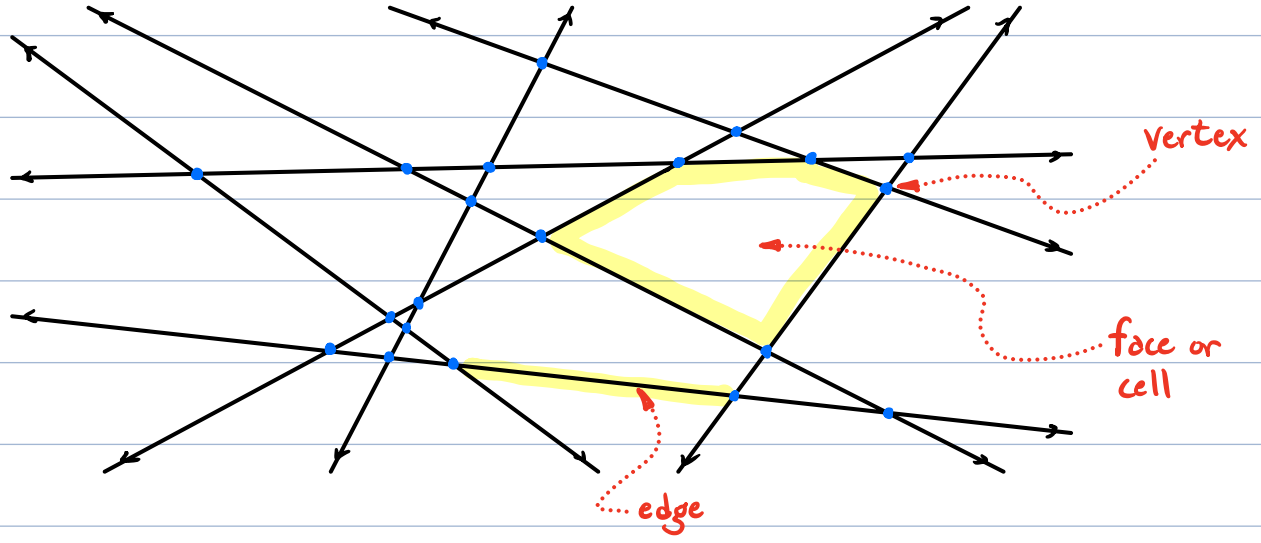
CMSC 754 - Computational Geometry
Lecture 13 - Line Arrangements + Applications

Arrangement:

Given a set $L = \{\ell_1, \ldots, \ell_n\}$ of lines in $\mathbb{R}^2$ they subdivide the plane into a cell complex (vertices, edges, faces), called an arrangement, $A(L)$



Combinatorics:

Lemma: Given n lines L in general position, $A(L)$ has:

(i) $\binom{n}{2} = \frac{1}{2} \cdot n(n-1)$ vertices

(ii) n^2 edges

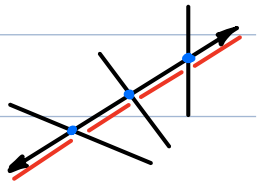
(iii) $\binom{n}{2} + n + 1 = \frac{1}{2}(n^2 + n + 2)$ faces

Proof:

(i) Each pair of lines intersect $\Rightarrow \binom{n}{2}$ vertices ✓

(ii) Each line intersects n-1 others

⇒ splits each line into n segments ✓

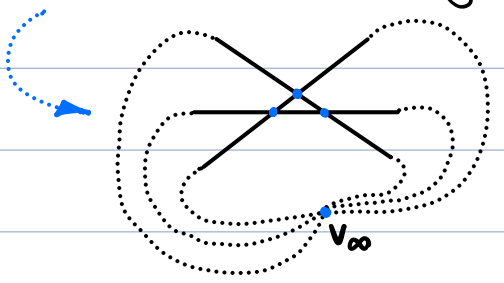


..... including 2 unbounded

To make this a true cell complex,

add vertex v_∞ and tie off unbounded edges

$$V = \binom{n}{2} + 1$$
$$e = n^2$$



By Euler's formula:

$v - e + f = 2$

$\Rightarrow \binom{n}{2} + 1 - n^2 + f = 2$

$\Rightarrow f = 2 - \frac{1}{2}n(n-1) - 1 + n^2$

$f = \frac{1}{2}(n^2 + n + 2)$ ✓ □

Total complexity = $O(n^2)$

Generalize: n hyperplanes in $\mathbb{R}^d$

(cell complex with faces of dim $0, 1, 2, \ldots, d$)

Total complexity = $O(n^d)$

Incremental Construction:

(randomization not needed)

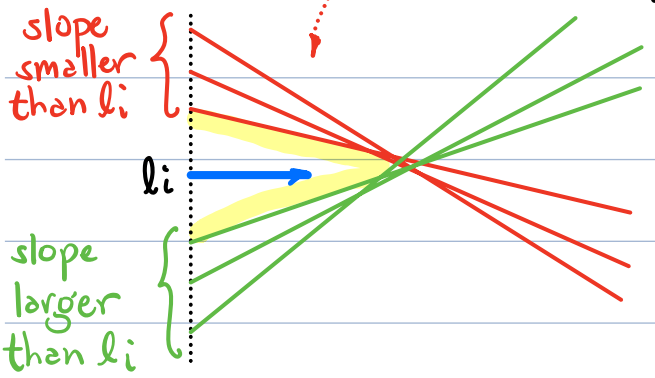
Intuition: Add lines one by one (any order)

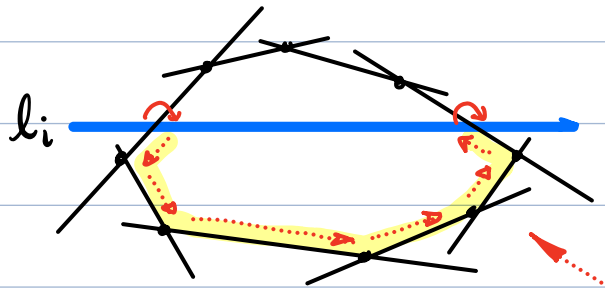
Update after each insertion

Notation: $L_i = \{l_1, \ldots, l_i\}$ $0 \le i \le n$

How to add the i^{th} line?

