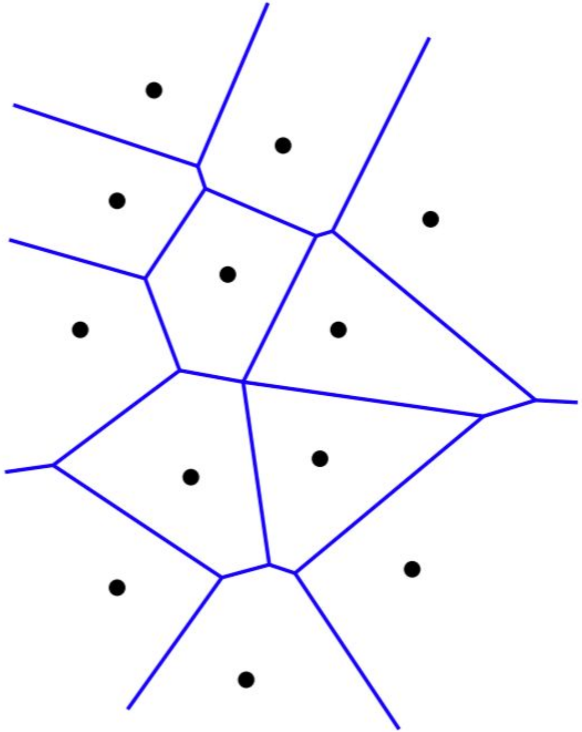
Delaunay Triangulation: Incremental Construction

Presented by **Pooja Bhagat**

IIT Guwahati

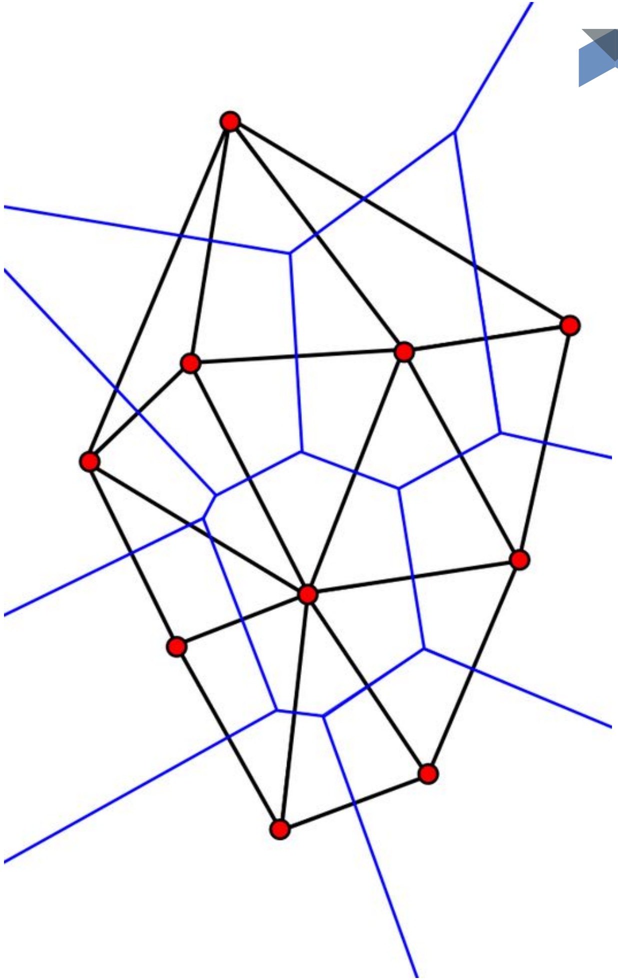
Voronoi Diagrams

- Subdivision of the plane where the faces correspond to the regions where one site is closest
- Every Voronoi vertex is the center of an empty circle through 3 or more sites
- Every point on a Voronoi edge is the center of an empty circle through 2 sites



Delaunay Triangulation

- Delaunay Triangulation is the graph dual of Voronoi Diagram(VoD).
- For each face of the primal graph (VoD), we create a vertex, and then we add an edge between two such vertices if their faces are adjacent in VoD.
- **Uniqueness:** The Delaunay triangulation is unique given no four sites are co-circular.

