TRIANGULATING POINT SETS

Vera Sacristán
Rodrigo Silveira

Universitat Politècnica de Catalunya

TRIANGULATING POINT SETS

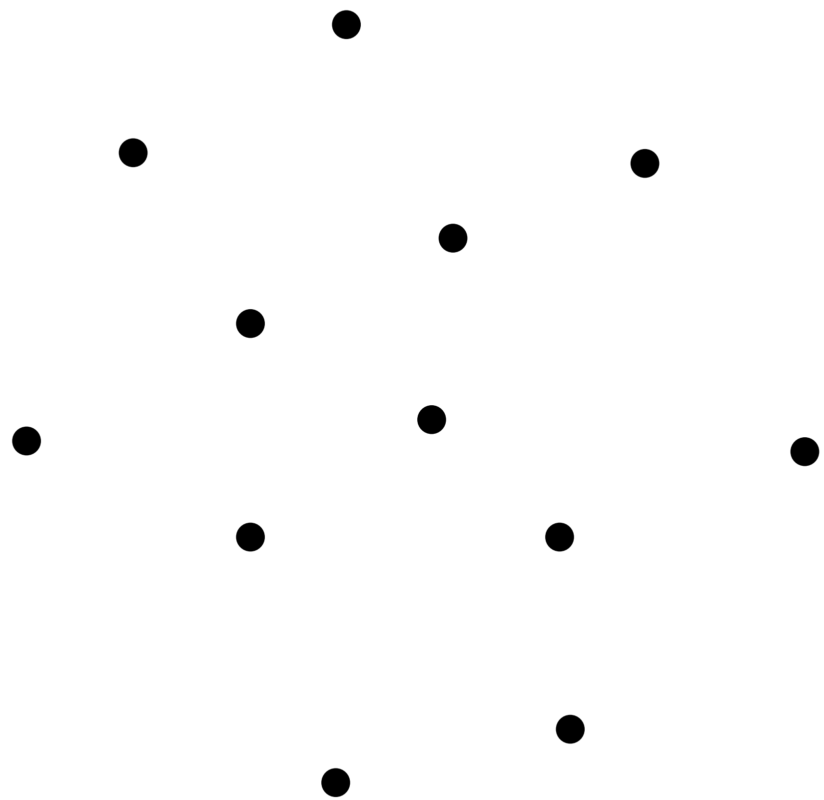
DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

TRIANGULATING POINT SETS

DEFINITION

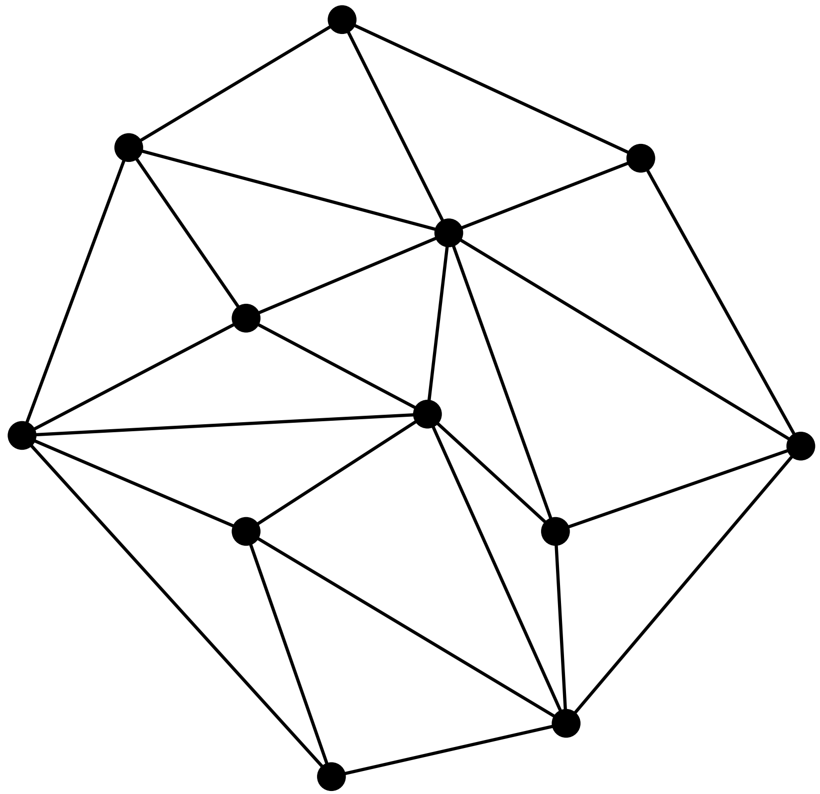
A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.



TRIANGULATING POINT SETS

DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

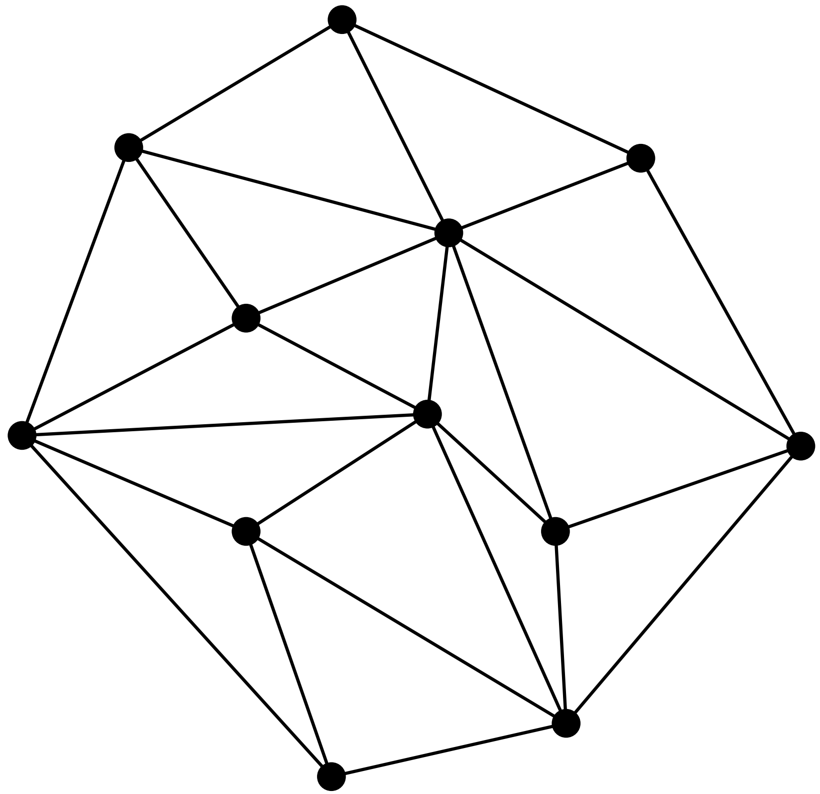


TRIANGULATING POINT SETS

DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

Corollary. All the faces of such a graph are triangles, except for the unbounded one, which is the exterior of the convex hull of P.



TRIANGULATING POINT SETS

DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

Corollary. All the faces of such a graph are triangles, except for the unbounded one, which is the exterior of the convex hull of P.

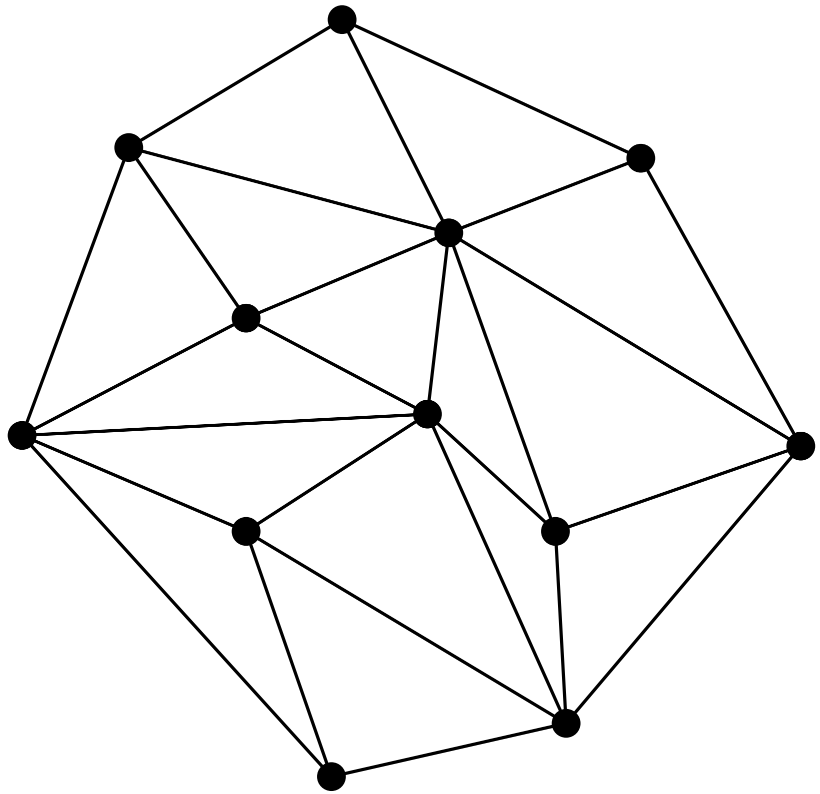
COMPLEXITY

Every triangulation of any set P of n points has:

$2n - h - 2$ triangles

$3n - h - 3$ edges

where h is the number of vertices of $ch(P)$.

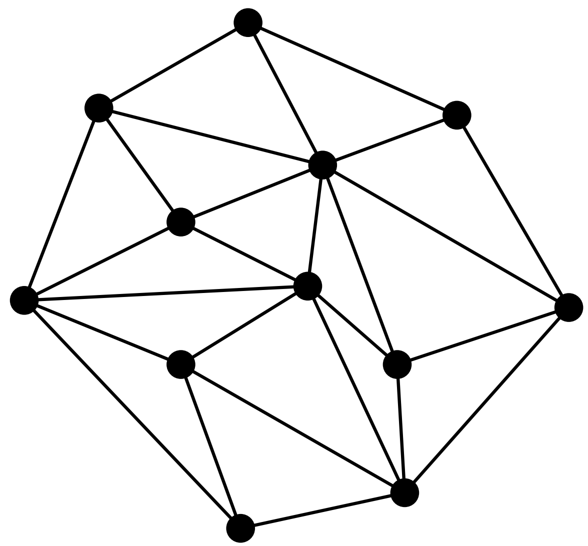


TRIANGULATING POINT SETS

DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

Corollary. All the faces of such a graph are triangles, except for the unbounded one, which is the exterior of the convex hull of P.



COMPLEXITY

Every triangulation of any set P of n points has:

$2n - h - 2$ triangles

$3n - h - 3$ edges

where h is the number of vertices of $ch(P)$.

Proof. Each triangle has exactly 3 edges. Each internal edge belongs to exactly 2 triangles. Each external edge belongs to exactly 1 triangle. Therefore, $3t = 2(e - h) + h = 2e - h$.
According to Euler's formula: $n + (t + 1) = v + f = e + 2$.
Combining both equations:

$$e = n + t - 1 \Rightarrow 3e = 3n + 3t - 3 = 3n + 2e - h - 3 \Rightarrow e = 3n - h - 3$$

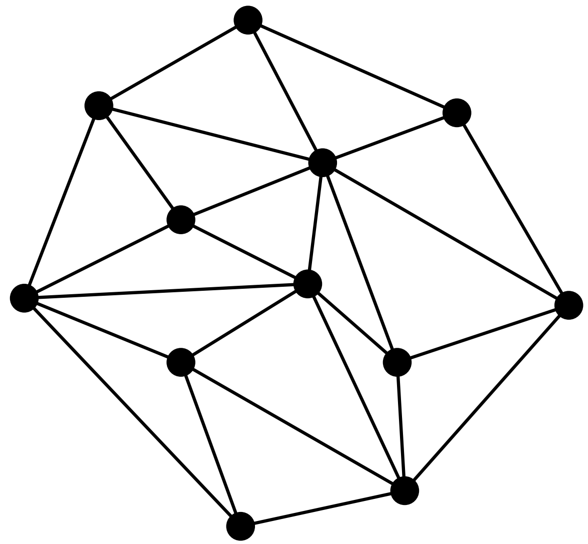
$$3t = 2e - h = 6n - 2h - 6 - h = 6n - 3h - 6 \Rightarrow t = 2n - h - 2$$

TRIANGULATING POINT SETS

DEFINITION

A triangulation of A set P of n points in the plane is a graph having P as set of vertices which is rectilinear, planar, and maximal in the number of edges.

Corollary. All the faces of such a graph are triangles, except for the unbounded one, which is the exterior of the convex hull of P.



COMPLEXITY

Every triangulation of any set P of n points has:

$2n - h - 2$ triangles

$3n - h - 3$ edges

where h is the number of vertices of $ch(P)$.

DEGENERACIES

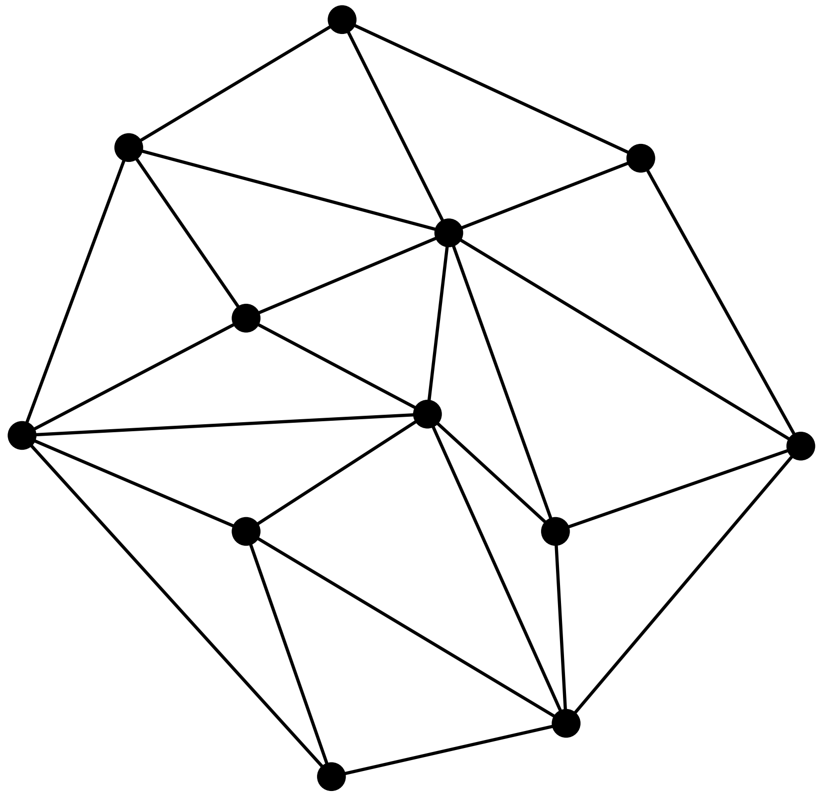
As you may have noticed, we are assuming that the set P does not contain three or more points on a line. The assumption is hold along the entire chapter.

TRIANGULATING POINT SETS

DATA STRUCTURE

We want to answer the most usual questions for any decomposition of the plane:

- For any given triangle, report its edges/vertices.
- For any given vertex, report the sorted list of edges/triangles incident to it.
- For any given edge, report its endpoints and its adjacent triangles.



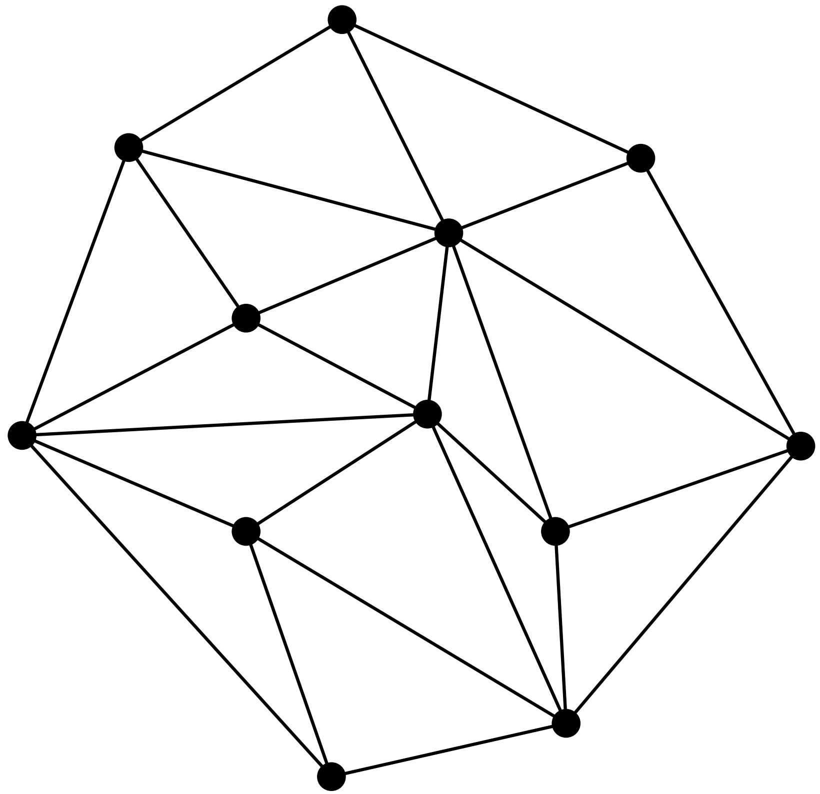
TRIANGULATING POINT SETS

DATA STRUCTURE

We want to answer the most usual questions for any decomposition of the plane:

- For any given triangle, report its edges/vertices.
- For any given vertex, report the sorted list of edges/triangles incident to it.
- For any given edge, report its endpoints and its adjacent triangles.

The appropiate structure is, once again, a **DCEL**:



TRIANGULATING POINT SETS

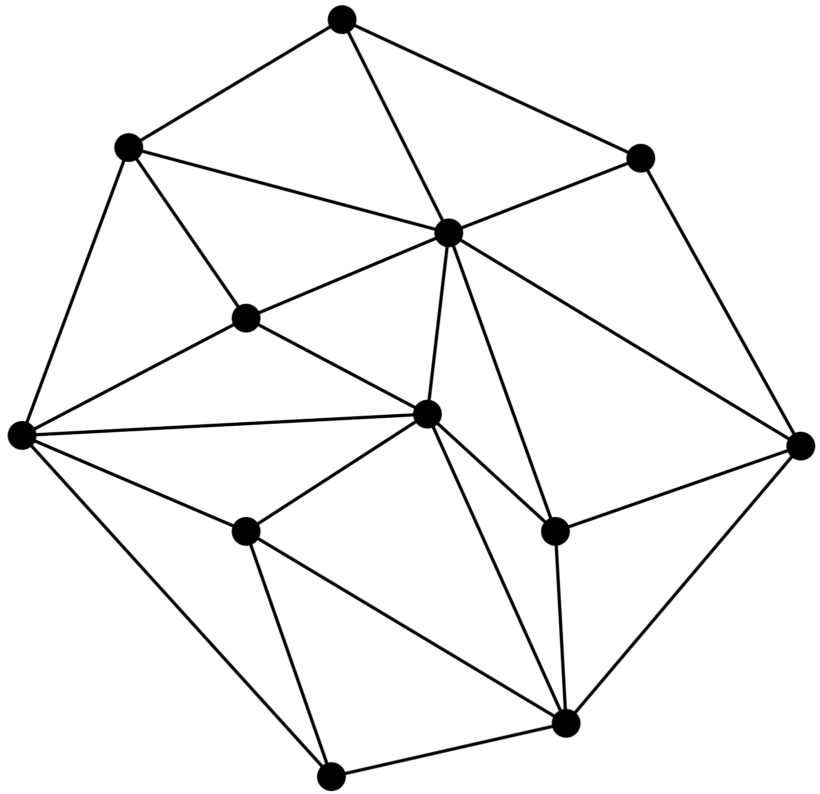
DATA STRUCTURE

We want to answer the most usual questions for any decomposition of the plane:
- For any given triangle, report its edges/vertices.
- For any given vertex, report the sorted list of edges/triangles incident to it.
- For any given edge, report its endpoints and its adjacent triangles.

The appropiate structure is, once again, a **DCEL**:

Table of vertices

v	x	y	e



DATA STRUCTURE

We want to answer the most usual questions for any decomposition of the plane:

- For any given triangle, report its edges/vertices.
- For any given vertex, report the sorted list of edges/triangles incident to it.
- For any given edge, report its endpoints and its adjacent triangles.

The appropiate structure is, once again, a **DCEL**:

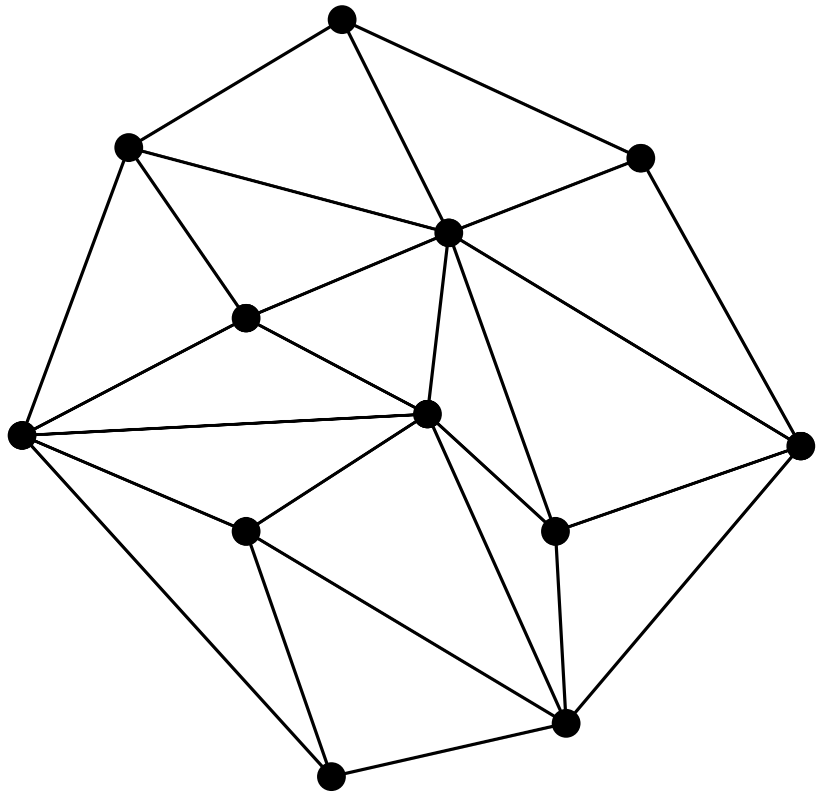


Table of vertices

v	x	y	e

Table of faces

t	e

DATA STRUCTURE

We want to answer the most usual questions for any decomposition of the plane:

- For any given triangle, report its edges/vertices.
- For any given vertex, report the sorted list of edges/triangles incident to it.
- For any given edge, report its endpoints and its adjacent triangles.

The appropiate structure is, once again, a **DCEL**:

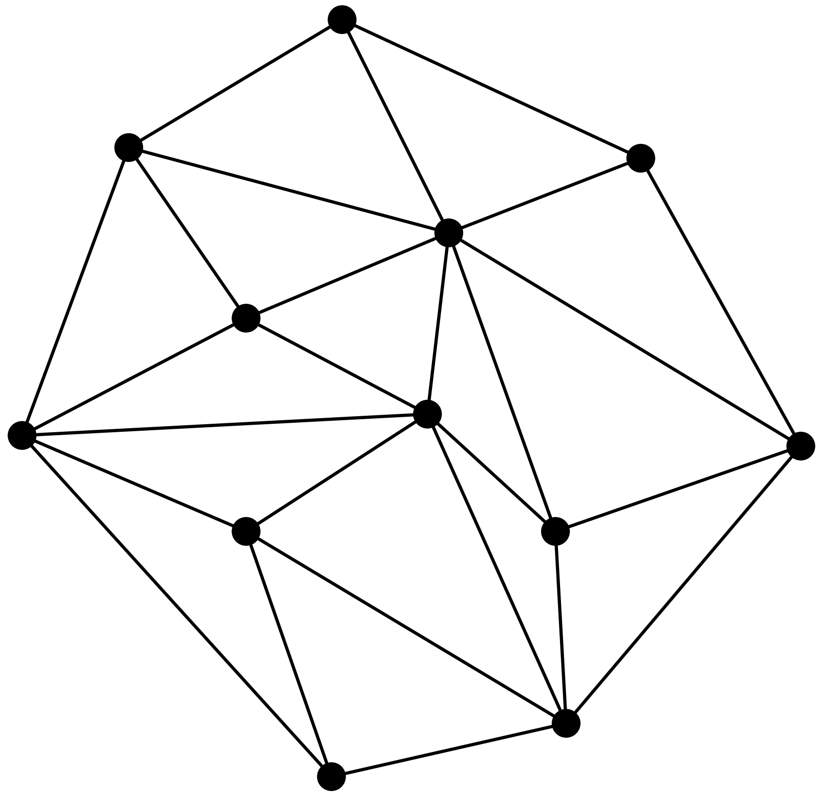


Table of vertices

v	x	y	e

Table of faces

t	e

DCEL

e	v_B	v_E	f_L	f_R	e_P	e_N

ALGORITHMS

ALGORITHMS

1. Incremental algorithms

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

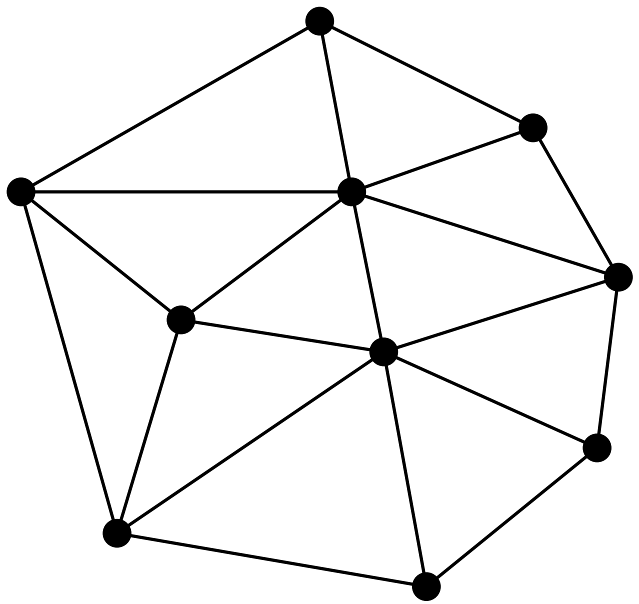
For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \dots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \dots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.



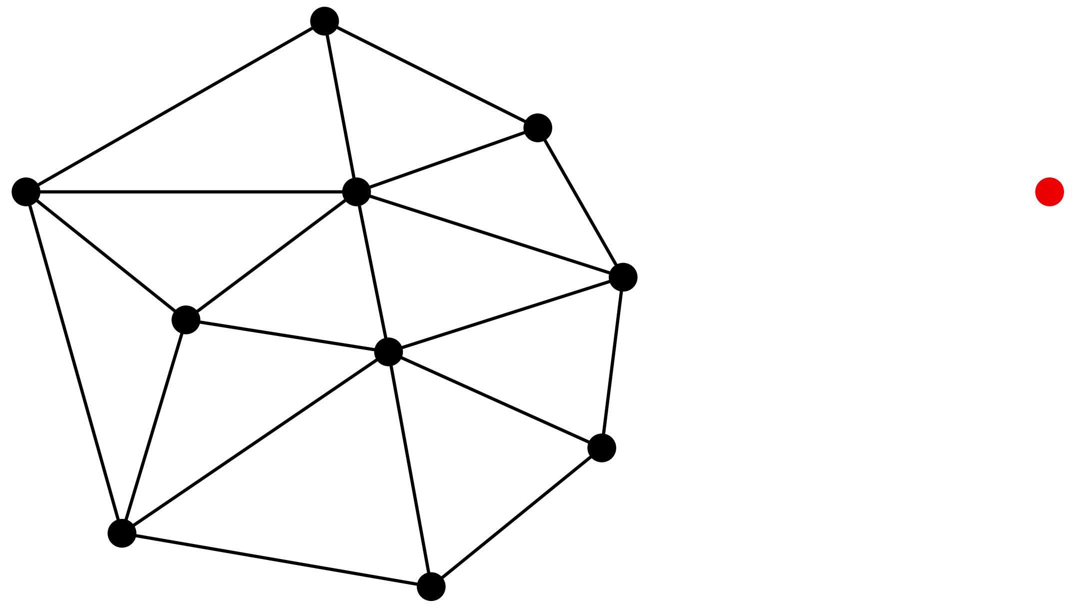
TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.



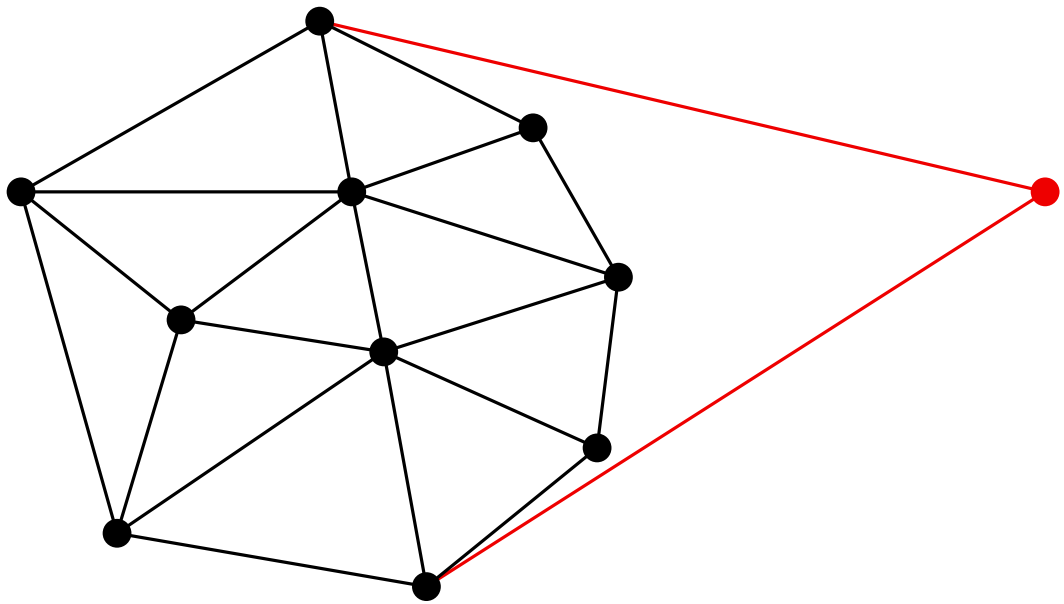
TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

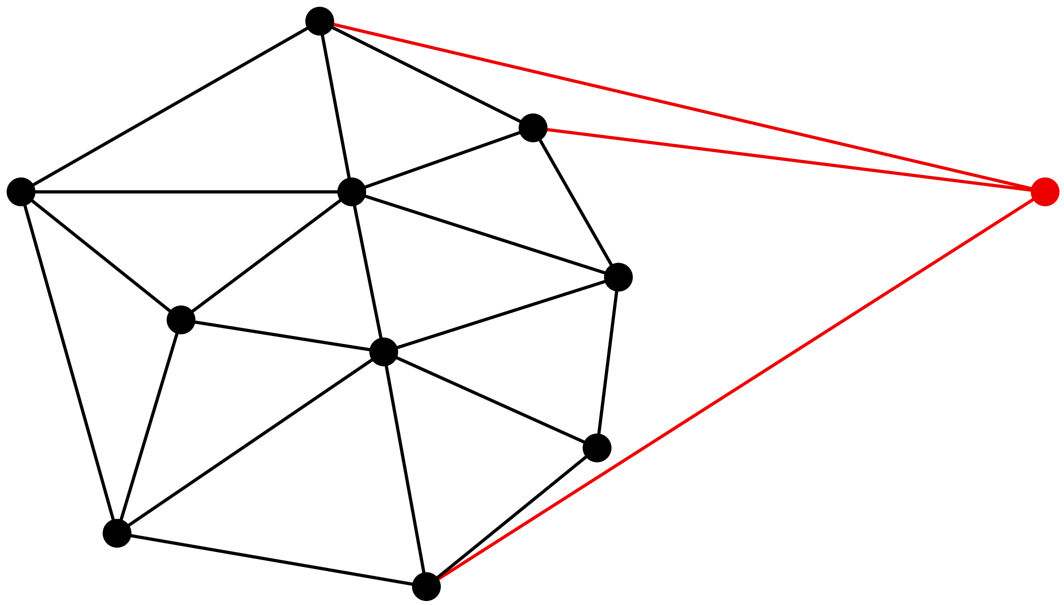


ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

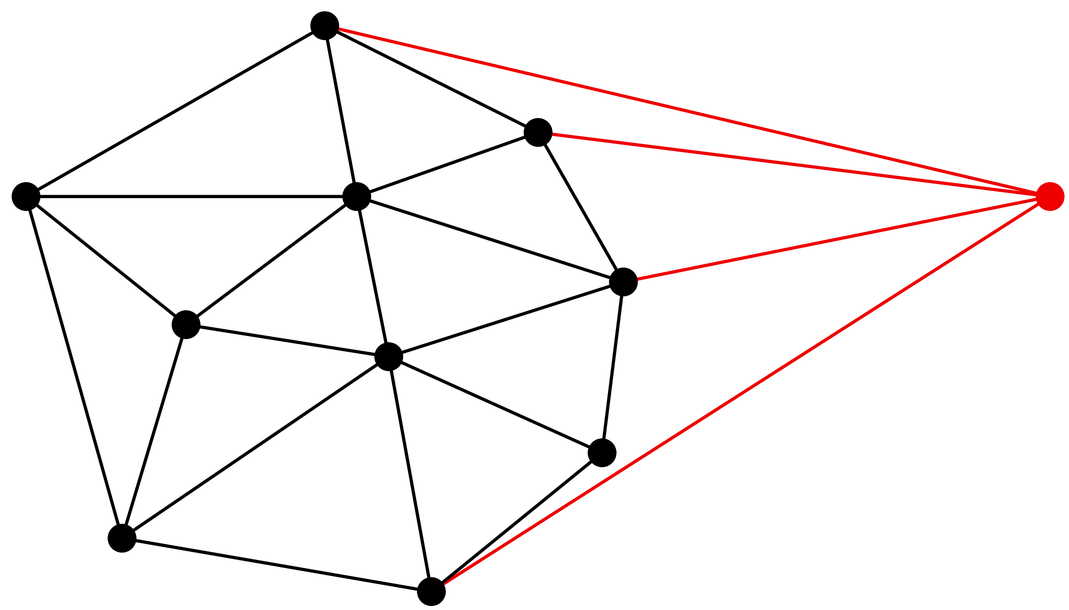


ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

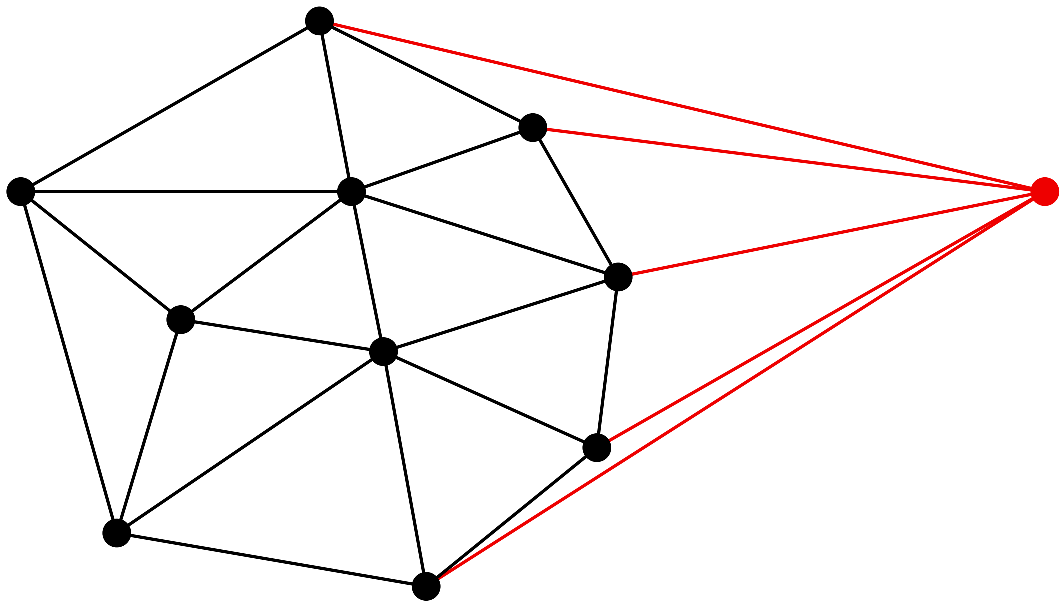


ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

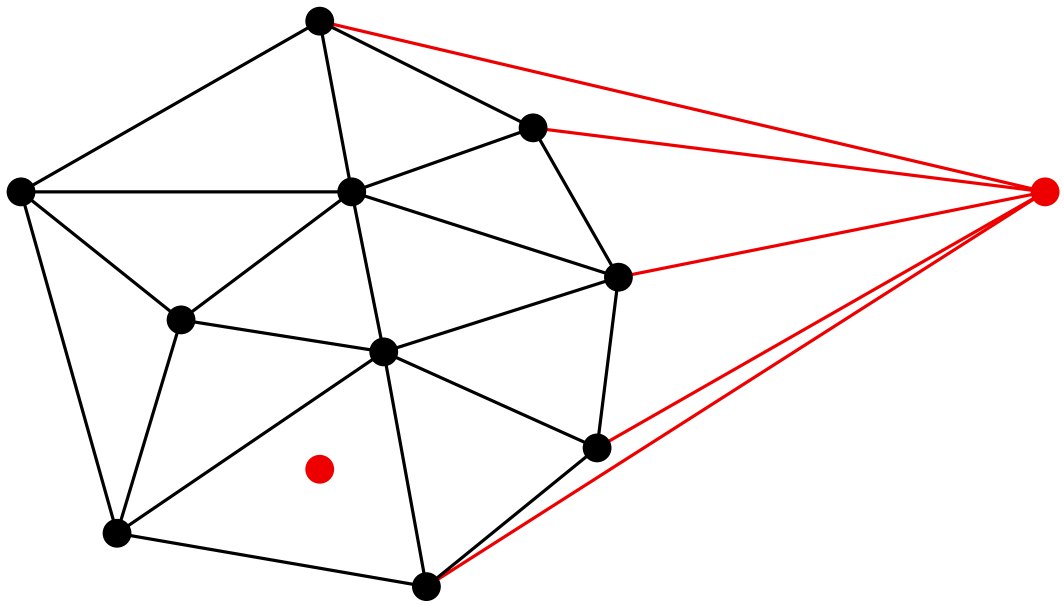


ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \dots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \dots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.