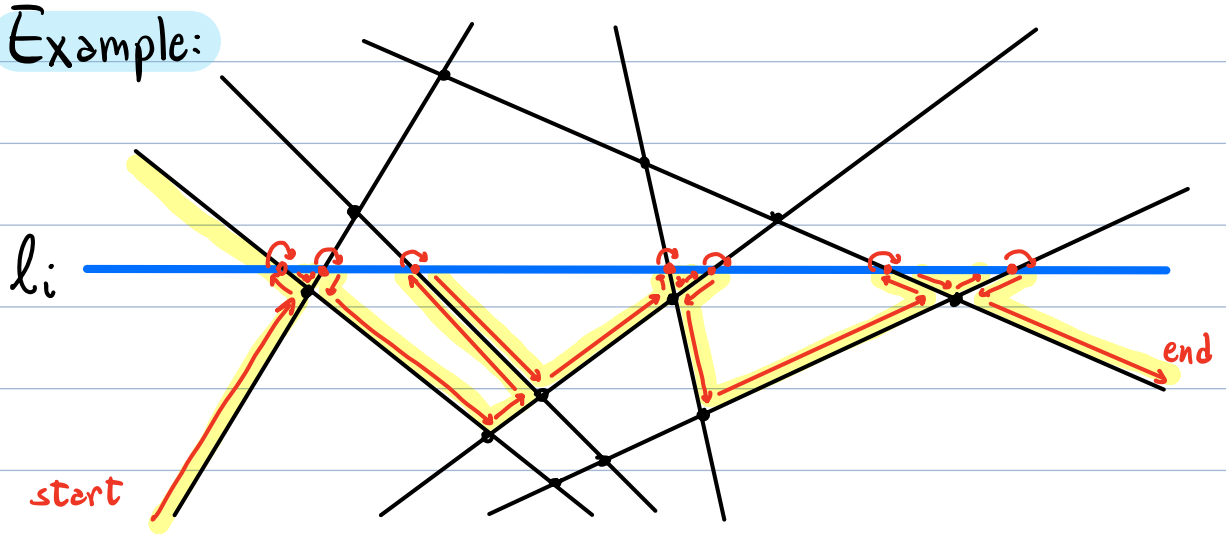
① Find the (unbounded) cell at $x = -\infty$ where ℓ_i starts. (Based on slope order)





② For each face of $A(L_{i-1})$ that intersects ℓ_i, walk along its lower boundary until finding the exit point

Example:



- Once we know the entry + exit points on each face, we can update the DCEL in $O(1)$ time per face $\Rightarrow$ $O(i)$ total.
- How long to crawl the faces?

Naïve Analysis (on adding ℓ_i)

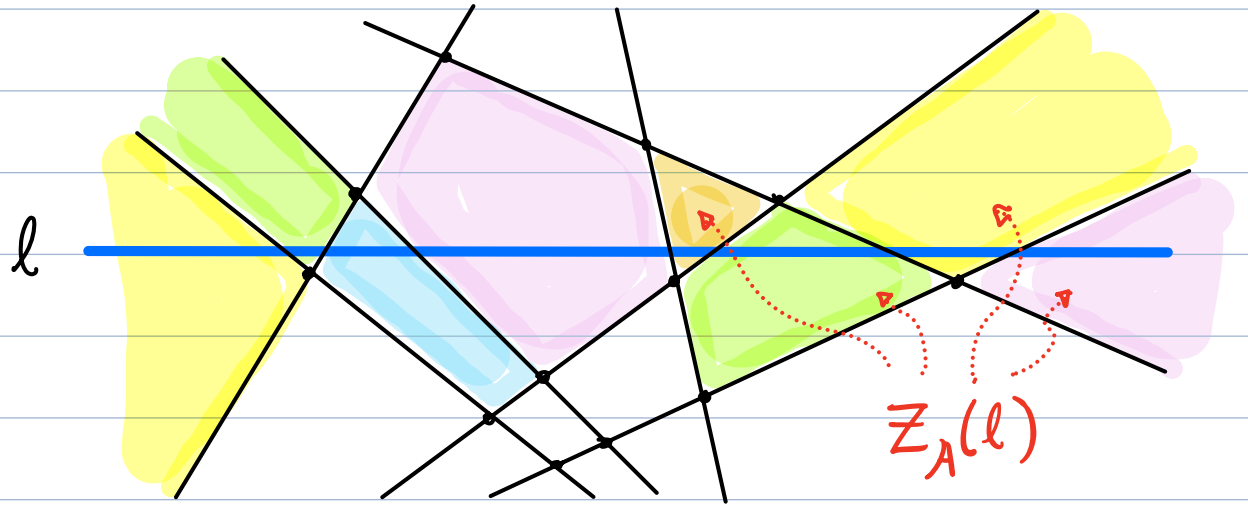
- ℓ_i crosses $i-1$ lines $\Rightarrow$ i faces
- Each face may have up to $i-1$ edges
- $O(1)$ time per edge $\Rightarrow$ $O(i(i-1)) = O(i^2)$ for ℓ_i

Total for all n lines:

$$\sum_{i=1}^{n} i^2 = O(n^3)$$

Can it be this bad?

Zone: Given an arrangement $A = A(L)$ + a line $\ell \notin L$, the **zone of ℓ in A**, $Z_A(\ell)$, is the **set of faces** of A that ℓ crosses.



Crawl time when inserting ℓ_i into $A(L_{i-1})$

$\leq$ no. edges on lower faces of $Z_{A(L_{i-1})}(\ell_i)$

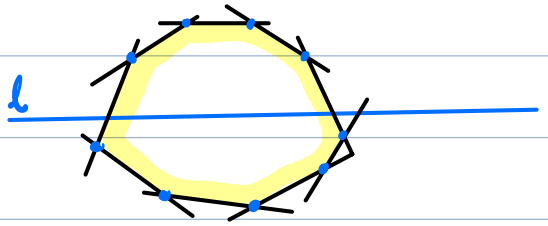
$\leq$ no. edges on $Z_{A(L_{i-1})}(\ell_i)$

We'll show this is $O(i)$, not $O(i^2)$

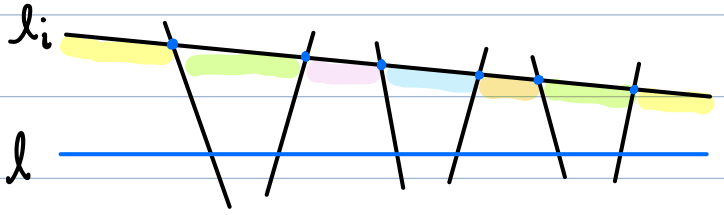
Theorem: (Zone Theorem) Given a line arrangement $A(L)$ of n lines, the number of edges in $Z_{A(L)}(l) \leq 6n$

How to prove this?