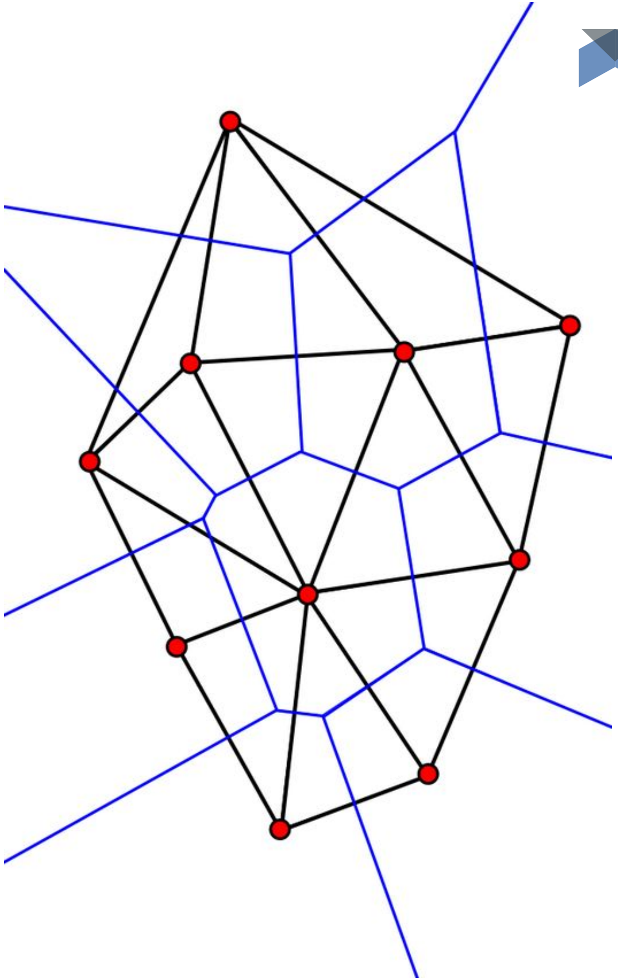


Property

If P has n points, of which k lie on the convex hull of P. Then, Delaunay triangulation of P (in fact, every triangulation) has $(2n - 2 - k)$ triangles and $(3n - 3 - k)$ edges.

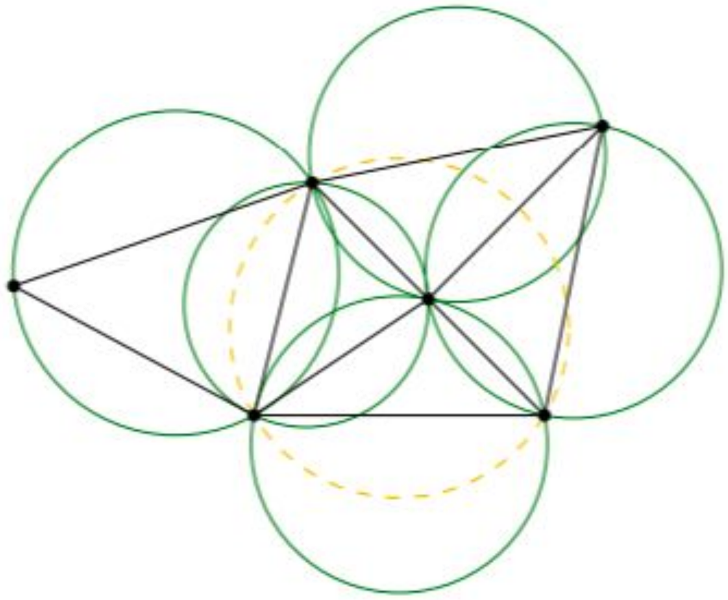
Proof by induction.