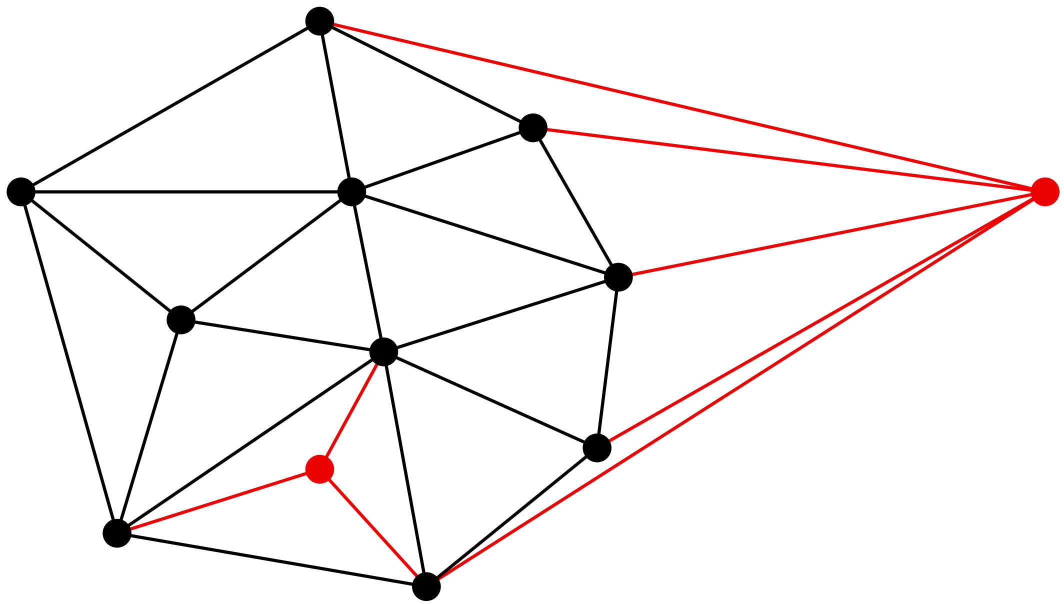


ALGORITHMS

1. Incremental algorithms

1.1. Without sorting

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.



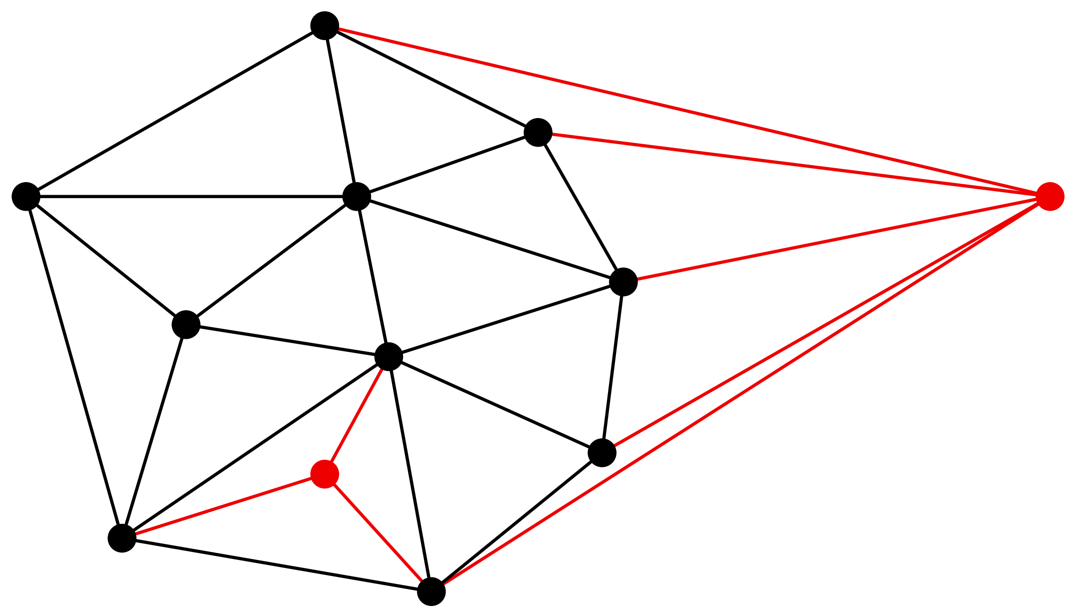
TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting **Running time:** $O(n^2)$

For each i, detect whether p_i lies in the interior or the exterior of $ch(p_1, \ldots, p_{i-1})$. If it is external, compute the supporting lines from p_i to $ch(p_1, \ldots, p_{i-1})$ and add all the intermediate diagonals to the triangulation. If it is internal, detect the triangle T containing p_i and partition T into 3 triangles.



ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting
   1.2. With sorting

Running time: $O(n^2)$

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting **Running time:** $O(n^2)$

1.2. With sorting

Start by sorting the points in lexicographical order in $O(n \log n)$ time. The information of the sorted order of the points allows to add the i diagonals in $O(i)$ time, so that the amortized cost of the insertion of all diagonals is done in $O(n)$ time.

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting **Running time:** $O(n^2)$

1.2. With sorting **Running time:** $O(n \log n)$

Start by sorting the points in lexicographical order in $O(n \log n)$ time. The information of the sorted order of the points allows to add the i diagonals in $O(i)$ time, so that the amortized cost of the insertion of all diagonals is done in $O(n)$ time.

ALGORITHMS

1. Incremental algorithms

- **1.1.** Without sorting — **Running time:** $O(n^2)$
- **1.2.** With sorting — **Running time:** $O(n \log n)$
- **1.3.** With hierarchical structure

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

1.3. With hierarchical structure

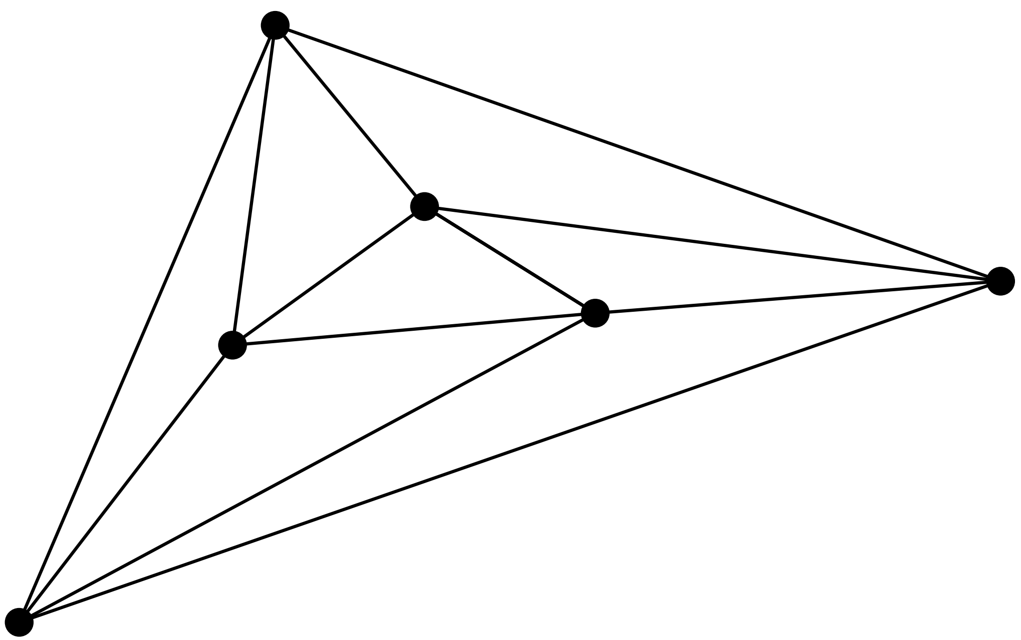
Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure

Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.

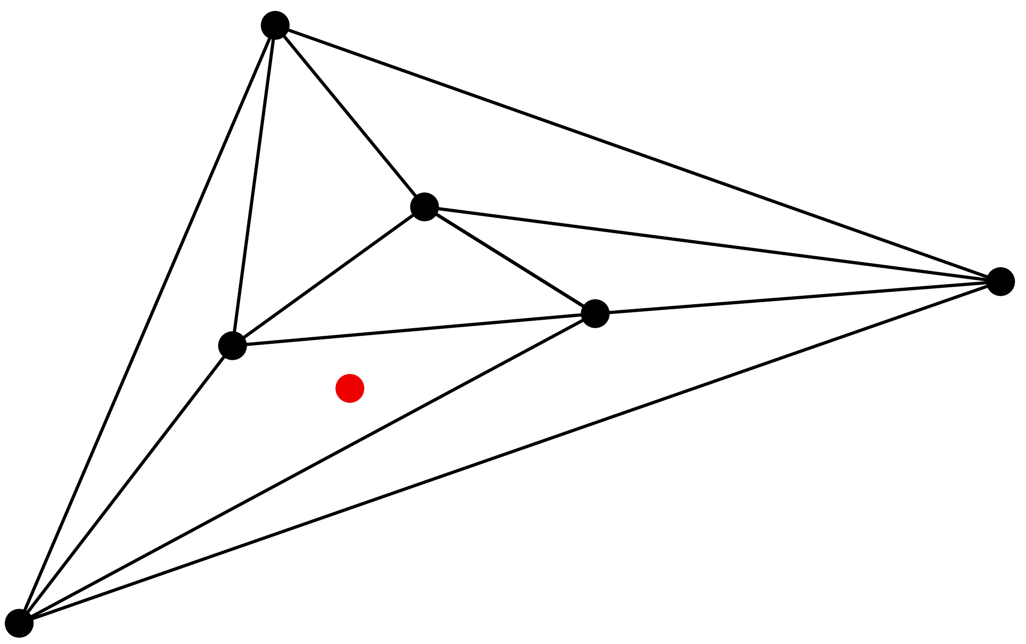


TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure

Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.



TRIANGULATING POINT SETS

ALGORITHMS

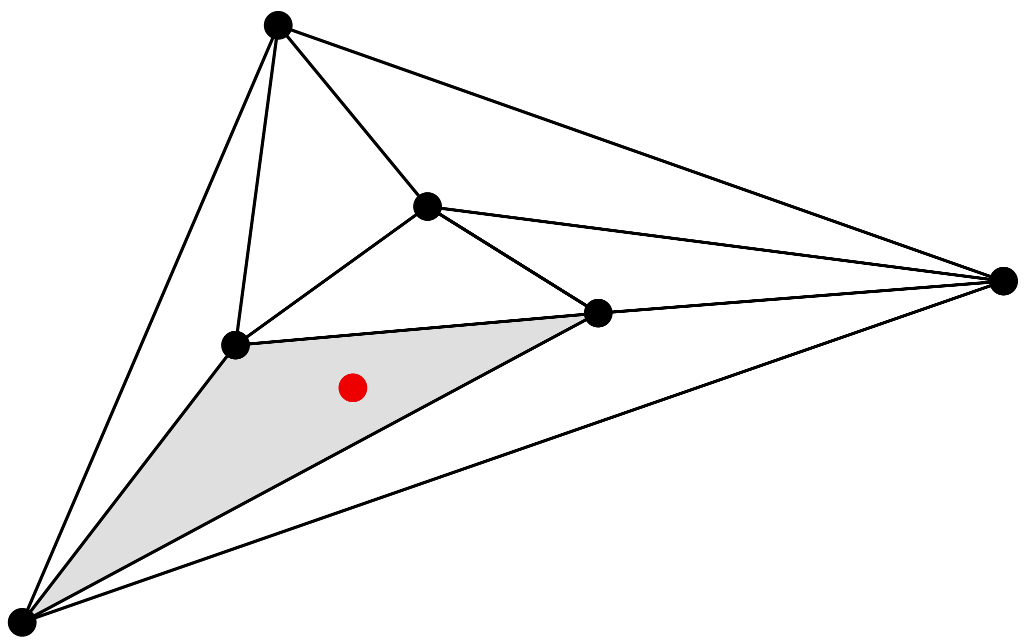
1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

1.3. With hierarchical structure

Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.



T

TRIANGULATING POINT SETS

ALGORITHMS

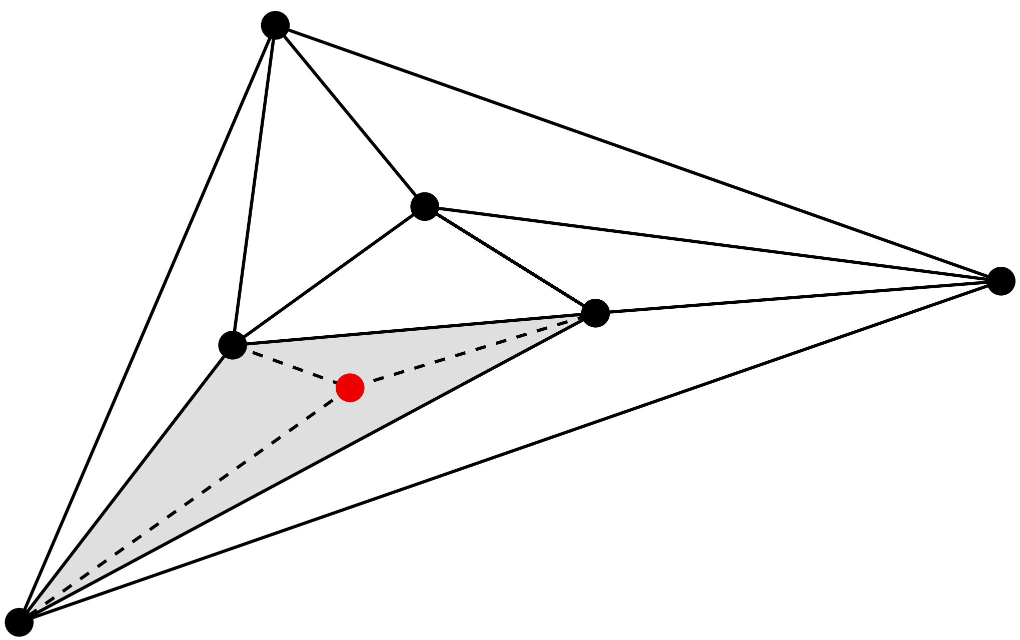
1. Incremental algorithms

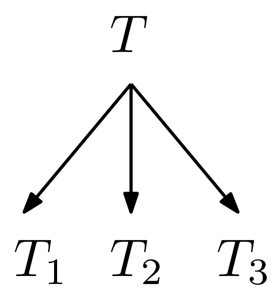
1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

1.3. With hierarchical structure

Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.





TRIANGULATING POINT SETS

ALGORITHMS

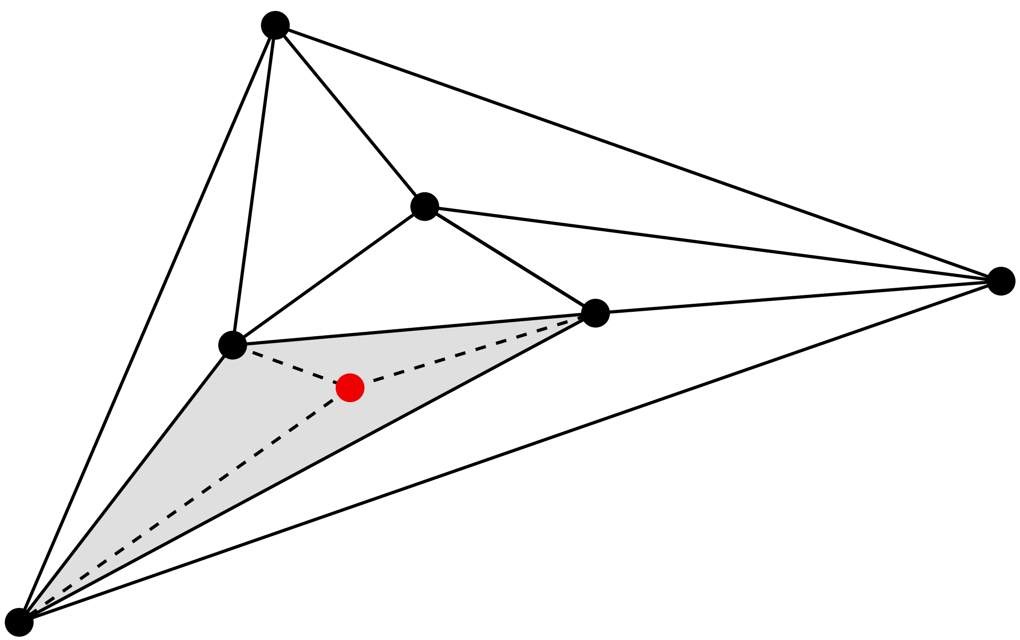
1. Incremental algorithms

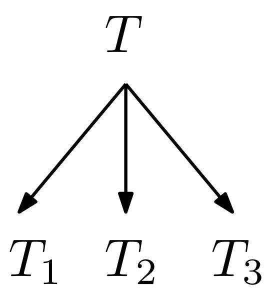
1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

Using an auxiliary enclosing triangle and a hierarchy of triangles: each time a new point is added, a triangle gets subdivided into three children.





ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1. **1.1.** Without sorting — **Running time:** $O(n^2)$
   2. **1.2.** With sorting — **Running time:** $O(n \log n)$
   3. **1.3.** With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   4. **1.4.** Randomized — **Running time:** $O(n \log n)$ expected
   5. **1.5.** With auxiliary point(s)

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

1.4. Randomized — **Running time:** $O(n \log n)$ expected

1.5. With auxiliary point(s)

A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
When inserting each point p_i:

- Scan the triangles stabbed by the segment $\overline{pp_i}$.
- Update, if necessary, the information of the triangle containing p.

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

   1.1. Without sorting — **Running time:** $O(n^2)$

   1.2. With sorting — **Running time:** $O(n \log n)$

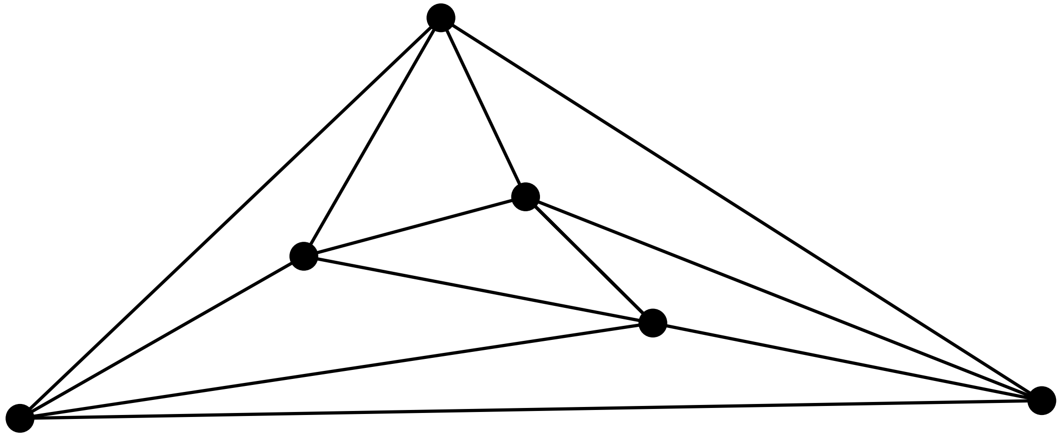
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

   1.4. Randomized — **Running time:** $O(n \log n)$ expected

   1.5. With auxiliary point(s)

   A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
   When inserting each point p_i:

   - Scan the triangles stabbed by the segment $\overline{pp_i}$.
   - Update, if necessary, the information of the triangle containing p.



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

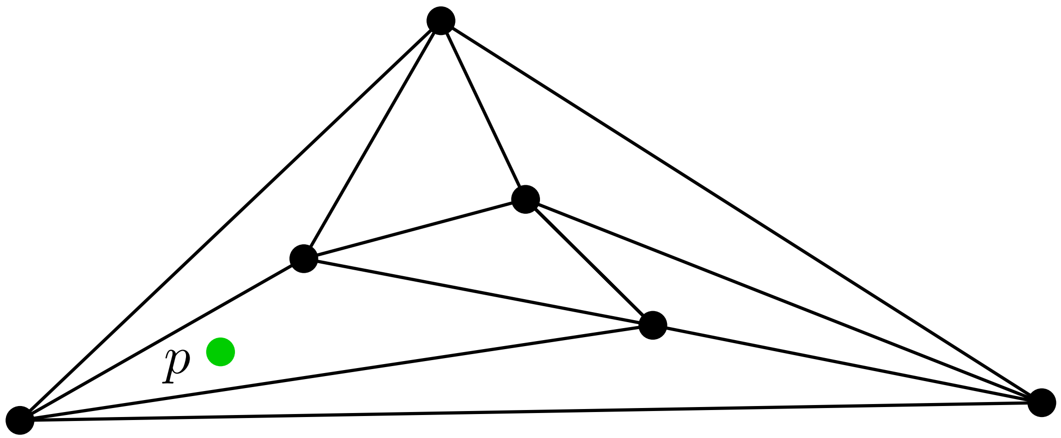
1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

1.4. Randomized — **Running time:** $O(n \log n)$ expected

1.5. With auxiliary point(s)

A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
When inserting each point p_i:

- Scan the triangles stabbed by the segment $\overline{pp_i}$.
- Update, if necessary, the information of the triangle containing p.



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

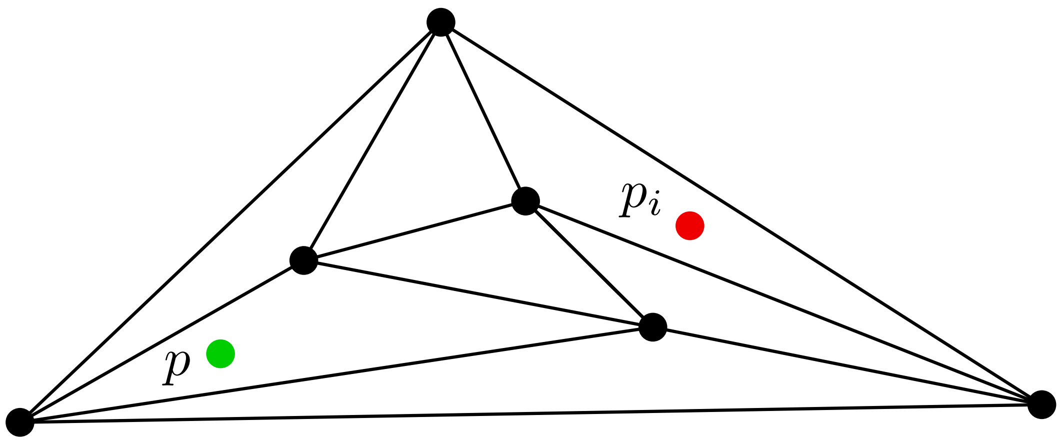
1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

1.4. Randomized — **Running time:** $O(n \log n)$ expected

1.5. With auxiliary point(s)

A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
When inserting each point p_i:

- Scan the triangles stabbed by the segment $\overline{pp_i}$.
- Update, if necessary, the information of the triangle containing p.



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

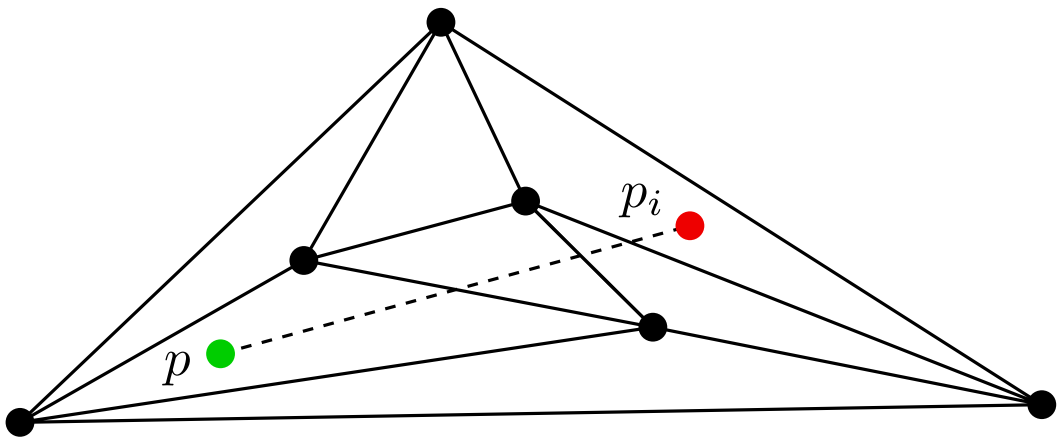
1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

1.4. Randomized — **Running time:** $O(n \log n)$ expected

1.5. With auxiliary point(s)

A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
When inserting each point p_i:

- Scan the triangles stabbed by the segment $\overline{pp_i}$.
- Update, if necessary, the information of the triangle containing p.



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

1.1. Without sorting — **Running time:** $O(n^2)$

1.2. With sorting — **Running time:** $O(n \log n)$

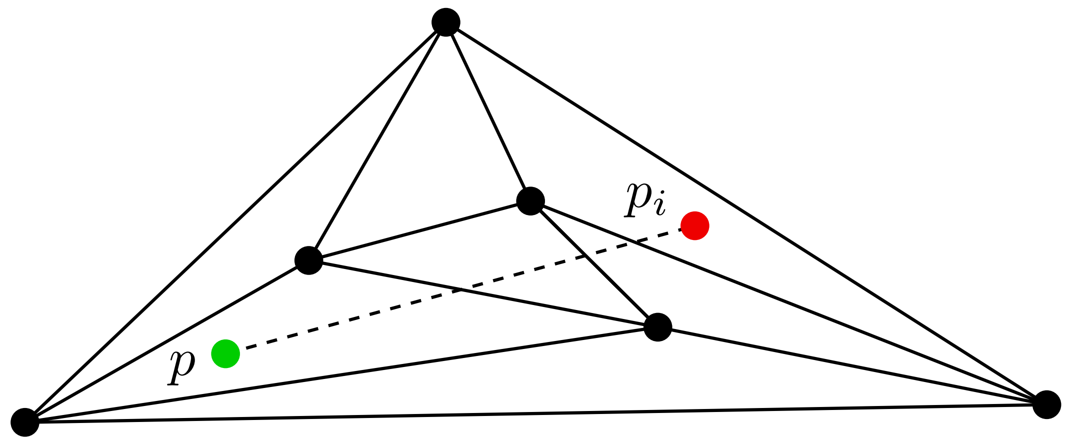
1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced

1.4. Randomized — **Running time:** $O(n \log n)$ expected

1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected

A fixed point p is used as a reference, and $P \cup \{p\}$ is enclosed in an auxiliary triangle.
When inserting each point p_i:

- Scan the triangles stabbed by the segment $\overline{pp_i}$.
- Update, if necessary, the information of the triangle containing p.



