Convex geometry

Throughout, the ambient space is the affine space $\mathbb{R}^d$.

⊙ Convex hulls:

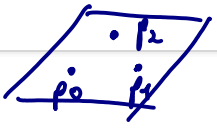
Def: Given $p_0, \dots, p_k \in \mathbb{R}^d$, an affine combination of $p_0, \dots, p_k$ is given by $\lambda_0 p_0 + \dots + \lambda_k p_k$, where $\lambda_0, \dots, \lambda_k \in \mathbb{R}$ and $\sum_{i=0}^{k} \lambda_i = 1$.

The combination is convex if, in addition, $\lambda_i \geq 0 \ \forall i$. (i.e. $\lambda_i \in [0,1]$)

Def: The affine hull of $p_0, \dots, p_k \in \mathbb{R}^d$ is $\mathrm{Aff}(p_0, \dots, p_k) := \left\{ \sum_{i=0}^{k} \lambda_i p_i \;\middle|\; \forall i\ \lambda_i \in \mathbb{R},\ \sum_{i=0}^{k} \lambda_i = 1 \right\}$

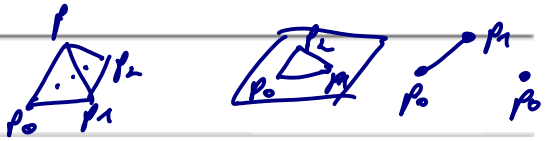
The convex hull of $p_0, \dots, p_k \in \mathbb{R}^d$ is $\mathrm{Conv}(p_0, \dots, p_k) := \left\{ \sum_{i=0}^{k} \lambda_i p_i \;\middle|\; \forall i\ \lambda_i \in [0,1],\ \sum_{i=0}^{k} \lambda_i = 1 \right\}$.

Def: $p_0, \dots, p_k$ are affinely independent if $\sum_{i=0}^{k} \lambda_i p_i = \sum_{i=0}^{k} \beta_i p_i \Rightarrow \forall i\ \lambda_i = \beta_i$.

Example / Obs.: 

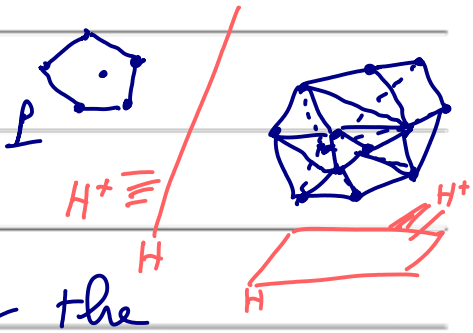
$\iff \mathrm{Aff}(p_0, \dots, p_k) \simeq \mathbb{R}^k$.

Def: An *affine k-simplex* is the convex hull of $k+1$ affinely independent points in some affine space $\mathbb{R}^d$.



→ More generally, the convex hull of $P \subset \mathbb{R}^d$ finite is a *polytope*, i.e., the intersection of finitely many closed half-spaces H^+ bounded by affine hyperplanes H:

$$\mathrm{Conv}(P) = \bigcap_{\substack{H \text{ hyperplane} \\ P \subseteq H^+}} H^+ .$$



The *bounding hyperplanes* are given by the subsets $\{p_0, \dots, p_d\} \subseteq P$ affinely indep. s.t. $P \subseteq \mathrm{Aff}(p_0, \dots, p_d)^+$.

↳ $$\mathrm{Conv}(P) = \bigcap_{\substack{p_0, \dots, p_d \in P \\ \text{aff. indep.} \\ P \subseteq \mathrm{Aff}(p_0, \dots, p_d)^+}} \mathrm{Aff}(p_0, \dots, p_d)^+ .$$

(assuming $\mathrm{Aff}(P) = \mathbb{R}^d$)

↳ $\partial\,\mathrm{Conv}(P)$ is made of convex pieces of the bounding hyperplanes, more precisely:

$$\partial\,\mathrm{Conv}(P) = \bigcup_{\substack{p_0, \dots, p_k \in P \\ \mathrm{conv}(p_0, \dots, p_k) \subset \text{ limit hyperplane}}} \mathrm{Conv}(p_0, \dots, p_k)$$

⇝ $\mathrm{Conv}(P)$ is a *combinatorial object*.