The k CH vertices create $k - 2$ triangles. Each of the remaining $(n - k)$ points destroys 1 and adds 3 new triangles, giving 2 additional triangles. The total is $(k - 2) + 2(n - k) = (2n - 2 - k)$.



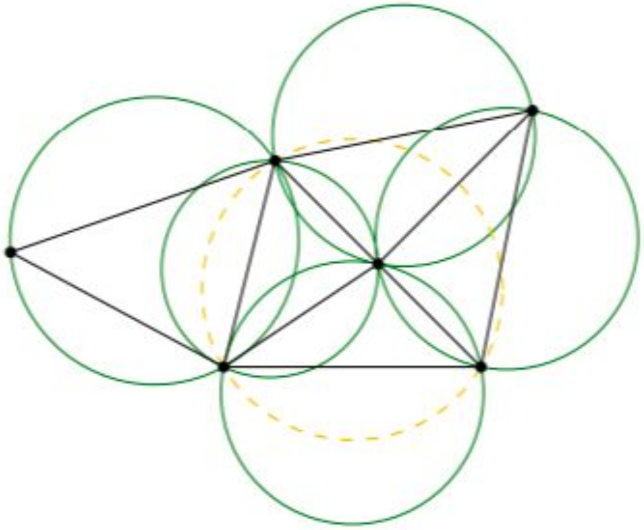
Circumcircle Criteria

- A triangulation of $N>2$ points is Delaunay if and only if the circumcircle of every interior triangle is point free. (Used for **inCircle()** test)



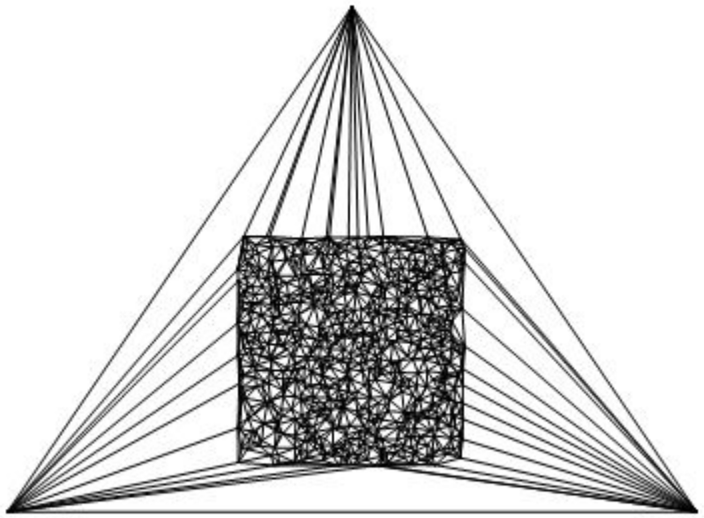
Edge Criteria

- The edge is in Delaunay Triangulation if and only if there exists an empty circle passing through its endpoints. An edge satisfying this property is said to be Delaunay.



The Algorithm : Base Case

- Start with a Delaunay Triangulation whose convex Hull contains all the given set of points.
- Add a **super triangle** large enough to completely contain all the points.
- Super triangle should not affect the global delaunay triangulation of inner points.



