(a few remarks on) Graphs of Polytopes

Lec 12
Sep 22

In $\mathbb{R}^{\geq 4}$ we see a new phenomenon:

- polytopes whose graph is K_n (not simplices)
- different polytopes with the same graph.

Let's talk about this graph $G(P) = (V, E)$ verts of P, edges of P

Def A linear fn. cx on $\mathbb{R}^d$ is in general position with respect to a polytope P if $cv_i \neq cv_j$ for any two vertices $v_i \neq v_j$

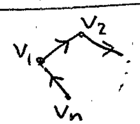
It induces an orientation of $G(P)$:

$$v_i \to v_j \quad \text{if} \quad cv_i < cv_j.$$

Prop The orientation of $G(P)$ induced by a general c

- is acyclic (no directed cycles)
- has a unique sink s (vx with no outgoing edges)

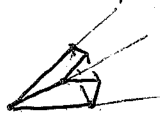
The function $c \cdot x$ is maximized at s.

Pf • If , $cv_1 < cv_2 < \cdots < cv_n < cv_1$. ✓

- Any acyclic graph has a sink. (Walk until you can't.)

Sup t is a sink.

Let $N(t) = \{v \in V : [t,v] \text{ edge}\} =$ "neighbors"
$= \{v_1, \ldots, v_k\} \leftrightarrow$ verts of P/t.


t

Then $c \cdot t > c \cdot v_i$ (all i)

$P \subset t + \text{cone}(v_1 - t, \ldots, v_n - t)$

so for $p \in P$, $p = t + \lambda_1(v_1 - t) + \cdots + \lambda_n(v_n - t)$: $(\lambda_i \geq 0)$

$c \cdot p = c \cdot t + \lambda_1 c \cdot (v_1 - t) + \cdots + \lambda_n c \cdot (v_n - t) \leq c \cdot t$ ∎

Linear Programming: maximize $c \cdot x$ subject to $Ax \leq b$

(i.e.: find the maximizing vertex)

"Easy"!

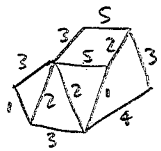
Dantzig's Simplex Algorithm:

- Start at a vertex v
- If v is a sink, done!
  If it is not, move to a $w \in N(v)$ with $v \to w$.
  Repeat!

With some tweaking, this works reasonably well in most applications.

Lec 13
Sep 24

Sample Application. (One of MANY.)



$G = (V, E)$ $\quad c: E \to \mathbb{R}_{\geq 0}$

Goal: Find the cheapest spanning tree (connected, no cycles, hits all the vertices)

Strategy: Use the spanning tree polytope:

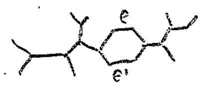
$$T \text{ spanning tree} \to \chi_T = (\chi_T(e))_{e \in E}$$
$\to$ 0 if $e \notin T$
1 if $e \in T$

$$ST(G) = \text{conv}(\chi_T : T \text{ spanning tree})$$

Then we are looking for the vertex of $ST(G)$ maximizing $c \cdot x$. "Just" use linear programming!

Well, we should know what the neighbors are:

$$T, T' \text{ neighbors} \iff T' = T \setminus e \cup e'$$



This turns into an efficient algorithm.
(Can be made to finish in $|V|-1$ steps, although $|ST(G)|$ can be as large as $|V|^{|V|-2}$)