Cell by cell? But many cells have $\Omega(n)$ complexity



Line by line? But some lines contribute $\Omega(n)$ edges to zone



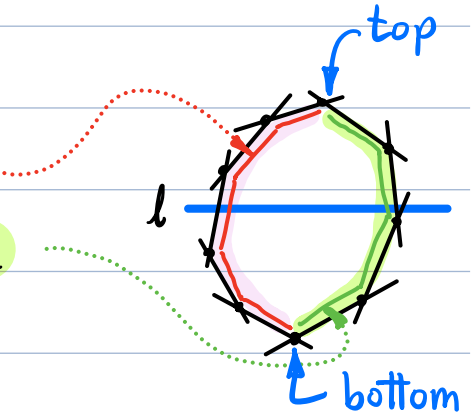
A smarter counting strategy:

- Partition edges into 2 types (left/right side)
- Show (by induction) that $\leq 3n$ of each

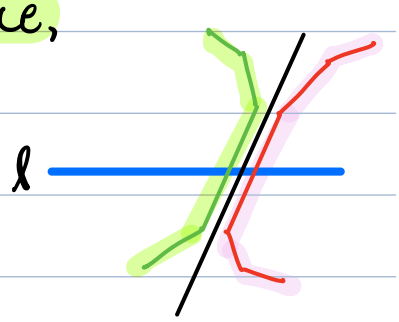
A zone edge is:

left bounding - on face's left side

right bounding - on face's right side



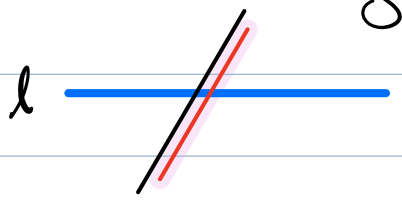
Note: Edges crossing ℓ are counted twice, once on left + once on right.



Lemma: In any zone of n lines, there are $\leq 3n$ left bounding edges.

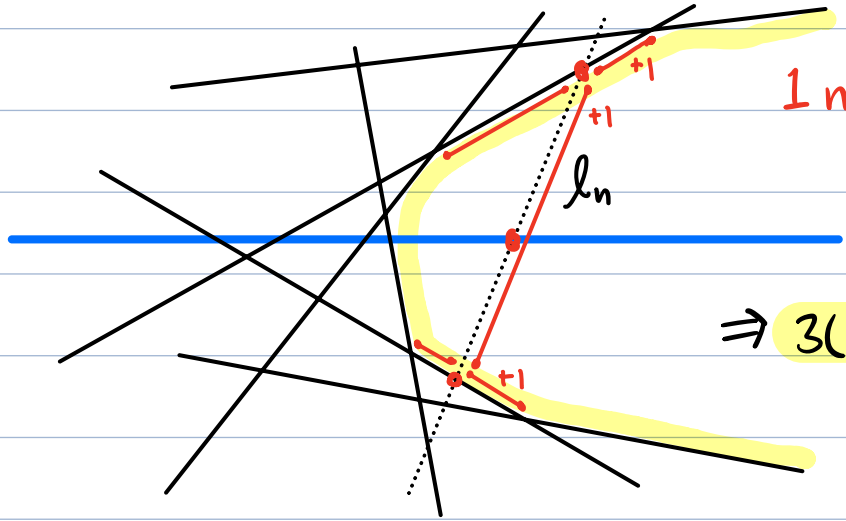
Proof: (by induction on n)

Base: $n = 1$ – Just one left bounding edge.



Step: $n \geq 2$

- Let ℓ_n be rightmost edge crossing ℓ
- Remove ℓ_n. By induction $\leq 3(n-1)$ left bounding edges in zone in $A(\ell_1, \ldots, \ell_{n-1})$
- Adding ℓ_n increases edge count by ≤ 3



1 new edge + 2 splits

$\Rightarrow 3(n-1)+3 = 3n$ total edges

□

Theorem: Given a set L of n lines in $\mathbb{R}^2$, the incremental algorithm builds $A(L)$ in $O(n^2)$ time

Proof: for $1 \le i \le n$, insert ℓ_i

- Existing arrangement has $i-1$ lines
- Zone of ℓ_i has complexity $\le 6(i-1)$
- Crawling + update takes $O(6(i-1)) = O(i)$
- Total time: $\sum_{i=1}^{n} i = O(n^2)$ □

Applications:

- Line arrangements have many applications.
- Most involve n points + duality → n lines
- Typical running time = $O(n^2)$

How to process an arrangement?

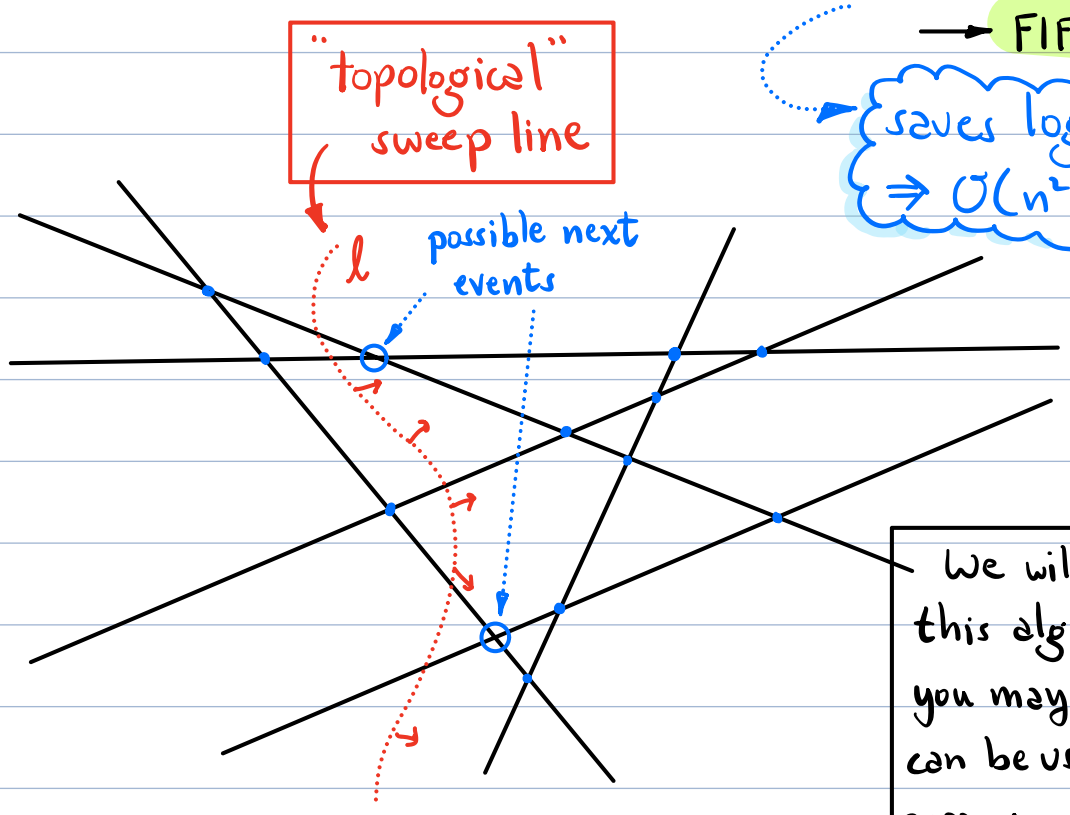
- Build it + traverse it like a graph
  ⇒ $O(n^2)$ time ☺ + $O(n^2)$ space ☹
- (Traditional) Plane Sweep
  ⇒ $O(n^2 \log n)$ time ☹ + $O(n)$ space ☺
- Topological Plane Sweep
  - A relaxed version of plane sweep
  - Only for line arrangements