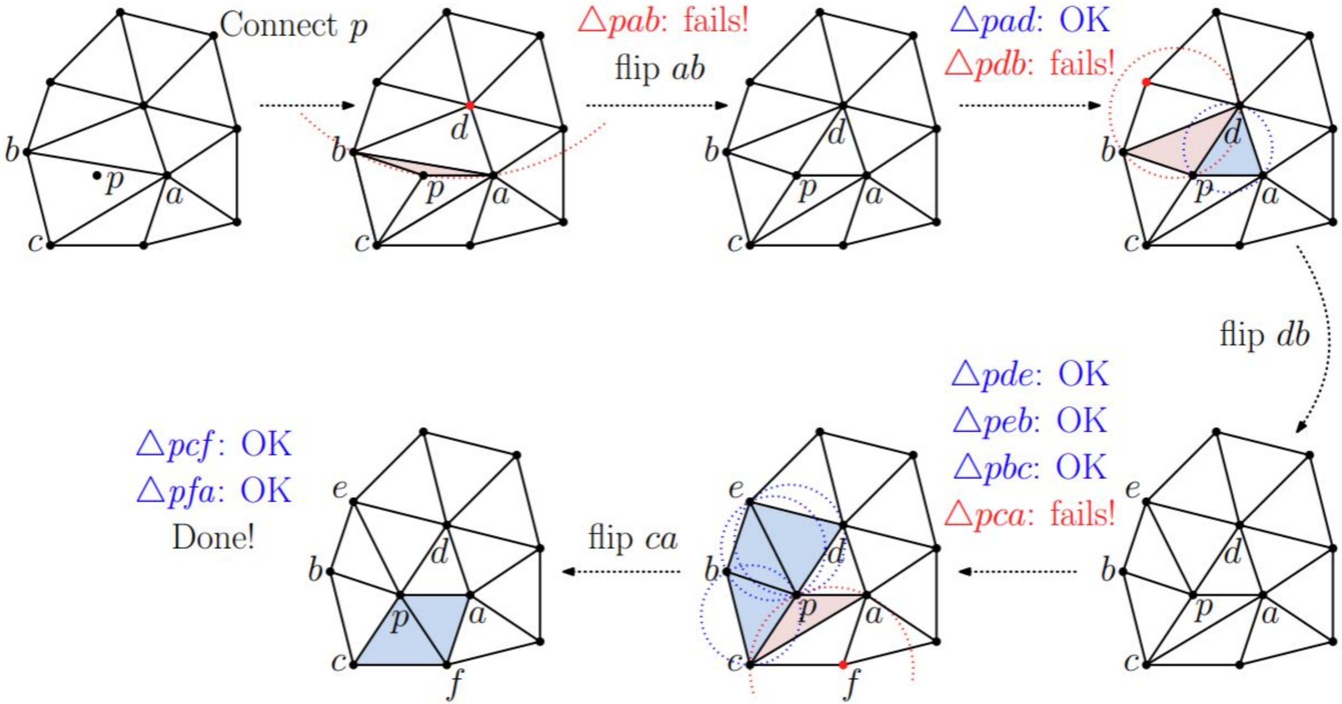
The Algorithm : Adding New Point

Randomized Incremental Delaunay Triangulation Algorithm

Insert(p) {
    Find the triangle $\triangle abc$ containing p
    Insert edges pa, pb, and pc into triangulation
    SwapTest(ab) // check/fix the surrounding edges
    SwapTest(bc)
    SwapTest(ca)
}

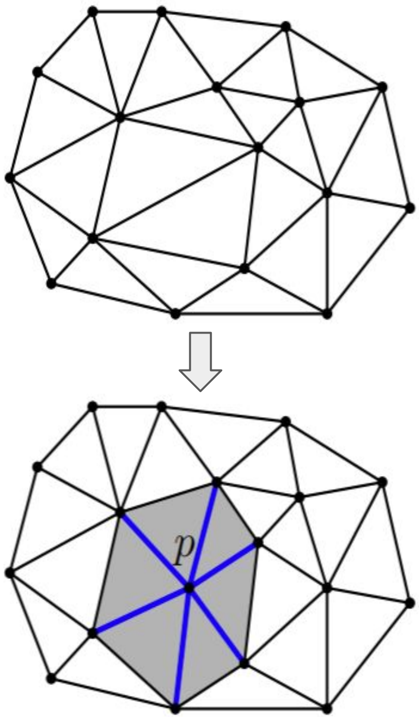
SwapTest(ab) {
    if (ab is an edge on the exterior face) return
    Let d be the vertex to the right of edge ab
    if (inCircle(b, p, a, d)) { // d violates the incircle test
        Flip edge ab // replace ab with pd
        SwaptTest(ad) // check/fix the new suspect edges
        SwaptTest(db)
    }
}

