{u\to s}) + \sum_{(u,v)\in E} x_{u\to v}(1 - f_{u\to v}) + \sum_{\omega\neq s,t} s_\omega \sum_u (f_{\omega\to u} - f_{u\to\omega})$$

$$\leq \min_{\substack{s_\omega \\ x_{u\to v}\geq 0 \\ s_s=1,\, s_t=0}} \max_{f\geq 0} \sum_{u\to v} x_{u\to v} + \sum_{u\to v} f_{u\to v}(-x_{u\to v} + s_u - s_v)$$

$$= \min_{\substack{x_{u\to v}\geq 0 \\ s_\omega \\ s_s=1,\, s_t=0}} \sum_{u\to v} x_{u\to v} \qquad \text{Min cut}$$

$$\text{s.t. } \quad x_{u\to v} - s_u + s_v \geq 0 \quad \forall\, u\to v$$

Proof of strong duality:

primal $\max_{x\geq 0} c\cdot x$, $Ax\leq b$ $\qquad$ dual $\min_{y\geq 0} b\cdot y$, $A^T y \geq c$

Assume (for simplicity) that the primal optimum $\vec{x}^* \in \mathbb{R}^n$ is unique and lies at the intersection of n faces

$$(A\vec{x}^*)_i = b_i \text{ for } i=1,\dots,n$$

$$(A\vec{x}^*)_j < b_j \text{ for } j>n$$

Let $V_{n\times n}$ be the matrix whose columns are the first n rows, Let $V_{n\times n}$ be the matrix whose columns are the first n rows, i.e., the face normals. $A = \begin{pmatrix} V^T \\ C \end{pmatrix}$

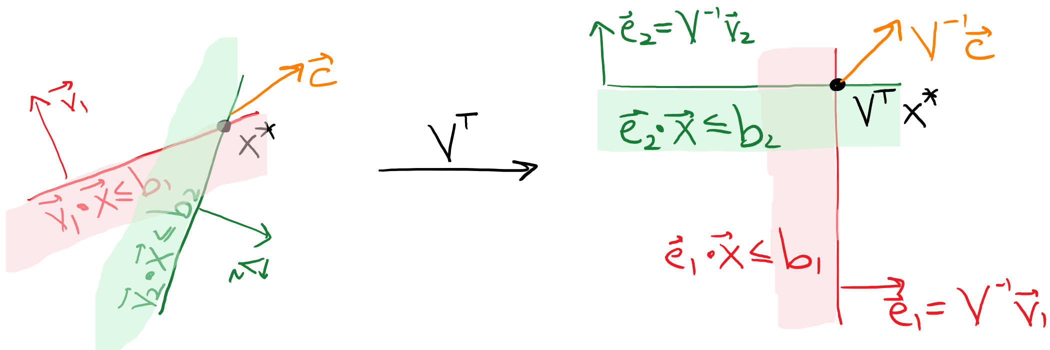
Since the optimum is unique, the normals are lin. indep.; they form a basis.

Change coordinates into this basis.

Apply V^{-1} to every normal (row of A) and to $\vec{c}$.
Apply V^T to every point $\vec{x}$. (this preserves vector-point products)

$$\max_{\substack{x \\ A\vec{x} \le \vec{b}}} \vec{c}\cdot\vec{x} = \max_{\substack{x \\ (AV^{T^{-1}})(V^T\vec{x}) \le \vec{b}}} (V^{-1}\vec{c})\cdot(V^T\vec{x})$$

$$= \max_{\substack{x \\ AV^{T^{-1}}\vec{x} \le \vec{b}}} (V^{-1}\vec{c})\cdot\vec{x}$$



Observe: Taking $b_i \to b_i + \varepsilon$, optimum increases by $(V^{-1}\vec{c})_i\,\varepsilon$.
$\vee$
0

Now let us relate this to the dual.

Let
$$\vec{y} = \begin{pmatrix} V^{-1}\vec{c} \\ 0 \end{pmatrix}$$

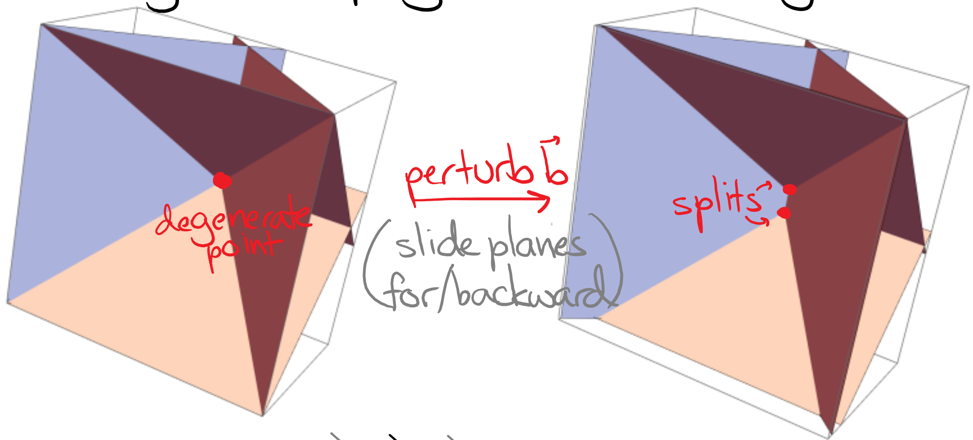
Then $\vec{y} \ge 0$.