Def: P is in general position if:

- $\# P \geq d+1$, and
- $\forall \{p_0, \dots, p_k\} \subseteq P$, $k \leq d$, $p_0, \dots, p_k$ are affinely independent.

Note: {general position} is generic (open, dense) in the space of point clouds of fixed size $\geq d+1$ in $\mathbb{R}^d$.

$\Rightarrow$ can be assumed up to infinitesimal perturbations of the pts.

Prop: P is in general position $\Rightarrow$ the faces of $\partial\, \mathrm{Conv}(P)$ are simplicial (ie., affine simplices).

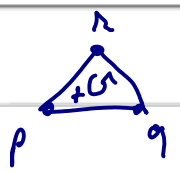
Outline of the lecture:

1. Convex hulls in 1-d: take min and max. ✓ (size $= O(1)$, running time $= O(\# P)$)
2. convex hulls in 2-d
3. convex hulls in higher d
4. miscellaneous.

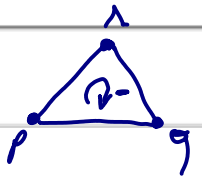
Preliminary: geometric tests:

-> for convex hulls, only the orientation test is used:

- in $\mathbb{R}^2$:



vs



$$(q-p)_x (r-p)_y - (q-p)_y (r-p)_x > 0 \qquad (q-p)_x (r-p)_y - (q-p)_y (r-p)_x < 0$$

$$\Leftrightarrow \begin{vmatrix} p_x & q_x & r_x \\ p_y & q_y & r_y \\ 1 & 1 & 1 \end{vmatrix} > 0 \qquad \Leftrightarrow \begin{vmatrix} p_x & q_x & r_x \\ p_y & q_y & r_y \\ 1 & 1 & 1 \end{vmatrix} < 0$$

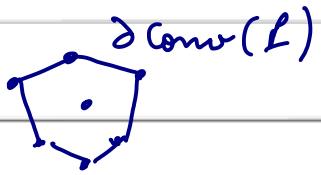
- in $\mathbb{R}^d$ $(d \geq 2)$:

$$\text{orient}(p_0, \dots, p_d) = \text{sign} \begin{vmatrix} (p_0)_1 & \dots & (p_d)_1 \\ & & \vdots \\ (p_0)_d & \dots & (p_d)_d \\ 1 & \dots & 1 \end{vmatrix}.$$

② Convex hulls in 2-d: (size and computation)

Let $P = \{p_1, \dots, p_n\} \subset \mathbb{R}^2$.

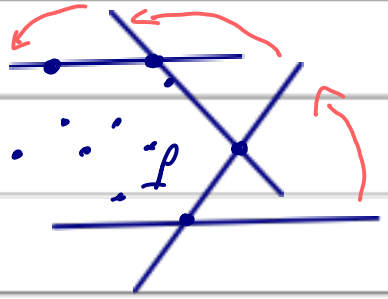
Size: $|\text{Conv}| := |\partial\text{Conv}| = O(n)$ (polygon with $\leq n$ vertices/edges).



Computation:

a) Gift-wrapping algo. [Jarvis '73]:

→ principle: place a horizontal line at the lowest point of P, then wrap the line around P counterclockwise.

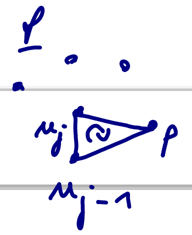


Pseudo-code: (assuming P in general position)

$u_1 := $ point of P with lowest y-coordinate (and x-coordinate if ties);

$(u_1 := \underset{p \in P}{\text{argmin}}\ p_y)$ ← $O(n)$

$j := 1$;



Do:

$j := j+1$; $u_j := p_1$;

For $i = 1$ to n do: ← $O(n)$

if $u_j = u_{j-1}$ or $\text{orient}(u_{j-1}, u_j, p_i) < 0$ then

$u_j := p_i$;