The Algorithm : Adding New Point



Runtime Analysis: Structural Changes

- Whenever an edge swap is performed, a new edge is added to p
- The total number of changes made in the triangulation for the insertion of p is proportional to the degree of p after the insertion is complete

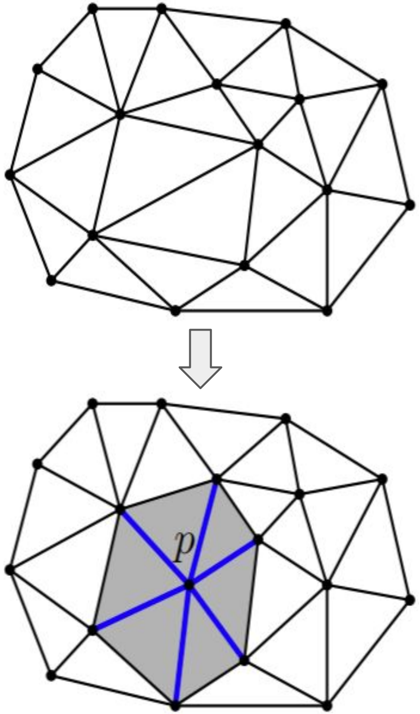


Runtime Analysis: Structural Changes

- If p_i is the i^{th} point to be inserted in the triangulation and d_i denotes the degree of the newly inserted site just after the i^{th} insertion

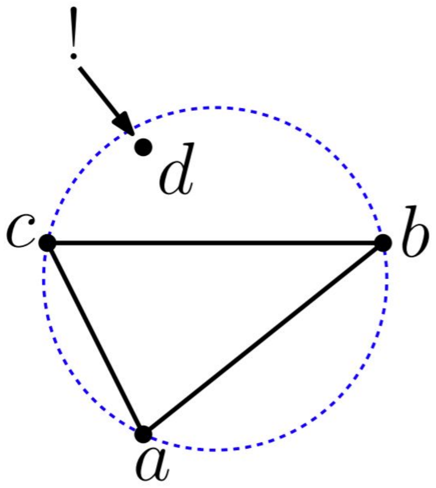
$$E[d_i] \;=\; \frac{1}{i}\sum_{j=1}^{i}\deg(p_i) \;\leq\; \frac{6i}{i} \;=\; 6$$

- Thus, $E[d_i] = O(1)$

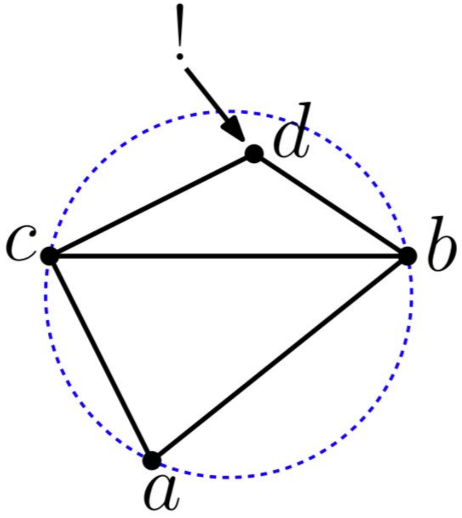


Correctness

Global Delaunay: The circumcircle of each triangle $4abc$ contains no other site d.



Local Delaunay: For each pair of neighboring triangles $4abc$ and $4acd$, d lies outside the circumcircle of $4abc$.