  ⇒ $O(n^2)$ time ☺ + $O(n)$ space ☺

Topological Plane Sweep:

- A relaxed version of plane sweep for arrangements
- No insert, delete, find ⇒ no dictionary → array
- Sweep based on partial order ⇒ no priority queue → FIFO queue

saves log n factor ⇒ $O(n^2)$ time



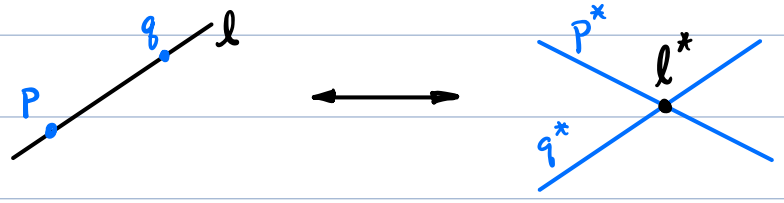
We will not cover this algorithm, but you may assume it can be used for any arrangement sweep.

Dual Transformation: (Recall)

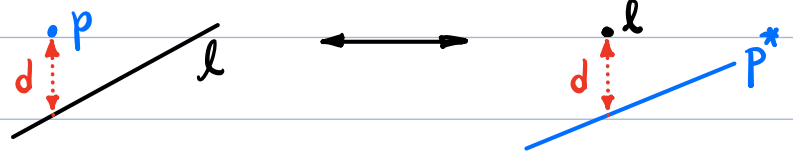
$p=(a,b)$ ⟷ $p^*: y=ax-b$

$\ell: y=cx-d$ ⟷ $\ell^*: (c,d)$

Incidence preserving

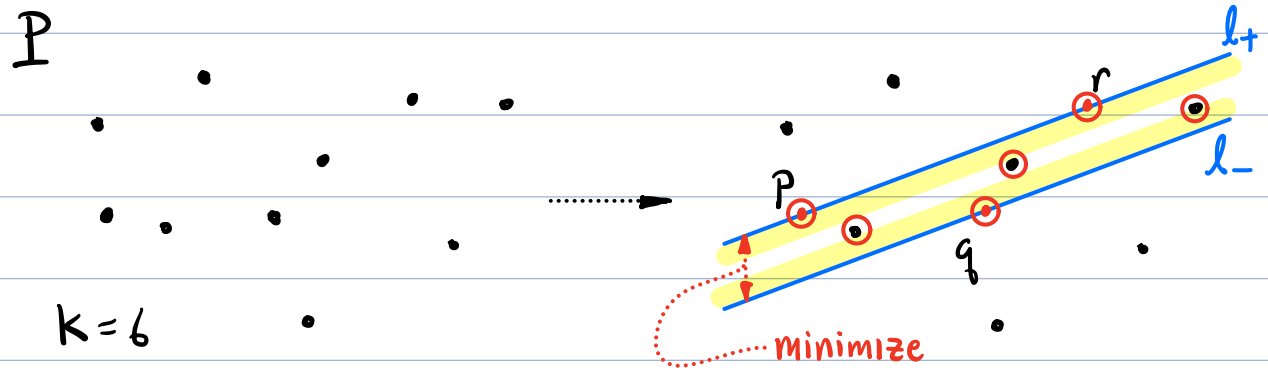


(Vertical) order reversing



Narrowest k-Corridor:

Given a set $P = \{p_1, \ldots, p_n\}$ in $\mathbb{R}^2$ + an integer k, $3 \le k \le n$, find the slab (region between 2 parallel lines) of minimum vertical width containing k pts.



Primal:

Let ℓ_+ + ℓ_- be upper + lower lines of the slab

$\ell_+ : y = ax - b_+$

$\ell_- : y = ax - b_-$

parallel ⇒ same slope

$b_+ \le b_-$

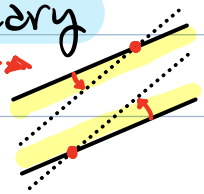
order reversal due to negation

- Vertical width: $b_- - b_+$
- k points of P in slab.
- Local optimality:

Claim: Assuming general position, 3 points of P lie on opt. slab boundary

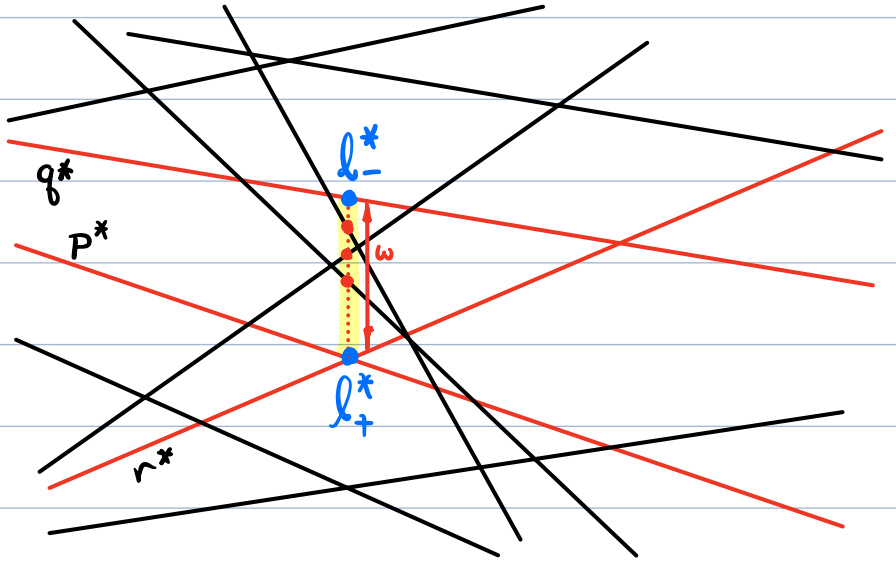
Proof: If 0, 1, or 2, can reduce width

4 cannot occur by gen'l position



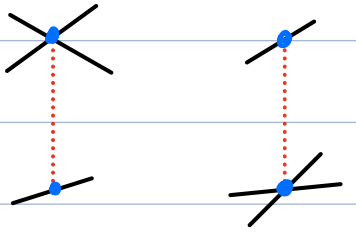
Dual:

- ℓ_+^* & ℓ_-^* are points (a, b_+), (a, b_-) — vertical
- vertical width = segment length: $b_- - b_+$
- k lines of P^* stab the segment $\overline{\ell_+^* \ell_-^*}$



Local optimality (Dual)

- 2 lines through one endpt and 1 through the other



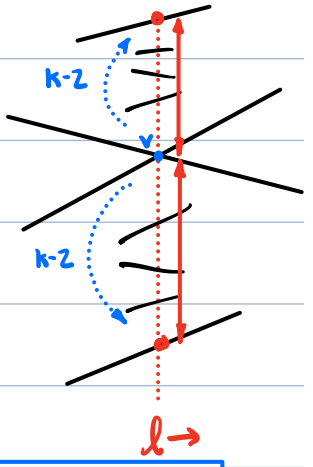
① Narrowest Corridor (P, k):

(1) $P^* \leftarrow$ dual lines of P

(2) Perform (topological) plane sweep

(3) On arriving at each vertex v, compute vertical distance to lines k-2 above & k-2 below

(4) Return smallest such distance



Store sweep line in an array

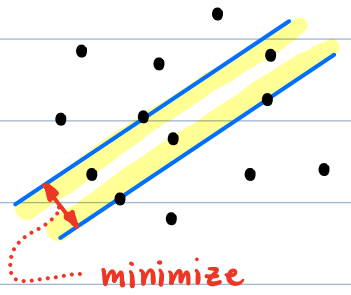
Correctness: Follows from duality + local opt. conditions

Time: $O(n^2)$
Space: $O(n)$
} topol. plane sweep

Aside: Easy to modify to compute perpendicular distance

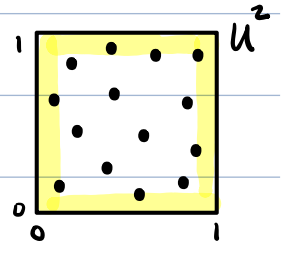
- If sweep line is $x = x_0$, then perp_dist = vert_dist / $\sqrt{1 + x_0^2}$

(Exercise in trigonometry)



② Halfplane Discrepancy

Given a set P of n points in the unit square $U^2 = [0,1] \times [0,1]$, how close is P to a uniform sample?

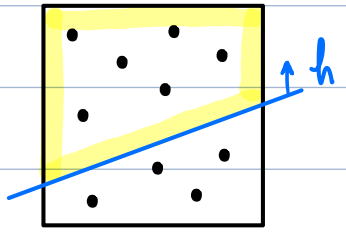


Idea: For any halfplane h, let

$\mu(h) = \text{area}(h \cap U^2)$ - true area

$\mu_P(h) = |h \cap P| / |P|$ - discrete area

Both quantities range from 0 to 1



$\mu(h) = 2/3 = 0.6666\ldots$

$\mu_P(h) = 6/10 = 0.6$

If P is nearly uniform, we expect
$|\mu(h) - \mu_P(h)|$ to be small, $\forall h$

For $0 < \Delta \leq 1$, we say P has halfplane discrepancy Δ if $|\mu(h) - \mu_P(h)| \leq \Delta$, $\forall h$

Questions:
- Given $P \subset U^2$, compute its max discrepancy, $\Delta(P)$
- Given n, what is the lowest possible discrepancy of any n-point set? ⟶ $\min_{P \subseteq U^2, |P|=n} \Delta(P)$
- Can we compute such a set given n?
- Why halfplanes? Discrepancy w.r.t. other shapes:
  - Triangles (simplices in $\mathbb{R}^d$)
  - Circles (Euclidean balls in $\mathbb{R}^d$)
  - Rectangles

Computing $\Delta(P)$ for $P \subset U^2$:

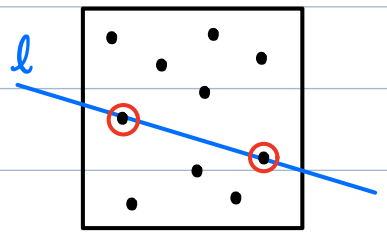
Outline:
- Identify $O(n^2)$ candidates for halfplanes (that includes the max discrepancy halfplane)
- Compute discrepancy for each - $O(1)$ each
- Return the max

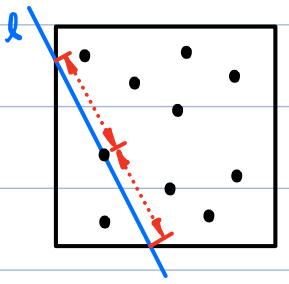
Candidates?

Lemma: Given $P \subset U^2$, let h be halfplane of max discrepancy + let ℓ be its bounding line. Either:

(i) ℓ passes through 2 points of P

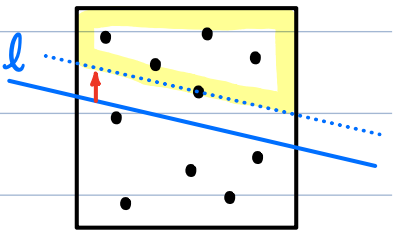
(ii) ℓ passes through 1 point at midpoint of $\ell \cap U^2$



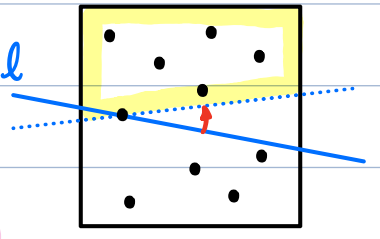


Proof: Take any line ℓ that fails both (i) + (ii)

- ℓ passes through no point of P
  - Translating ℓ up/down changes area $(h \cap U^2)$ without changing $|P \cap h|$.
  - Move ℓ to increase discrepancy



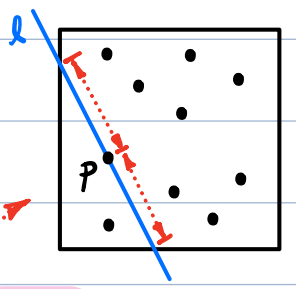
- ℓ passes through 1 point $p \in P$, not midpoint
  - Rotating ℓ about p increases/decreases area $(h \cap U^2)$ without changing $|P \cap h|$.
  - Rotate ℓ to increase discrepancy



□

Computing $\Delta(P)$:

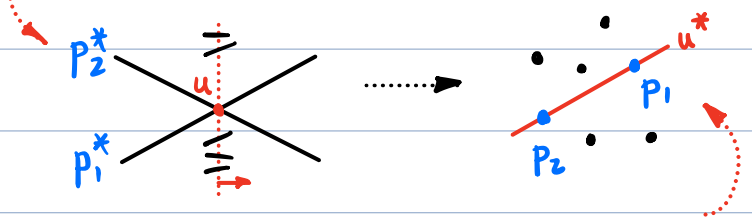
Type (ii): for each $p \in P$, compute the line ℓ through p s.t. p is midpoint of $\ell \cap U^2$.