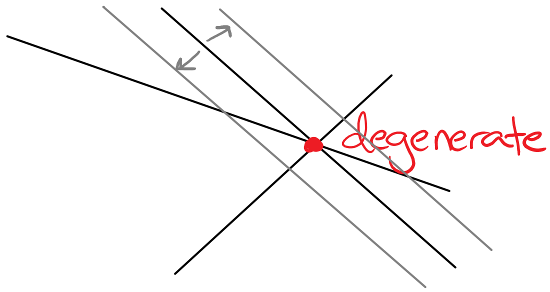
$$A^T\vec{y} = \begin{pmatrix} V & C^T \end{pmatrix}\begin{pmatrix} V^{-1}c \\ 0 \end{pmatrix} = \vec{c}$$

$$b\cdot y = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}\cdot V^{-1}\vec{c} = \left[V^{T^{-1}}\begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}\right]\cdot\vec{c} = x^*\cdot c$$

Hence $\vec{y}$ satisfies the dual constraints, and matches the primal objective. By weak duality, it's optimal. $\square$

Remark: If you randomly perturb A and/or $\vec{b}$, you'll get a nondegenerate program. (This is "generic.")

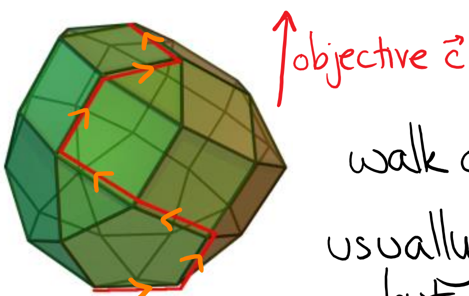




How computers solve LPs

https://en.wikipedia.org/wiki/Linear_programming#Algorithms

Simplex method



walk on polytope edges to improve objective

usually good in practice,
but exponentially slow in worst case

"Interior point" methods are provably polynomial time, and can be practical

Remarks:

* Linear programs can be generalized, eg., semi-definite programs (SDPs)
* But optimizing integer programs is NP-hard
  ↳ solutions must be integers

More examples:

Exercise: Compute the dual LP for

$$\min_{\vec{x}} \quad \vec{c}\cdot\vec{x}$$

$$\text{subject to} \quad \begin{aligned} A\vec{x} &\leq \vec{a} \\ B\vec{x} &= \vec{b} \\ \vec{x} &\geq \vec{0} \end{aligned}$$

Answer:

$$= \min_{\vec{x}\geq 0} \max_{\substack{\vec{y}\geq 0 \\ \vec{z}}} \vec{c}\cdot\vec{x} - \vec{y}\cdot(\vec{a}-A\vec{x}) + \vec{z}\cdot(B\vec{x}-\vec{b})$$

$$\geq \max_{\substack{\vec{y}\geq 0 \\ \vec{z}}} \min_{\vec{x}\geq 0} \left[\begin{aligned} &-\vec{y}\cdot\vec{a} - \vec{z}\cdot\vec{b} \\ &\quad + \vec{x}\cdot(\vec{c} + A^T\vec{y} + B^T\vec{z}) \end{aligned} \right]$$

$$= \max_{\substack{\vec{y}\leq 0 \\ \vec{z}}} \vec{a}\cdot\vec{y} + \vec{b}\cdot\vec{z} \qquad (\text{switching } \vec{y} \text{ with } -\vec{y},\ \vec{z} \text{ w/ } -\vec{z})$$

$$\text{s.t.} \quad A^T\vec{y} + B^T\vec{z} \leq \vec{c}$$

Check in Matlab:

```
>> m1 = 5; m2 = 6; n = 15;
>> A = randn(m1, n); B = randn(m2, n);
>> a = randn(m1, 1); b = randn(m2, 1); c = randn(n, 1);
>> cvx_begin
>> variables x(n, 1)
>> minimize (c' * x)
>> A * x <= a
>> B * x == b
>> x >= 0
>> cvx_end
```

Primal program:

```
------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): -18.4292
```

```
>> cvx_begin
>> variables y(m1, 1)
>> variables z(m2, 1)
>> maximize (a' * y + b' * z)
>> A' * y + B' * z <= c
>> y <= 0
>> cvx_end
```

Dual program:

```
------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): -18.4292
```