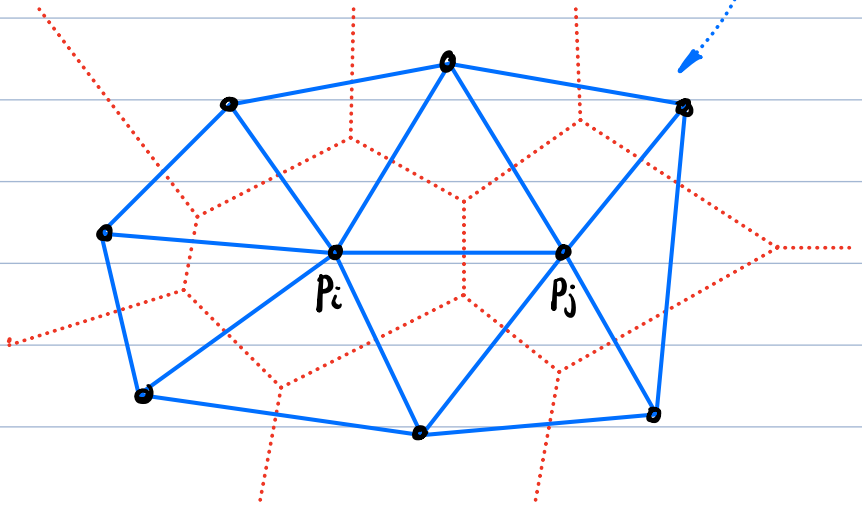
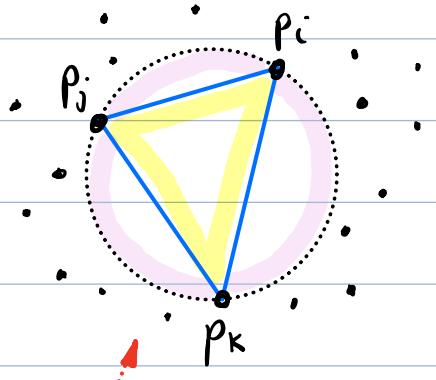
CMSC 754 - Computational Geometry
Lecture 12: Delaunay Triangulations II (Construction)

Recall from last lecture: Given a set of sites $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^2$, the **Delaunay Triangulation, DT(P)**, is the **dual** of the **Voronoi diagram**.





Empty Circumcircle Property:

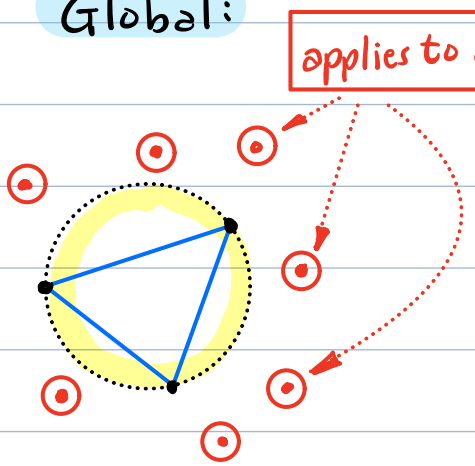
$\Delta p_i p_j p_k \in DT(P)$ iff this triangle's **circumcircle** contains no other site of P

Local vs. Global Delaunay:

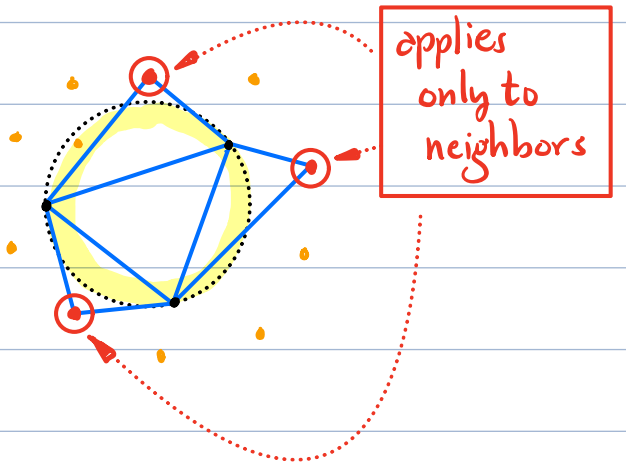
Globally Delaunay: Circumcircle property applies to **all sites of P**

