Linear programming.

- define
- assumptions:
  - nonempty, bounded polyhedron
  - minimizing x_1
  - unique minimum, at a vertex
  - exactly d constraints per vertex
- definitions:
  - hyperplanes H
  - **basis** $B(H)$ of hyperplanes that define optimum
  - optimum value $O(H)$
- Simplex
  - exhaustive polytope search:
  - walks on vertices
  - runs in $O(n^{\lceil d/2 \rceil})$ time in theory
  - often great in practice
- polytime algorithms exist (ellipsoid)
- but bit-dependent (weakly polynomial)!
- OPEN: strongly polynomial LP
- goal today: polynomial algorithms for small d

Random sampling algorithm

- Goal: find $B(H)$
- Plan: random sample
  - solve random subproblem
  - keep only violating constraints V
  - recurse on leftover
- problem: violators may not contain all of $B(H)$
- bf BUT, contain **some** of $B(H)$
  - opt of sample better than opt of whole

- but any point feasible for $B(H)$ no better than $O(H)$
- so current opt not feasible for $B(H)$
- so some $B(H)$ violated

- revised plan:
  - random sample
  - discard useless planes, add violators to "active set"
  - repeat sample on whole problem while keeping active set
  - claim: add one $B(H)$ per iteration

- Algorithm **SampLP**:
  - set S of "active" hyperplanes.
  - if $n < 9d^2$ do simplex ($d^{d/2+O(1)}$)
  - pick $R \subseteq H - S$ of size $d\sqrt{n}$
  - $x \leftarrow$ **SampLP**$(R \cup S)$
  - $V \leftarrow$ hyperplanes of H that violate x
  - if $V \leq 2\sqrt{n}$, add to S

- Runtime analysis:
  - mean size of V at most $\sqrt{n}$
  - each iteration adds to S with prob. 1/2.
  - each successful iteration adds a $B(H)$ to S
  - deduce expect $2d$ iterations.
  - $O(dn)$ per phase needed to check violating constraints: $O(d^2n)$ total
  - recursion size at most $2d\sqrt{n}$

$$T(n) \leq 2dT(2d\sqrt{n}) + O(d^2n) = O(d^2n\log n) + (\log n)^{O(\log d)}$$

  (Note valid use of linearity of expectation)

Must prove claim, that mean $V \leq \sqrt{n}$.

- Lemma:
  - suppose $|H - S| = m$.
  - sample R of size r from $H - S$
  - then expected violators $d(m - r - 1)/(r - d)$

- **book broken: only works for empty** S

- Let C_H be set of optima of subsets $T \cup S$, $T \subseteq H$
- Let C_R be set of optima of subsets $T \cup S$, $T \subseteq R$
- note $C_R \subseteq C_H$, and $O(R \cup S)$ is only point violating no constraints of R
- Let v_x be number of constraints in H violated by $x \in C_H$,
- Let i_x indicate $x = OPT(R \cup S)$

$$\begin{aligned} E[|V|] &= E[\sum v_x i_x] \\ &= \sum v_x \Pr[i_x] \end{aligned}$$

- decide $\Pr[v_x]$
  - $\binom{m}{r}$ equally likely subsets.
  - how many have optimum x?
  - let q_x be number of planes defining x **not** already in S
  - must choose q_x planes to define x
  - all others choices must avoid planes violating x. prob.

$$\begin{aligned} \binom{m - v_x - q_x}{r - q_x} / \binom{m}{r} &= \frac{(m - v_x - q_x) - (r - q_x) + 1}{r - q_x} \binom{m - v_x - q_x}{r - q_x - 1} / \binom{m}{r} \\ &\le \frac{(m - r + 1)}{r - d} \binom{m - v_x - q_x}{r - q_x - 1} / \binom{m}{r} \end{aligned}$$

  - deduce

$$E[V] \le \frac{m - r + 1}{r - d} \sum v_x \binom{m - v_x - q_x}{r - q_x - 1} / \binom{m}{r}$$

  - summand is prob that x is a point that violates exactly one constraint in r.
    - must pick q_x constraints defining x
    - must pick $r - q_x - 1$ constraints from $m - v_x - q_x$ nonviolators
    - must pick one of v_x violators
  - therefore, sum is expected number of points that violate exactly one constraint in R.
  - but this is only d (one for each constraint in basis of R)

Result:

- saw sampling LP that ran in time $O((\log n)^{O(\log d)} + d^2 n \log n + d^{O(d)}$
- key idea: if pick r random hyperplanes and solve, expect only dm/r violating hyperplanes.

Iterative Reweighting

Get rid of recursion and highest order term.