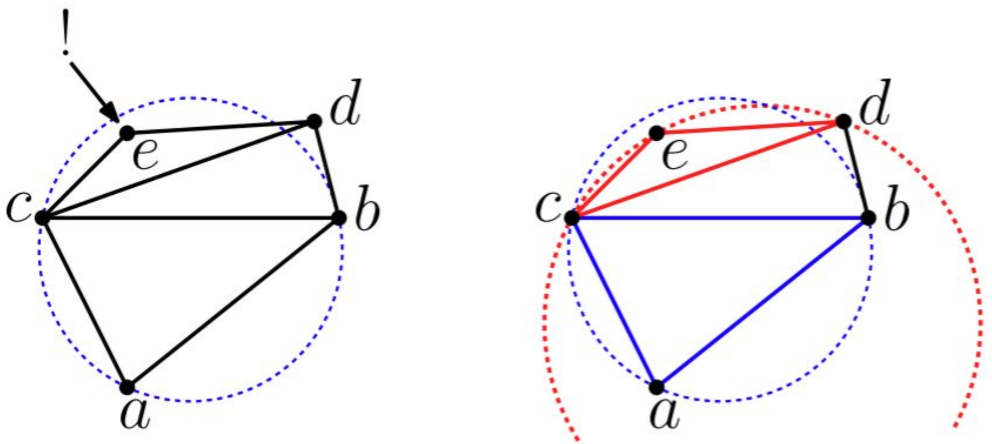


Correctness

Globally Delaunay ⇔ Locally Delaunay

Proof by contradiction:

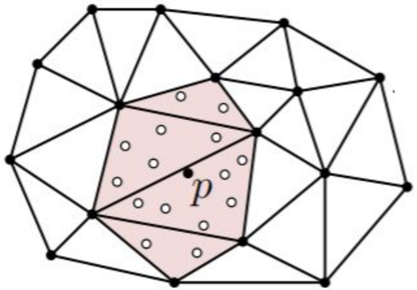
Assume the triangulation
to be locally delaunay
but not globally delaunay

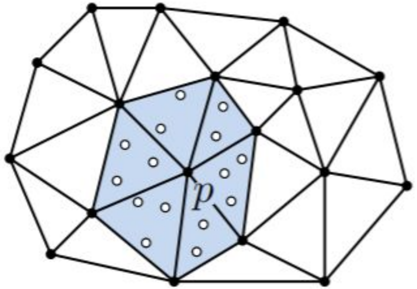


Finding Triangle containing p

Bucketing Approach

- Each triangle of the current triangulation is considered as a bucket that holds the sites that lie within this triangle and have yet to be inserted
- On adding a new site p, some new triangles are created and some destroyed
- Each point in triangles destroyed need to be rebucketed.



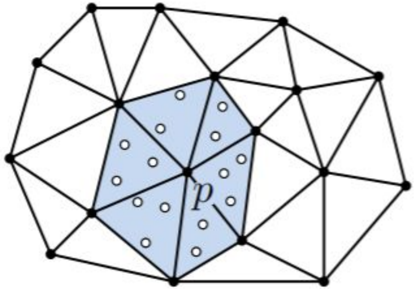


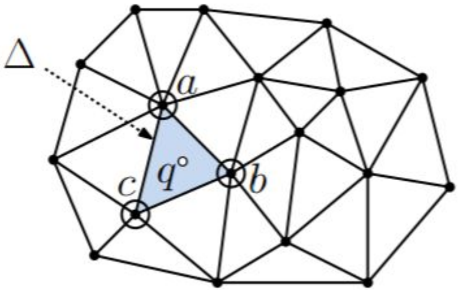
Runtime Analysis: Rebucketing

- Cost of rebucketing one point on inserting $p \propto deg(p) = O(1)$
- Letting B(q) denote the average number of times that q is rebucketed throughout the algorithm, we have

$$B(q) \leq \sum_{i=1}^{n} \frac{3}{i} = 3 \sum_{i=1}^{n} \frac{1}{i}$$

$$B(q) \leq 3 \cdot \ln n = O(\log n)$$





Runtime Analysis

1. On addition of a new point $O(1)$ structural changes are made to the triangulation (in expectation)
2. $O(\log n)$ time is spent determining which triangle contains each newly inserted site (in expectation).