End For

while $u_j \neq u_1$; ← $O(|\text{Conv}(P)|)$ iterations

Return $(u_1, \ldots, u_j)$; ($u_j = u_1$)

Complexity: $O(n)$ space, $O(n . |\text{Conv}(P)|)$ time

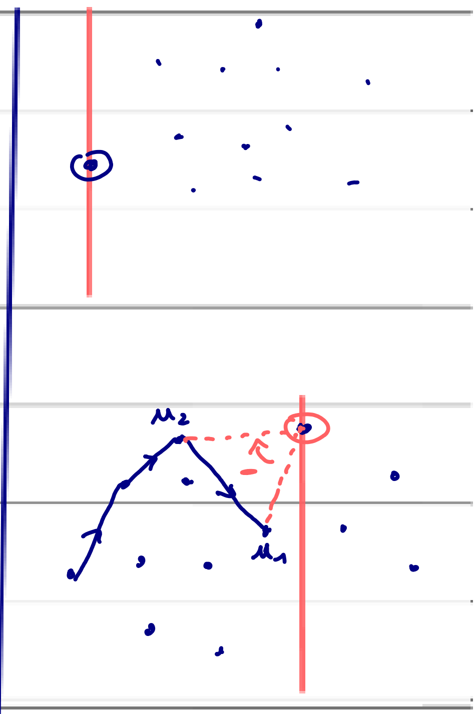
b) Sweep algorithm [Graham '72] [Andrew '79]:

→ Principle:

- sort points by increasing abcissa

- sweep points horizontally by increasing abcissa, and update the convex hull along the way via orientation tests.

- invariant, the current point is on ∂Conv, as an x-extreme point.



Pseudo-code: (computing the upper convex hull boundary)

sort the points of P by increasing abscissa: $(p_1)_x \leq \dots \leq (p_n)_x$; ← $O(n \log n)$

Stack $U = [p_1, p_2]$; // stack storing the upper convex hull boundary from left to right

For $i = 3$ to n do: ← $O(n)$ iterations

  Let u_1, u_2 be at the top of the stack U ;

  While orient$(p_i, u_1, u_2) < 0$ // p_i, u_1, u_2 oriented cw ← $O(i)$ iterations

    U.pop() ;

    If U.size() < 2 then break ;

    Let u_1, u_2 be at the top of the stack U ;

  End While

  U.push(p_i) ;

End For

Return U ;

Complexity bounds: $O(n)$ space

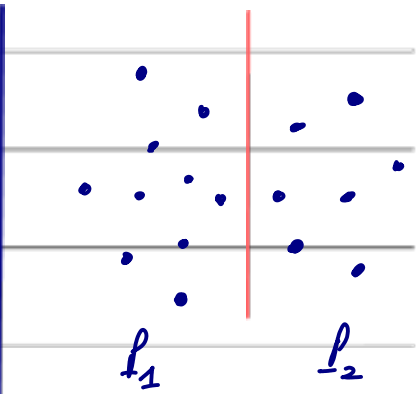
Time:
- naive: $O(n^2)$
- obs: each pt of P is inserted ≤ 1 time in U
  $\Rightarrow$ at most $O(n)$ push() / pop() in total
  $\Rightarrow$ amortized complexity in $O(n)$ for the For loop

$\rightsquigarrow$ $O(n \log n + n) = O(n \log n)$ in total.

c) Divide-and-conquer algorithm:

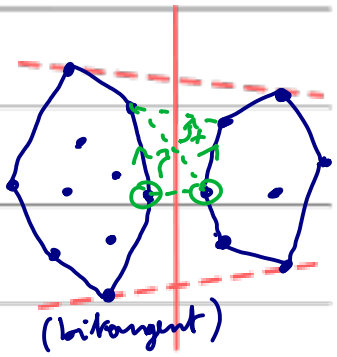
→ principle:

① divide $\mathcal{P}$ into two subsets $\mathcal{P}_1, \mathcal{P}_2$ by a vertical line passing through the median in abscissa



② recursive calls on $\mathcal{P}_1, \mathcal{P}_2$ to compute their convex hulls

③ merge the results: start from horizontally extremal points then march along the boundaries to find the upper and lower bitangents.