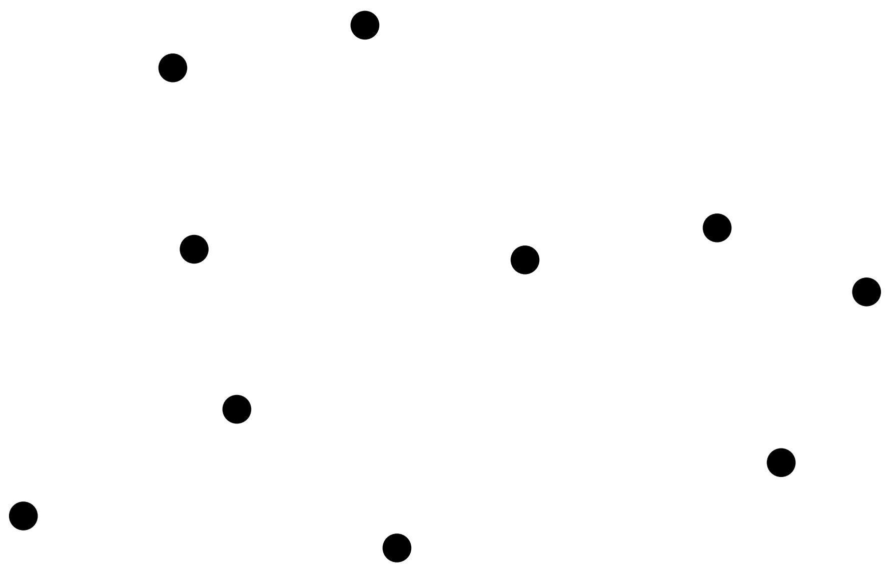
ALGORITHMS

1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



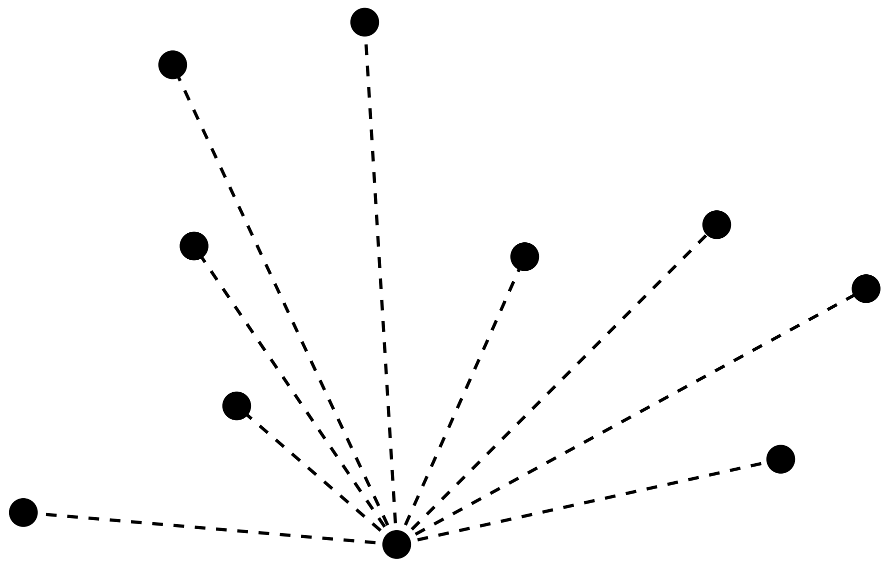
TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms

- **1.1.** Without sorting — **Running time:** $O(n^2)$
- **1.2.** With sorting — **Running time:** $O(n \log n)$
- **1.3.** With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
- **1.4.** Randomized — **Running time:** $O(n \log n)$ expected
- **1.5.** With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected

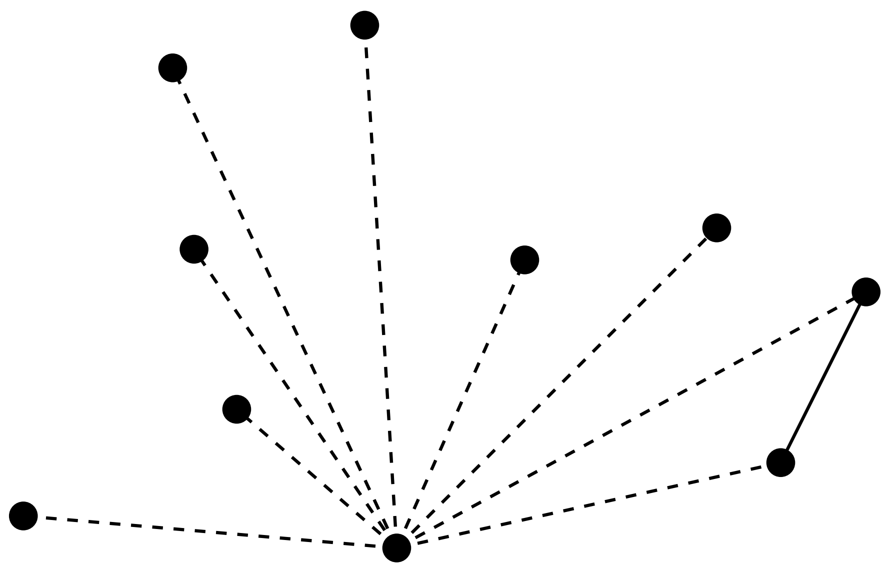
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

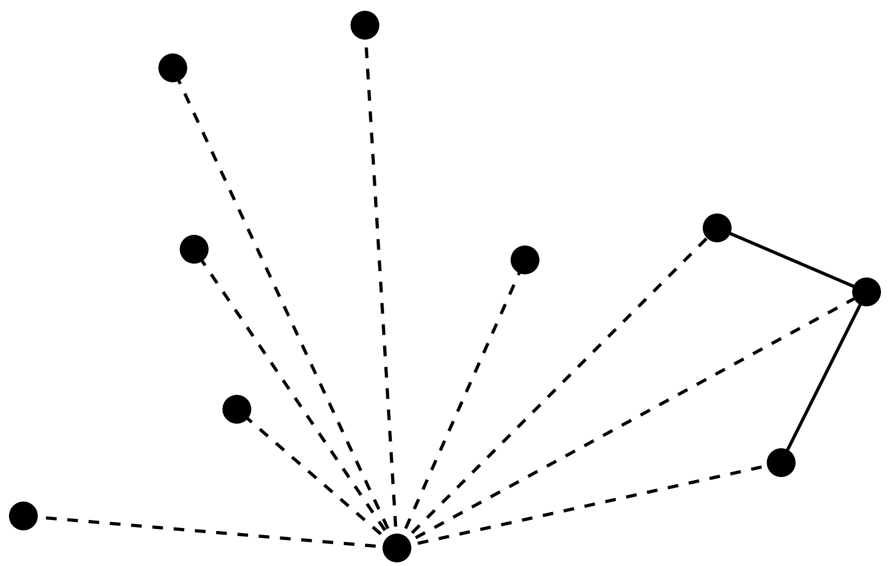
1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

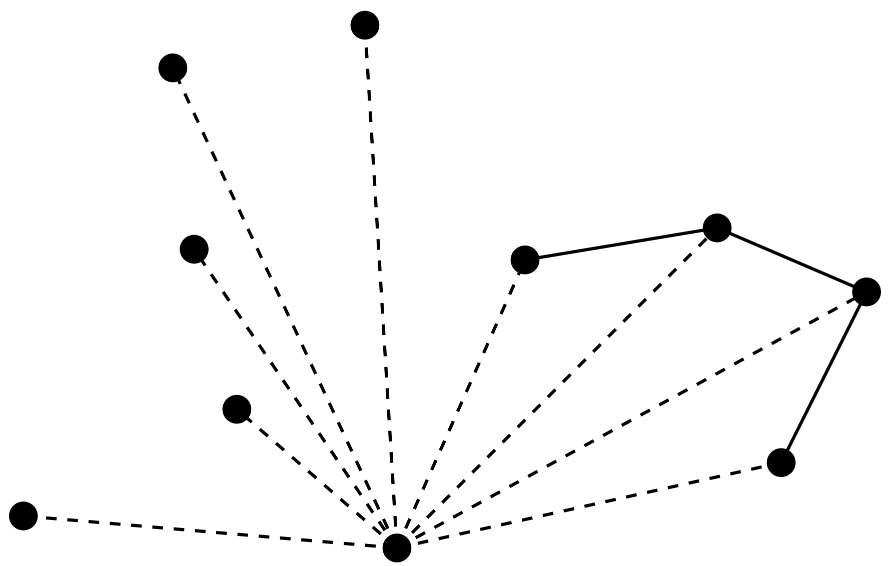
1. Incremental algorithms
    1. 1.1. Without sorting — **Running time:** $O(n^2)$
    2. 1.2. With sorting — **Running time:** $O(n \log n)$
    3. 1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    4. 1.4. Randomized — **Running time:** $O(n \log n)$ expected
    5. 1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

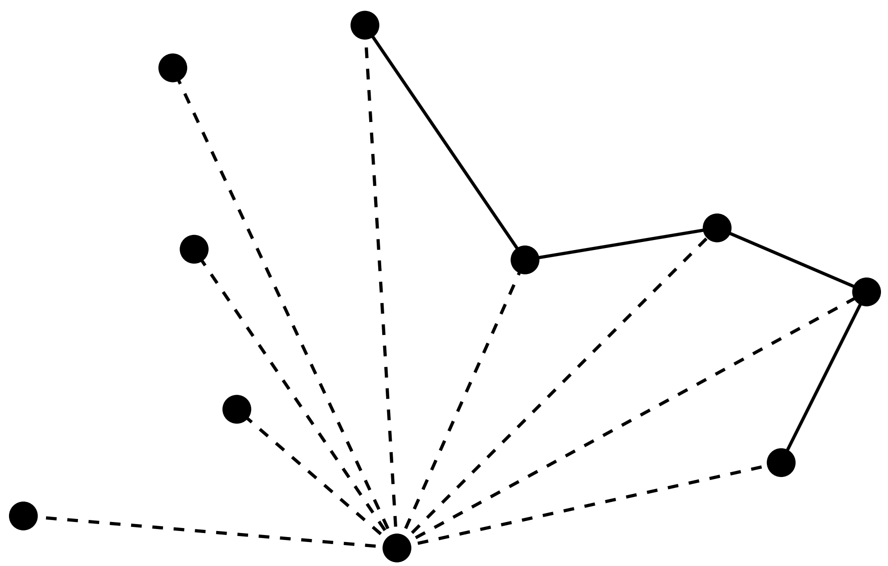
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

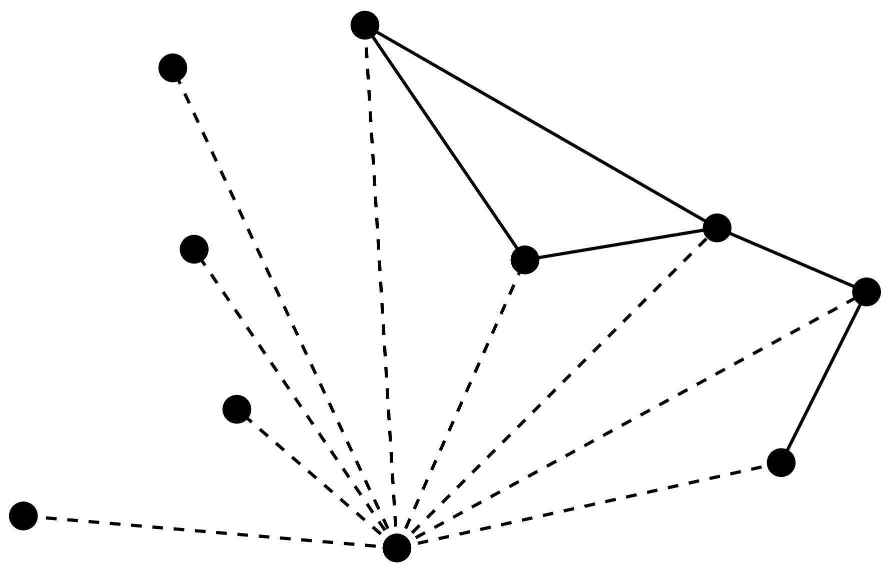
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

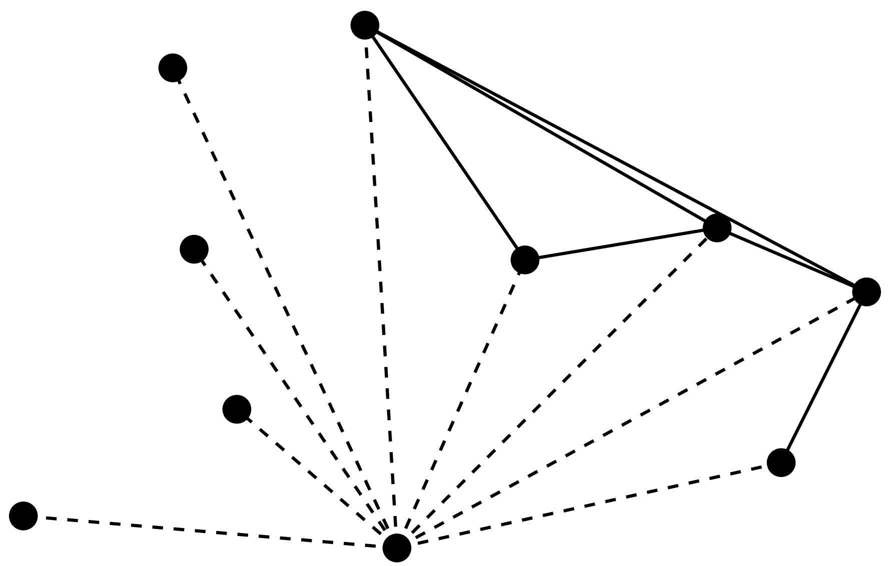
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

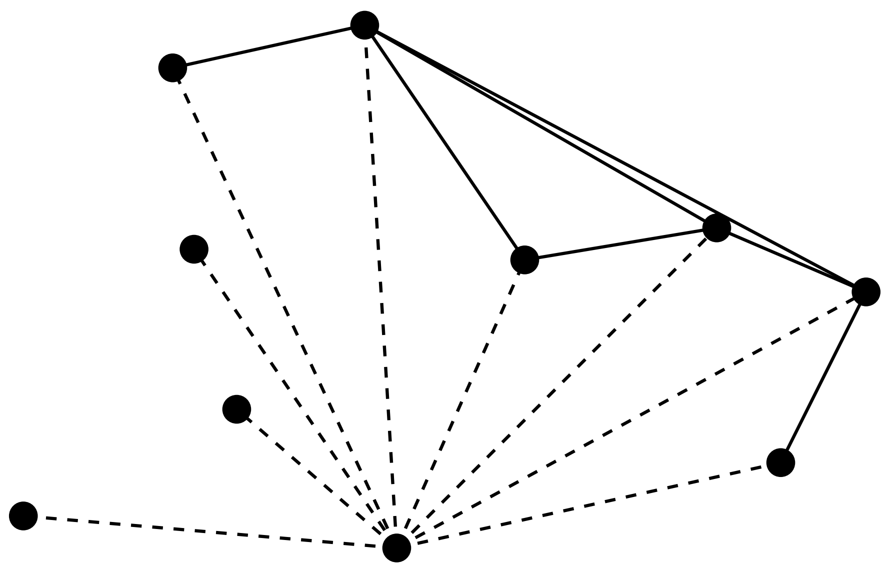
1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

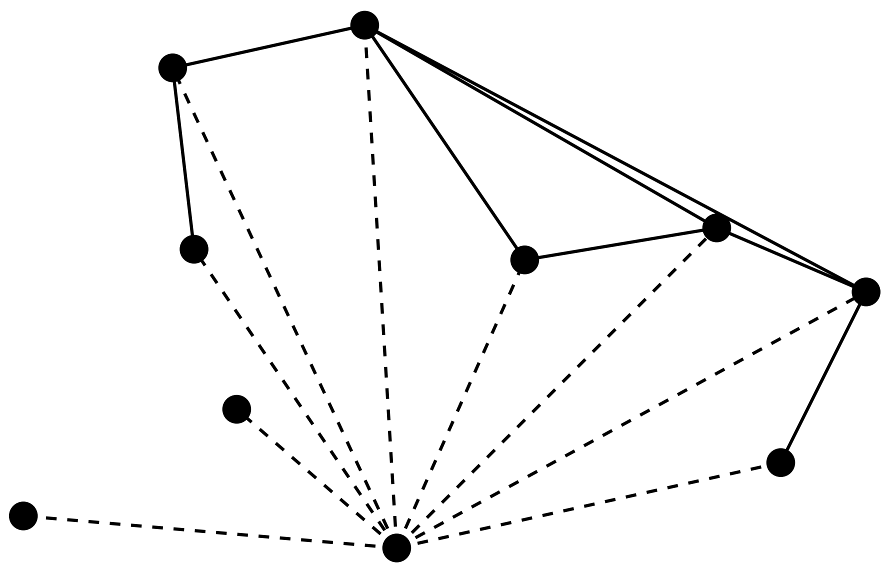
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

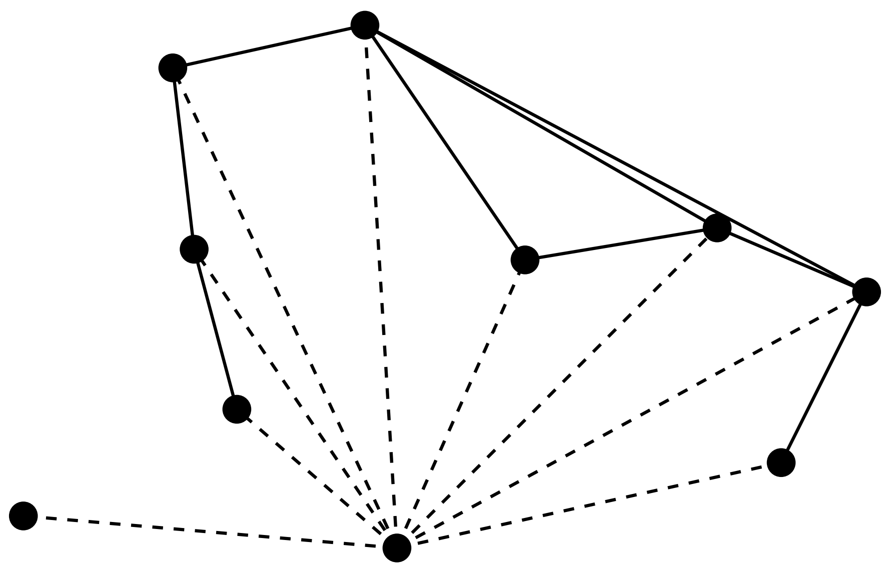
1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

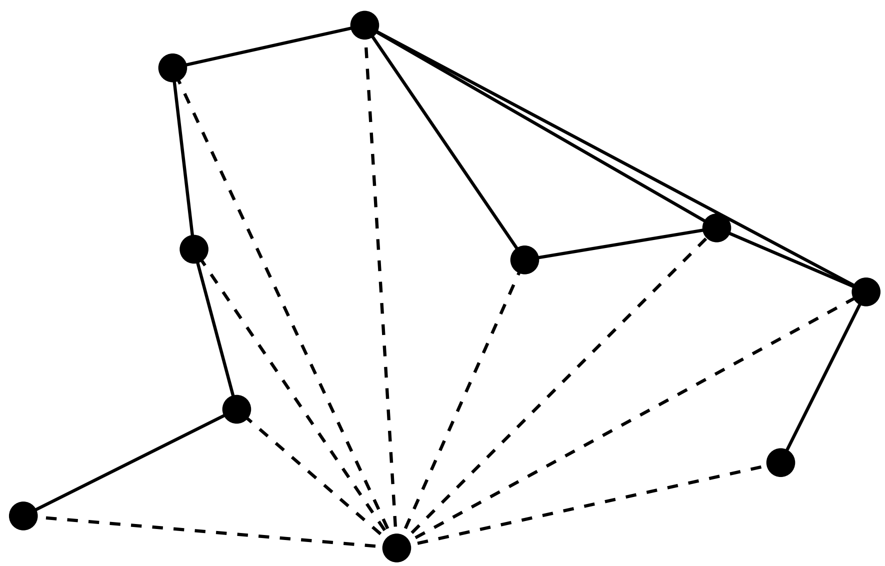
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

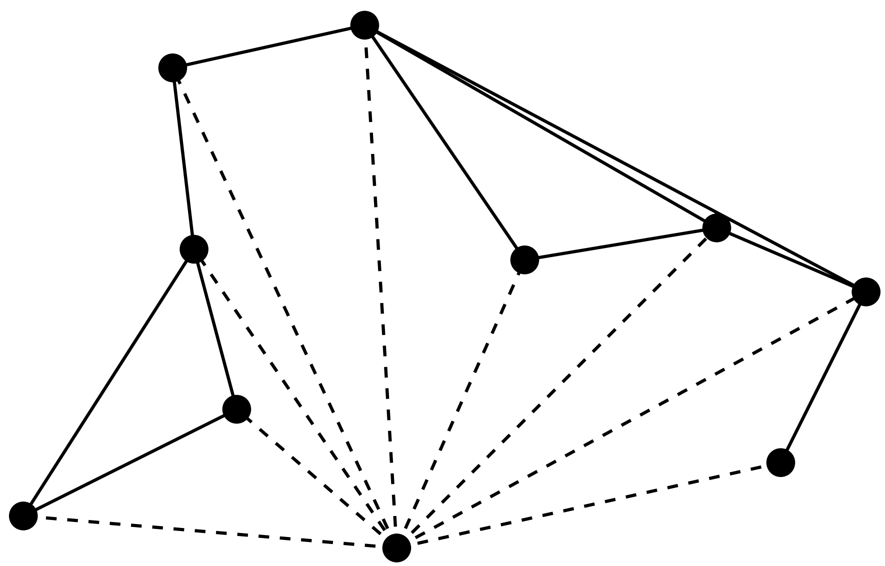
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

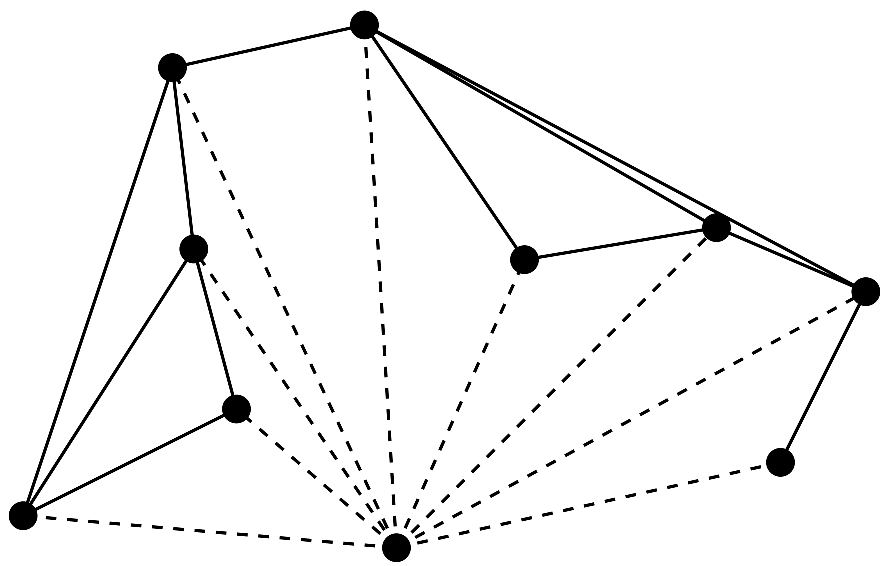
1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

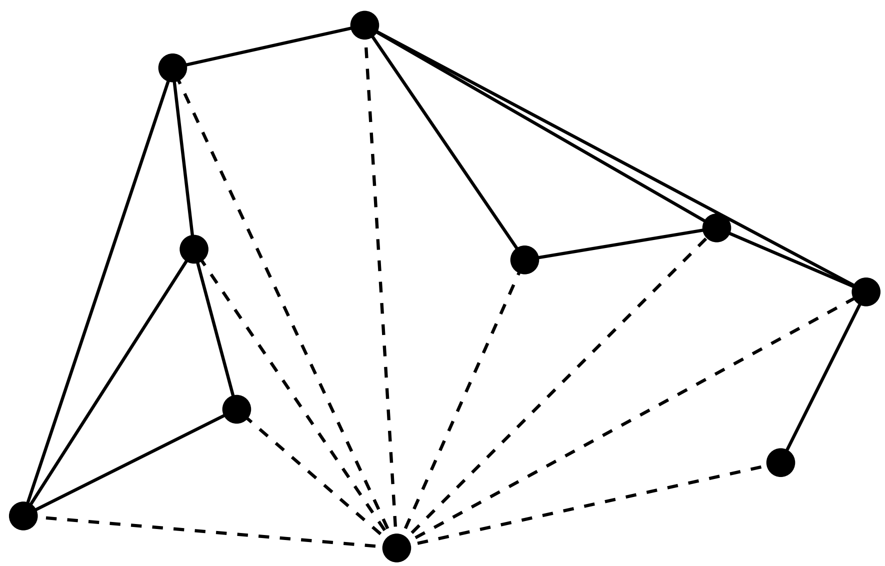
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm — **Running time:** $O(n \log n)$



TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
    1.1. Without sorting — **Running time:** $O(n^2)$
    1.2. With sorting — **Running time:** $O(n \log n)$
    1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
    1.4. Randomized — **Running time:** $O(n \log n)$ expected
    1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm — **Running time:** $O(n \log n)$
3. Dividide and conquer

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm — **Running time:** $O(n \log n)$
3. Dividide and conquer

Initialization

Sort the points by abscissa

Advance

- Partition: divide the points into roughly two vertically separated halfs
- Recursion: recursively triangulate each half
- Fusion: compute the external common tangents and triangulate the intermediate space

TRIANGULATING POINT SETS

ALGORITHMS

1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm — **Running time:** $O(n \log n)$
3. Dividide and conquer — **Running time:** $O(n \log n)$

Initialization

Sort the points by abscissa

Advance

- Partition: divide the points into roughly two vertically separated halfs
- Recursion: recursively triangulate each half
- Fusion: compute the external common tangents and triangulate the intermediate space

TRIANGULATING POINT SETS

ALGORITHMS

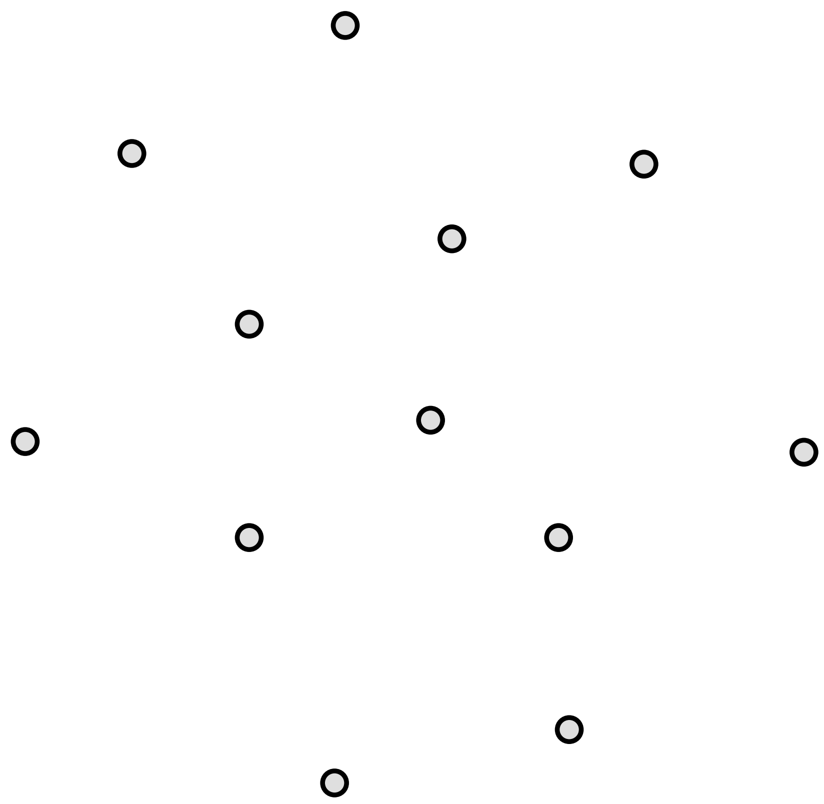
1. Incremental algorithms
   1.1. Without sorting — **Running time:** $O(n^2)$
   1.2. With sorting — **Running time:** $O(n \log n)$
   1.3. With hierarchical structure — **Running time:** $O(n^2)$ worst case, $O(n \log n)$ if balanced
   1.4. Randomized — **Running time:** $O(n \log n)$ expected
   1.5. With auxiliary point(s) — **Running time:** $O(n^2)$ worst case, $O(n^{3/2})$ expected
2. Graham's algorithm — **Running time:** $O(n \log n)$
3. Dividide and conquer — **Running time:** $O(n \log n)$

LOWER BOUND

This problem has an $\Omega(n \log n)$ lower bound, since the convex hull of the set of points can be trivially obtained in $O(n)$ time from the triangulation.