- Count no. of points + areas on either side of ℓ } $O(n)$
- Save the side with larger discrepancy
- Time: $n \cdot O(n) = O(n^2)$

Type (i):

- $P^* \leftarrow$ n dual lines of P
- Sweep arrangement $A(P^*)$
- For each vertex u, maintain count of no. lines above/below



$\Rightarrow$ no. points below/above u^*

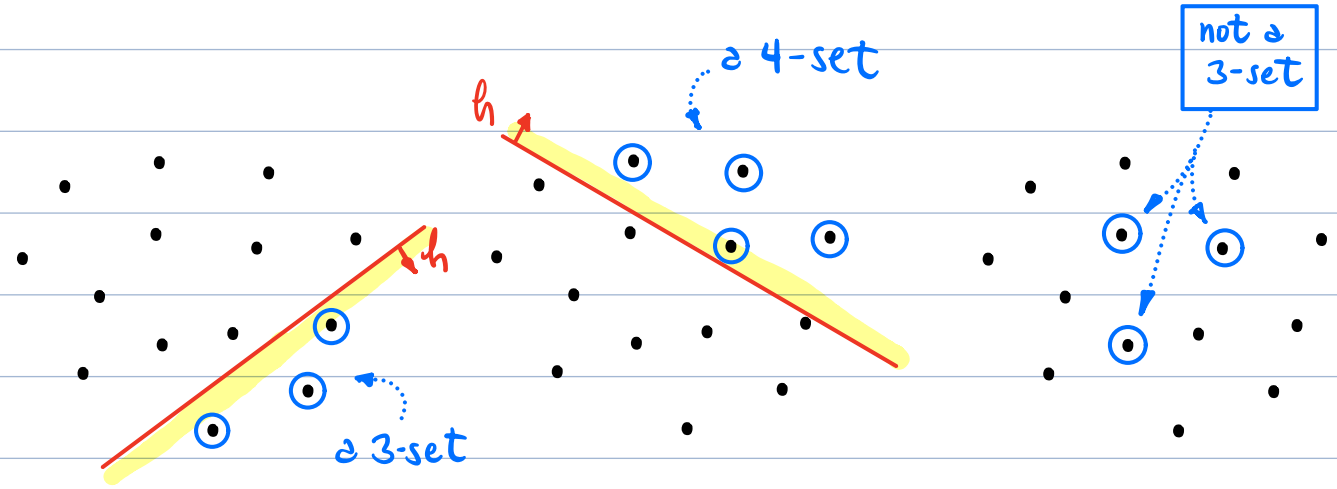
(can update counts in $O(1)$ time per event)

- Given u^* + counts, can compute discrepancy of u^* in $O(1)$ time

Total time: $O(n^2)$ for topol. plane sweep.

③ Computing k-Sets:

Given $P = \{p_1, \ldots, p_n\}$ in $\mathbb{R}^2$ + integer k $(1 \le k \le n-1)$ a **k-set** is a k-element subset of the form $P \cap h$, for some halfplane h



Problem: Given P + k, report/count all k-sets of P.

How many k-sets?

$\binom{n}{k} = O(n^k)$? - Naive

$O(n^2)$ - Better

$O(n \cdot k^{1/3})$ - Best upper bound } Big gap!

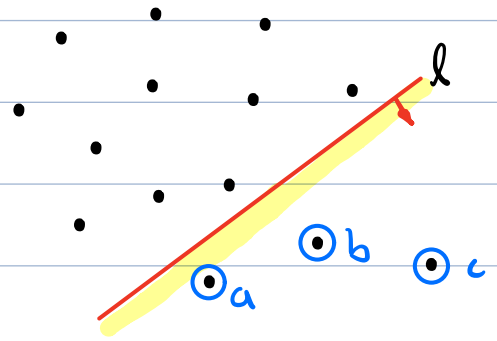
$\Omega(n \cdot c^{\sqrt{\log k}})$ - Best lower bound }

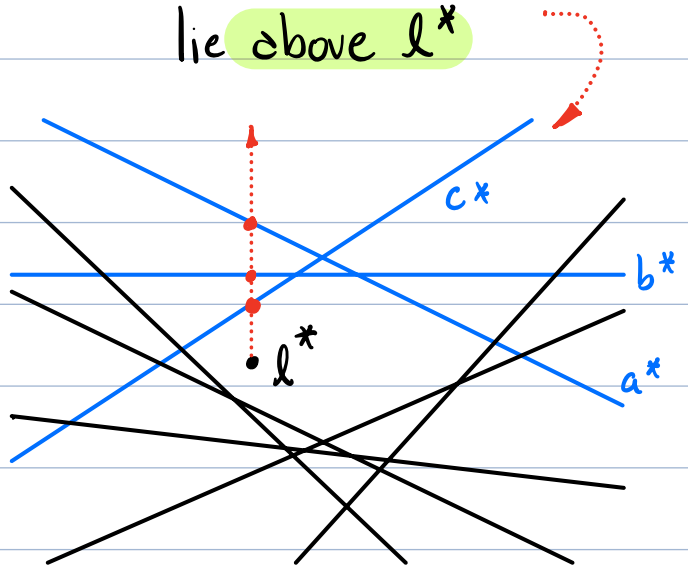
We will show that all k-sets can be computed in $O(n^2)$ time.

Dual Equivelence: Let's just consider lower halfplanes.

By order reversal:

k points of P lie below ℓ iff k lines of P^* lie above ℓ^*

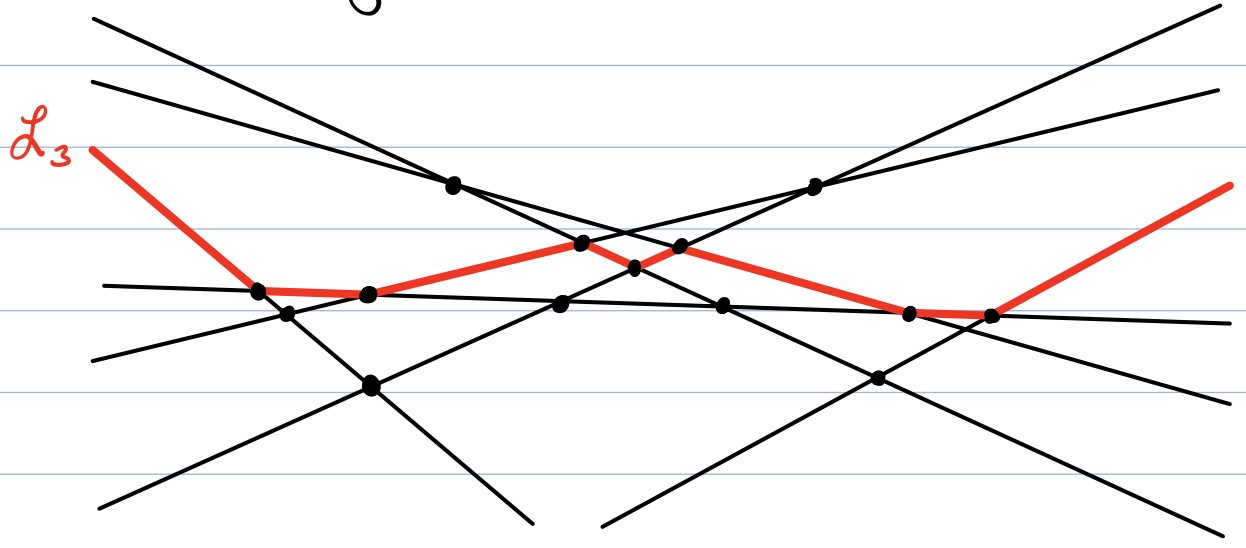




lower k-sets of P in primal $\equiv$ edges of $A(P^*)$ with k lines on or above

A useful concept is a level:

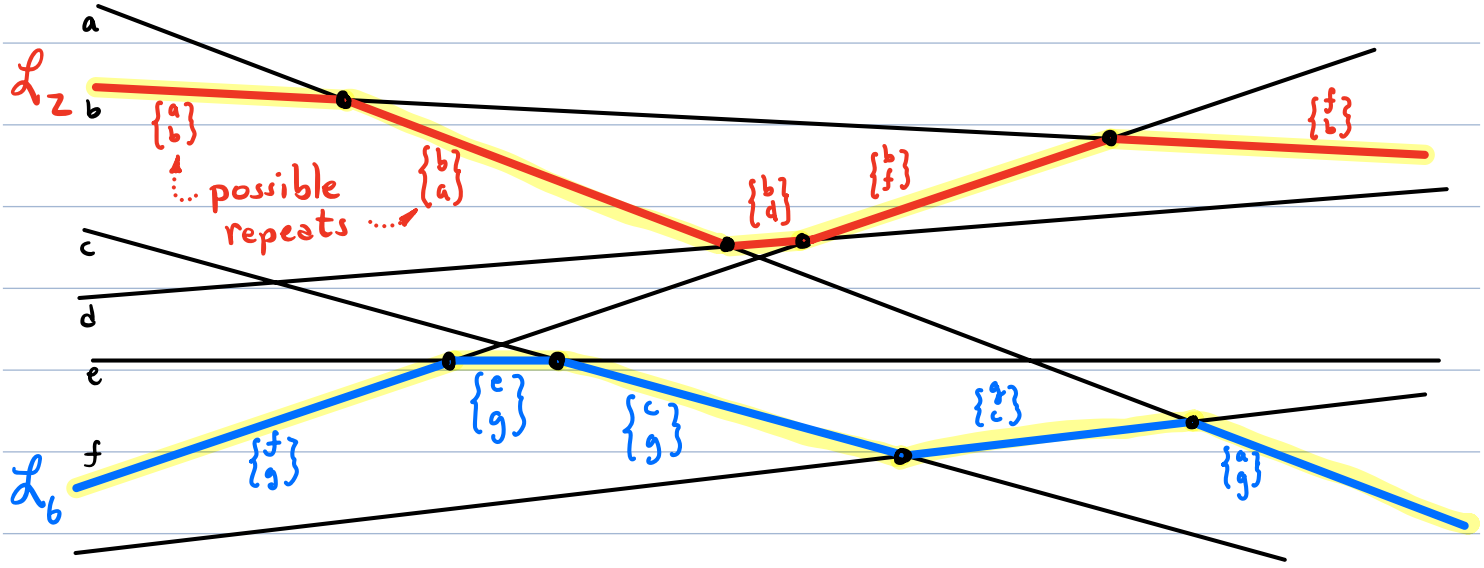
Def: Given an arrangement A of n lines and $1 \le k \le n$, level-k or $\mathcal{L}_k(A)$, is the set of edges of A with k lines on or above.



Claim: The set of k-sets of P are associated (with possible duplicates) with the edges of $\mathcal{L}_k + \mathcal{L}_{n-k+1}$ in $A(P^*)$.

lower — $\mathcal{L}_k$; upper — $\mathcal{L}_{n-k+1}$

Example: $k=2$, $n=7$



Enumerating all k-sets:

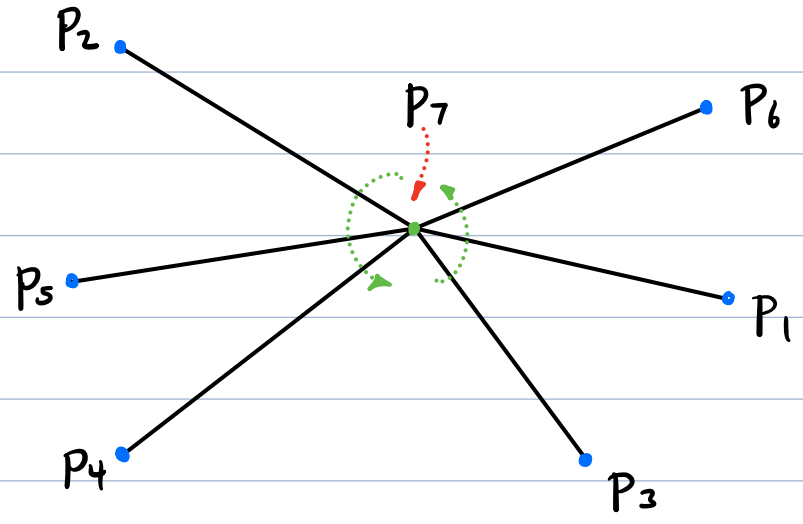
- P^* ← dual of P
- sweep $A(P^*)$ keeping track edges at levels k & $n-k+1$
- At each vertex of either level, swap one point in & one point out of current k-set.

Topol. plane sweep ⇒ $O(n^2)$ time, $O(n)$ space.