Locally Delaunay: Circumcircle property applies only to **vertices of neighboring triangles**

Global:



Local:



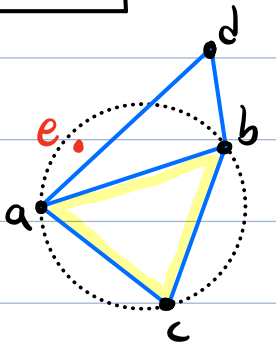
Why do we care?

- The local Delaunay test is more efficient. (Just test 3 sites, rather than n-3.)
- Suppose every triangle passes the local test. Could the global condition fail? → No!

Delaunay's Theorem: A triangulation is globally Delaney iff it is locally Delaunay.

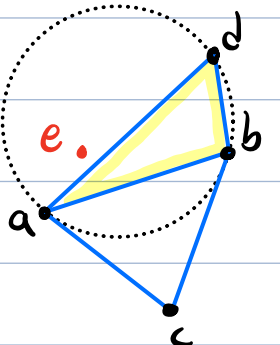
Proof: (See lecture notes.)

Local test doesn't detect e's violation



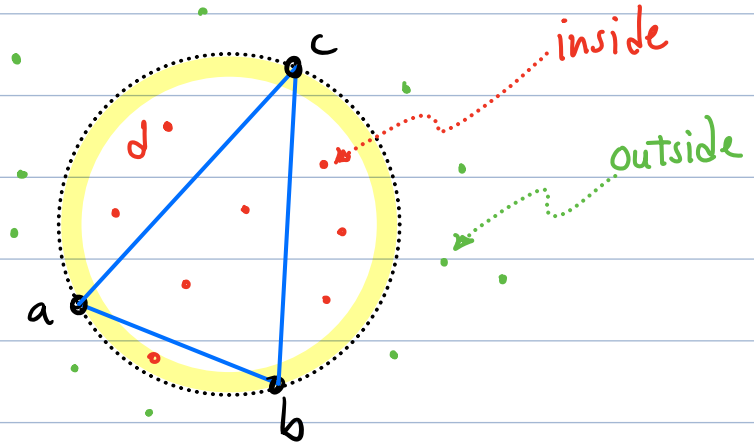
Intuition: Suppose triangulation is locally Delaunay but not globally.

But another triangle Δabd does.



Then some other triangle will witness the Delaunay violation.

Incircle Test: Given points $a, b, c, d \in \mathbb{R}^2$, does d lie in circumcircle of triangle Δabc (given in counterclockwise order)



inCircle(a,b,c,d):

d is inside iff

$$\det \begin{bmatrix} a_x & a_y & a_x^2+a_y^2 & 1 \\ b_x & b_y & b_x^2+b_y^2 & 1 \\ c_x & c_y & c_x^2+c_y^2 & 1 \\ d_x & d_y & d_x^2+d_y^2 & 1 \end{bmatrix} > 0$$