→ details:

① sort $\mathcal{P}$ initially (preprocessing) by abcissa ⇝ $O(n \log n)$

② nothing particular to do ⇝ $2 \cdot \text{Time}(\#\mathcal{P}/2)$

③ similar configuration as sweep algo.

↳ encode sup. and inf. convex hull boundaries in stacks and pop() during walks ⇝ $O(n)$

complexity bounds: $O(n)$ space

$$\text{Time}(n) = O(n) + 2 \cdot \text{Time}(n/2) = O(n)$$

$O(n \log n)$ preprocessing.

d) Lower bound: Size: $\Omega(n)$ (need to store P).

Thm: In the algebraic decision tree model, computing the convex hull of n points in general position in $\mathbb{R}^2$ (i.e., computing the sorted list of the vertices of $\partial \mathrm{Conv}(p_1, \dots, p_n)$) takes $\Omega(n \log n)$ time.

→ proof: by a linear reduction from sort:

given $x_1, \dots, x_n \in \mathbb{R}$ to be sorted,

let $p_1 = (x_1, x_1^2), \dots, p_n = (x_n, x_n^2) \in \mathbb{R}^2$.

Then, compute and store $\mathrm{Conv}(p_1, \dots, p_n)$.

Then, walk along the lower boundary of $\mathrm{Conv}(p_1, \dots, p_n)$, which contains all the points p_i, sorted by abscissa.

This yields the sorted list of reals x_i. ∎

③ Convex hulls in dimension $d \geq 2$:

Size:

Thm: (Upper Bound) [M. Mullen '70]

Every polytope with n vertices (hence, the convex hull of those vertices) in $\mathbb{R}^d$ has $O(n^{\lfloor \frac{d}{2} \rfloor})$ faces in total. Furthermore, this bound is tight.

Tight configuration:

points on moments curve $t \mapsto (t, t^2, t^3, \ldots, t^d)$, yielding so-called cyclic polytopes with size $\Omega(n^{\lfloor d/2 \rfloor})$

⇝ linear size in $\mathbb{R}^2/\mathbb{R}^3$, quadratic size in $\mathbb{R}^4/\mathbb{R}^5$, etc.

Computation:

Example: sweep algo. in $\mathbb{R}^3$: (reminder: size of ∂Conv in $\Theta(n)$)



the insertion
each pt on the line kills

$\Theta(n)$ faces of ∂Conv and then creates $\Theta(n)$ faces.

$\Rightarrow$ time complexity in $\Theta(n^2)$.

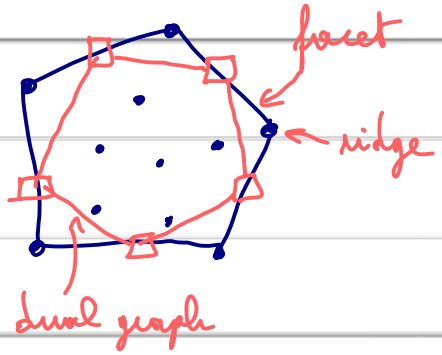
⇝ issue: insertion order is too strict.

Ideas:

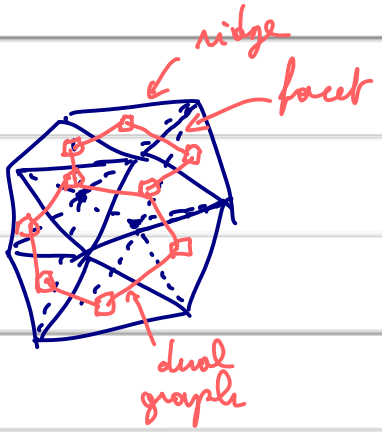
- use random insertion order on the points of P.
- requires a data structure for $\text{Conv}(P)$ that allows for point location w.r.t. $\partial\,\text{Conv}(P)$ and update of $\partial\,\text{Conv}$.
- assume constant d (complexity will be exponential in d) and general position of the points.
  $\Rightarrow$ faces of $\partial\,\text{Conv}(P)$ are simplicial.