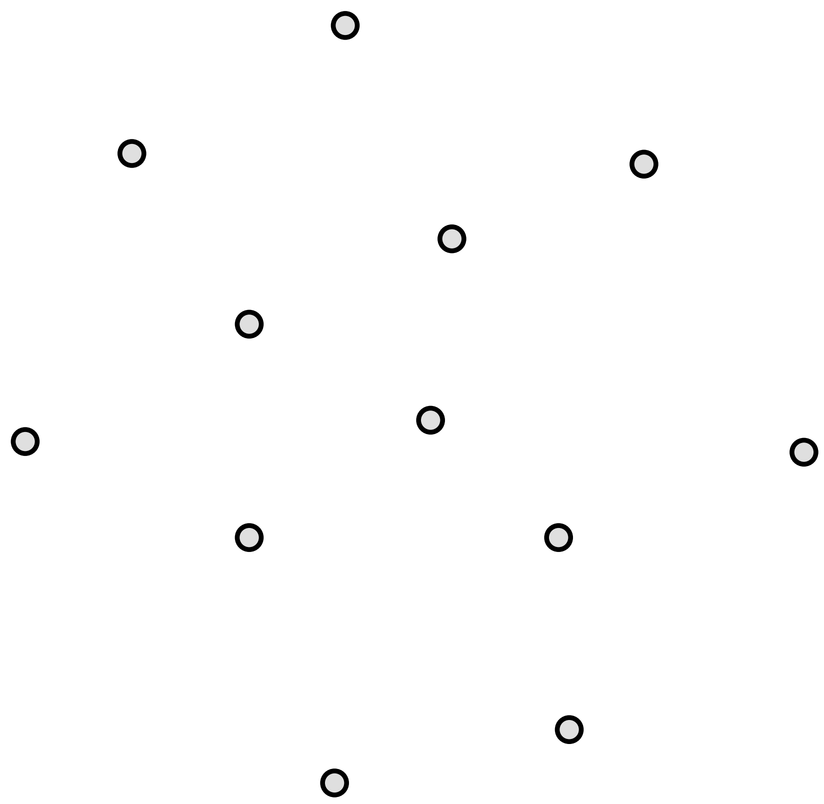
Quality of a triangulation

Quality of a triangulation



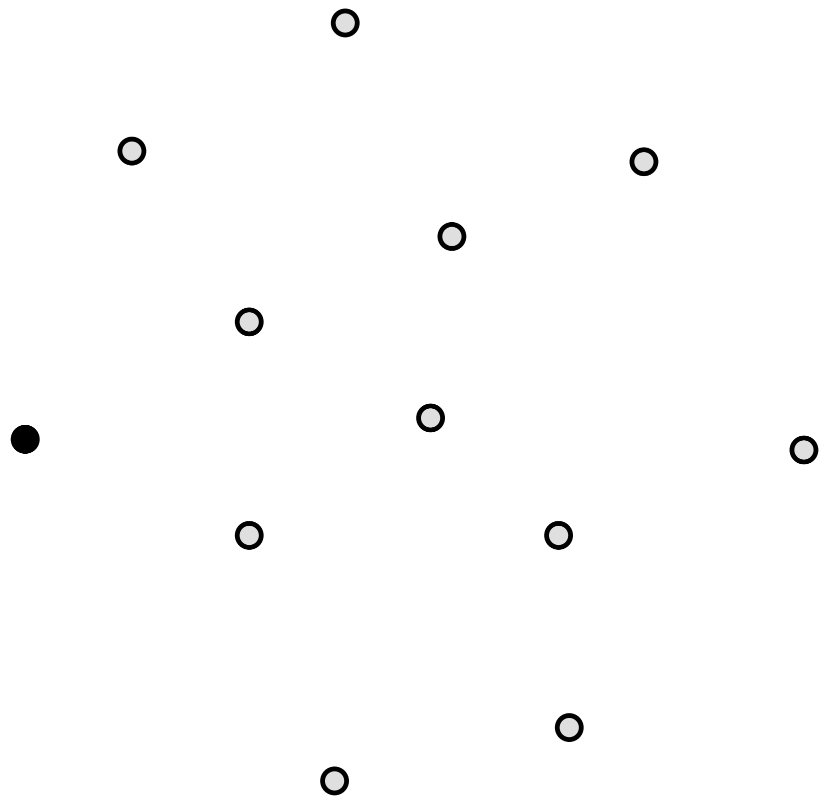
Quality of a triangulation

Incremental, without sorting



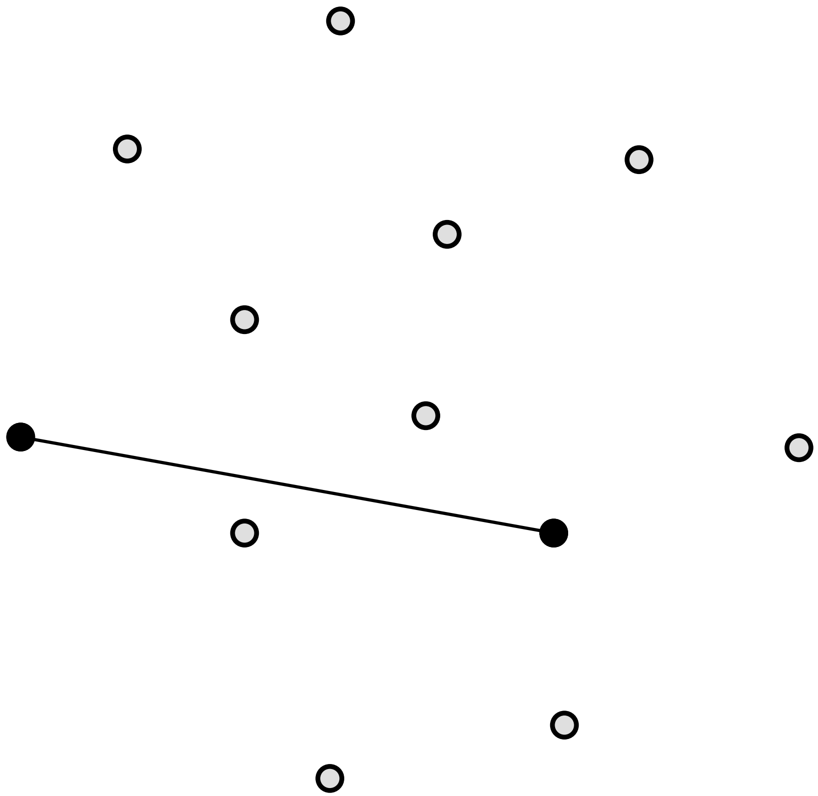
Quality of a triangulation

Incremental, without sorting



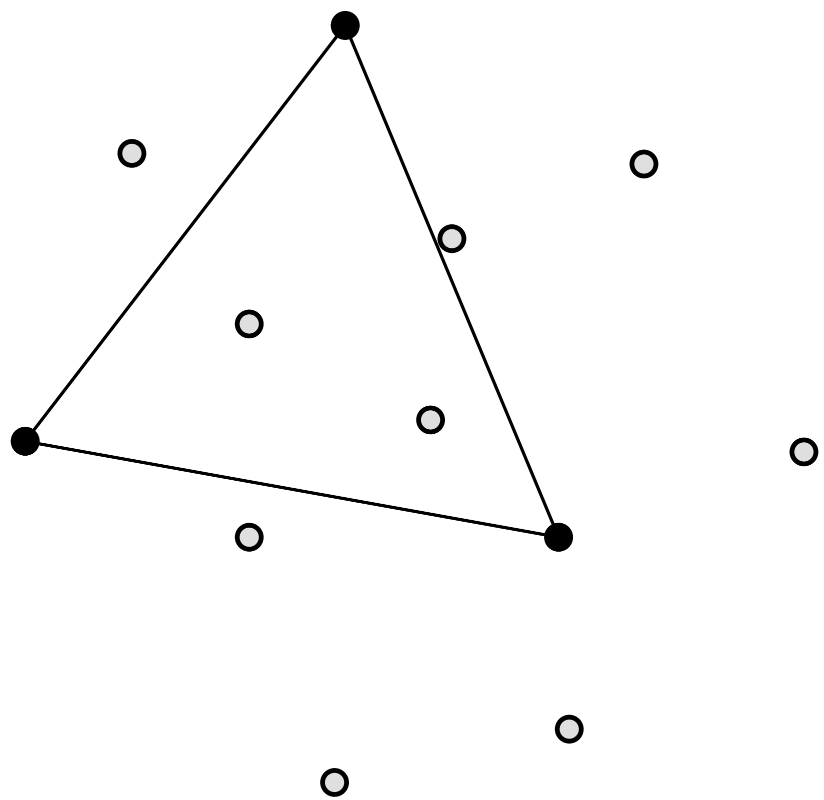
Quality of a triangulation

Incremental, without sorting



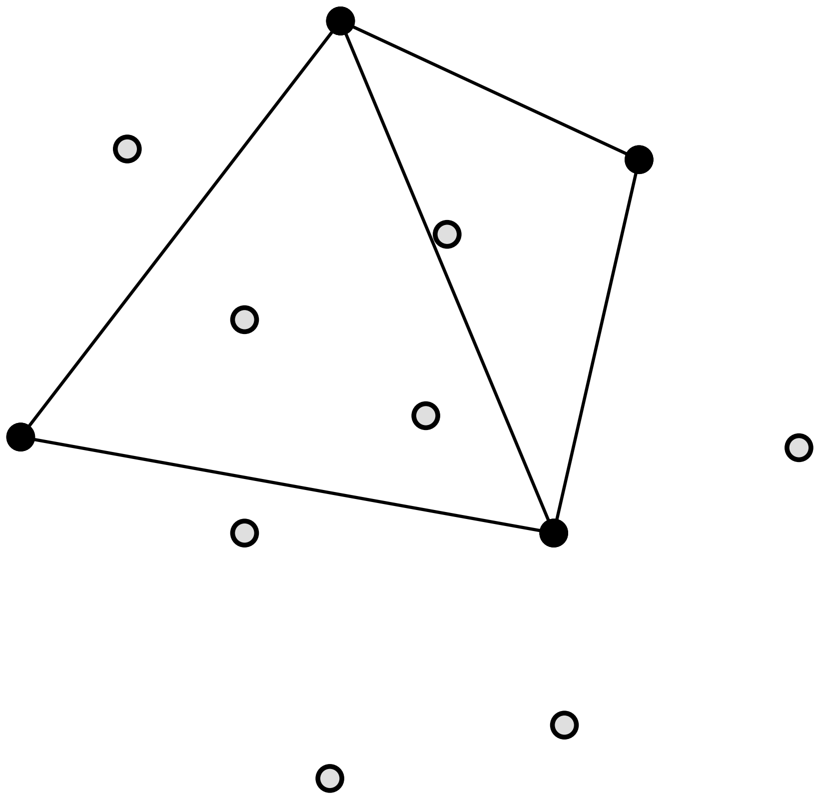
Quality of a triangulation

Incremental, without sorting



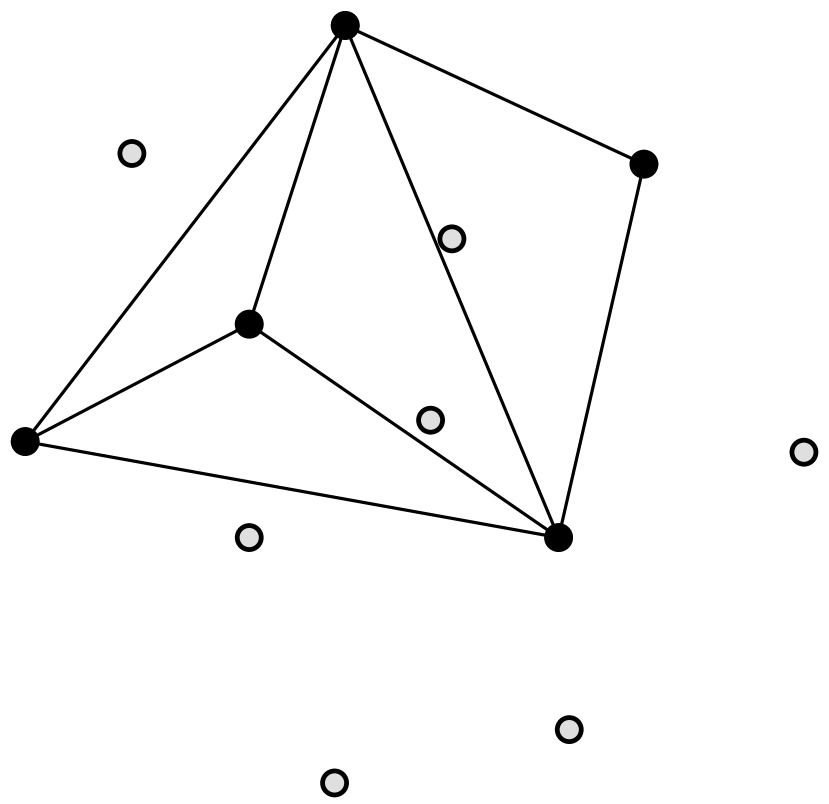
Quality of a triangulation

Incremental, without sorting



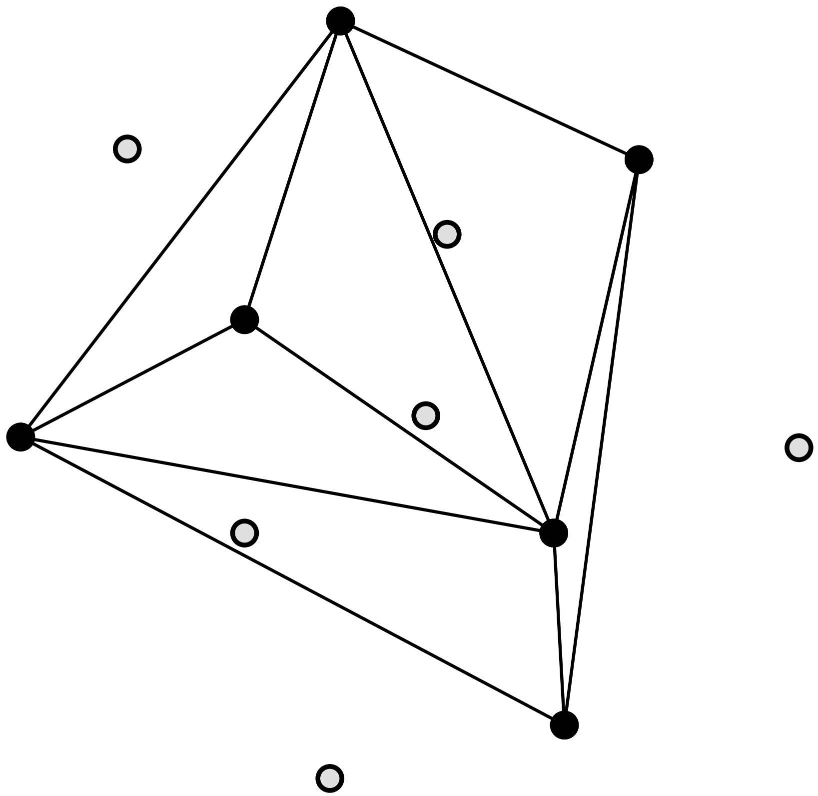
Quality of a triangulation

Incremental, without sorting



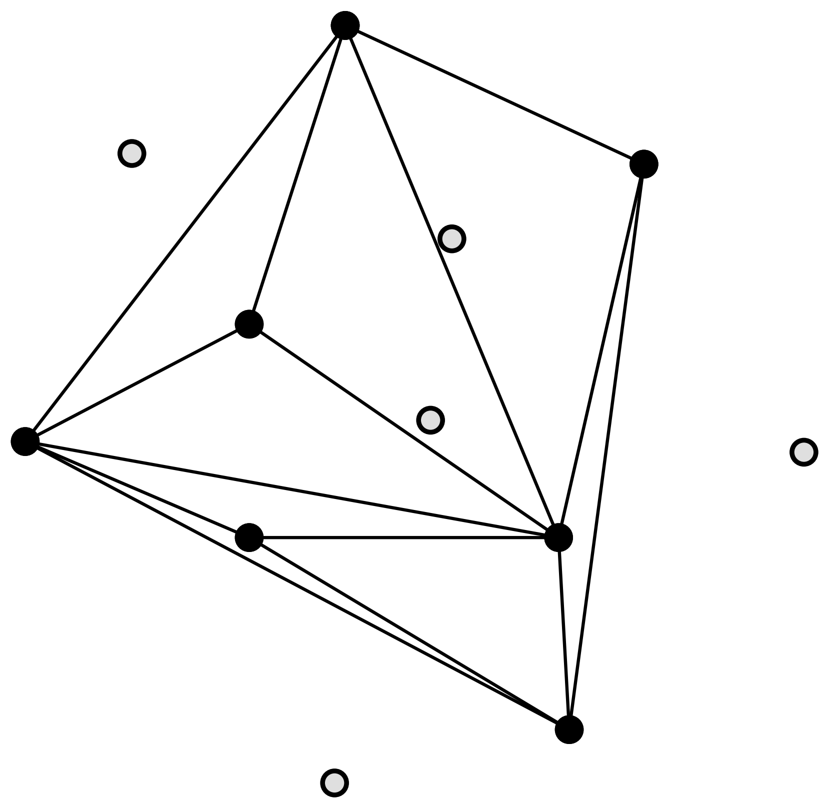
Quality of a triangulation

Incremental, without sorting



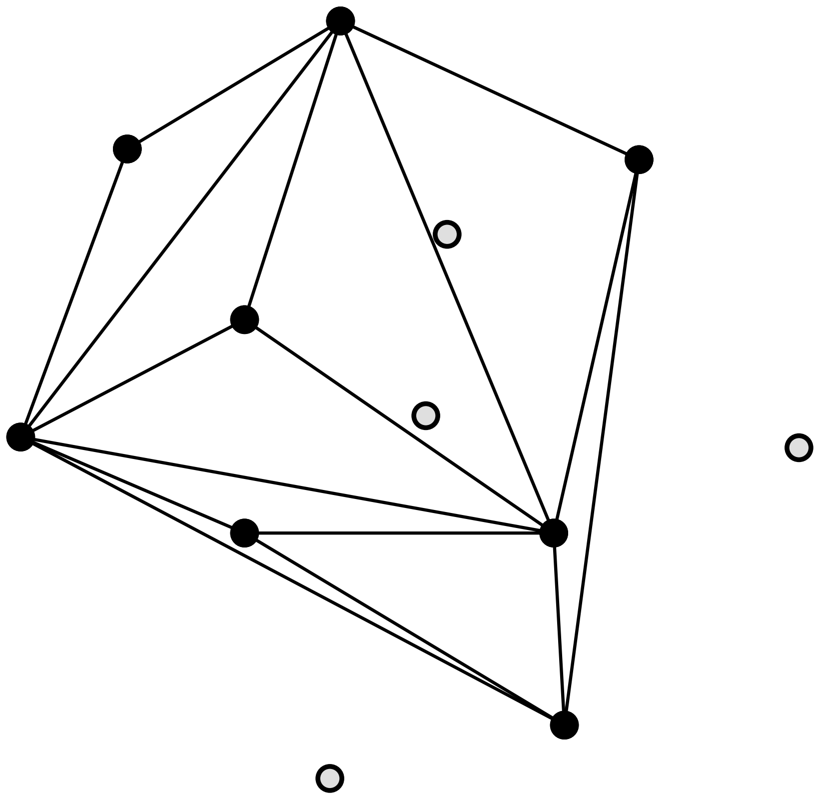
Quality of a triangulation

Incremental, without sorting



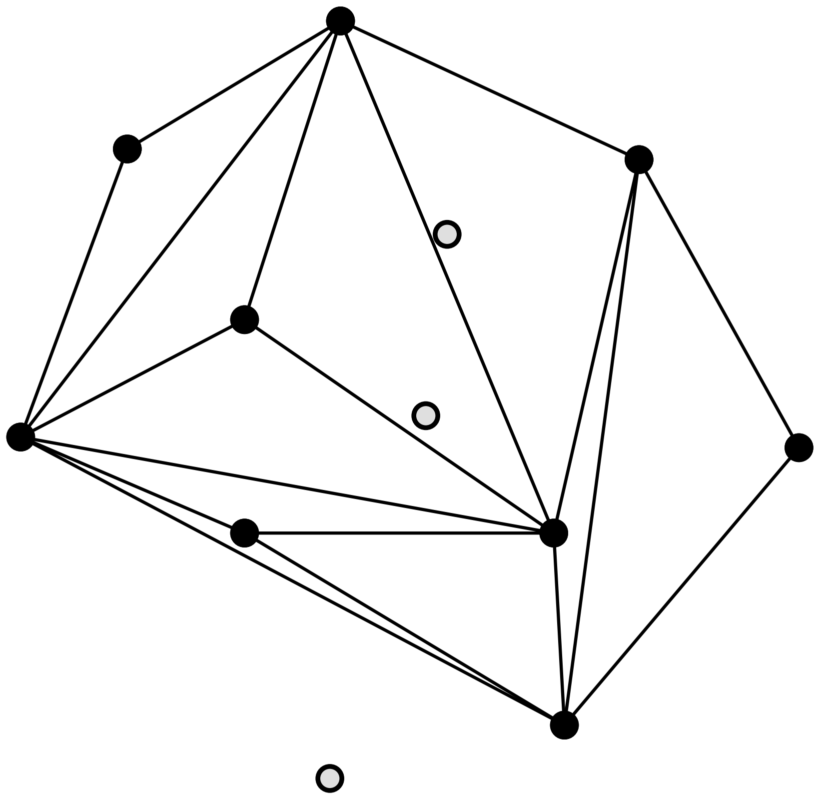
Quality of a triangulation

Incremental, without sorting



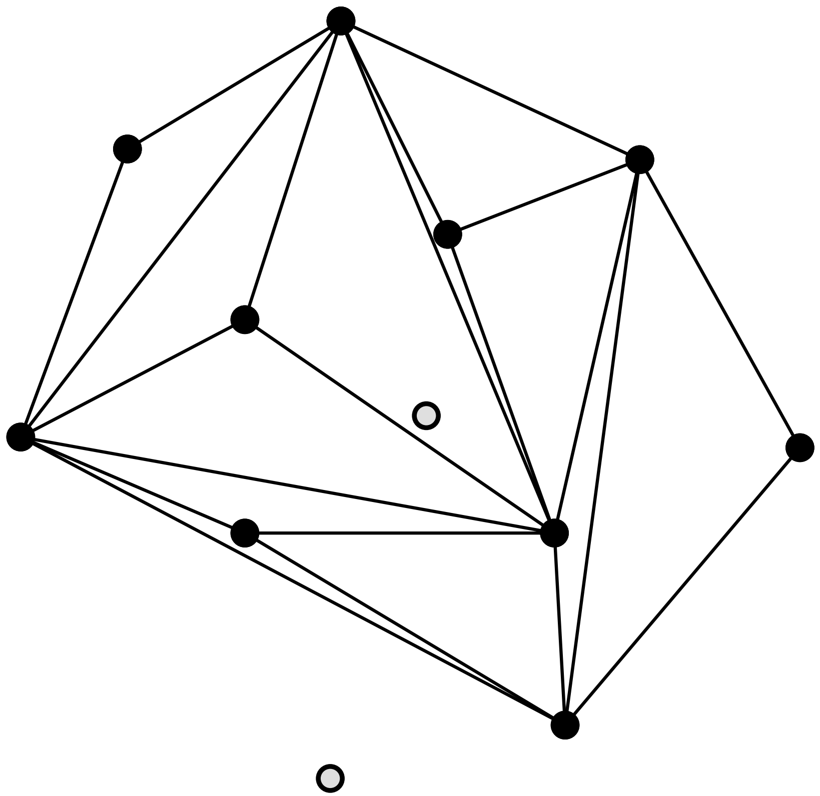
Quality of a triangulation

Incremental, without sorting



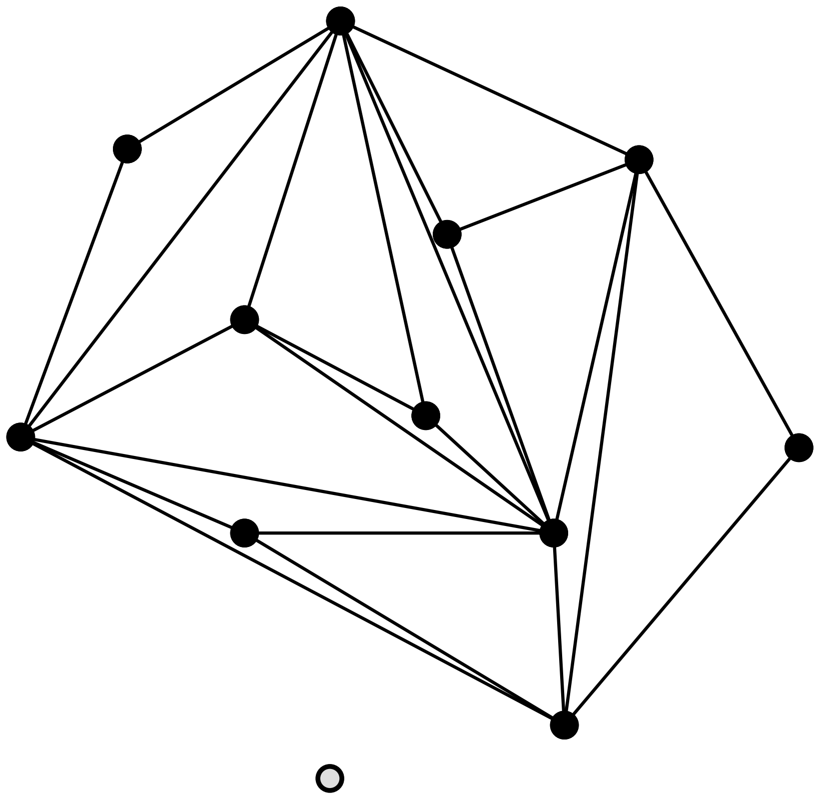
Quality of a triangulation

Incremental, without sorting



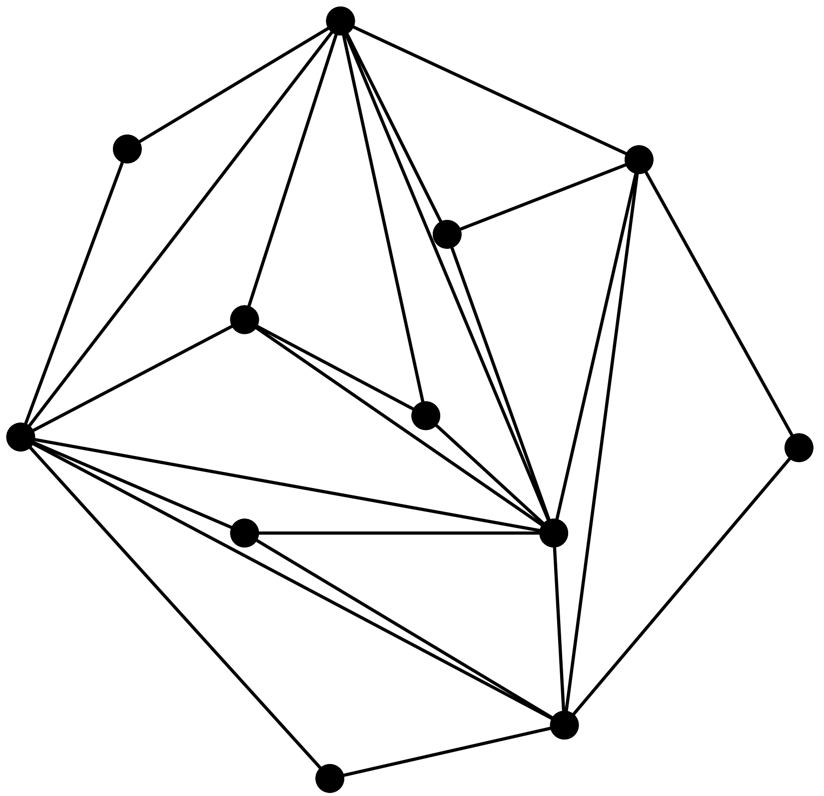
Quality of a triangulation

Incremental, without sorting



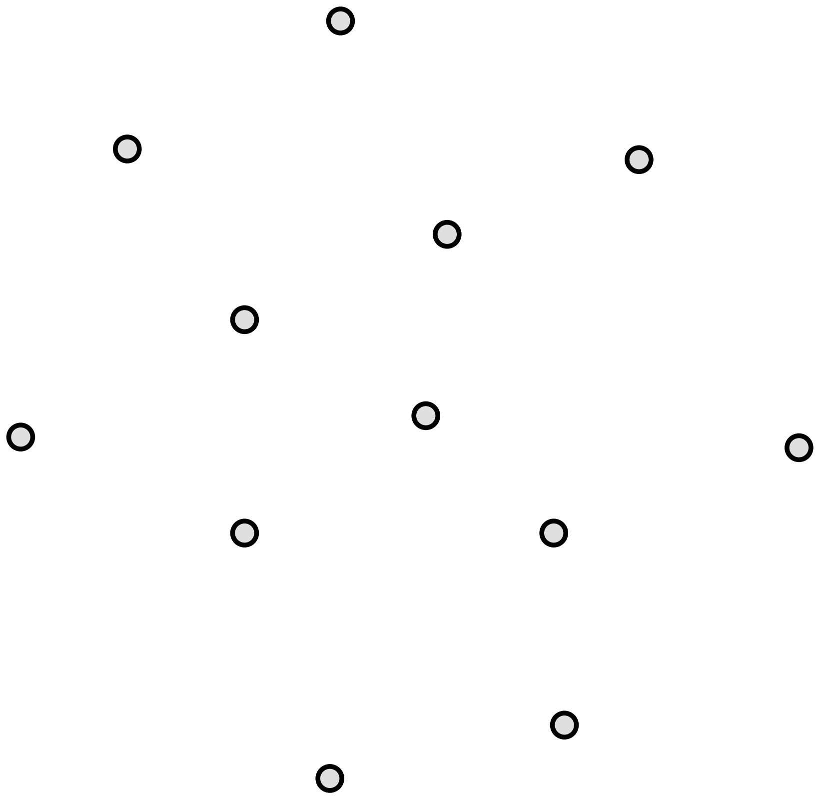
Quality of a triangulation

Incremental, without sorting



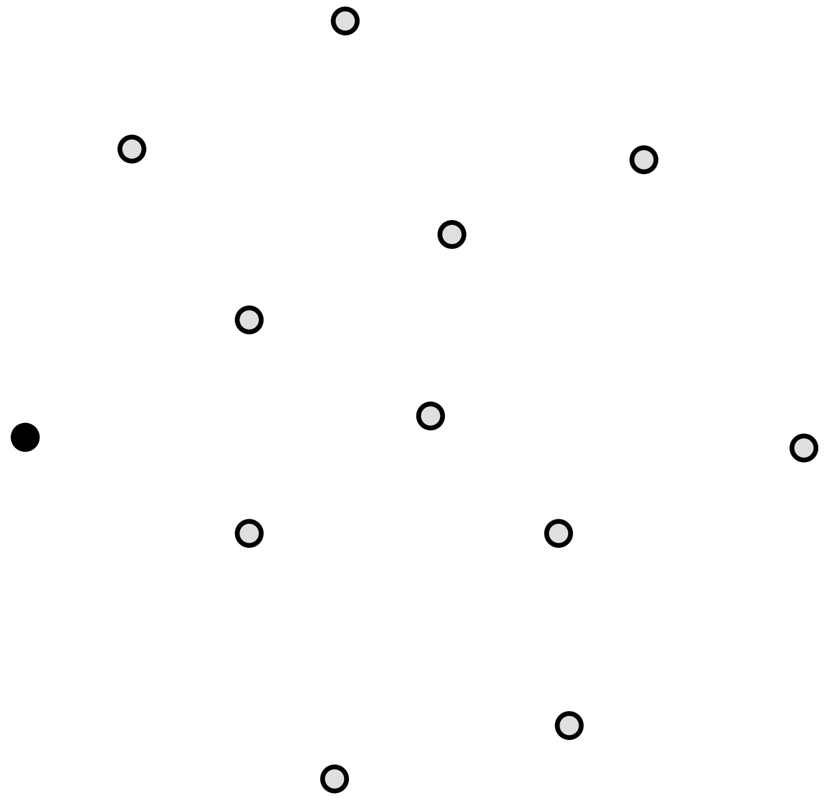
Quality of a triangulation

Incremental, sorting



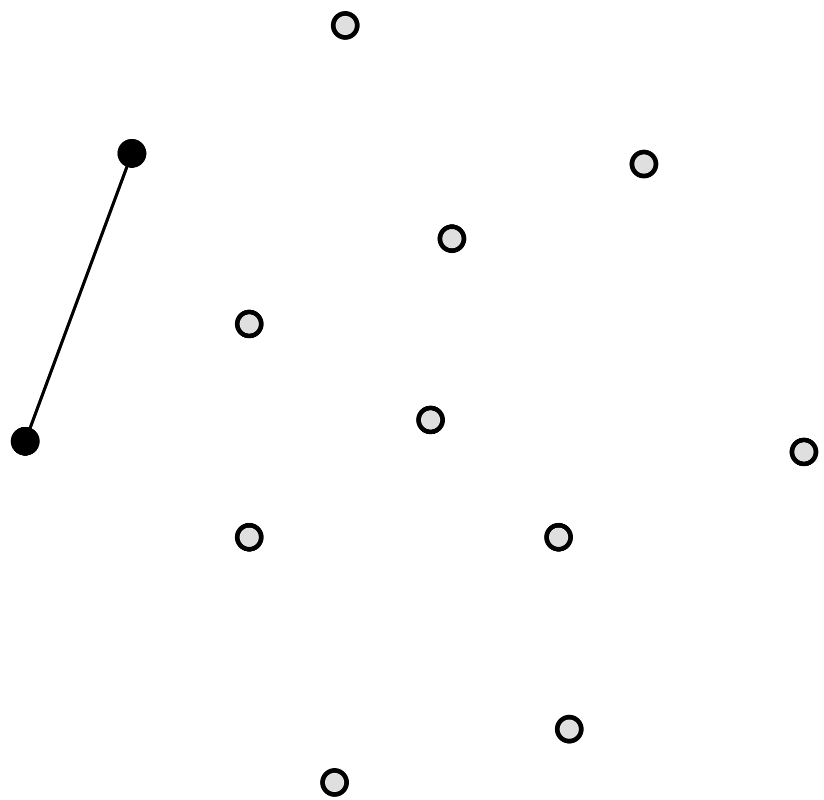
Quality of a triangulation

Incremental, sorting