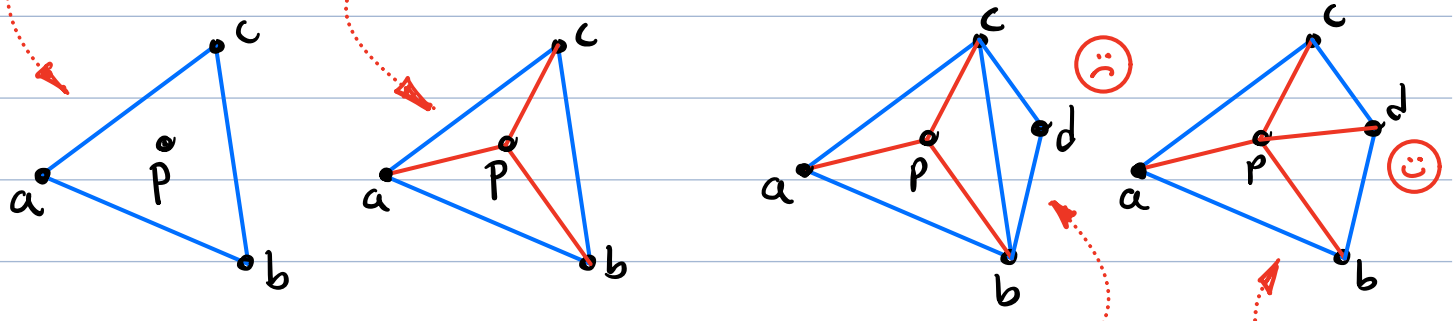
Observations:

- This is orientation test of points lifted to paraboloid in $\mathbb{R}^3$
- Generalizes to $d+2$ points in $\mathbb{R}^d$
- Computable (exactly) in $O(1)$ time in any fixed dimension

Randomized Incremental Construction (RIC):

- Insert sites in random order + update

1. Find the triangle containing the new site p → Δabc
2. Add edges connecting p to a, b, + c

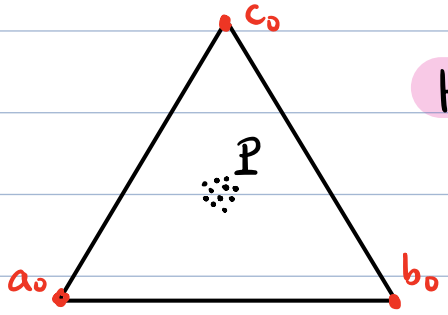


3. Check neighboring triangles for violation of local Delaunay property

4. Apply edge flips to correct these

5. Repeat until all neighbors are locally Delaunay

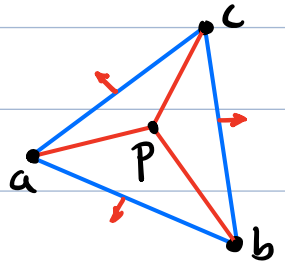
Sentinel Sites: To get the process started, we need a single triangle containing all the sites of P.



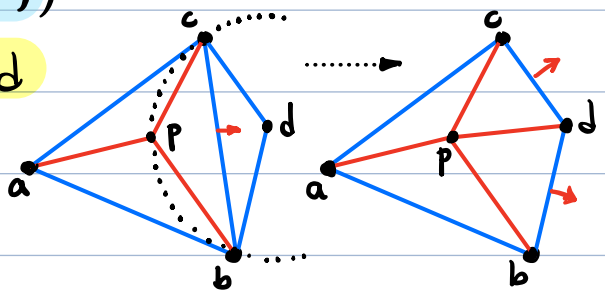
How big? No circumcircle from P should include a_0, b_0 or c_0 (see text)

```
BuildDelaunay(P={p1,...,pn})
    Create sentinel triangle Δa0b0c0
    Randomly permute P
    for i ← 1 to n  Insert(pi)
```