Data structure: store the $(d-1)$-dimensional faces (called "facets") and the $(d-2)$-dimensional faces (called "ridges") via their dual incidence graph (1 vertex per facet, 1 edge per ridge).

→ in 2-d:



in 3-d:

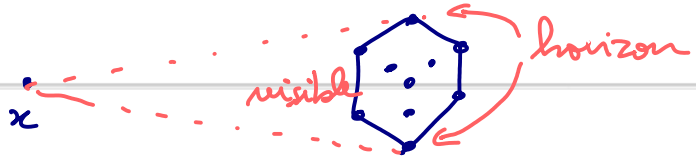


Def: A facet $\text{Conv}(p_{i_0}, \dots, p_{i_d})$ of $\partial\,\text{Conv}(P)$ is visible from $x \in \mathbb{R}^d$ if $\text{Aff}(p_{i_0}, \dots, p_{i_d})$ separates P from x. Otherwise, the facet is called invisible.

note: "visible" means that there is no other facet in-between the facet and x

Def: - A ridge incident to 2 visible facets is visible
- ——— 2 invisible ——— invisible
- ——— 1 visible facet and 1 invisible

facet is called a horizon ridge.



General Algo:

Input: $P \subset \mathbb{R}^d$ finite, of size n, in general position

Preprocessing: sort P according to some order (to be defined).
↳ sequence of points $p_1, \dots, p_n$.

Init: let $C := \text{Conv}(p_1, \dots, p_{d+1})$ // d-simplex

For $i = d+2$ to n do: (hyp: $C = \text{Conv}(p_1, \dots, p_{i-1})$)

(a) find a facet $F \in \partial C$ that is visible from p_i;
if no such facet exist then skip p_i and proceed to p_{i+1}.

(b) walk along ∂C from F
to find and delete all visible facets and ridges.
↳ eventually reach all horizon ridges.

(c) for each horizon ridge r, create the facet $(r * p_i)$,
i.e. a new vertex in the dual graph.

(d) create the new ridges (incident to p_i) by creating
the corresponding edges in the dual graph.

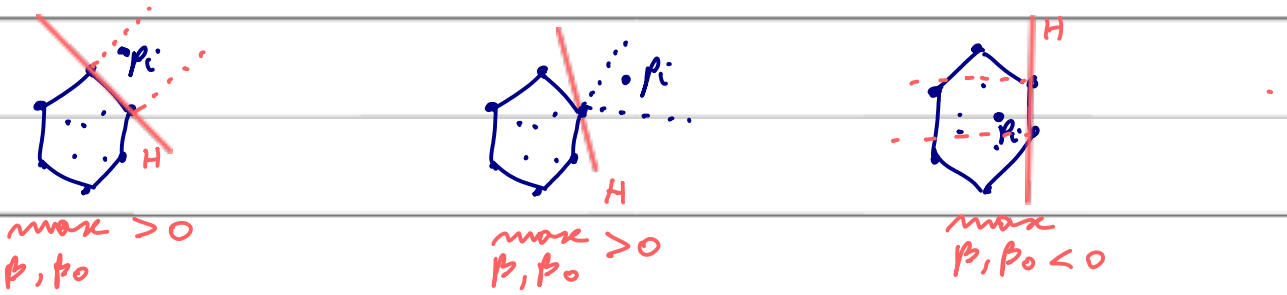
End For

Return the facets and ridges of $\text{Conv}(p_1, \dots, p_n)$, more precisely their dual graph.

Details:

(a) use linear programming: find a hyperplane H of equation $\beta^T x - \beta_0 = 0$ ($\beta \in \mathbb{R}^d$, $\beta_0 \in \mathbb{R}$) that is tangent to $\text{Conv}(p_1, \dots, p_{i-1})$ and leaves p_i "as far away as possible" on the other side.