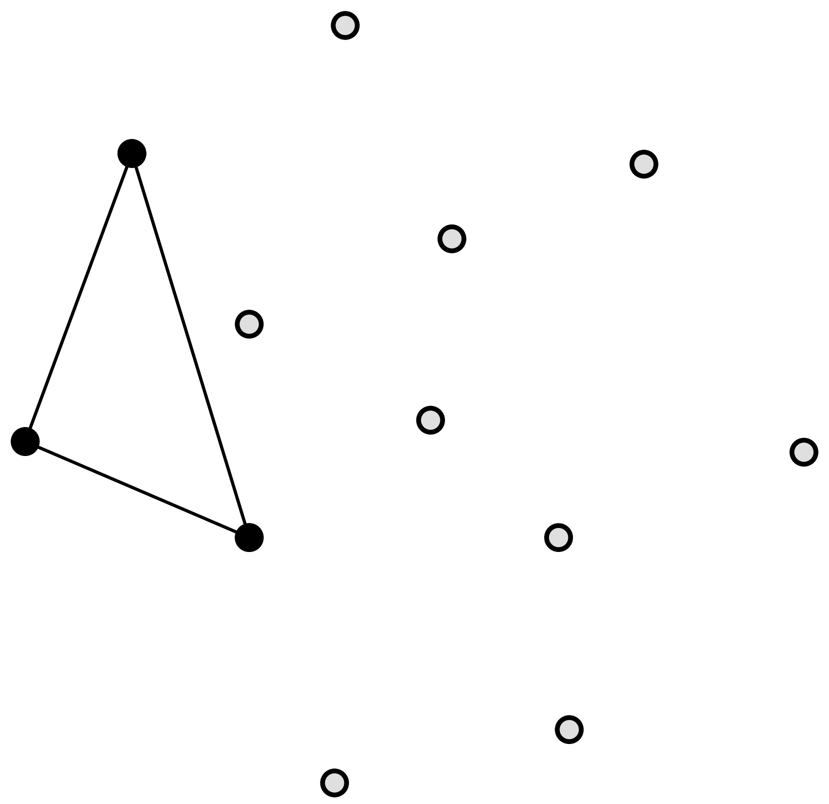
Quality of a triangulation

Incremental, sorting



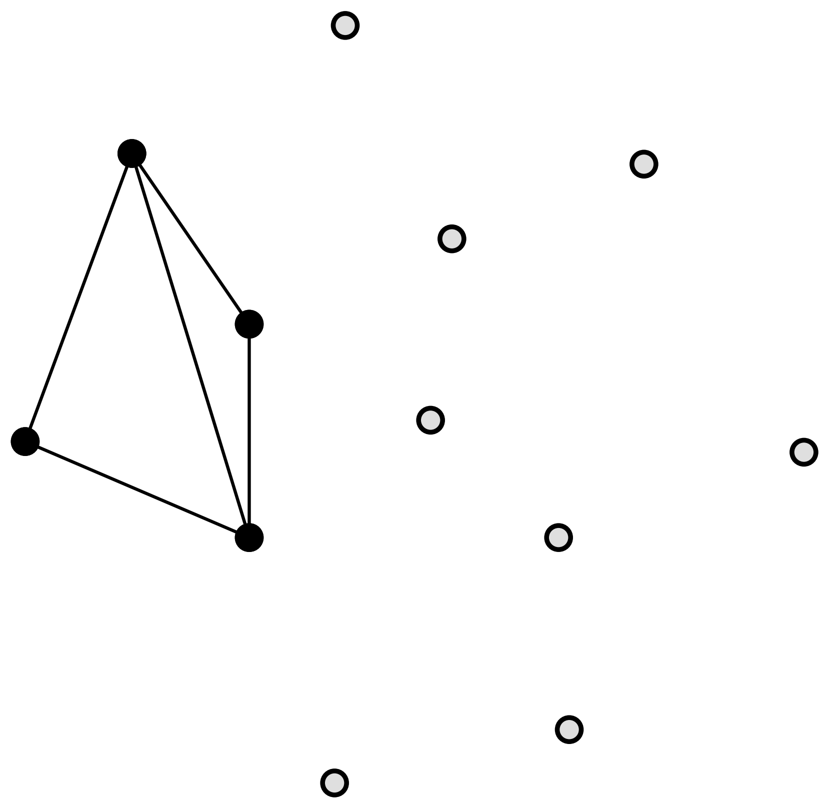
Quality of a triangulation

Incremental, sorting



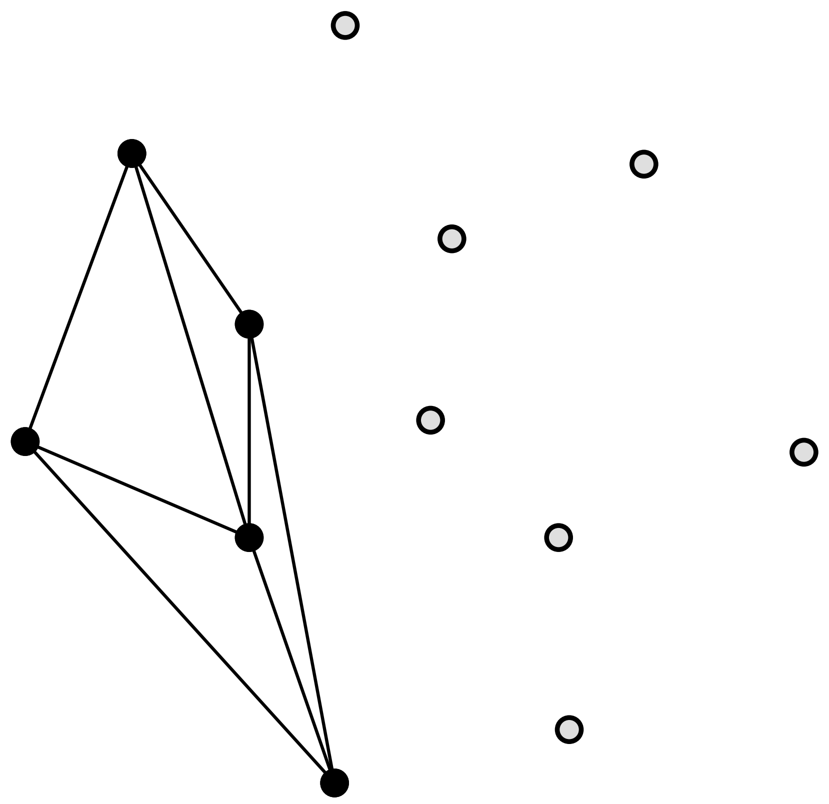
Quality of a triangulation

Incremental, sorting



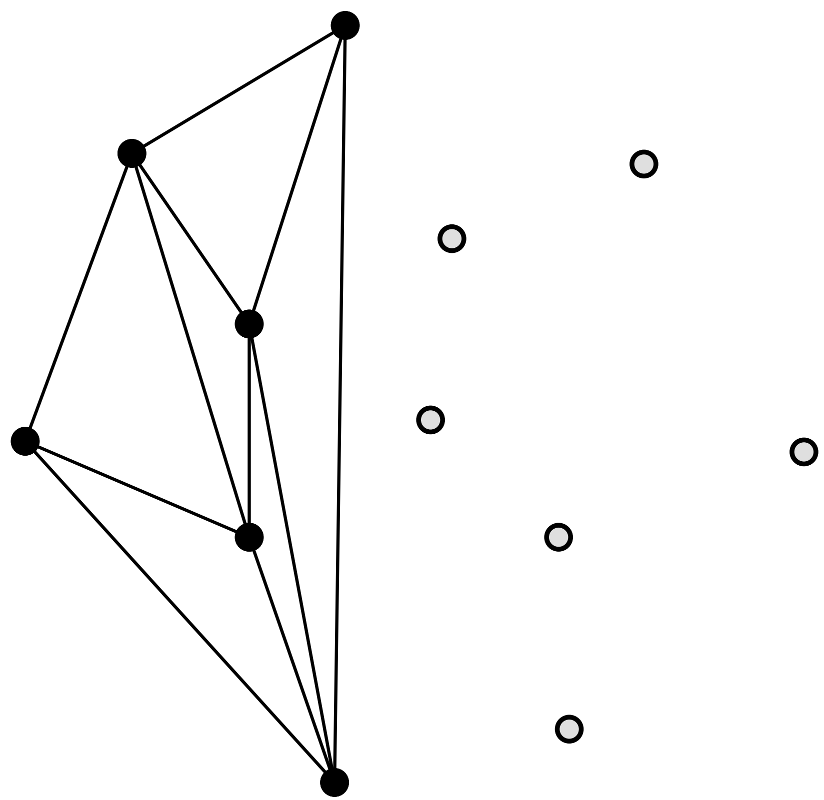
Quality of a triangulation

Incremental, sorting



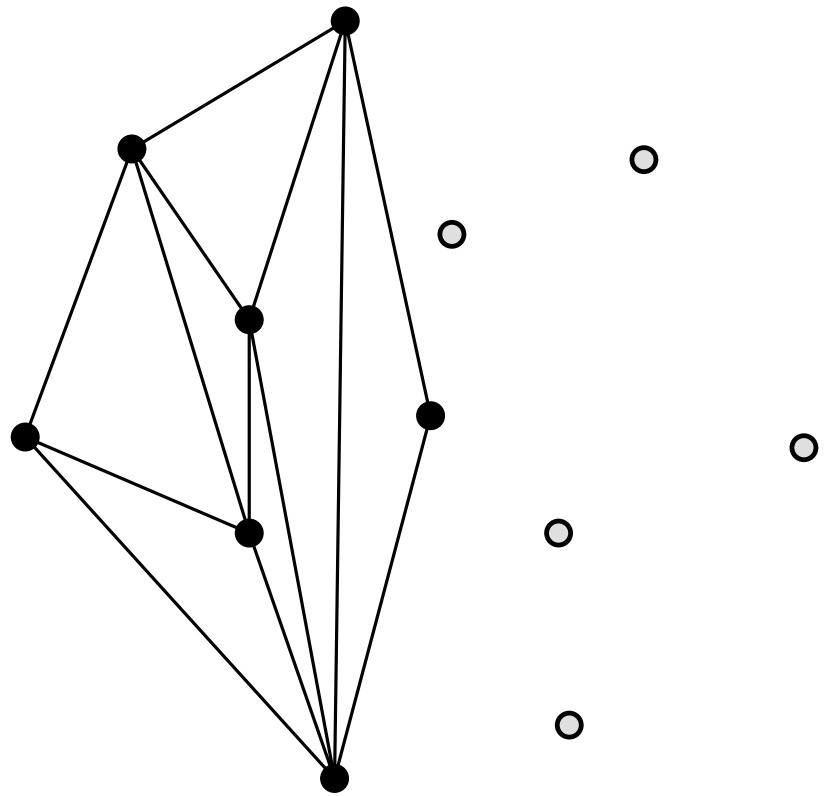
Quality of a triangulation

Incremental, sorting



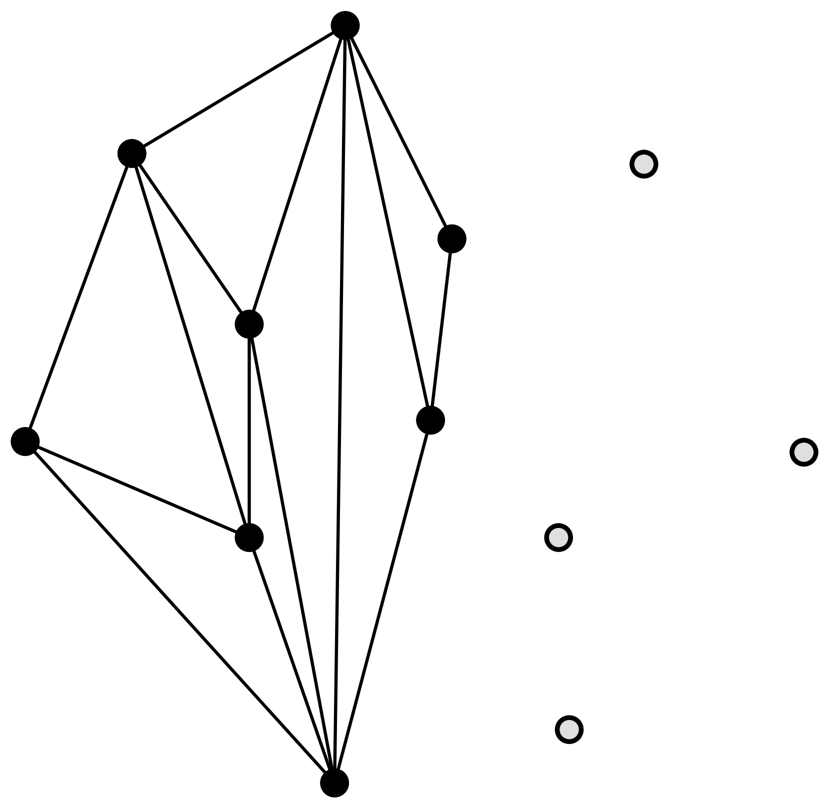
Quality of a triangulation

Incremental, sorting



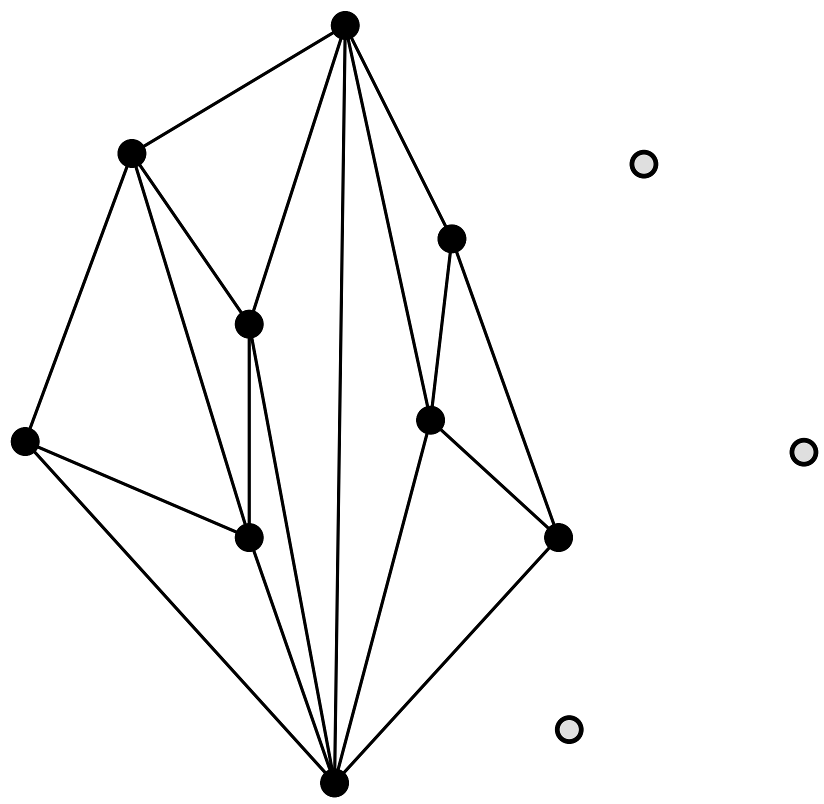
Quality of a triangulation

Incremental, sorting



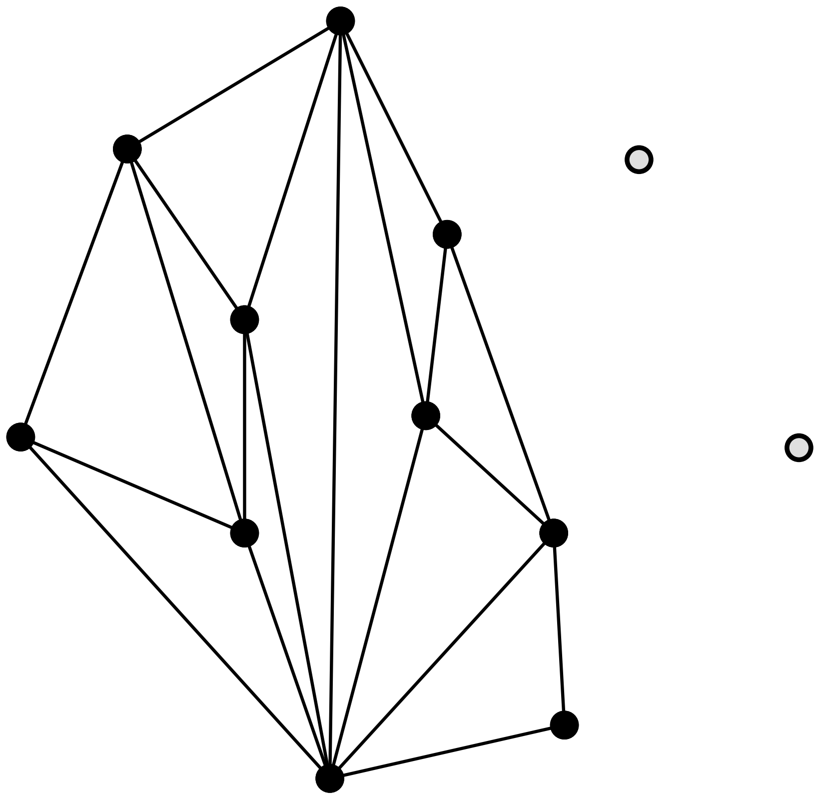
Quality of a triangulation

Incremental, sorting



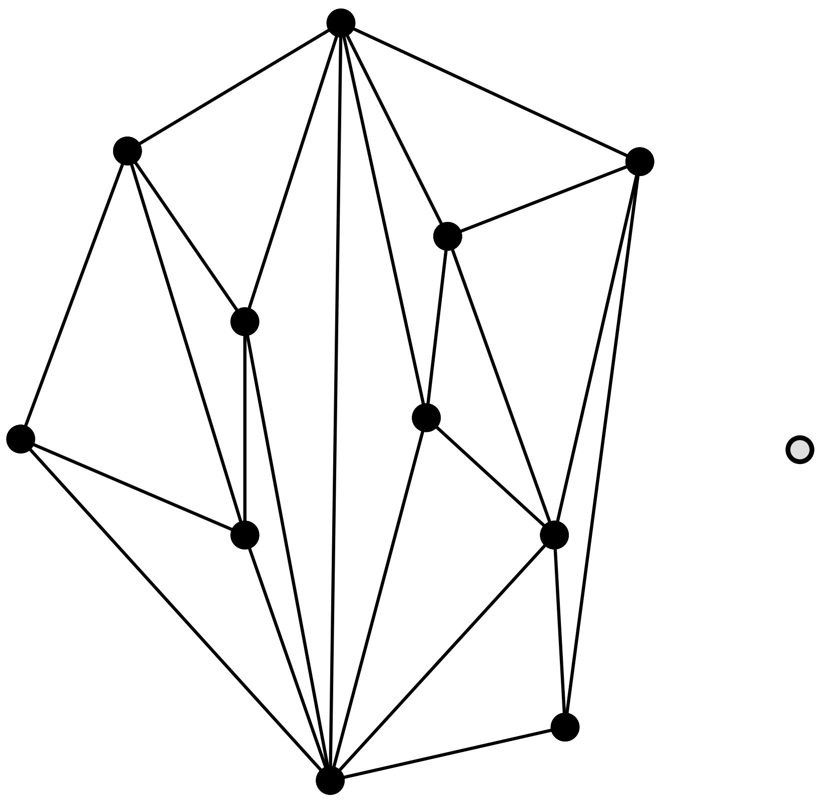
Quality of a triangulation

Incremental, sorting



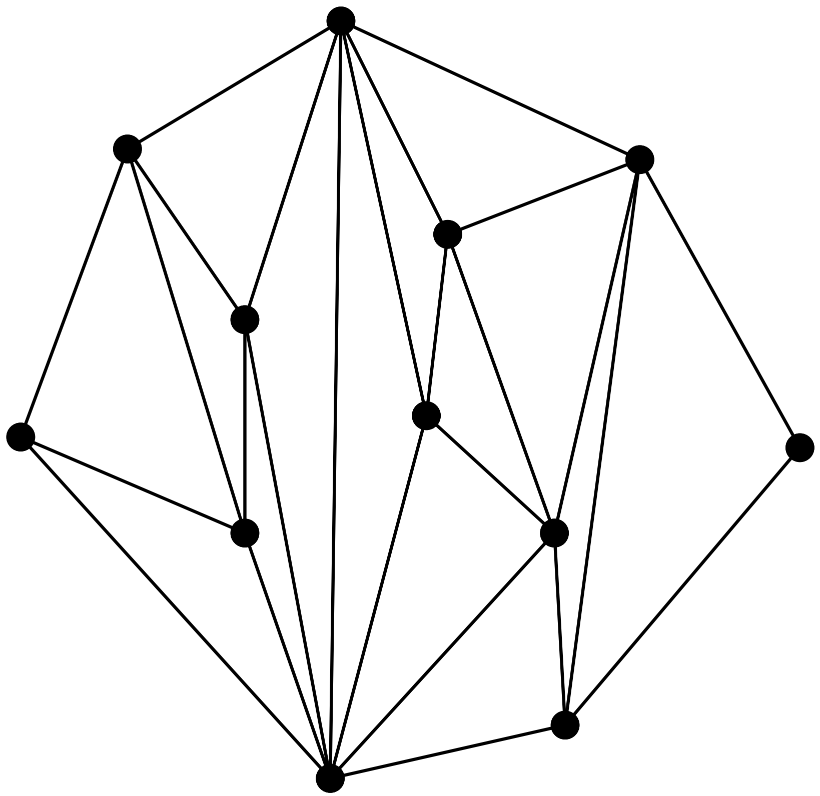
Quality of a triangulation

Incremental, sorting



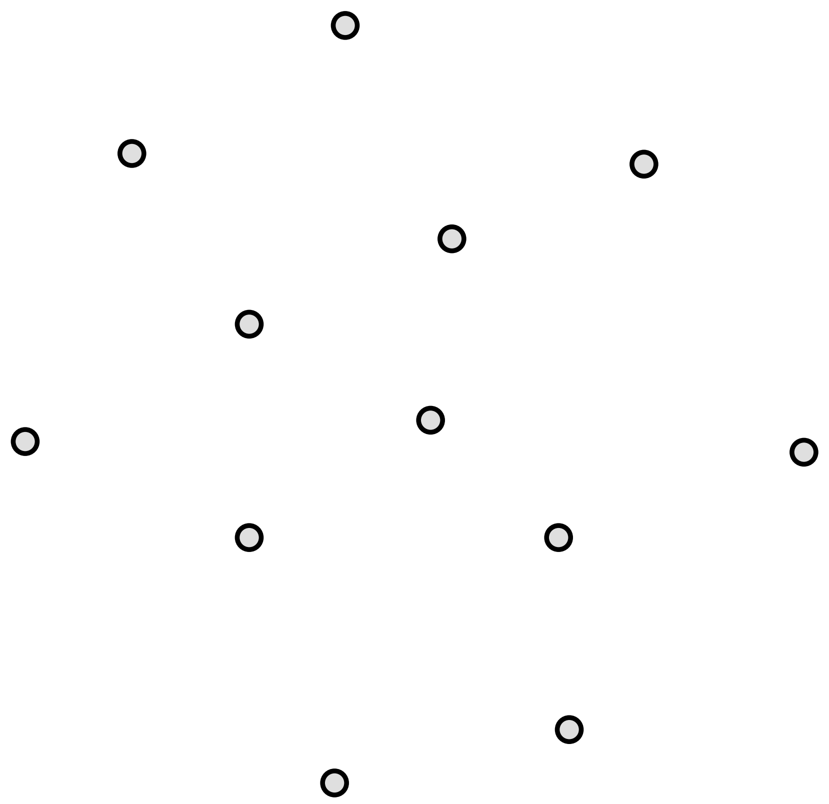
Quality of a triangulation

Incremental, sorting



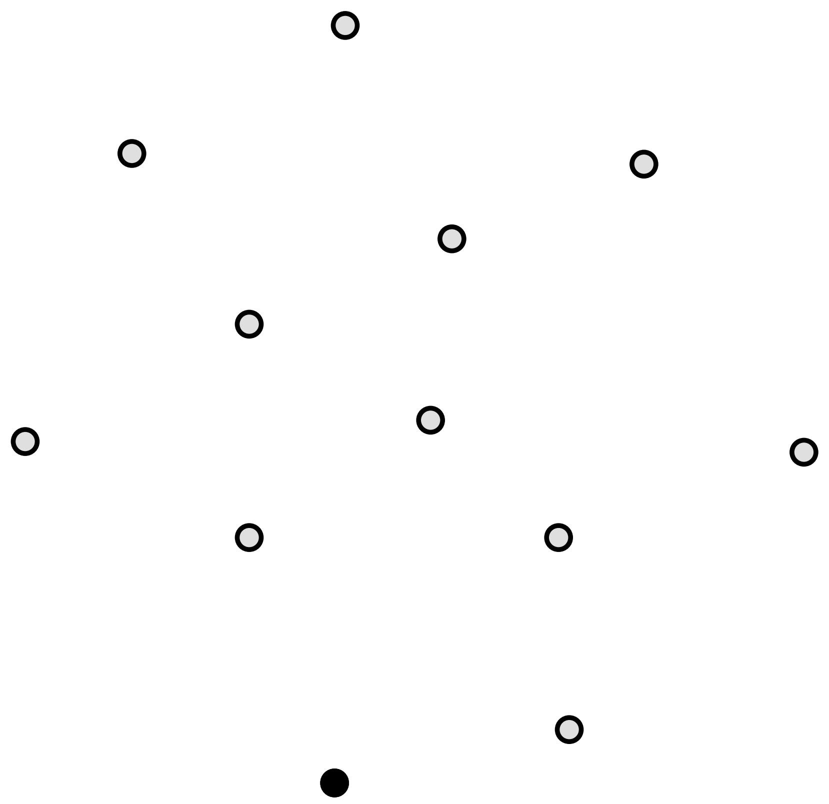
Quality of a triangulation

Graham's



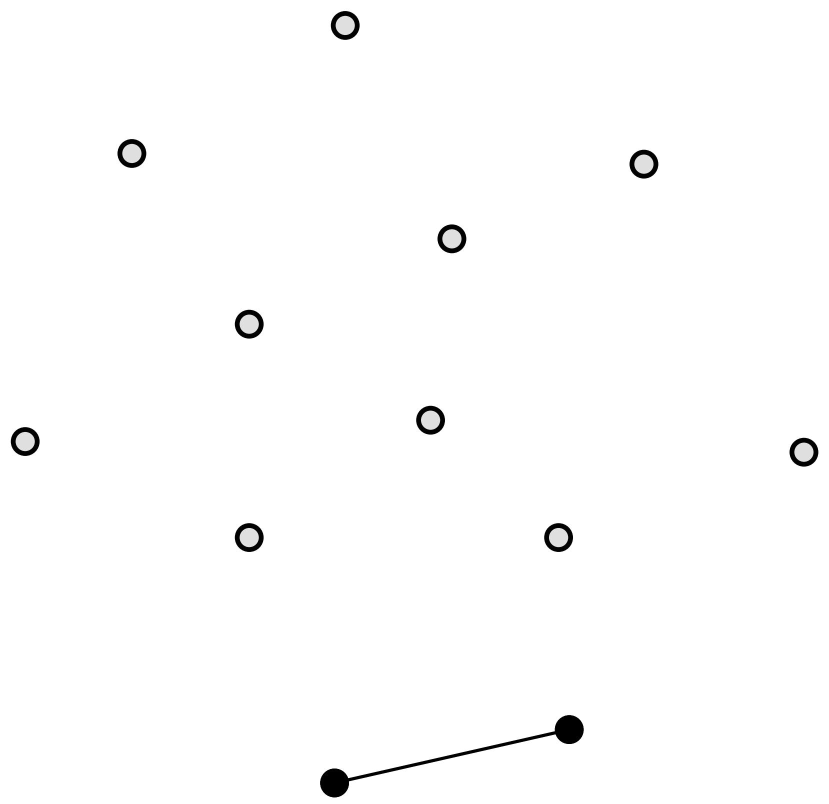
Quality of a triangulation

Graham's



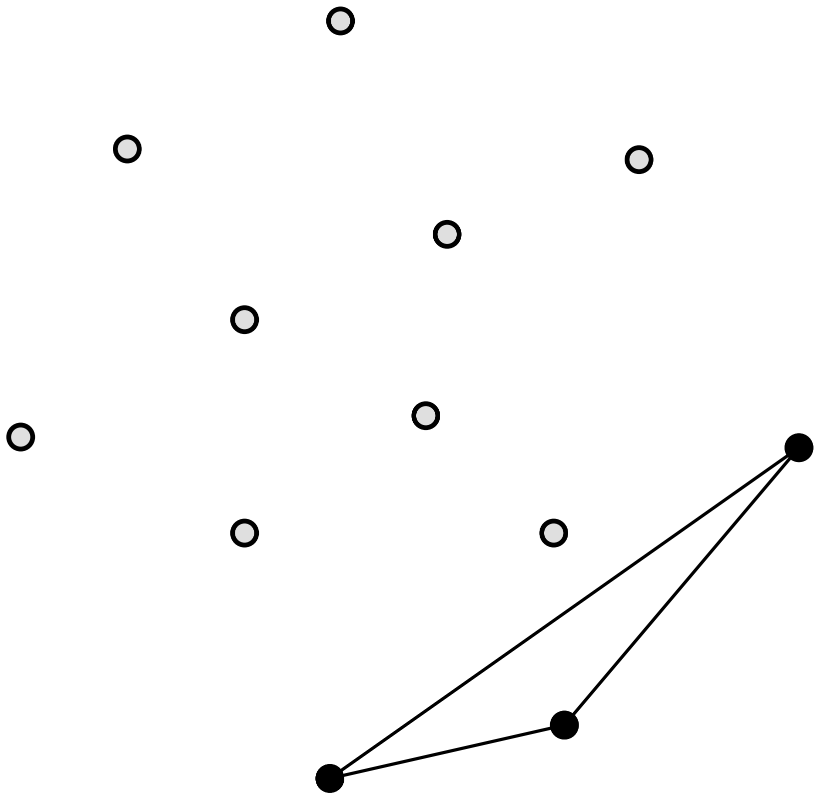
Quality of a triangulation

Graham's



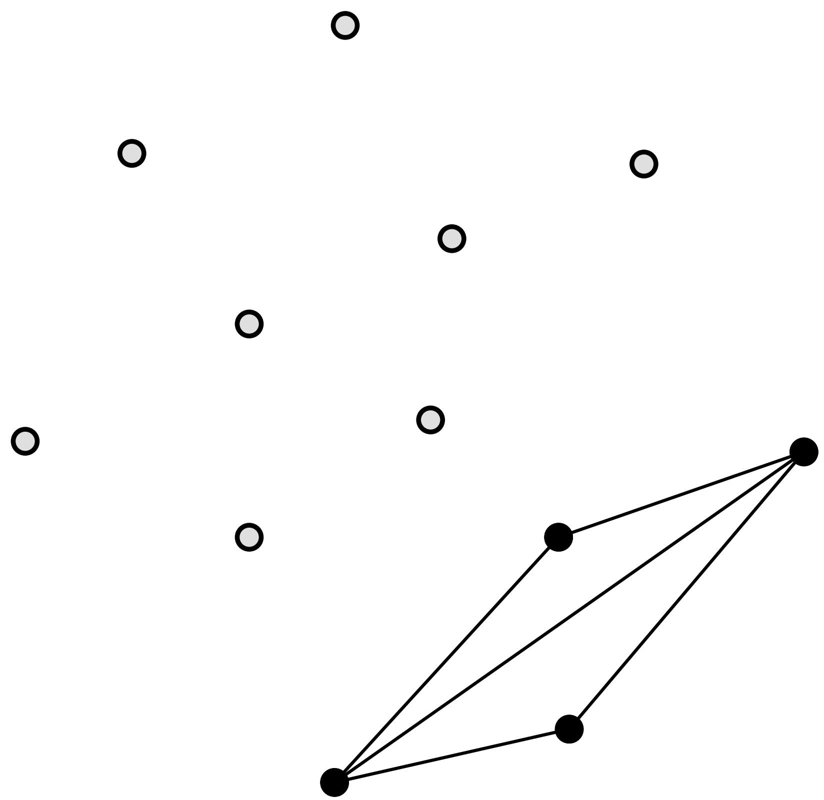
Quality of a triangulation

Graham's



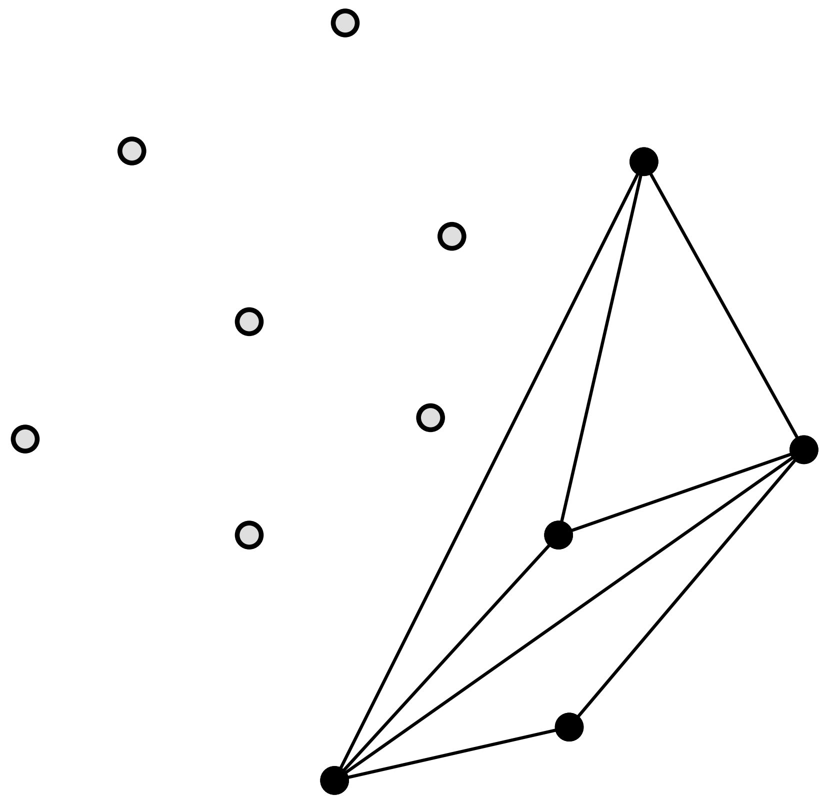
Quality of a triangulation

Graham's



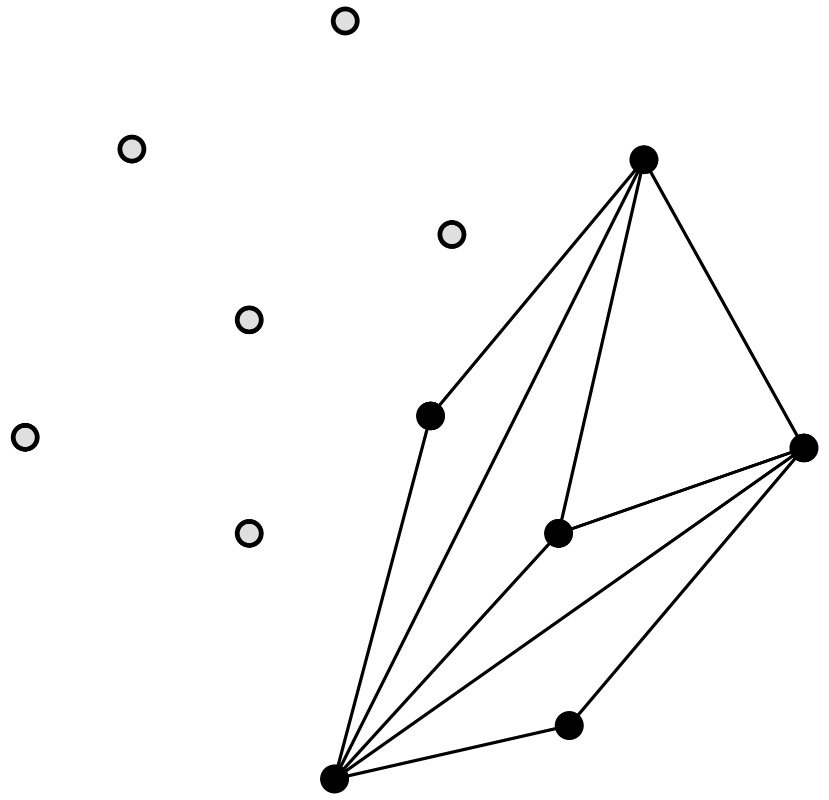
Quality of a triangulation

Graham's



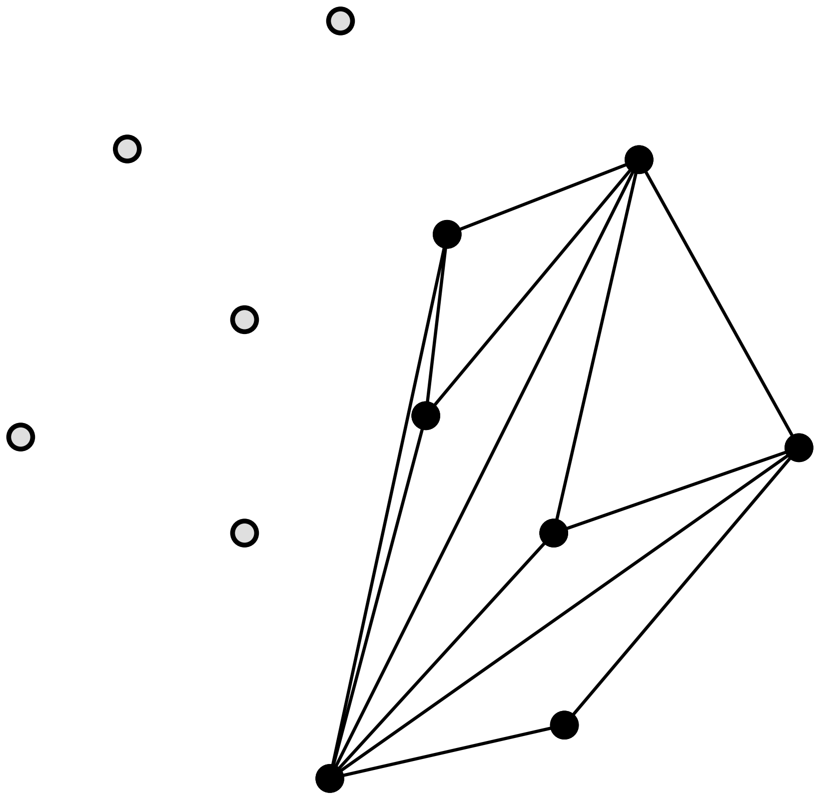
Quality of a triangulation