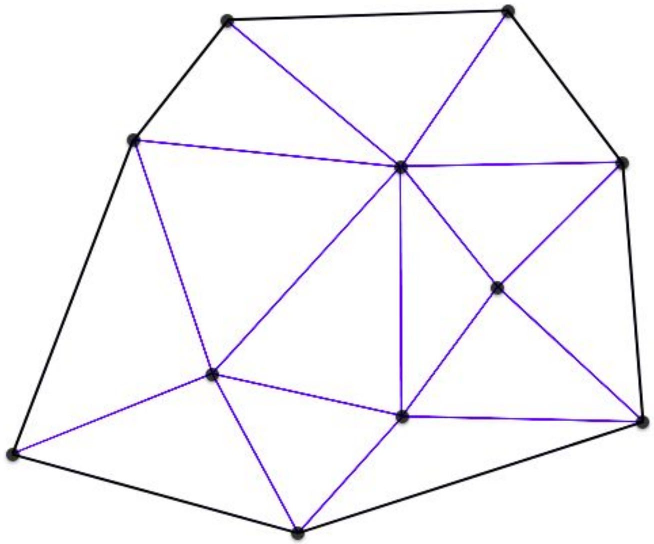
Overall **Expected Running Time**

$$= n * O(1) + n * O(\log n)$$

$$= O(n \log n)$$

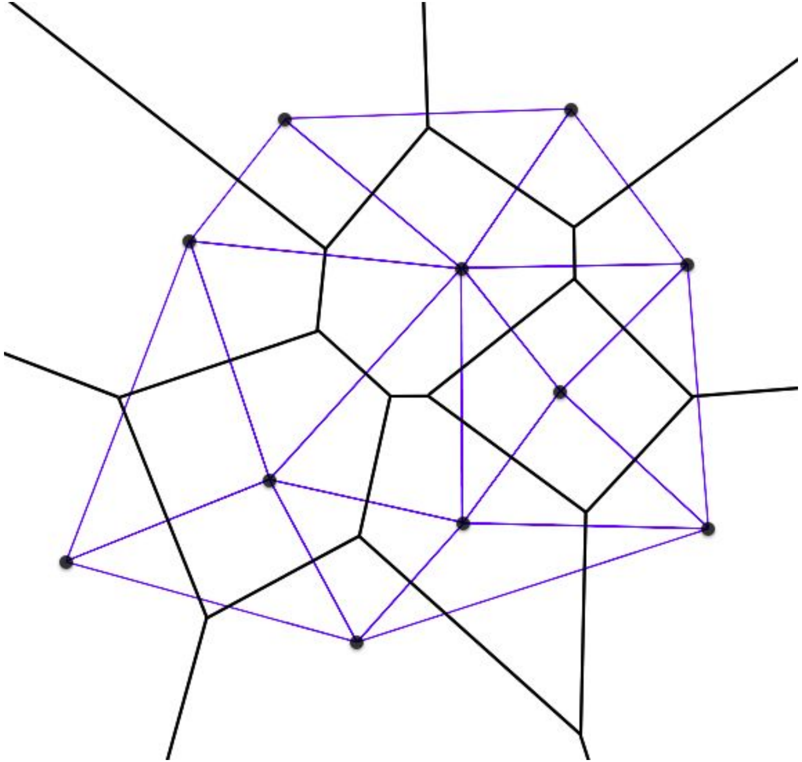
Convex Hull

Every edge that lies on the boundary of the convex hull of a vertex set and has no vertex in its interior is Delaunay.



Voronoi Diagram

- Vertices in Voronoi Diagram are the circumcenters of the triangles in Delaunay Triangulation.
- If we join the circumcenters of neighbouring triangles in Delaunay Triangulation, we get the Voronoi Diagram.



Demo

https://poojab01.github.io/RandomisedDelaunay/