④ Sorting Angular Sequences:

Given a set $P = \{p_1, \ldots, p_n\}$ in $\mathbb{R}^2$, for each p_i, sort other $n-1$ points about p_i in angular order (Output: n sorted sequences)

$\Rightarrow O(n^2)$ size



T:

2
5
4
3
1
6

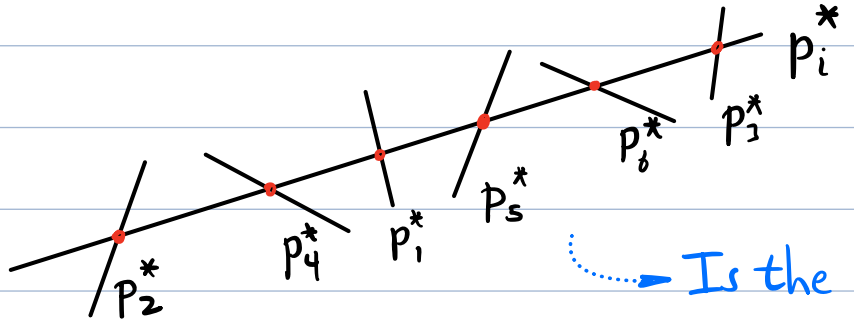
Naive: Perform n separate sorts of $n-1$ angles

$\Rightarrow O(n \cdot ((n-1)\log(n-1)) = O(n^2 \log n)$

We'll show how to do this $O(n^2)$ time using arrangements.

Smarter (?)

- Dualize $\longrightarrow$ p_i^* is a line in $A(P^*)$
- Sort intersections with other lines p_j^* $(j \neq i)$ — Is this the answer?



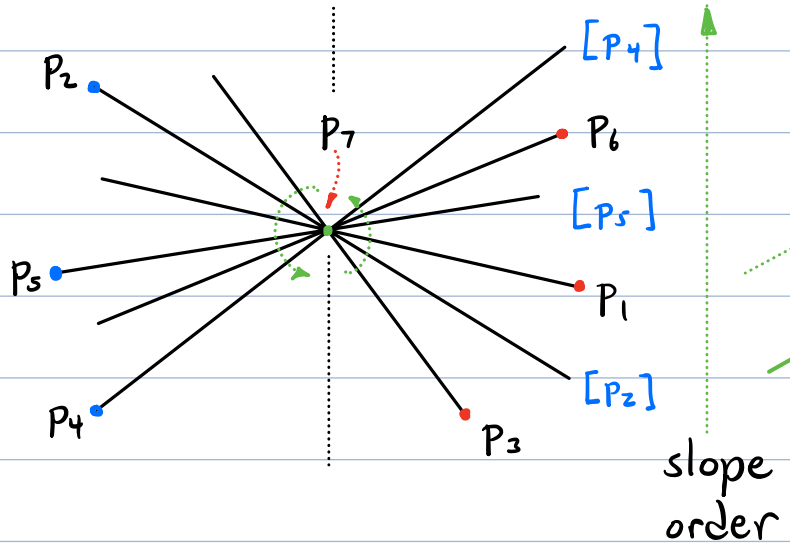
→ Is the answer $[2,4,1,5,6,3]$?

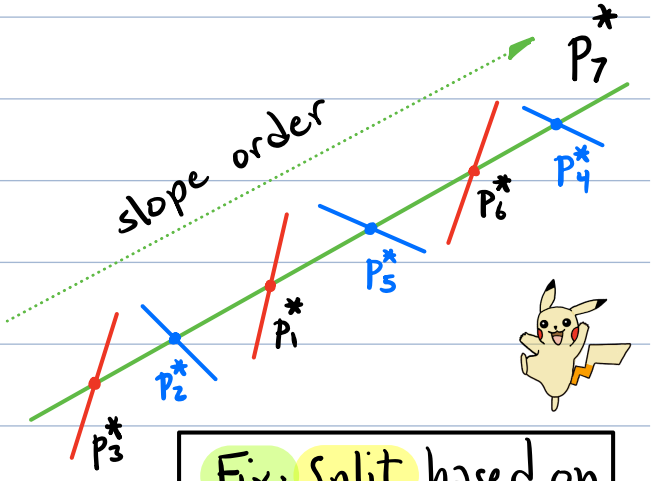
No!



But this not far off.

- The intersection points are sorted by x
  ⇒ Ordered by slope of line $\overline{p_i p_j}$
- But slope is not same as angle



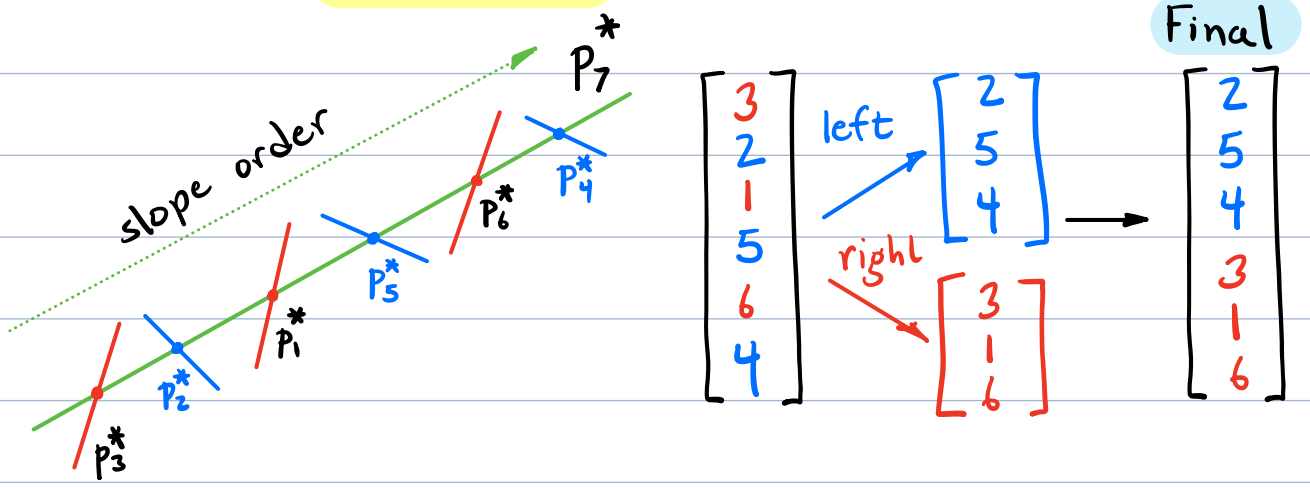


Fix: Split based on whether point is left/right of p_i + concatenate.

Sorting Angular Sequences:

- P^* ← duals of P
- sweep $A(P^*)$
- for each line p_i^*, maintain a list of intersections – sorted by increasing x

- After sweep finishes, for each p_i:
  - split its slope-sorted sequence into points to its left + points to right
  - concatenate these two lists.



Summary:

- Arrangements of lines in $\mathbb{R}^2$
- Complexity = $O(n^2)$
- Constructable in $O(n^2)$ time (incremental, deterministic)
- Zone Theorem: Total zone complexity is $O(n)$. [$O(n^{d-1})$ in $\mathbb{R}^d$]
- Topological plane sweep $O(n^2)$ time ; $O(n)$ space

- Applications:
  - Narrowest k-corridor
  - Halfplane discrepancy
  - k-sets + levels
  - Sorting angular sequences