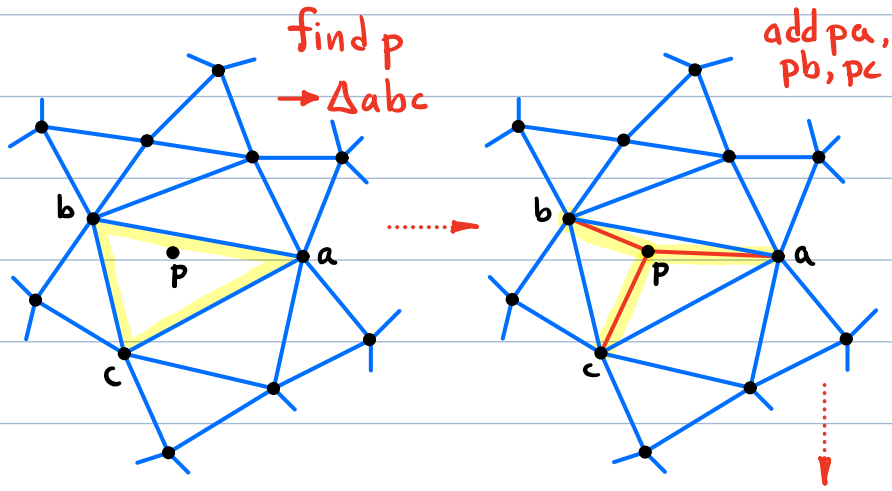
```
Insert(p)
    Find triangle containing p → Δabc
    Add edges pa, pb, pc
    SwapTest(ab)
    SwapTest(bc)
    SwapTest(ca)
```