LP: $\underset{\beta, \beta_0}{\text{argmax}} \; \beta^T p_i - \beta_0 \quad \text{s.t.} \quad \beta^T \cdot p_j - \beta_0 \leq 0 \quad \forall 1 \leq j < i$



→ if max < 0 then no visible facet exist

if max > 0 then H is a supporting hyperplane of a visible face (by the LP formulation).

notes: • H supports the facet or ridge whose "normal cone" contains p_i

• LP solver navigates along the dual graph of ∂ Conv $(p_1, \ldots, p_{i-1})$ (cf. simplex algo.)

• if H supports a ridge then return any incident facet.

(b) do a dfs or bfs on dual graph, starting from the dual vertex of F.

↳ every visible facet / ridge encountered is deleted from the dual graph.

↳ the dfs or bfs eventually reaches all horizon ridges.

(c) iterate over horizon ridges r and create the new vertices in the dual graph corresponding to the facets $(r * p_i)$.

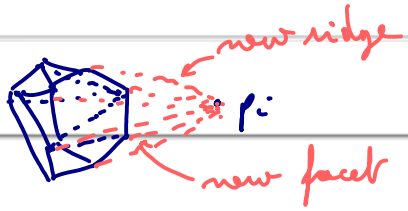
(ol) ∀ new facet $(r * p_i)$, we iterate over its $(d-1)$ new ridges (note: the facet is $(d-1)$-dimensional hence has d ridges in total, including r itself).

↳ identify each ridge by its sorted list of $(d-1)$ vertices, and attach to it a pointer to the new facet it is on the boundary of.

↳ store all the new ridges in a hash table.

↳ when we iterate over the ridges of a new facet, we check for each ridge whether it has a copy in the hash table: if so then we create the edge in the dual graph connecting the nodes dual to the new facets.

in 3-d,



Analysis: • $\forall i$, let N_i be the number of ridges on the horizon at iteration i (when p_i is inserted)

• We treat d as a constant, hidden in big-O notations.

(c) Takes $O(N_i)$ time.

(b) sorting the vertices of a ridge takes $O(d \log d) = O(1)$ time

Inserting the new ridges in the hash table (≤ 2 copies each) takes $O(N_i)$ time. Each facet has $O(d) = O(1)$ ridges to check, so $O(N_i)$ operations needed to iterate over the ridges of all new facets and creating the corresponding edges in the dual graph.

$\Rightarrow O(N_i)$ time in total for iteration i.

(b) Amortized cost is $O\left(\sum_{i=d+2}^{n} N_i\right)$, because a deleted facet or ridge has to be created first and cannot be re-created after deletion.

$\Rightarrow$ Total amortized cost of steps (b)-(c)-(d): $O\left(\sum_{i=d+2}^{n} N_i\right)$.

Q Can we bound this quantity?

Case 1: initial sort of $p_1, \dots, p_n$ is by increasing abcissa, as in the 2-d sweep algo.

↳ in worst case, $N_i = \Omega(\text{size of } \text{Conv}(p_1, \dots, p_{i-1})) = \Omega\left(i^{\lfloor d/2 \rfloor}\right)$

$$\Rightarrow \sum_{i=d+1}^{n} \Omega\left(i^{\lfloor d/2 \rfloor}\right) = \Omega\left(n \cdot n^{\lfloor d/2 \rfloor}\right) = \Omega\left(n^{\lceil d/2 \rceil}\right).$$

Case 2: initial sort of $p_1, \dots, p_n$ is by random permutation with uniform probability over the permutations.

↳ size of $\text{Conv}(p_1, \dots, p_i) = O\left(i^{\lfloor d/2 \rfloor}\right)$ and each facet of $\partial\,\text{Conv}(p_1, \dots, p_i)$ has exactly d vertices (general position hypothesis).

$$\Rightarrow \mathbb{E}[N_i] = \sum_{q \notin \partial\,\text{Conv}(p_1, \dots, p_i) \cap \{p_1, \dots, p_i\}} \Pr(q = p_i) \times 0 \quad \text{(no new facet incident to } p_i\text{)}$$