Graham's



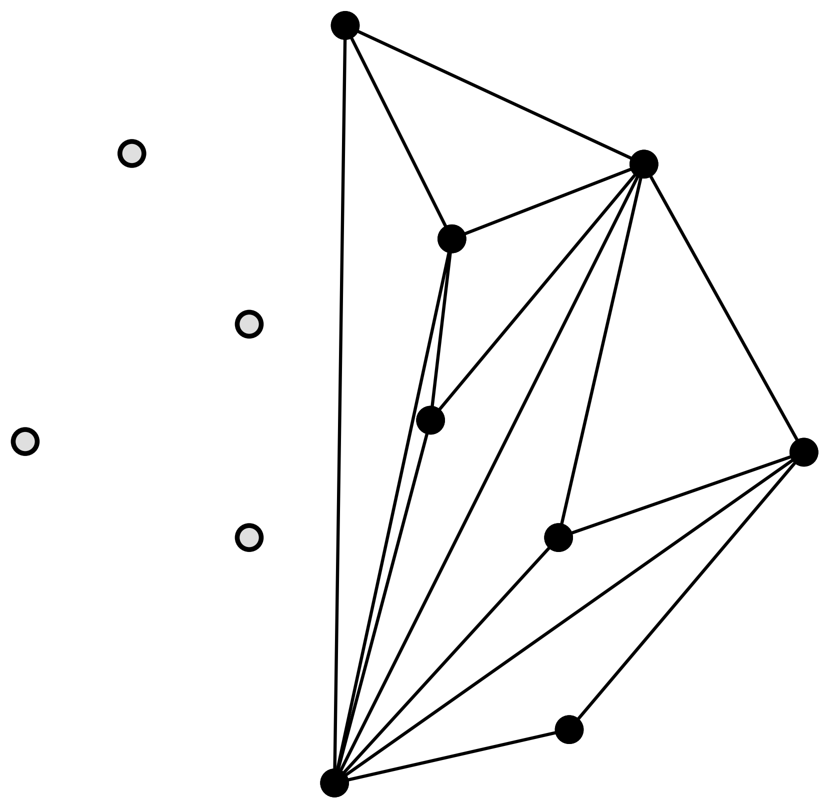
Quality of a triangulation

Graham's



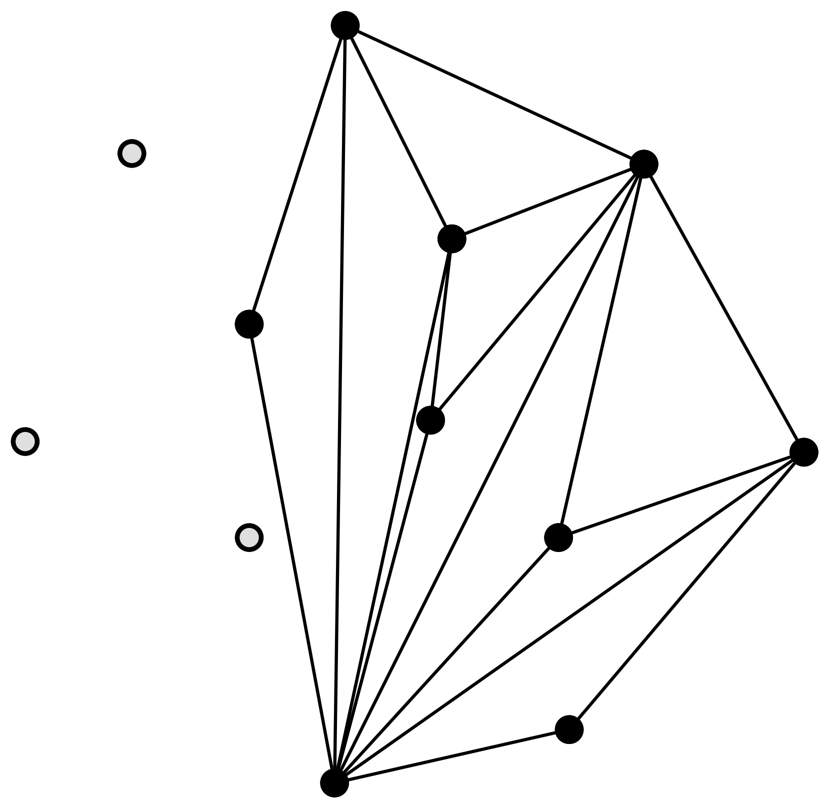
Quality of a triangulation

Graham's



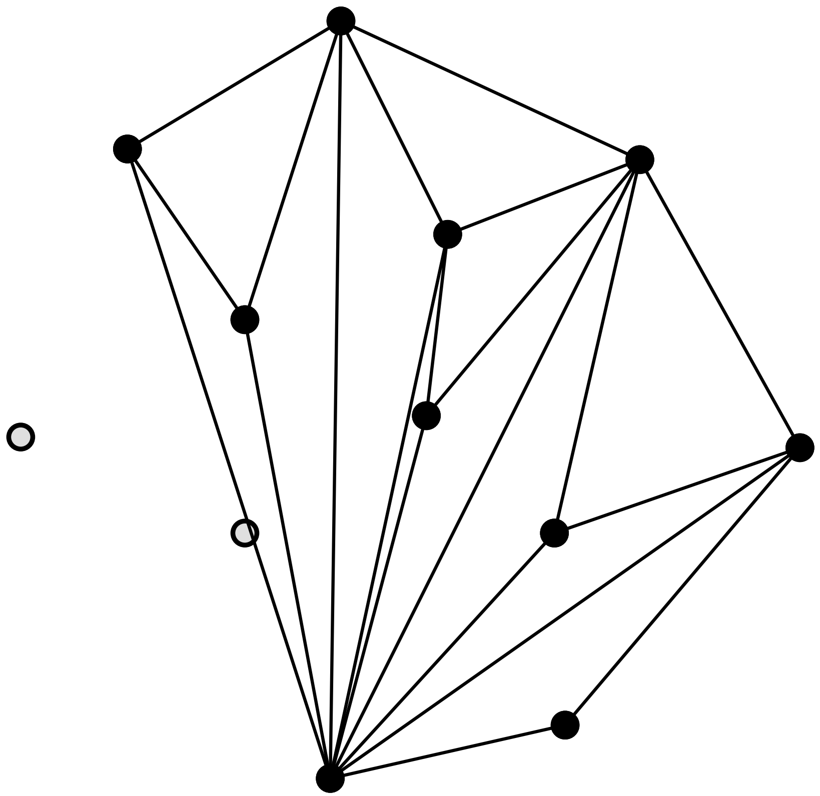
Quality of a triangulation

Graham's



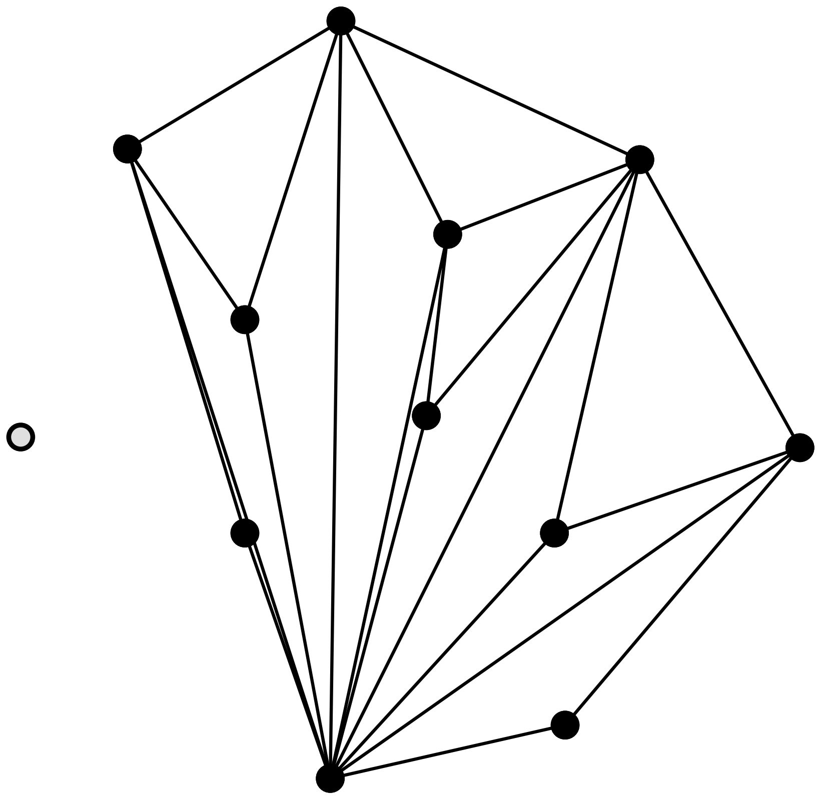
Quality of a triangulation

Graham's



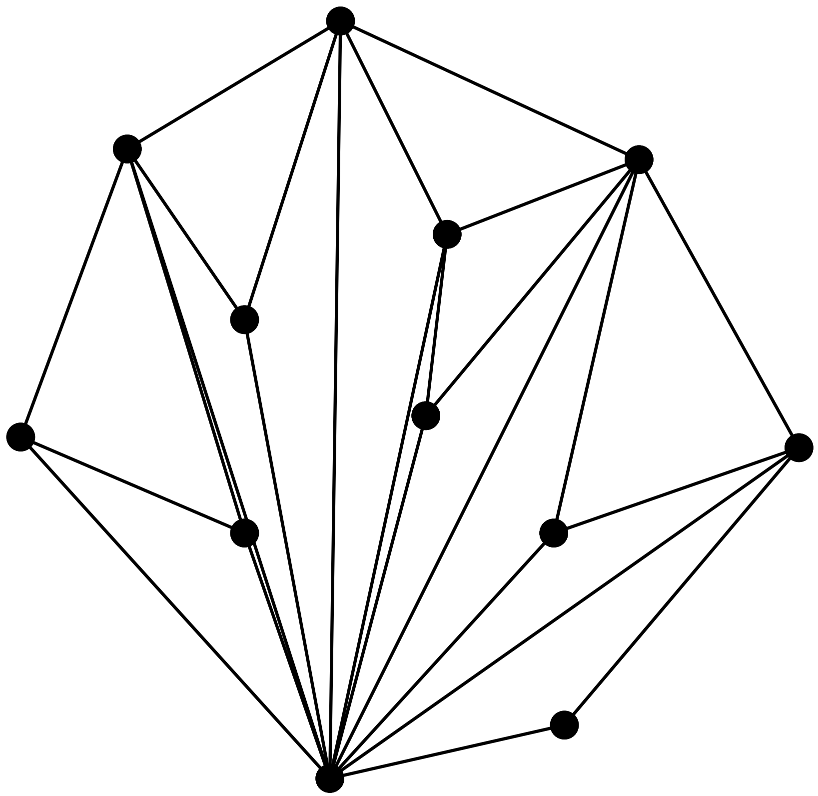
Quality of a triangulation

Graham's



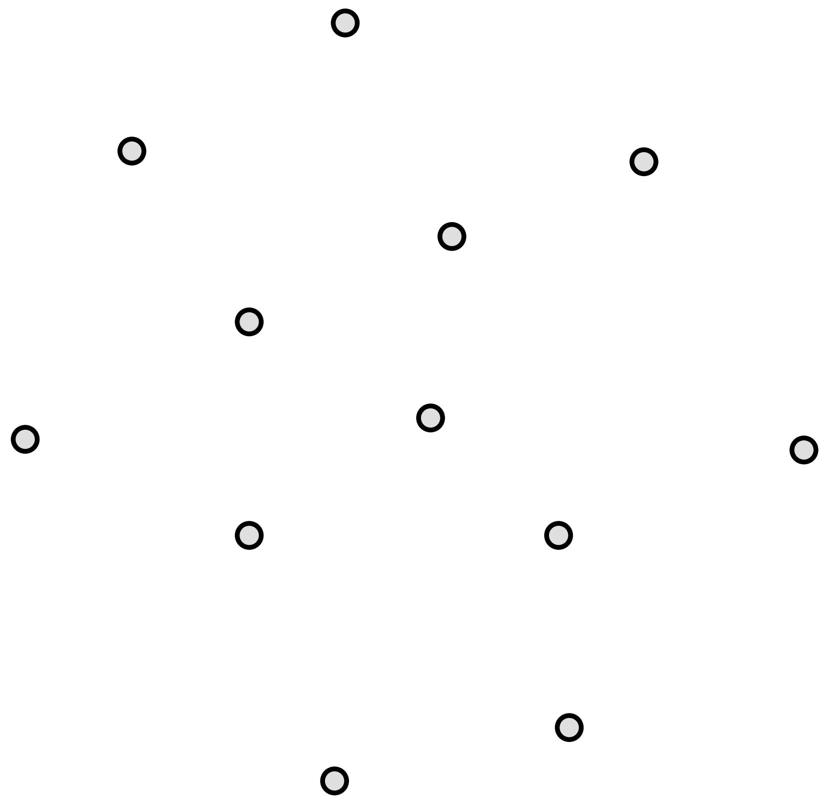
Quality of a triangulation

Graham's



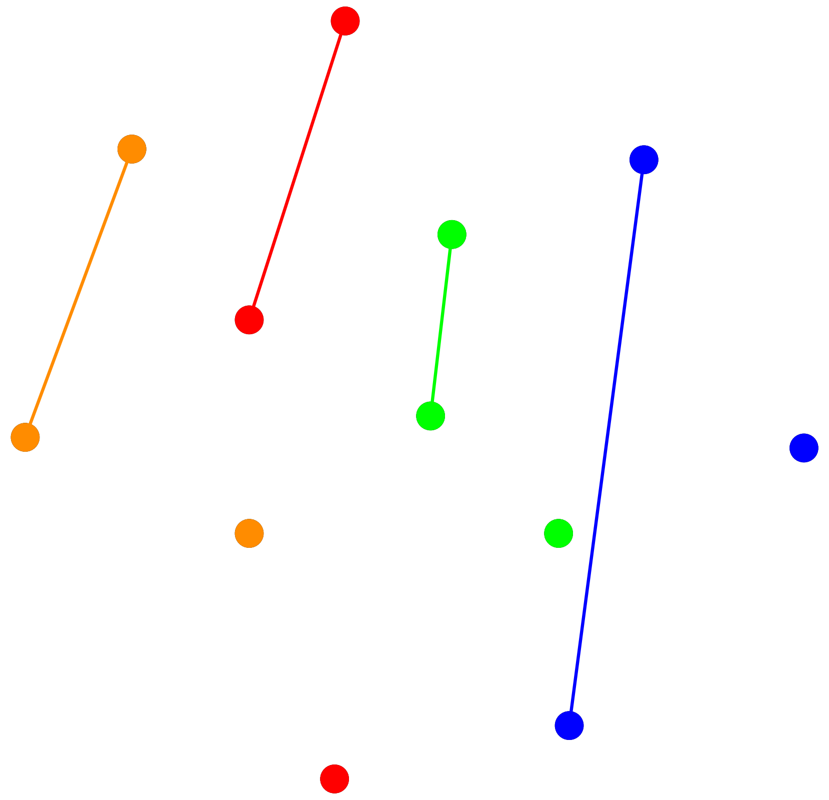
Quality of a triangulation

Divide and conquer



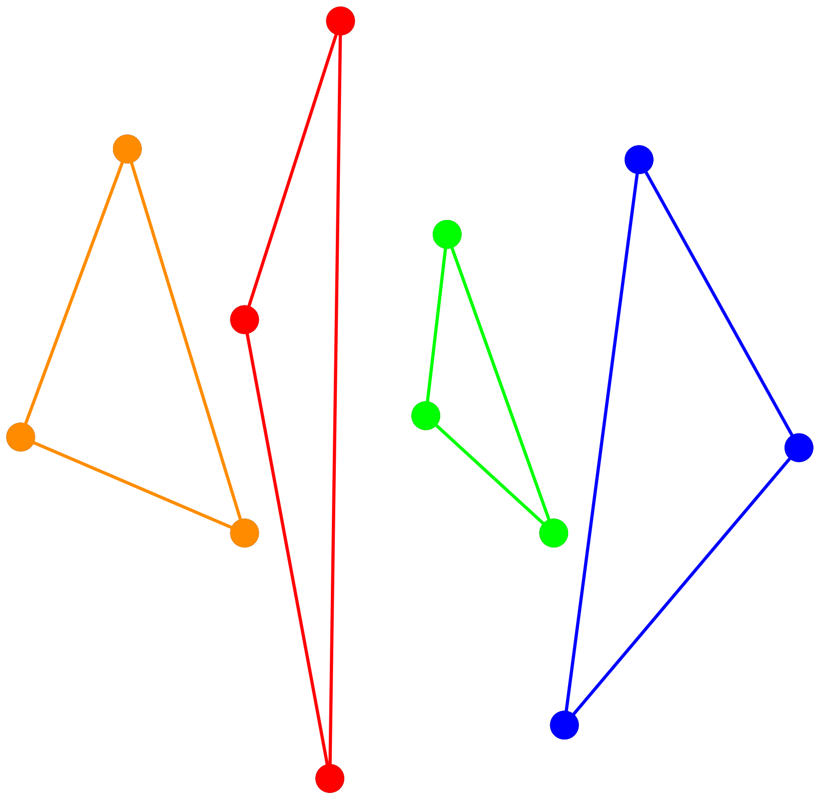
Quality of a triangulation

Divide and conquer



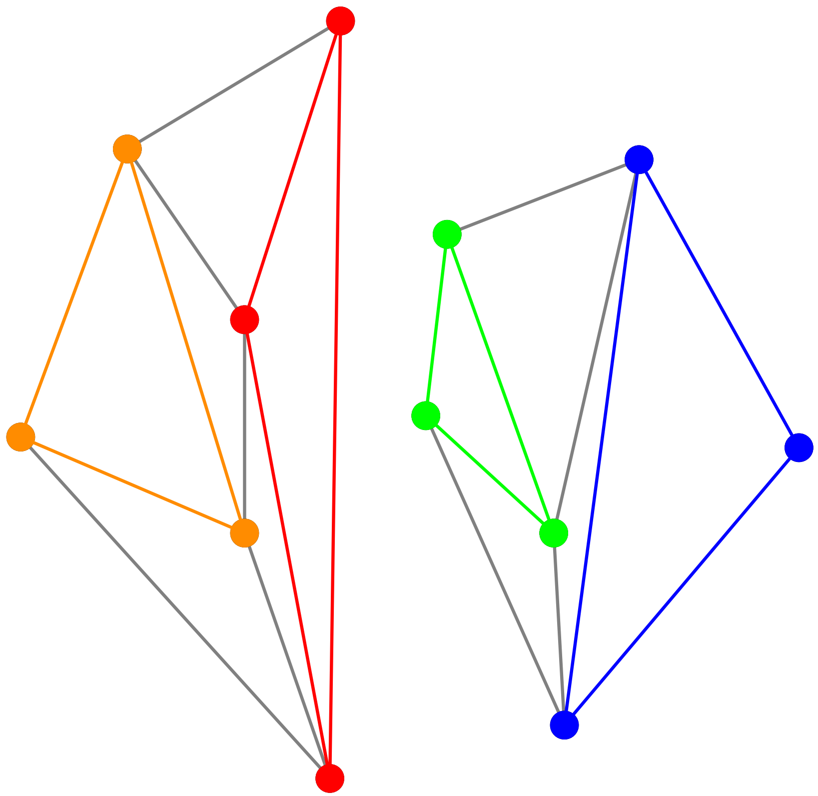
Quality of a triangulation

Divide and conquer



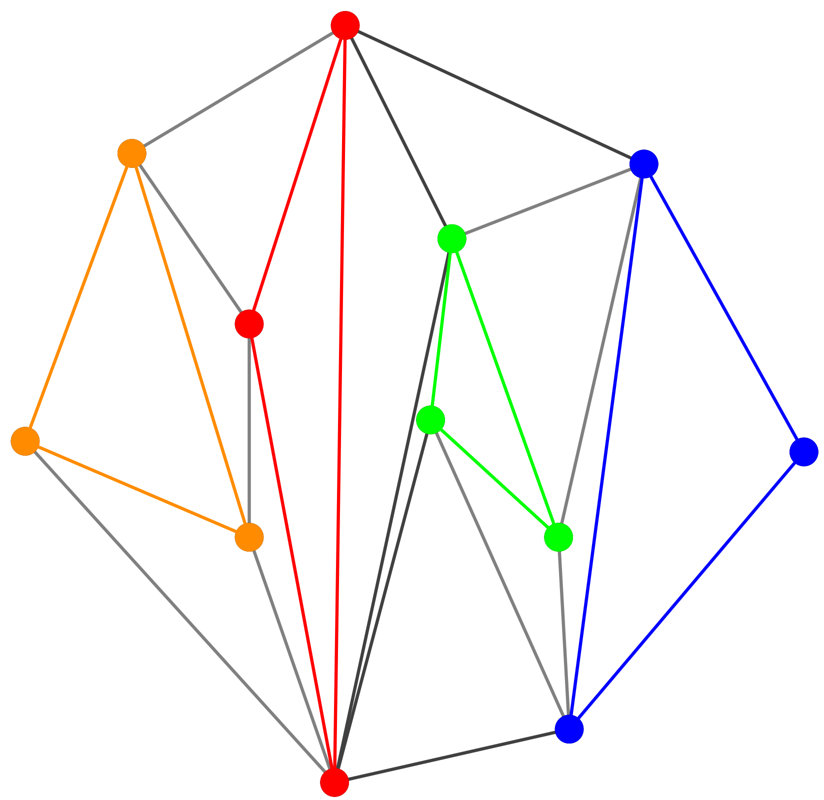
Quality of a triangulation

Divide and conquer



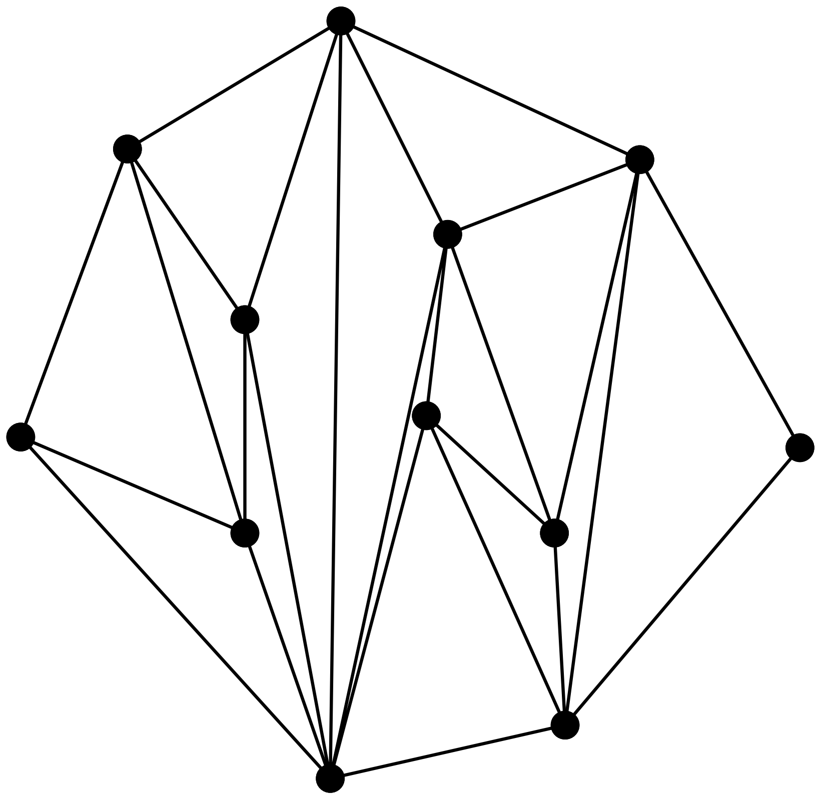
Quality of a triangulation

Divide and conquer



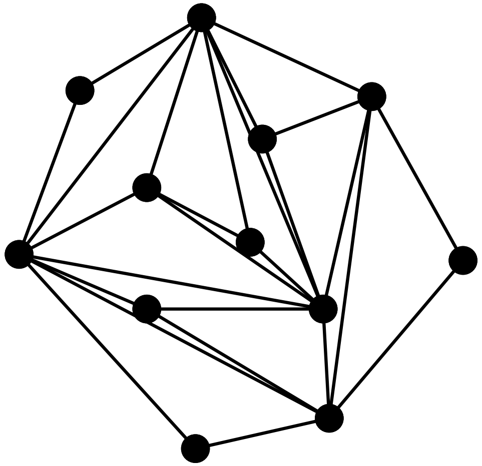
Quality of a triangulation

Divide and conquer

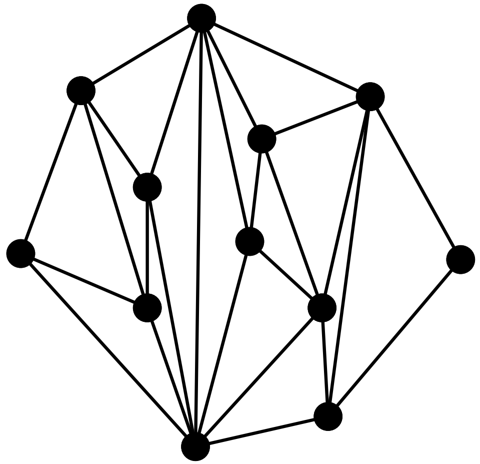


TRIANGULATING POINT SETS

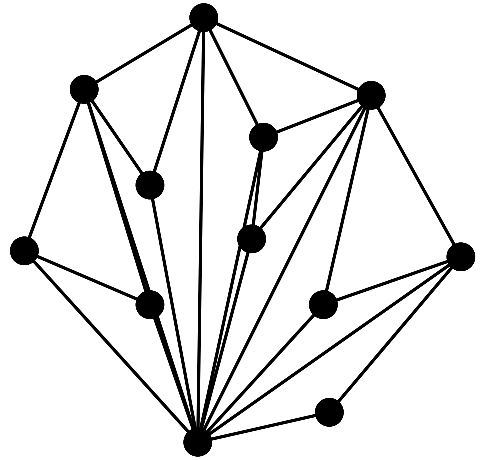
Quality of a triangulation



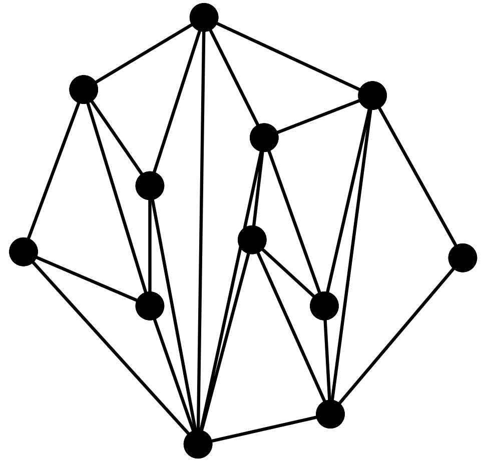
Incremental, unsorted



Incremental, sorted



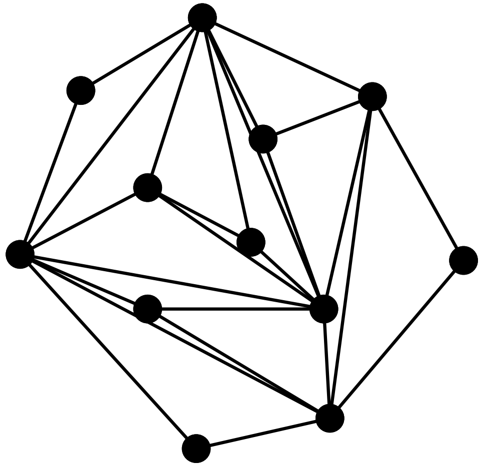
Graham's scan



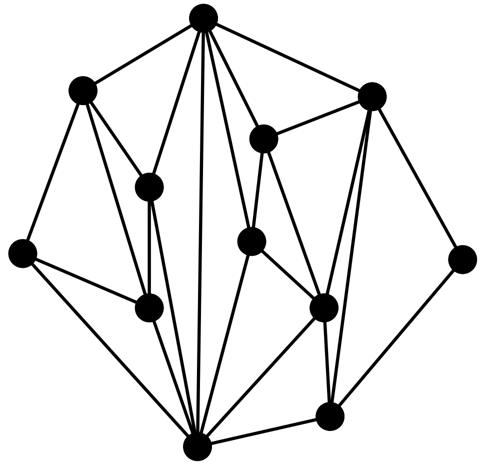
Divide and conquer

TRIANGULATING POINT SETS

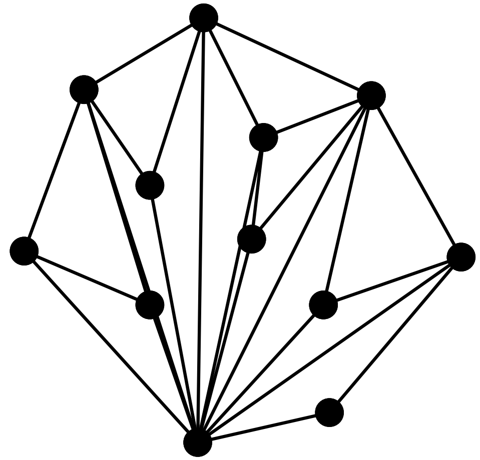
Quality of a triangulation



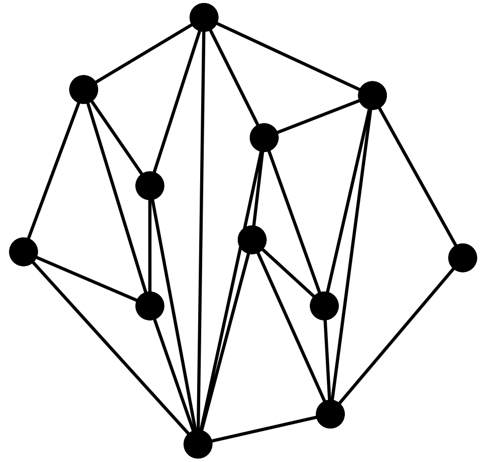
Incremental, unsorted



Incremental, sorted