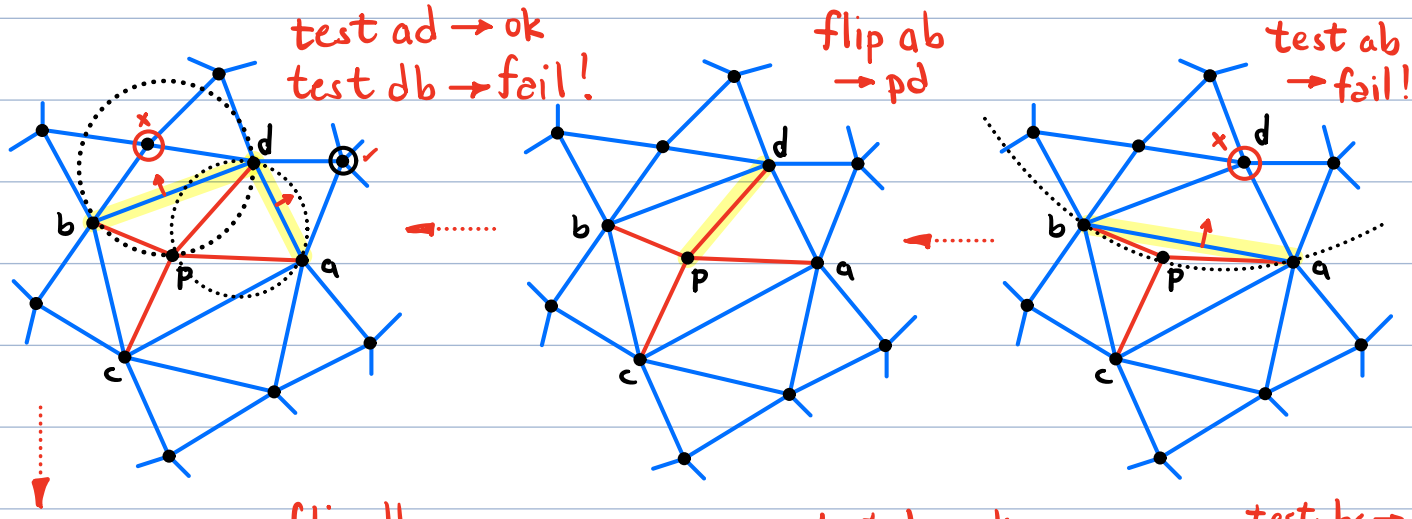


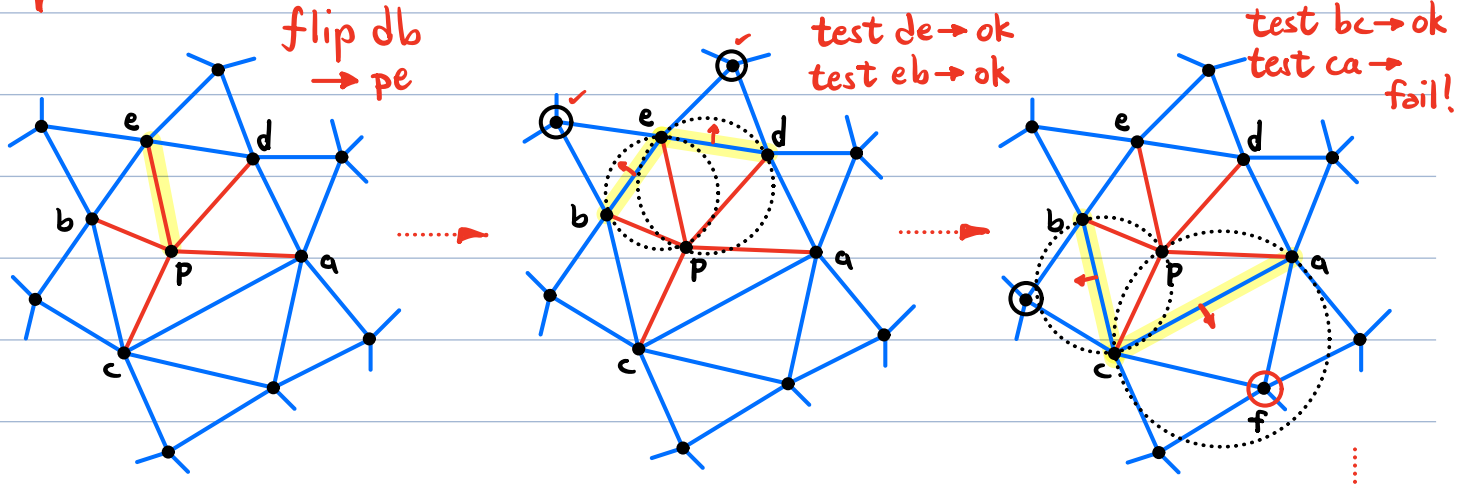

```
SwapTest(ab)
    if (ab is not a sentinel triangle edge)
        d ← vertex on opposite side of ab
        if (inCircle(p,a,b,d))
            flip edge ab → pd
            SwapTest(ad)
            SwapTest(db)
```



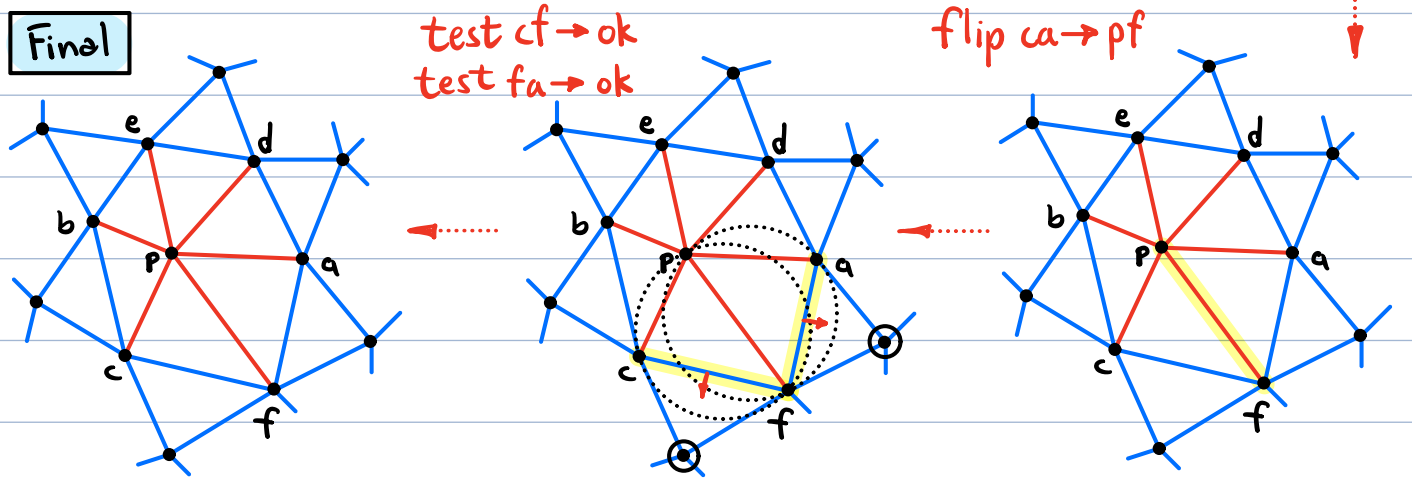
Example:







Final



Correctness:

- By induction – Assume triangulation is Delaunay prior to p's insertion
- The only triangles that might violate local Delaunay condition are incident to p – We check all
- By Delaunay's Thm, local ⇒ global

Running Time:

Time (assuming random)

- For each of n sites, p:
  - Locate triangle containing p — $O(\log n)$
  - Apply tests + update — $O(1)$

Total expected: $O(n \log n)$

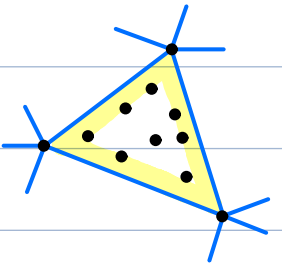
> **Lemma:** Expected time for inCircle tests + updates = $O(1)$

Proof: (by backwards analysis)

- Each update takes $O(1)$ time + adds an edge incident to p
  ⇒ update time ~ degree of p after insertion
- Since insert order is random, each vertex is equally likely last
- Total degree = $\sum_{p \in P} \deg(p) = 2(\text{no. edges}) \leq 2(3n) = 6n$
- Average degree = $(1/n) \cdot$ total-degree $\leq (1/n)\,6n = 6$
  ⇒ Expected update time = $O(1)$ □

Point Location:

- Could build history DAG (ala. trap. maps)
- Instead, we'll do this by bucketing
- Intuition:
  - Each triangle stores future sites that lie in this triangle
  - To locate a new site, just look up the triangle (bucket) containg it
  - When updates occur, rebucket sites





Lemma: Let p be any site. The probability that p is rebucketed during the i^{th} insertion is $\leq 3/i$.

Proof: (by backwards analysis)

- A site is rebucketed only if it lies in a new triangle after the insertion.
- All new triangles are incident to the last site inserted.
- Every existing site is equally likely to be last.
- Let Δabc be the triangle containg p

  $\Rightarrow$ p is rebucketed only if a, b, or c was last

$\Rightarrow$ Prob(p is rebucketed after i^{th} insertion)

$\leq$ Prob(a, b, or c was last among i)

$\leq 3/i$ □

Lemma: Total number of rebucketing operations over all sites is $O(n \log n)$ in expectation.

Proof: Expected number of rebucketings

$$= \sum_{p \in P} \sum_{i=1}^{n} 1 \cdot \text{Prob}(p \text{ was rebucketed after } i^{th} \text{ insert})$$

$$\leq \sum_{p \in P} \sum_{i=1}^{n} 3/i$$

$$= 3 \cdot \sum_{p \in P} \sum_{i=1}^{n} 1/i \approx 3 \cdot \sum_{p \in P} \ln n \quad \text{(Harmonic series)}$$

$$= 3 \cdot n \ln n = O(n \log n)$$ □

* Formally, this does not imply that run time is $O(n \log n)$

Need to show expected rebucket time per insert is $O(1)$

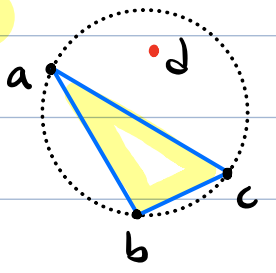
Summary:

- Local vs. global Delaunay

  Delaunay's Thm: local $\Rightarrow$ global

- Incircle test

  - Orientation test in $\mathbb{R}^2$



- Randomized Incremental Construction (RIC) for DT
  - Update time - $O(1)$ per pt. expected
  - Point location - $O(\log n)$ per pt. "