(expectation over choices of permutations)

$$+ \sum_{q \in \partial\,\text{Conv}(p_1, \dots, p_i) \cap \{p_1, \dots, p_i\}} \Pr(q = p_i) \times \#\{\text{facets of } \partial\,\text{Conv}(p_1, \dots, p_i) \text{ incident to } q\}$$

$$= \sum_{q \in \partial\,\text{Conv}(p_1, \dots, p_i) \cap \{p_1, \dots, p_i\}} \frac{1}{i} \times \#\{\text{facets of } \partial\,\text{Conv}(p_1, \dots, p_i) \text{ incident to } q\}$$

$$= \frac{1}{i} \cdot \sum_{q \in \partial\,\text{Conv}(p_1, \dots, p_i) \cap \{p_1, \dots, p_i\}} \#\{\text{facets of } \partial\,\text{Conv}(p_1, \dots, p_i) \text{ incident to } q\}$$

$$= O\left(\frac{1}{i} \cdot d \cdot i^{\lfloor d/2 \rfloor}\right) = O\left(i^{\lfloor d/2 \rfloor - 1}\right).$$

(with multiplicity, each facet counted $\leq d$ times)

Hence, $\mathbb{E}\left[\sum_{i=d+2}^{n} N_i\right] = \sum_{i=d+2}^{n} \mathbb{E}[N_i] = \sum_{i=d+2}^{n} O(i^{\lfloor d/2 \rfloor - 1})$

$= O(n^{\lfloor d/2 \rfloor})$.

(a) **Thm:** [Seidel '91]

Under a uniform random permutation, the LP problem with a constant number of variables $(d+1)$ and i constraints can be solved in $O_d(i)$ time.

$\Rightarrow$ Total (average) cost of main loop of algo.: $O(n^{\lfloor d/2 \rfloor})$ time.

cost of preprocessing: random permutation via random integer numbers assignment to the points then sort.

$\rightarrow O(n \log n)$ time.

cost of initialization: conv $(p_1, \ldots, p_{d+1})$ computed in $O(d^{\lceil d/2 \rceil}) = O(1)$ time by, e.g., sweep algorithm.

$\Rightarrow$ Total (average) running time of algo.: $\boxed{O(n \log n + n^{\lfloor d/2 \rfloor})}$.

Historical notes:

- This algo. is by Raimund Seidel [Seidel '91]
- it is a variant of the original optimal randomized algo. [Clarkson, Shor '89], working in the primal directly instead of the dual (via point/hyperplane duality).
- the algo. was derandomized around the same time [Chazelle '91].

References:

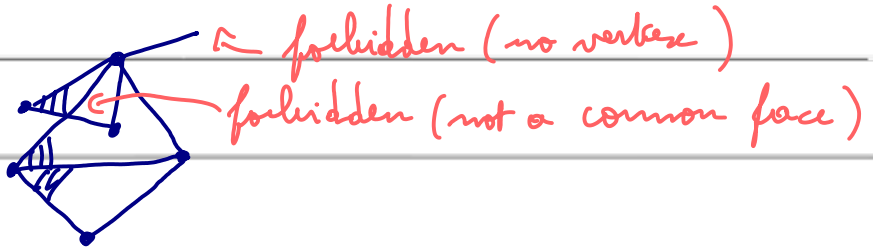
- R. Seidel (1991): Small-dimensional linear programming and convex hulls made easy, Discrete and Computational Geometry.
- K. Clarkson, P. Shor (1989): Applications of random sampling in computational geometry II, D.C.G.
- B. Chazelle (1993): An optimal convex hull algorithm in any fixed dimension, D.C.G.

④ Miscellaneous :

Def: An *affine simplicial complex* in $\mathbb{R}^d$ is a (possibly) infinite set K of affine simplices, such that:

(i) $\forall\ \sigma = \mathrm{Conv}(p_0, \dots, p_k) \in K$, $\forall\ \gamma$ *face* of σ (i.e., $\gamma = \mathrm{Conv}(q_0, \dots, q_\ell)$ for some subset $\{q_0, \dots, q_\ell\} \subseteq \{p_0, \dots, p_k\}$), $\gamma \in K$;

(ii) $\forall\ \sigma, \gamma \in K$, $\sigma \cap \gamma$ is either empty or a face of both σ and γ.