- idea: be "softer" regarding mistakes
- plane in V gives "evidence" it's in $B(H)$
- Algorithm:
  - give each plane weight one
  - pick $9d^2$ planes with prob. proportional to weights
  - find optimum of R
  - find violators of R
  - if
  $$\sum_{h \in V} w_h \leq (2 \sum_{h \in H} w_h)/(9d-1)$$
  then double violator weights
  - repeat till no violators
- Analysis
  - show weight of basis grows till rest is negligible.
  - claim $O(d \log n)$ iterations suffice.
  - claim iter successful with prob. 1/2
  - deduce runtime $O(d^2 n \log n) + d^{d/2+O(1)} \log n$.
  - proof of claim:
    * after each iter, double weight of some basis element
    * after kd iterations, basis weight at least $d2^k$
    * total weight increase at most $(1 + 2/(9d-1))^{kd} \leq n \exp(2kd/(9d-1))$
  - after $d \log n$ iterations, done.
- so runtime $O(d^2 n \log n) + d^{O(d)} \log n$
- Can improve to linear in n

Randomized incremental algorithm

$$T(n) \leq T(n-1, d) + \frac{d}{n}(O(dn) + T(n-1, d-1)) = O(d!n)$$

Incomparable to prior bound.
Improvement to Seidel:

- Silly to discard previous info on recursion

- tested basis B, violated by H
- start from basis of $B \cup \{h\}$
- Intuition: forms good starting point for recursive call
- "hidden dimension" is how many of true basis hyperplanes are in current bases
- show hidden dimension rises quickly
- improves bound to $O(d^4 2^d N)$ (see book)

Followups:

- Kalai achieved $n^{O(\sqrt{d \log d})}$ (subexponential)
- led to more careful analysis above: $nd^{\sqrt{d \log n}}$
- combined with above to $O(d^2 n + b^{\sqrt{d \log d} \log n})$

Is polynomial possible?

- these are all simplex algorithms
- cannot do better than diameter of graph
- Kalai and Kleitman proved $n^{2+\log d}$
- must better than best algs, but still not poly

1 Voronoi Diagram

Goal: find nearest athena terminal to query point.
Definitions:

- point set p
- $V(p_i)$ is space closer to p_i than anything else
- for two points, $V(P)$ is bisecting line
- For 3 points, creates a new "voronoi" point
- And for many points, $V(p_i)$ is intersection of halfplanes, so a convex polyhedron
- And nonempty of course.
- but might be infinite
- Given VD, can find nearest neighbor view planar point location:
- $O(\log n)$ using persistent trees

Space complexity:

- VD is a **planar graph**: no two voronoi edges cross (if count voronoi points)
- add one point at infinity to make it a proper graph with ends
- Euler's formula: $n_v - n_e + n_f = 2$
- (n_v is voronoi points, not original ones)
- But $n_f = n$
- Also, every voronoi point has degree at least 3 while every edge has two endpoints.
- Thus, $2n_e \geq 3(n_v + 1)$
- rewrite $2(n + n_v - 2) \geq 3(n_v + 1)$
- So $n - 2 \geq (n_v + 3)/2$, ie $n_v \leq 2n - 7$
- Gives $n_e \leq 3n - 6$

Summary: $V(P)$ has linear space and $O(\log n)$ query time.
Which voronoi points and lines survive?

- if no other point inside circle containing them, then survive

Delaunay Triangulation

For interpolation

- Given values at set of points
- interpolate elsehwere by convex combination
- eg, topographical map with heights at given points.

Goal: no skinny triangles

- Consider 4 points in convex
- two triangulations
- one makes fatter triangles
- it's the one with no points inside those triangles
- Delaunay triangles: triples with no points inside circles

Voronoi and Delunay

- Define planar dual graph
- argue based on containined circles

Construction

Several methods

- Voronoi is projection of convex hull of lift
- Or, build Delaunay, take dual

To build Delaunay:

- Find "illegal edge", flip

Incremental construction

- Insert point (inside some triangle)
- draw 3 lines
- flip illegal edges till stable
- Claim: Illegal edges only at changes, so can propogate from insertion
- Claim: All flips produce edges incident on new point, which are Delaunay

Analysis:

- Each flip takes constant time, so proportional to number of flips
- So proportional to final number of edges on inserted point
- RIC. Average degree constant
- So flip work per insert constant
- So $O(n)$ flip work

Detail:

- Need to know which triangle point goes in
- Use point location like TD
- When destroy triangles, point their (leaf) nodes to subtrangles
- Point location search by testing all (at most 3) children
- RIC: expected depth $O(\log n)$