Def: The *dimension* of a simplex is the dimension of its affine hull, i.e., the number of its vertices minus 1.

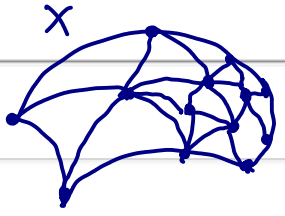
- 0-dim. simplices are called vertices
- 1-dim. ——— edges
- 2-dim. ——— triangles
- 3-dim. ——— tetrahedra
- etc.

Def: The standard k-simplex $\Delta_k \subset \mathbb{R}^{k+1}$ is $\mathrm{Conv}(e_1, \dots, e_{k+1})$.



Def: A geometric simplicial complex is a topological space X together with a family of continuous maps $\left(f_i : \Delta_{d_i} \longrightarrow X\right)_{i \in I}$ such that:

- $X = \bigsqcup_{i \in I} \mathrm{Im}\, f_i |_{\mathring{\Delta}_{d_i}}$; (note: big convention, $\mathring{\Delta}_0 = \Delta_0$)
- each f_i is a homeomorphism $\Delta_{d_i} \longrightarrow \mathrm{Im}\, f_i$;
- $\forall i \in I$, $\forall \delta$ face of Δ_{d_i}, $\exists j \in I$ s.t. $f_i|_\delta = f_j \circ \varepsilon^\delta_{d_j}$, where $\varepsilon^\delta_{d_j}$ is the natural homeomorphism $\delta \longrightarrow \Delta_{d_j}$.
- $U \subseteq X$ is open in X iff $f_i^{-1}(U)$ is open in Δ_{d_i} for all $i \in I$.



Note: affine simplicial complexes are geometric simplicial complexes.

Def: An <u>abstract simplicial complex</u> on a vertex set V is a subset $K \subseteq 2^V$ such that:

- $\forall \sigma \in K$, σ is finite;
- $\forall \sigma \in K$, $\forall \tau \subseteq \sigma$, $\tau \in K$.

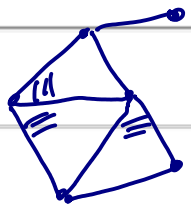
Note: no need to control "$\sigma \cap \tau$" for $\sigma, \tau \in K$ since all simplices are abstract hence "$\sigma \cap \tau$" is actually the simplex $(\sigma \cap \tau) \subseteq 2^V$, which by def. belongs to K.

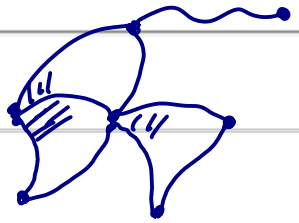
Def:

- $\forall \sigma \in K$, $\dim \sigma := \#\sigma - 1 \in \mathbb{N}$.
- $K_k := \{\sigma \in K \mid \dim \sigma = k\}$ (k-skeleton of K)
  - ↳ **note:** $K_0 \subseteq V$ and one usually ignores V itself.
- $\tau \subseteq \sigma$ is called a <u>face</u> of σ.
- $\tau \subseteq \sigma$ with $\dim \tau = \dim \sigma - 1$ is called a <u>facet</u> of σ.
- $\dim K := \sup_{\sigma \in K} \dim \sigma \in \mathbb{N} \cup \{+\infty\}$.

Def: A geometric realization of an abstract simplicial complex K is a geometric simplicial complex $(f_i : \Delta_{d_i} \to X)_{i \in I}$ together with a bijection $\gamma : K \to I$ such that:

- $\forall \sigma \in K$, $d_{\gamma(\sigma)} = \dim \sigma$
- $\forall \sigma \in K$, $\forall \delta$ face of $\Delta_{d_{\gamma(\sigma)}}$, $f_{\gamma(\sigma)}|_{\delta} = f_{\gamma(\tau)} \circ \varepsilon^{\delta}_{d_{\gamma(\sigma)}}$.





Prop: Every abstract simplicial complex admits geometric realizations.

Prop: All the geometric realizations of a given simplicial complex K are homeomorphic.

↳ The underlying space $|K|$ is any geometric realization of K.