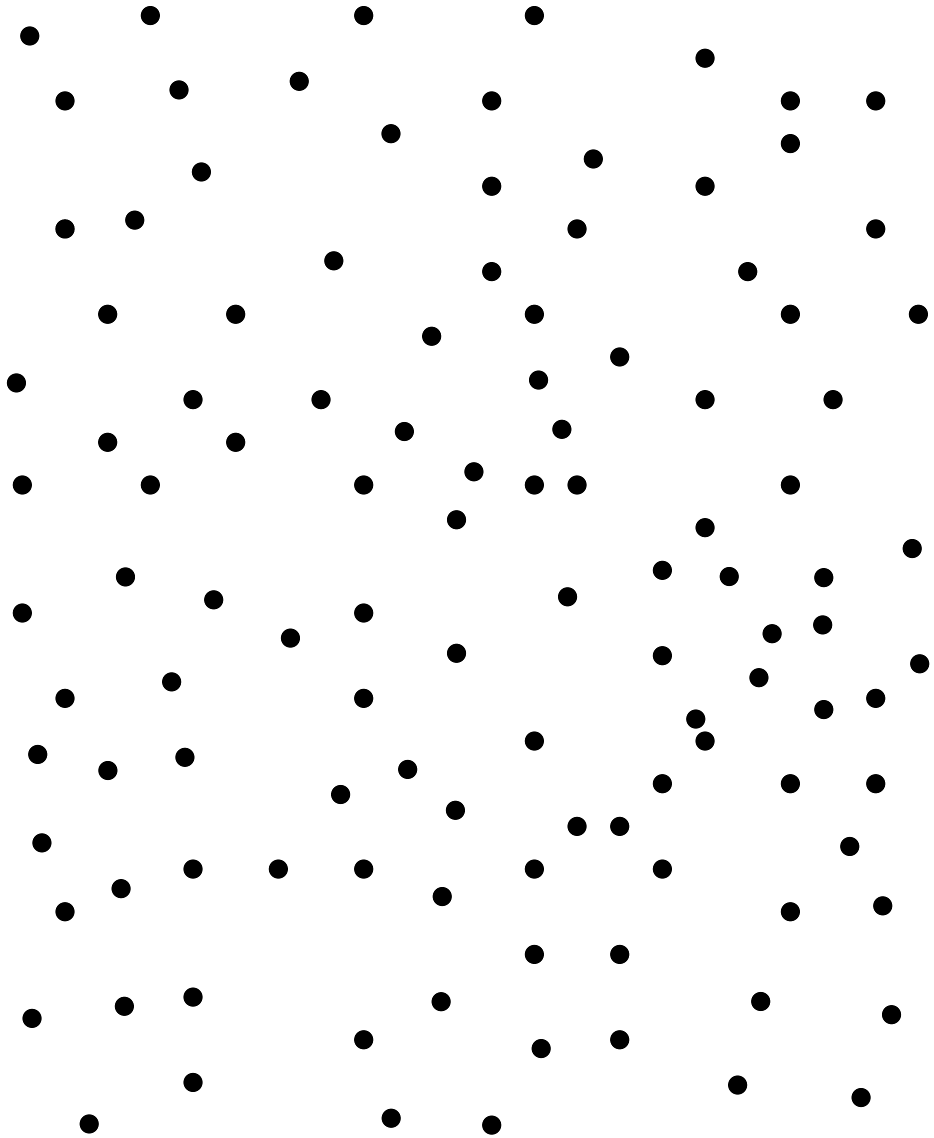
Graham's scan



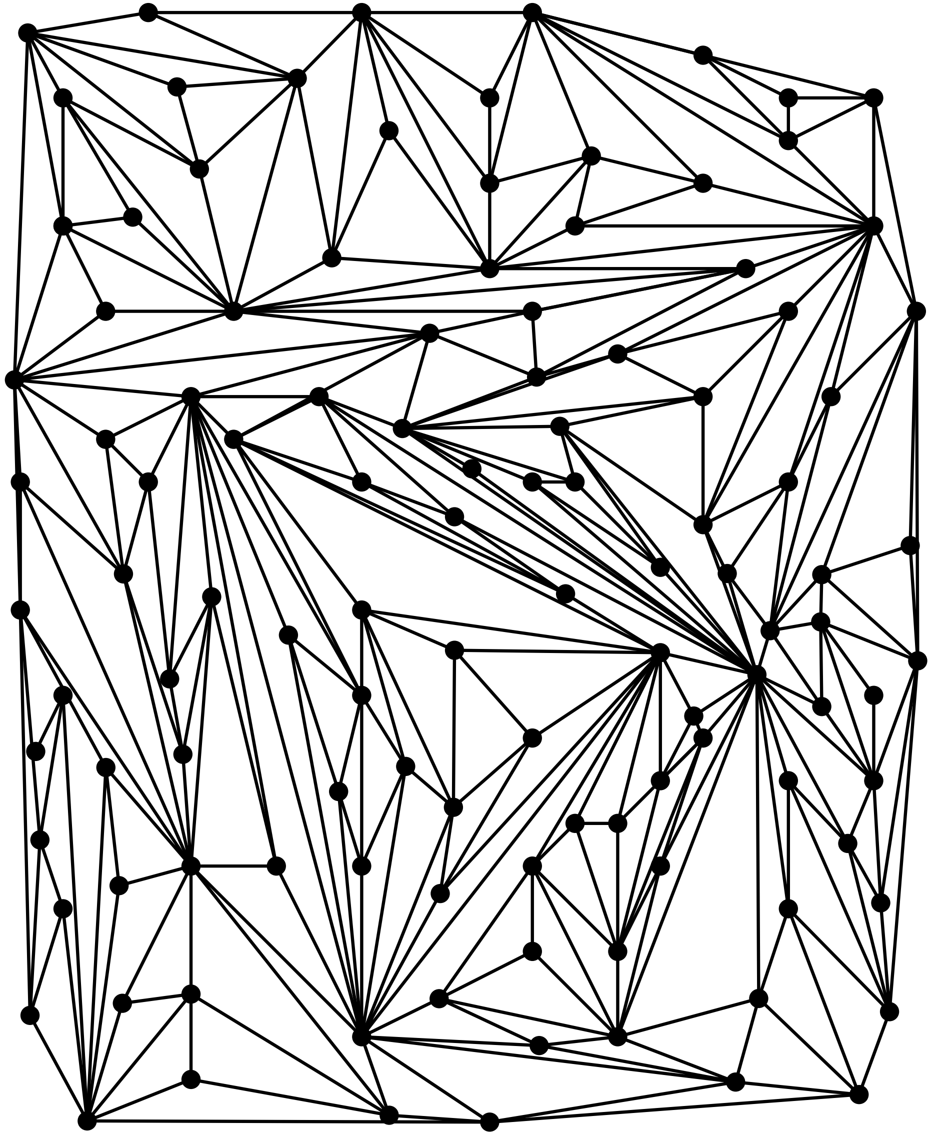
Divide and conquer

Not *all* triangulations are the same

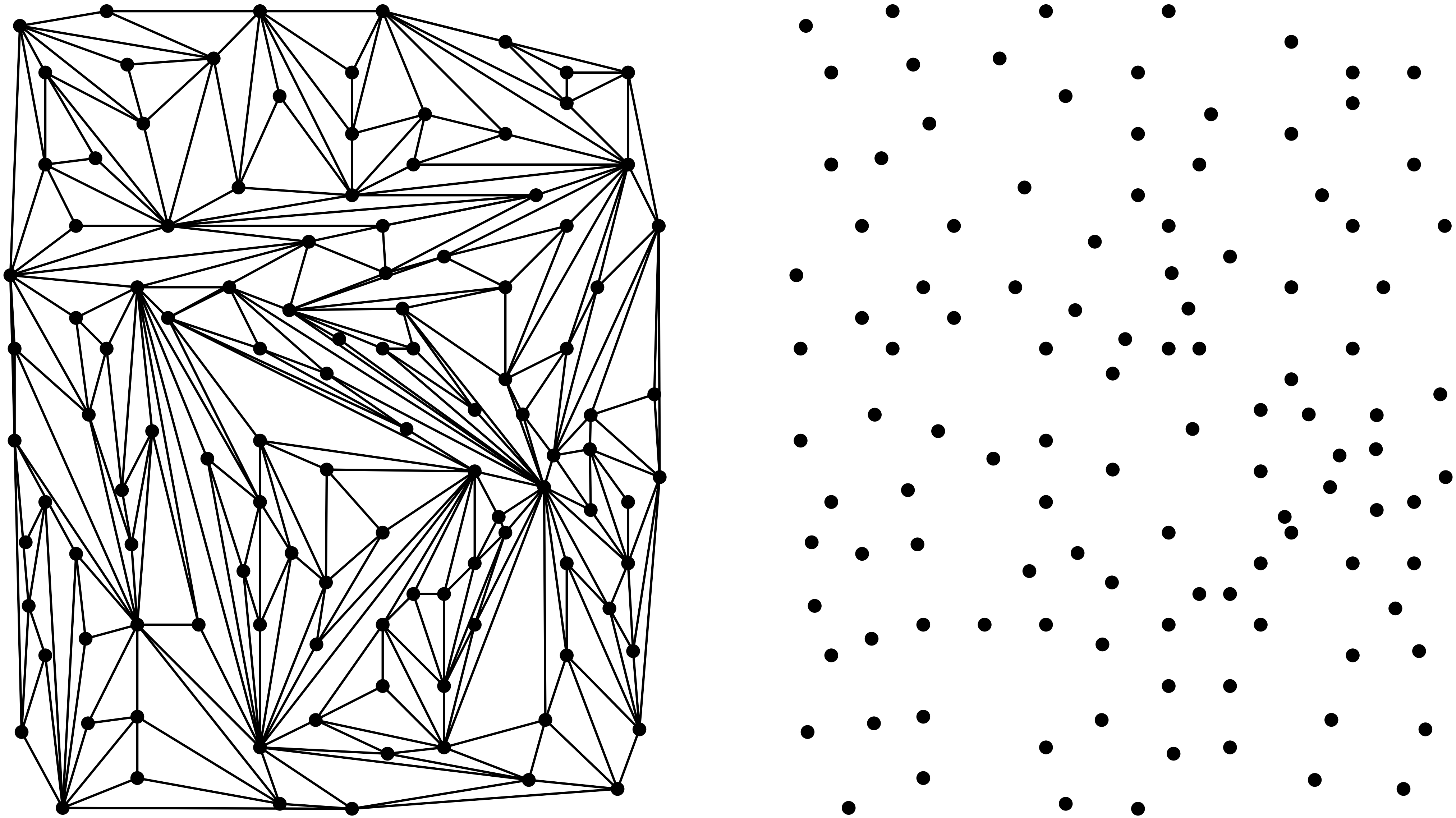
TRIANGULATING POINT SETS



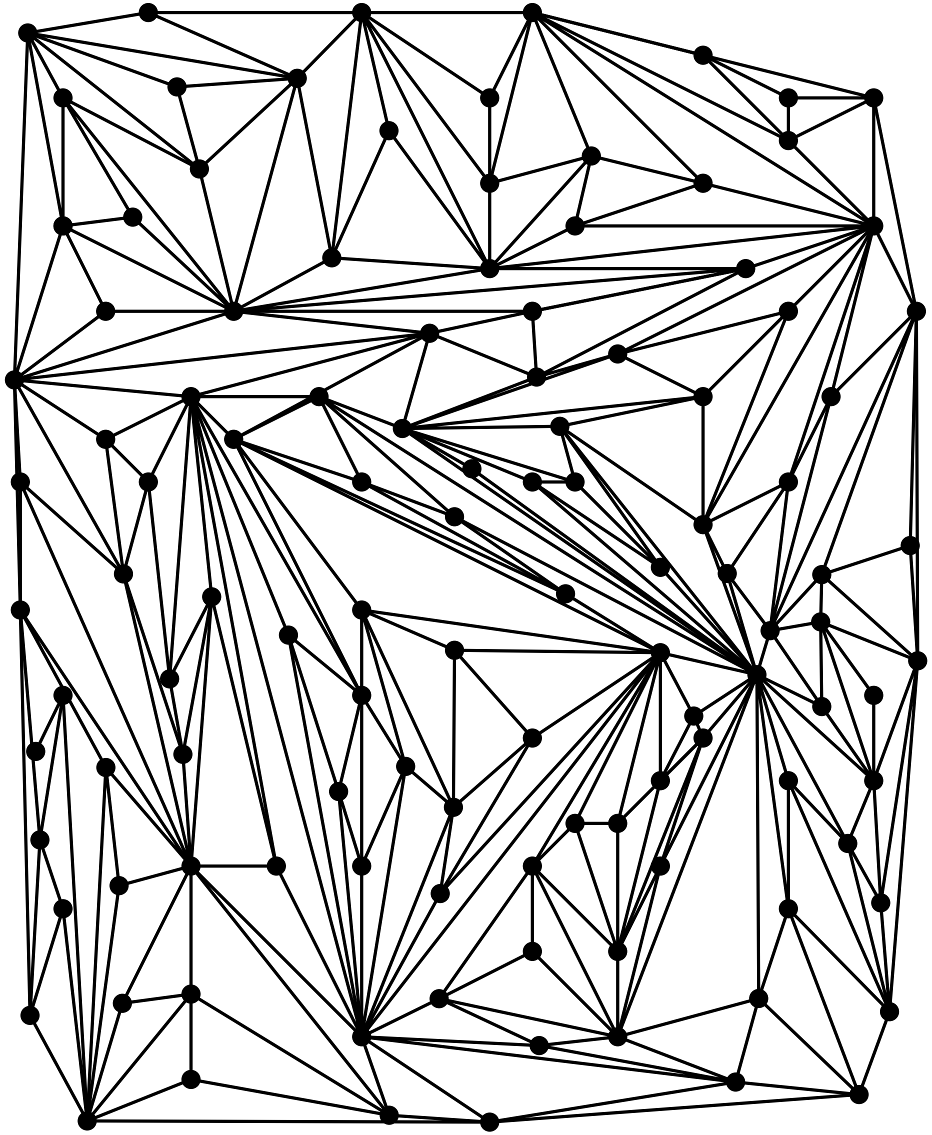
TRIANGULATING POINT SETS

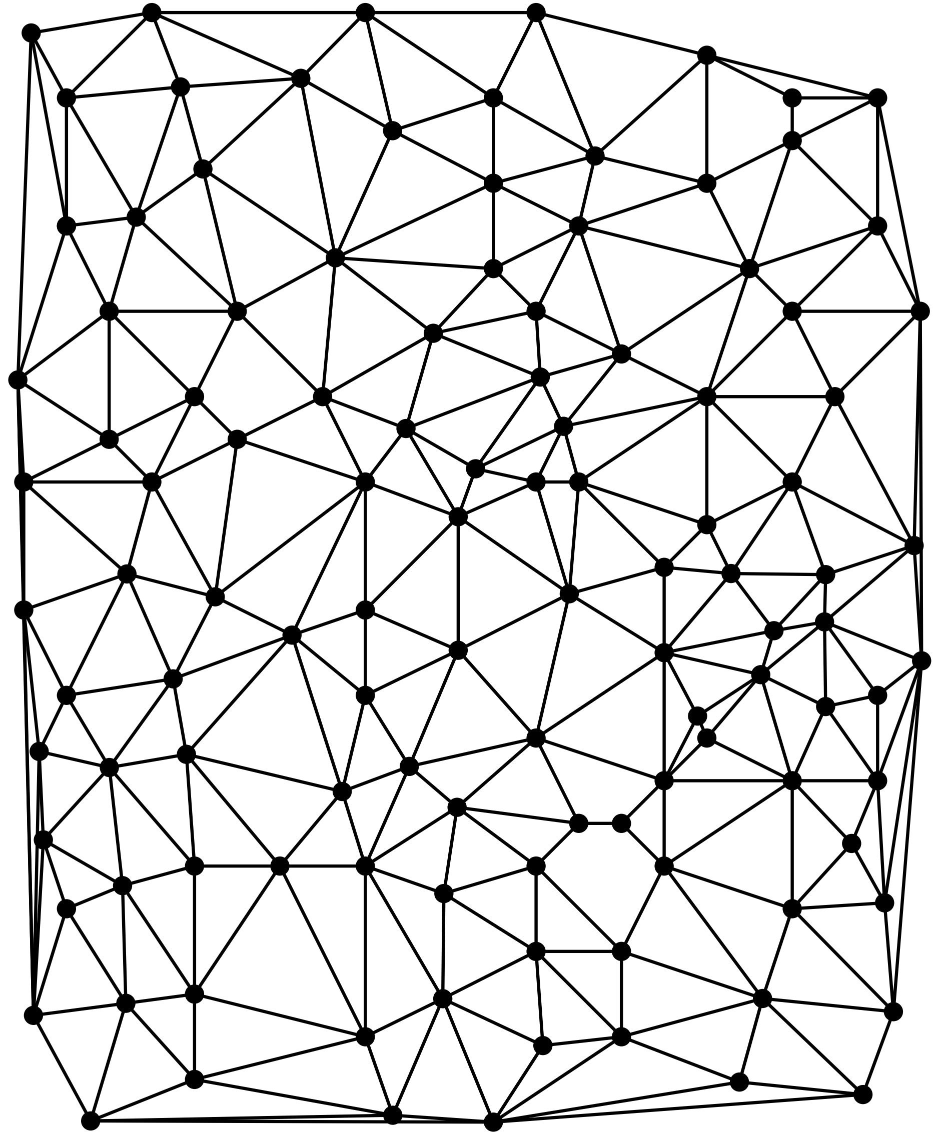


TRIANGULATING POINT SETS

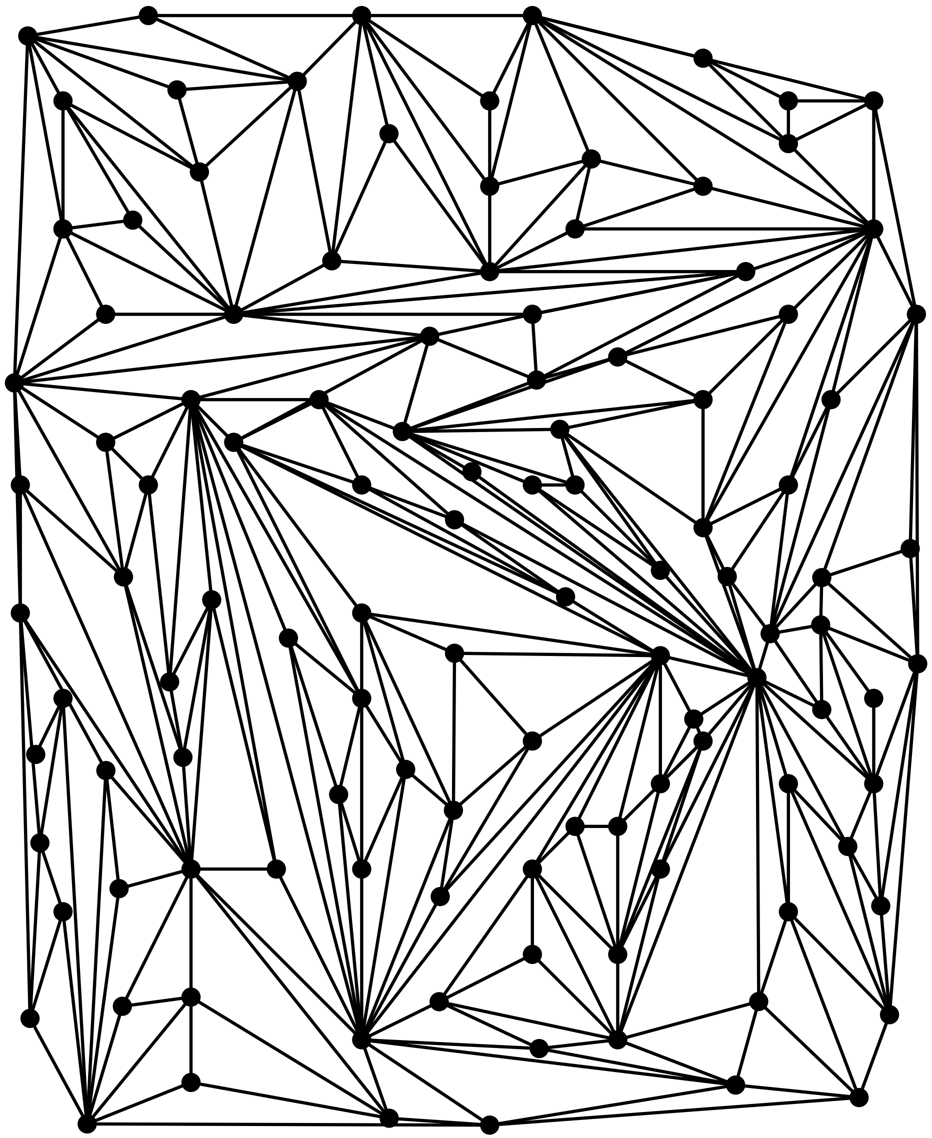


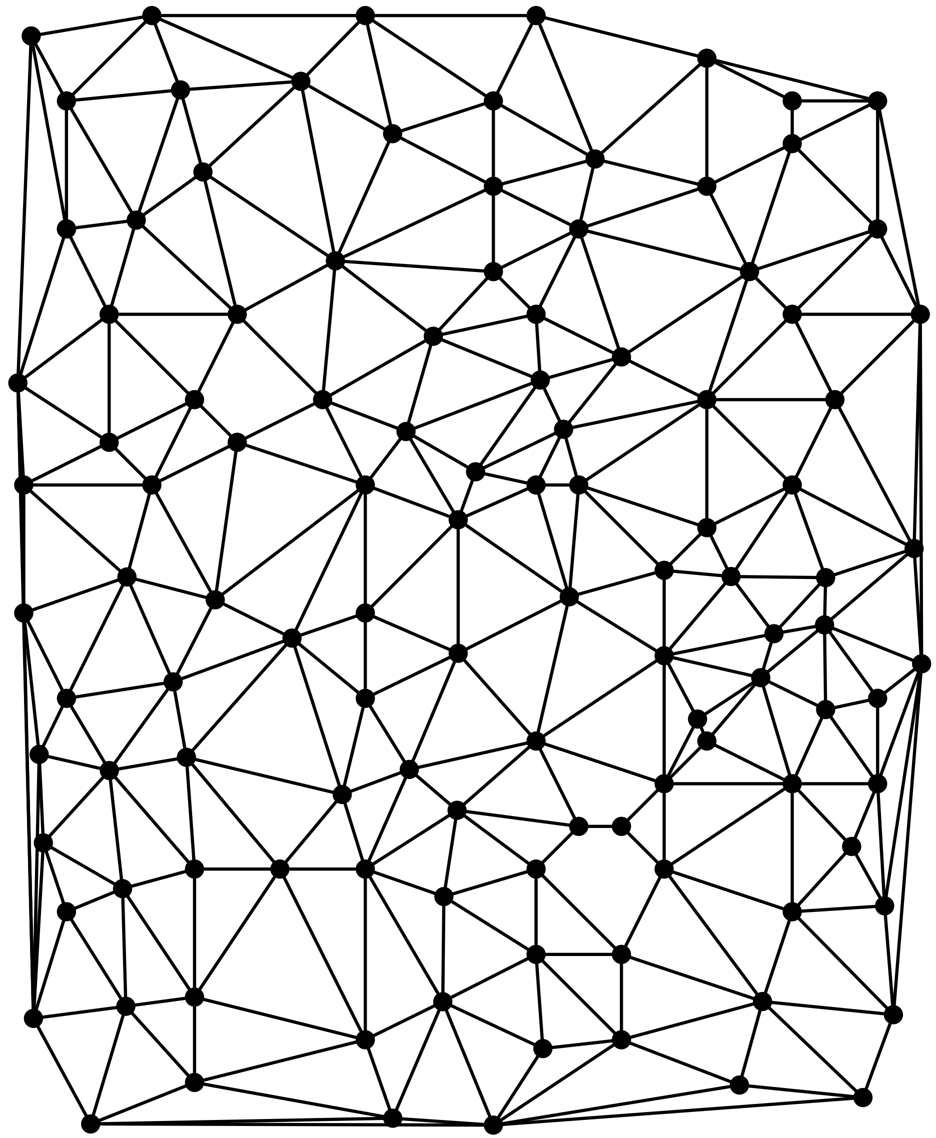
TRIANGULATING POINT SETS





TRIANGULATING POINT SETS





Delaunay triangulation

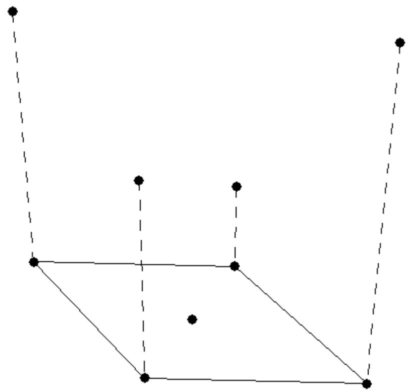
Delaunay Triangulation

DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION

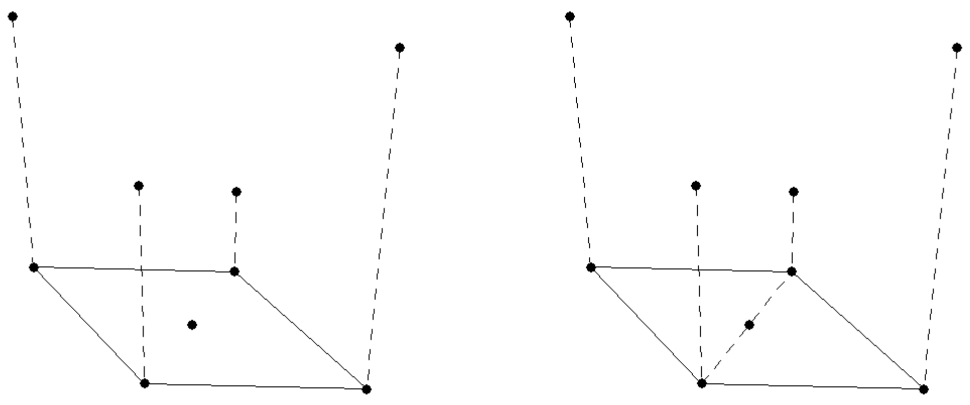
DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION



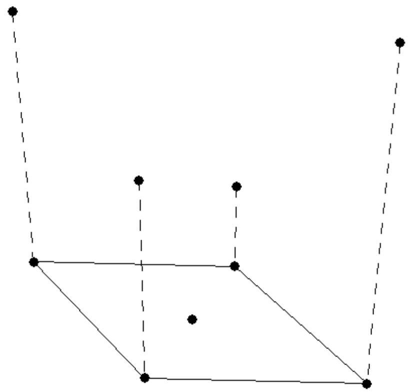
DELAUNAY TRIANGULATION

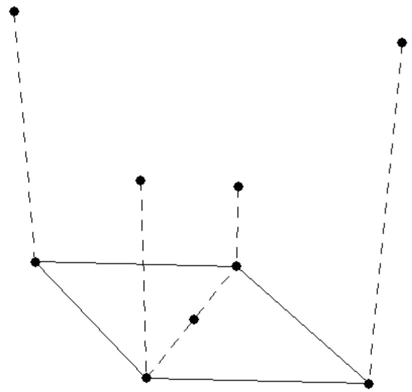
A TOOL FOR INTERPOLATION

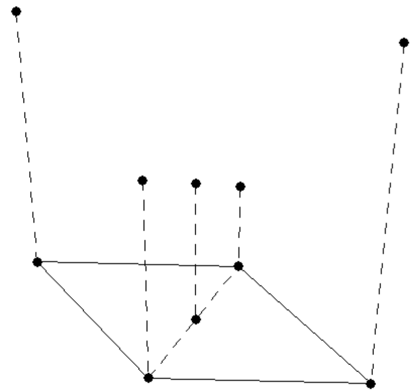


DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION

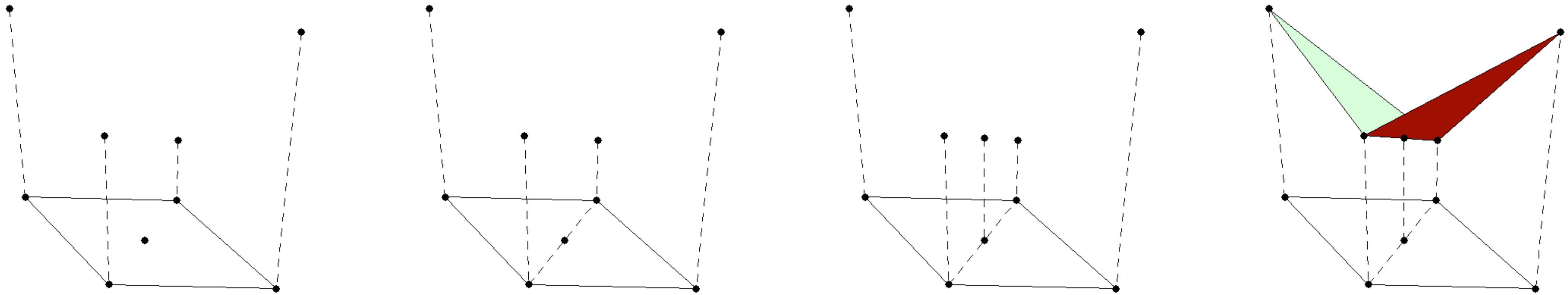






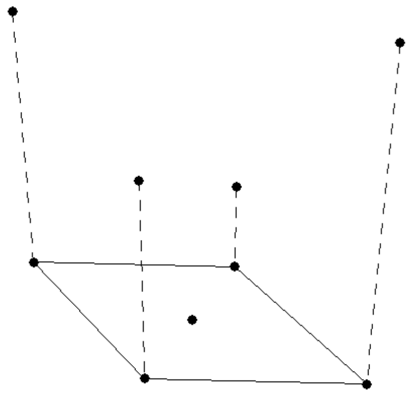
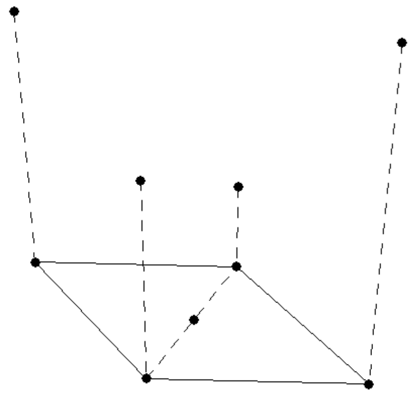
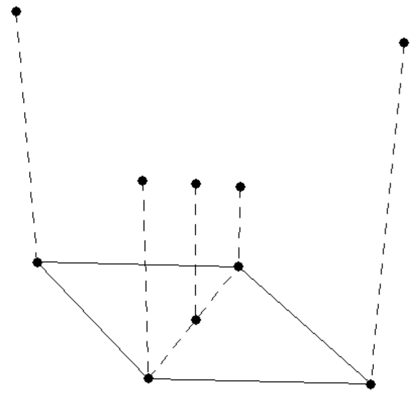
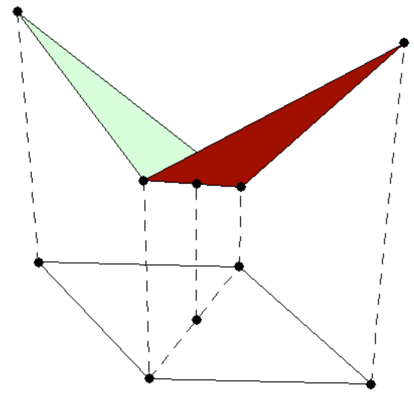
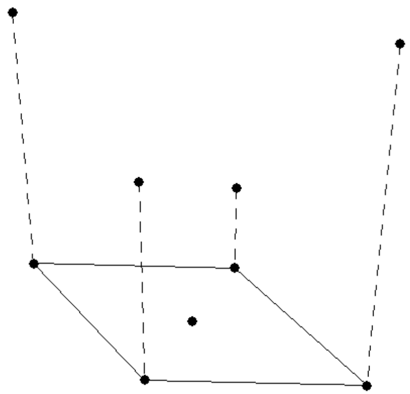
DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION



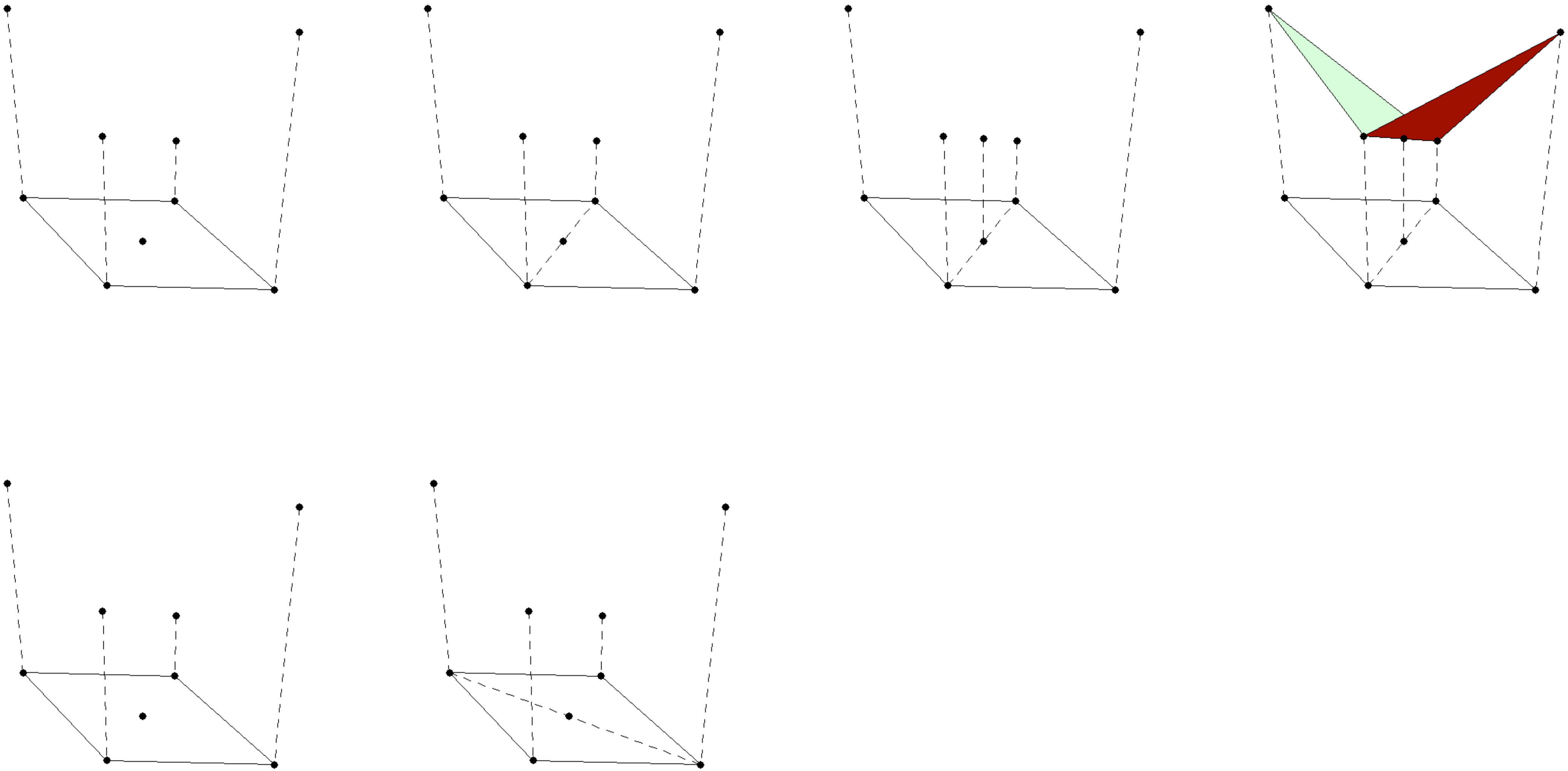
DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION

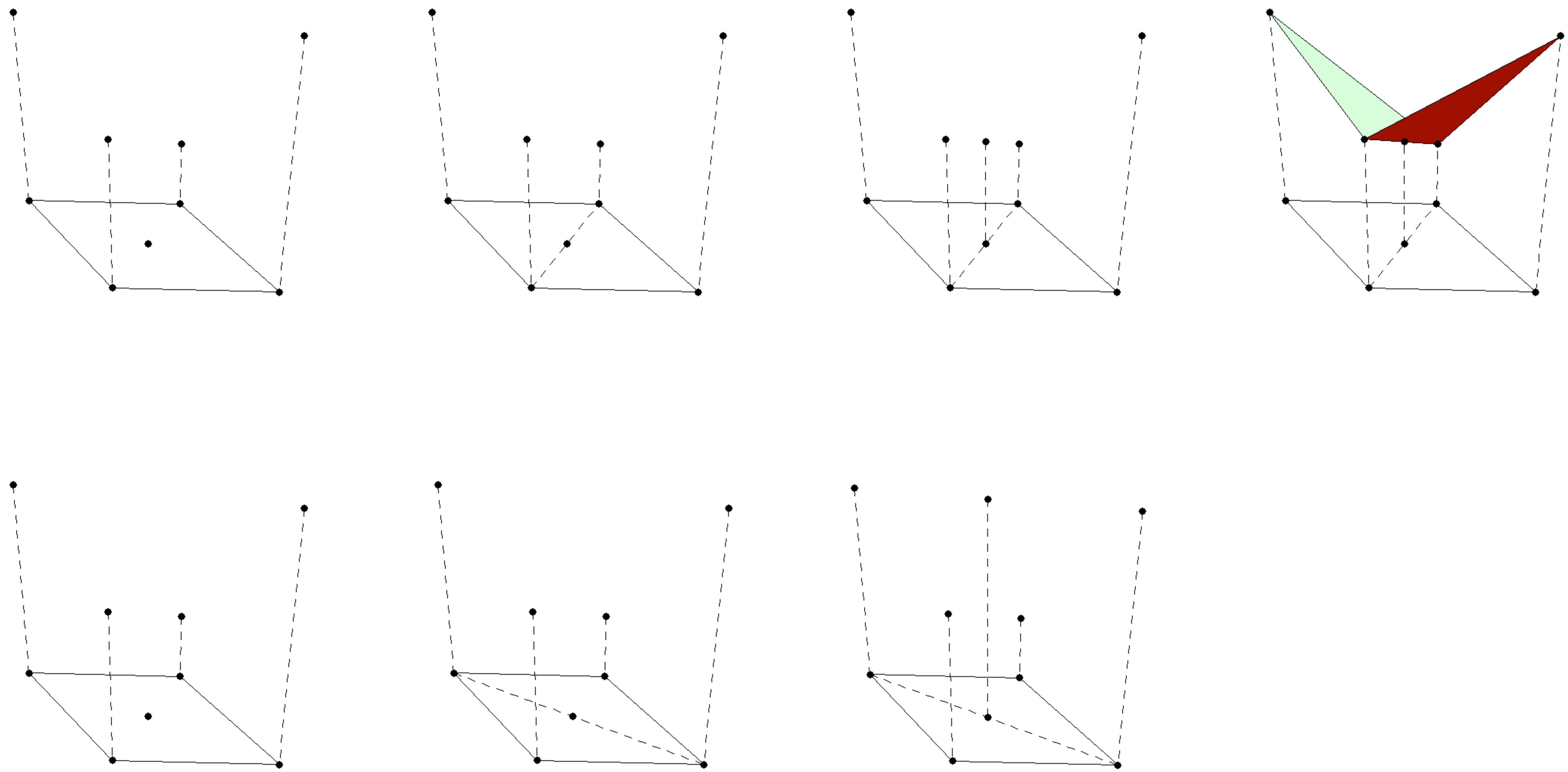
DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION



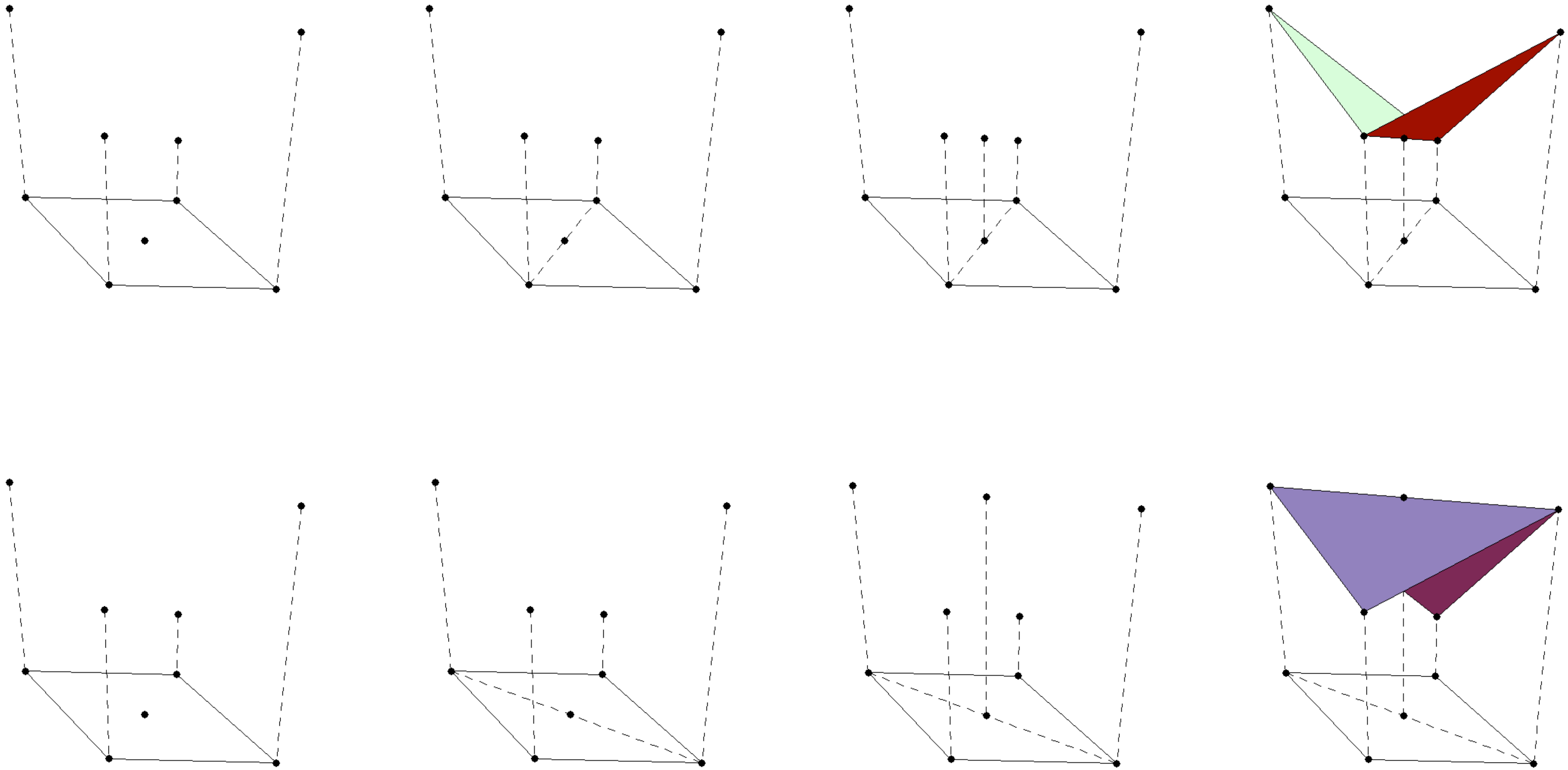
DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION



DELAUNAY TRIANGULATION

A TOOL FOR INTERPOLATION



DELAUNAY TRIANGULATION

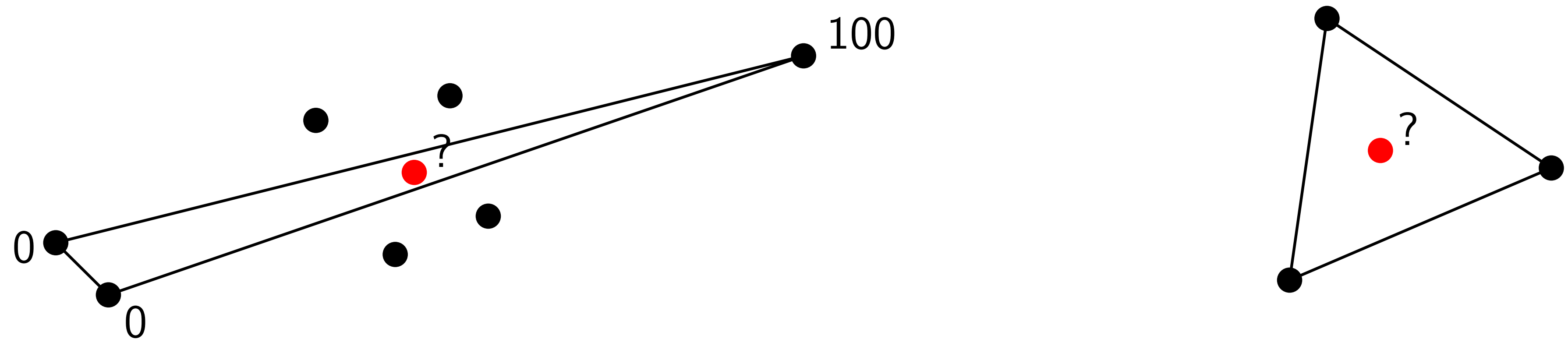
A TOOL FOR INTERPOLATION

Motivation

For many interpolation applications, it makes more sense to interpolate from points that are nearby than from points that are far away

To do that, the shape of the triangles matters:

- Long and skinny triangles produce interpolation from points that are far
- Equilateral-like triangles tend to give interpolations from points closer and well distributed



DELAUNAY TRIANGULATION

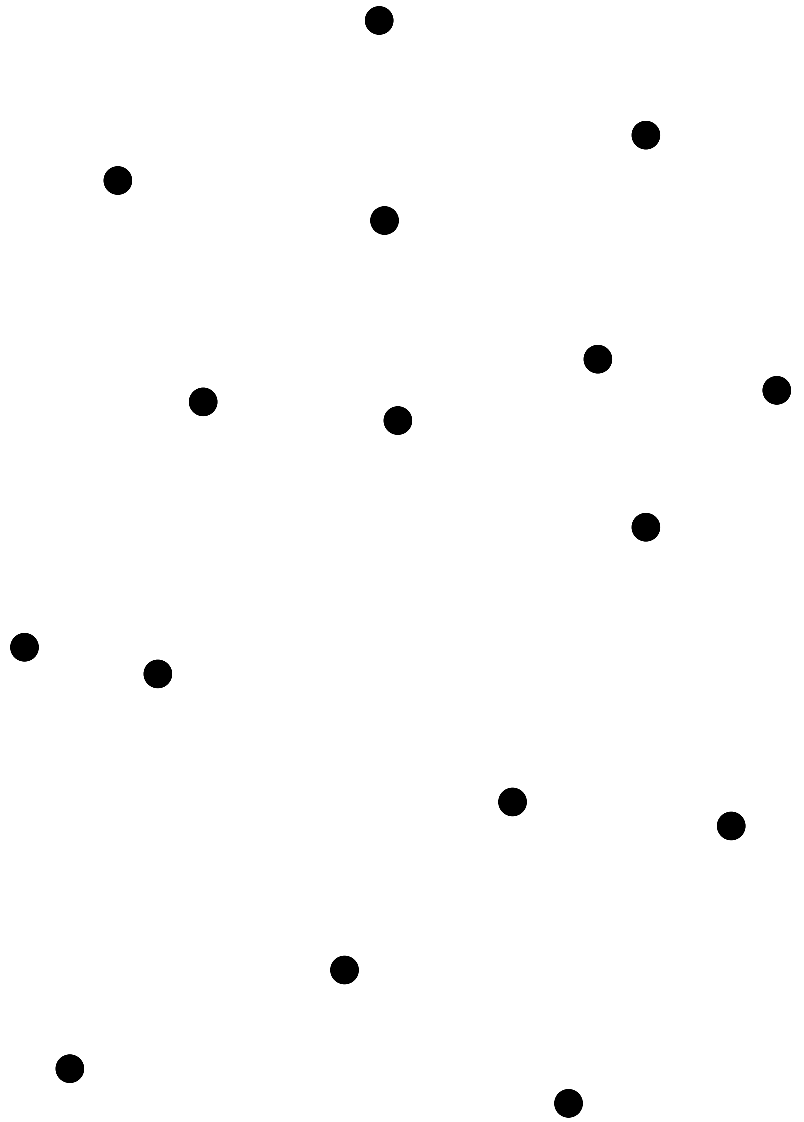
DEFINITION AND PROPERTIES

DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

Given a set P with n points in the plane...



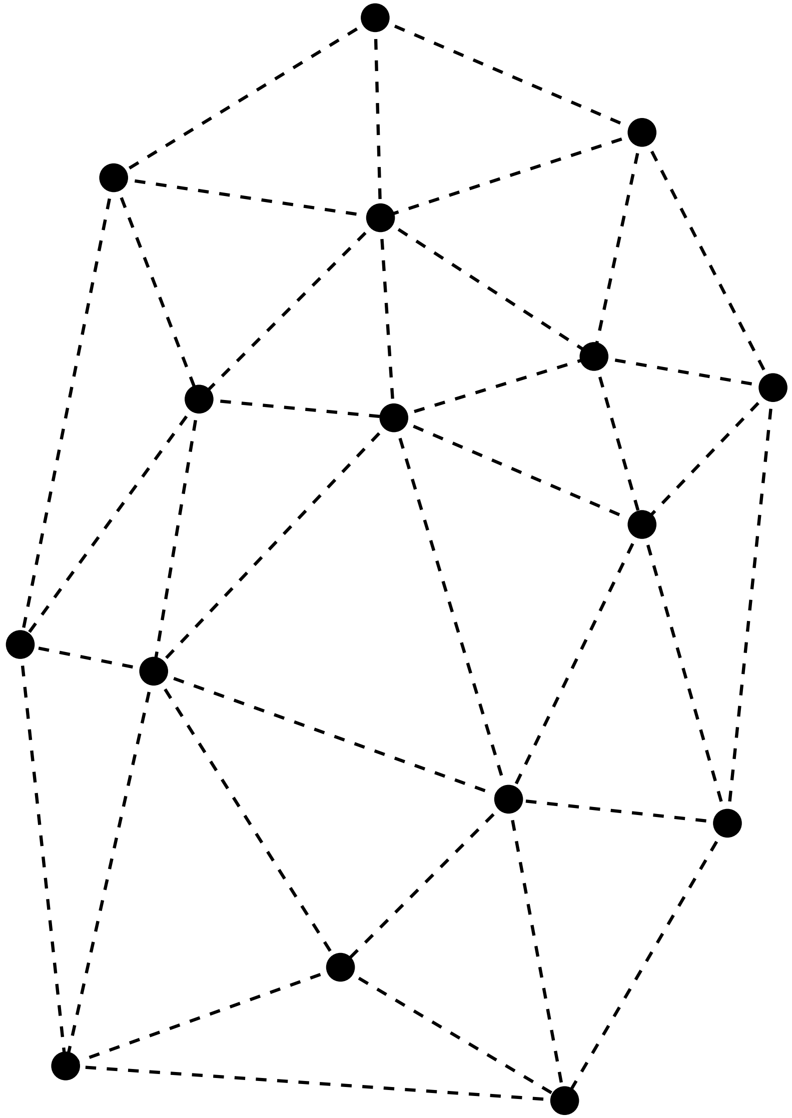
DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Three points p_i, p_j, p_k form a *Delaunay triangle* (in general, are vertices of a face) if and only if the circle through them does not contain any point of P in its interior.



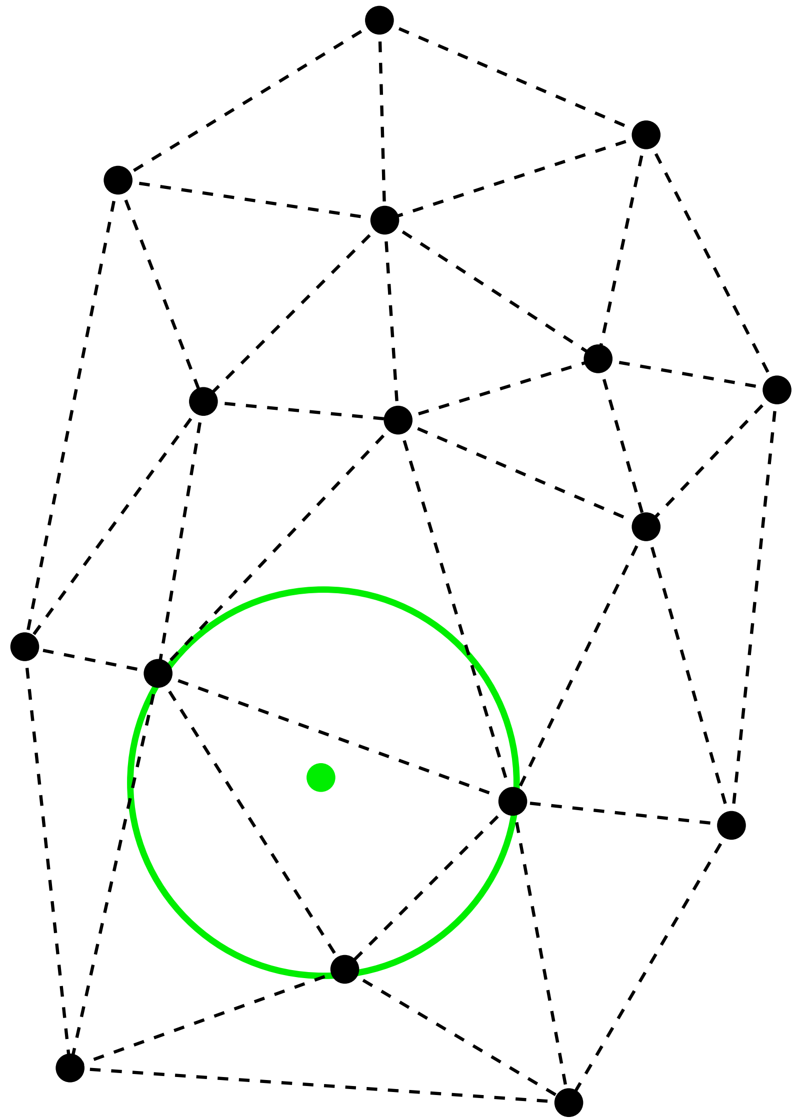
DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Three points p_i, p_j, p_k form a *Delaunay triangle* (in general, are vertices of a face) if and only if the circle through them does not contain any point of P in its interior.



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

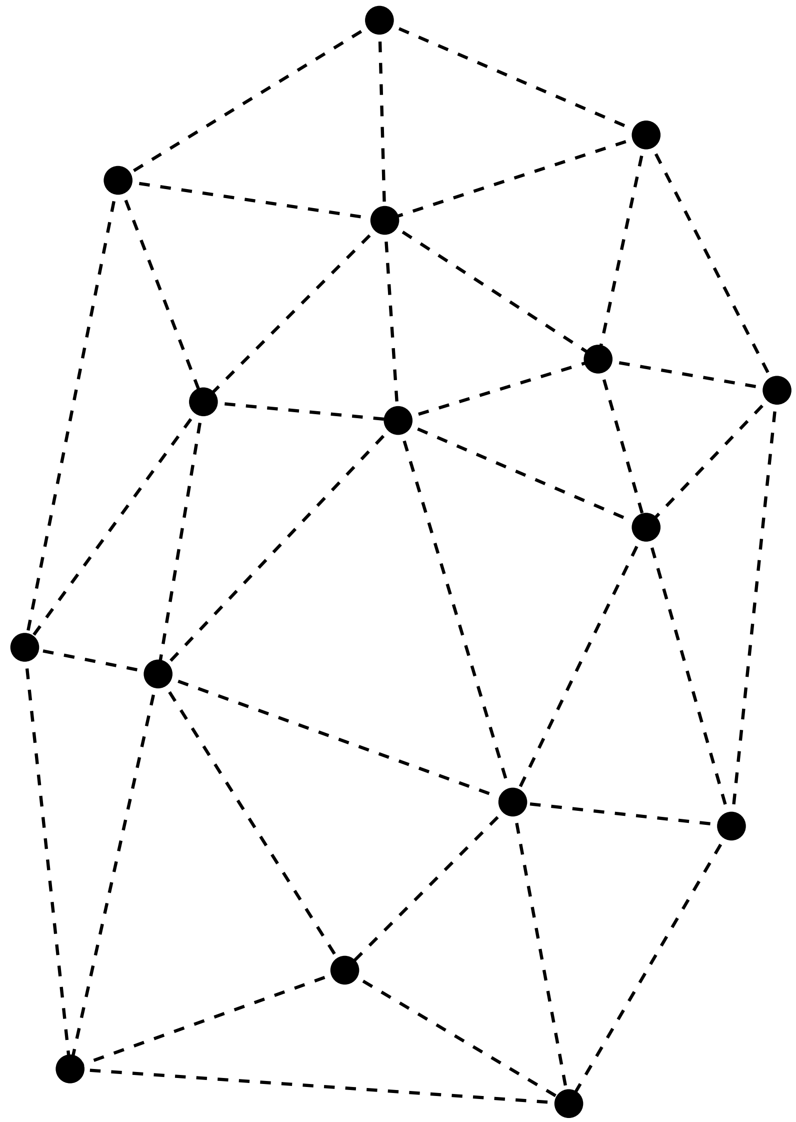
Definition (triangle-based)

Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Three points p_i, p_j, p_k form a *Delaunay triangle* (in general, are vertices of a face) if and only if the circle through them does not contain any point of P in its interior.

Equivalent characterization (edge-based)

- Two points $p_i, p_j \in P$ form a Delaunay edge if and only if there exists a circle through p_i and p_j that does not contain any point of P in its interior.



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

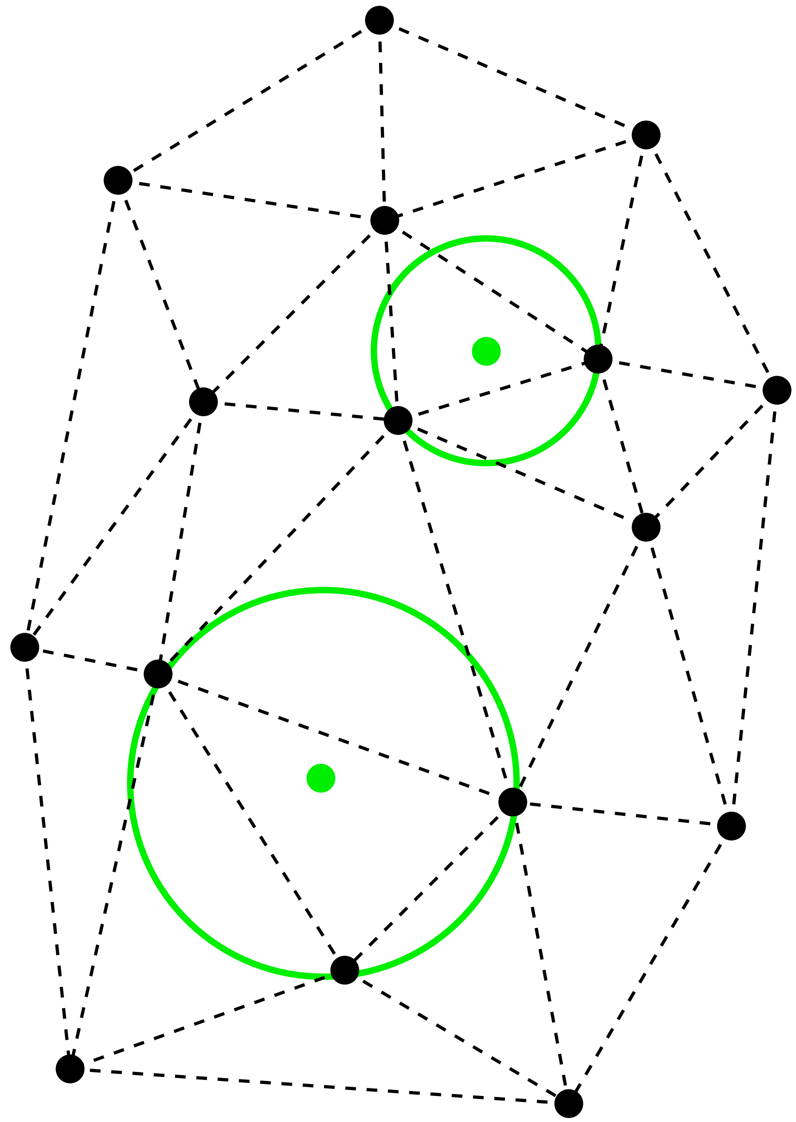
Definition (triangle-based)

Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Three points p_i, p_j, p_k form a *Delaunay triangle* (in general, are vertices of a face) if and only if the circle through them does not contain any point of P in its interior.

Equivalent characterization (edge-based)

- Two points $p_i, p_j \in P$ form a Delaunay edge if and only if there exists a circle through p_i and p_j that does not contain any point of P in its interior.



DELAUNAY TRIANGULATION

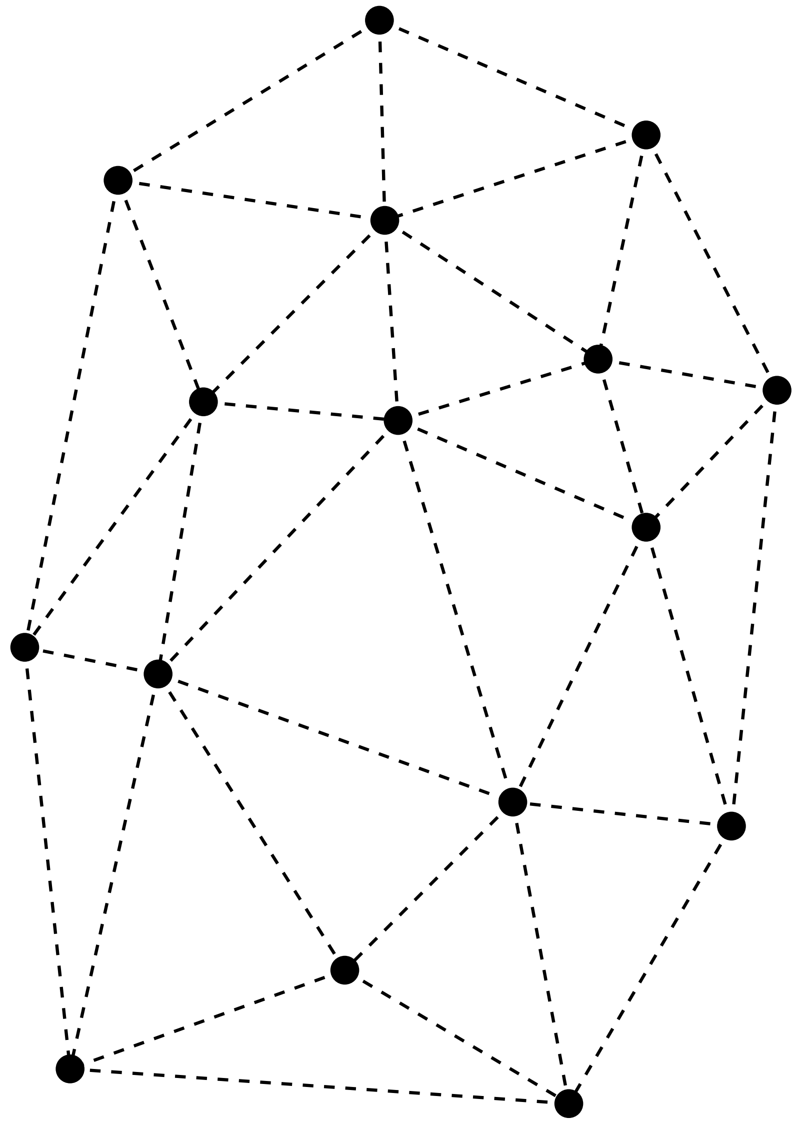
DEFINITION AND PROPERTIES

Definition (triangle-based)

Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

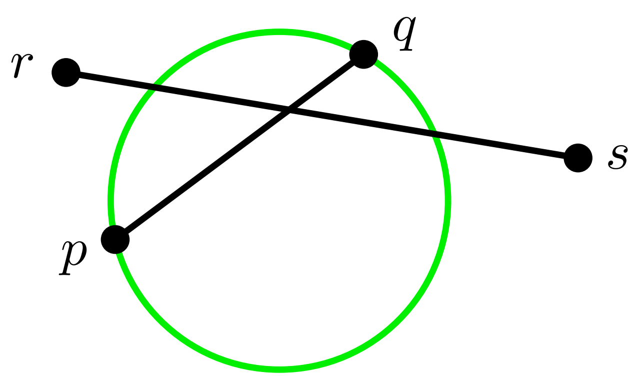
Definition (triangle-based)

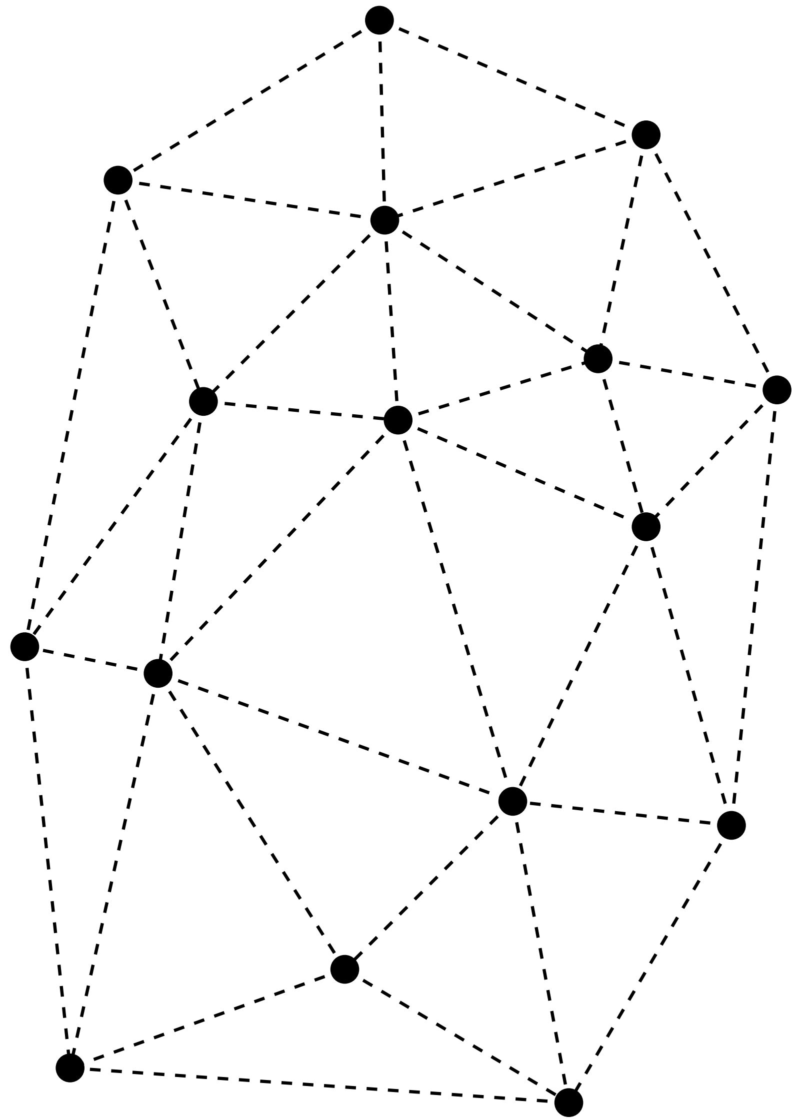
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

If $\overline{pq}$ is a Delaunay edge, there exists an empty circle through p and q. If a segment $\overline{rs}$ intersects $\overline{pq}$, then every circle through r and s contains at least one of p or q.





DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

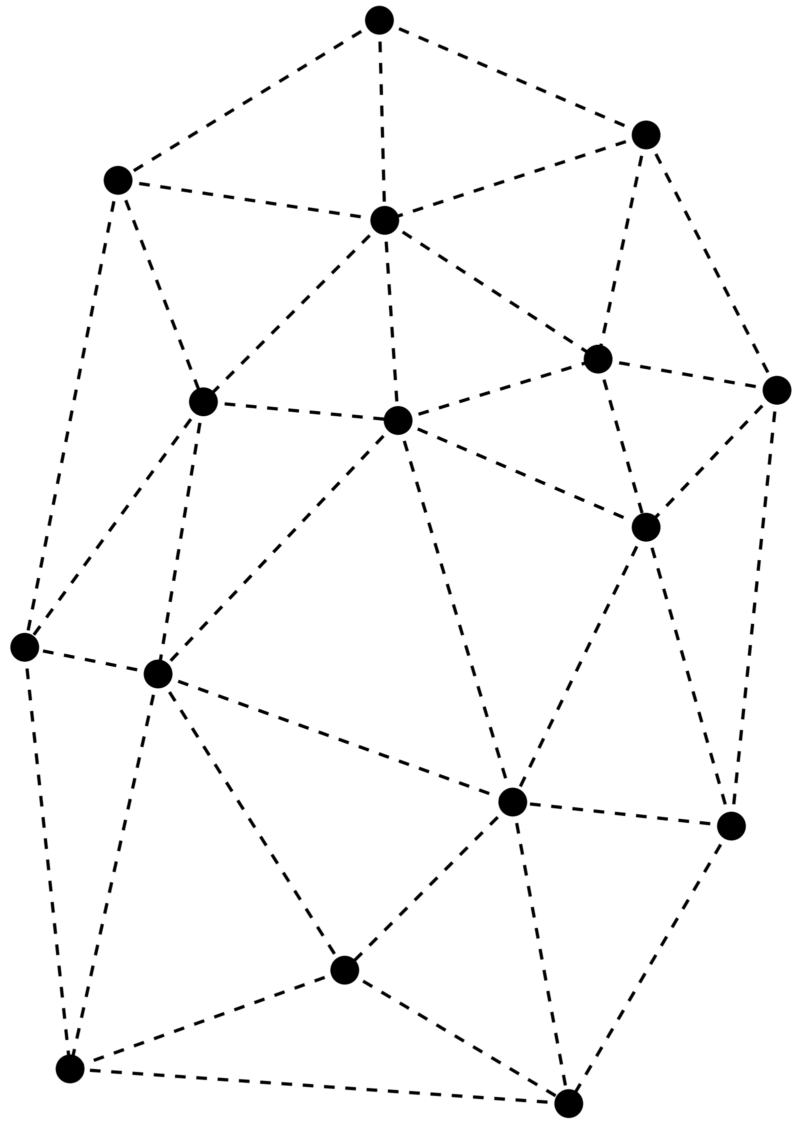
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

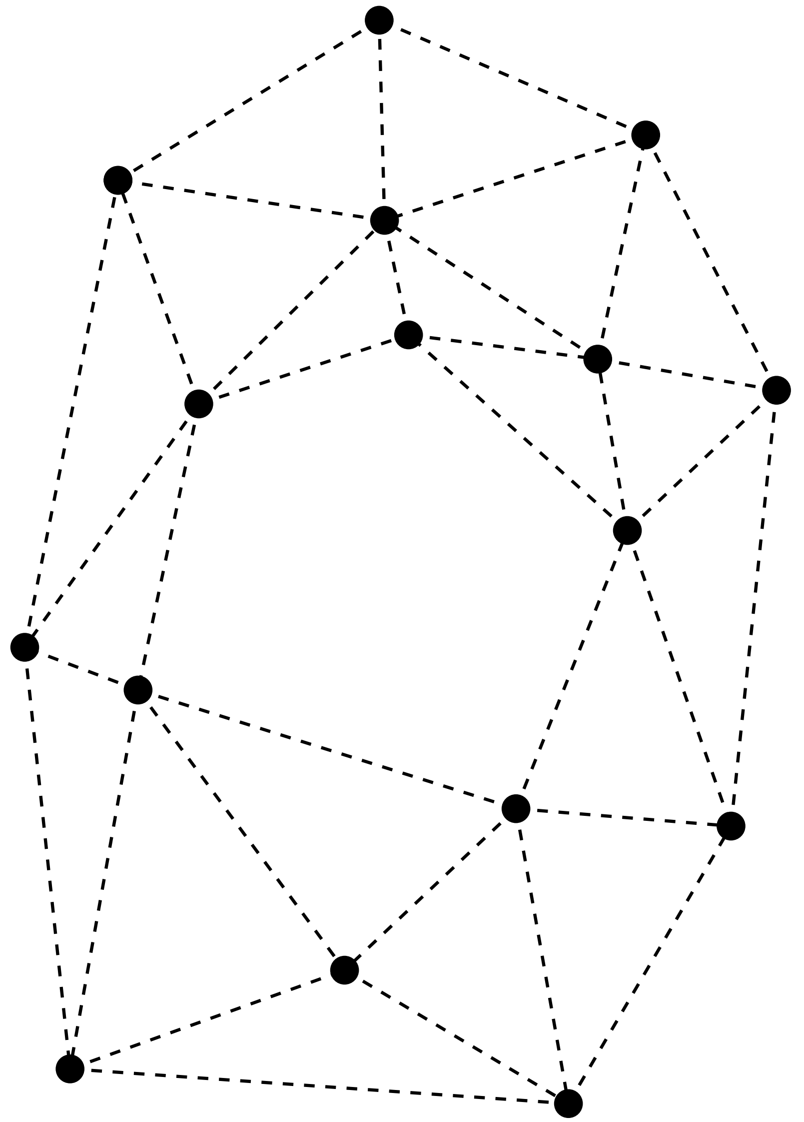
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be done trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

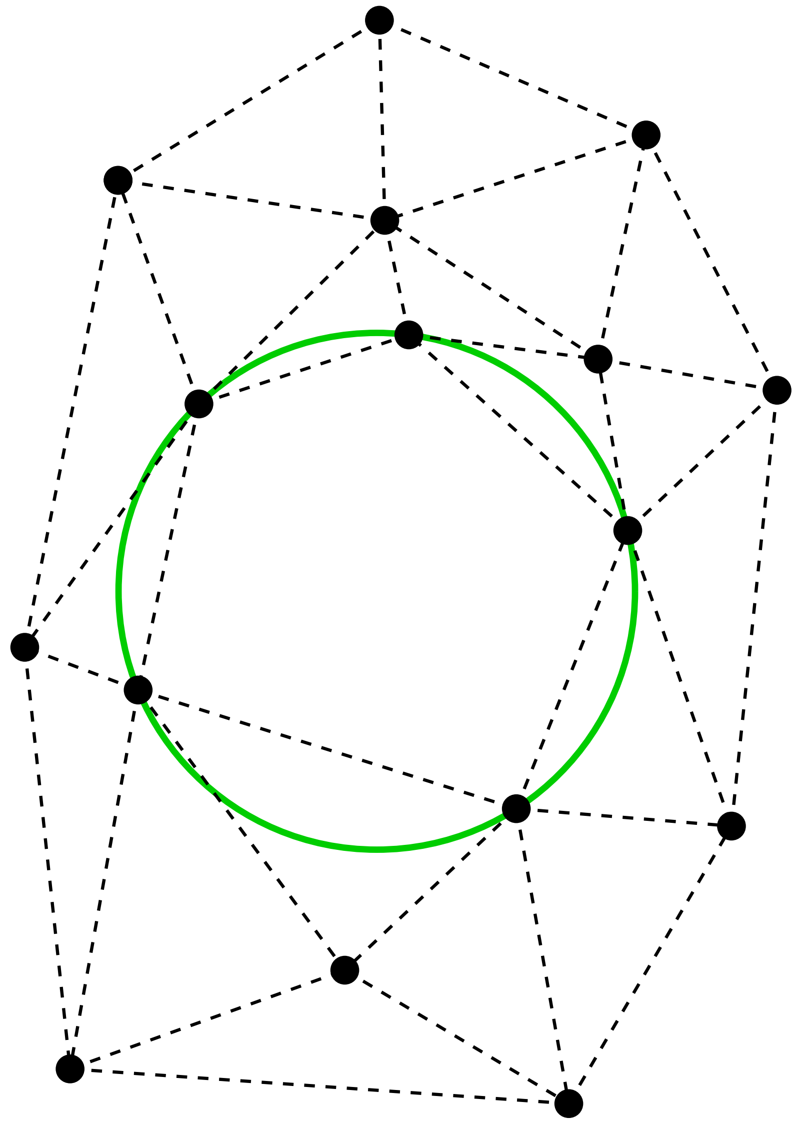
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

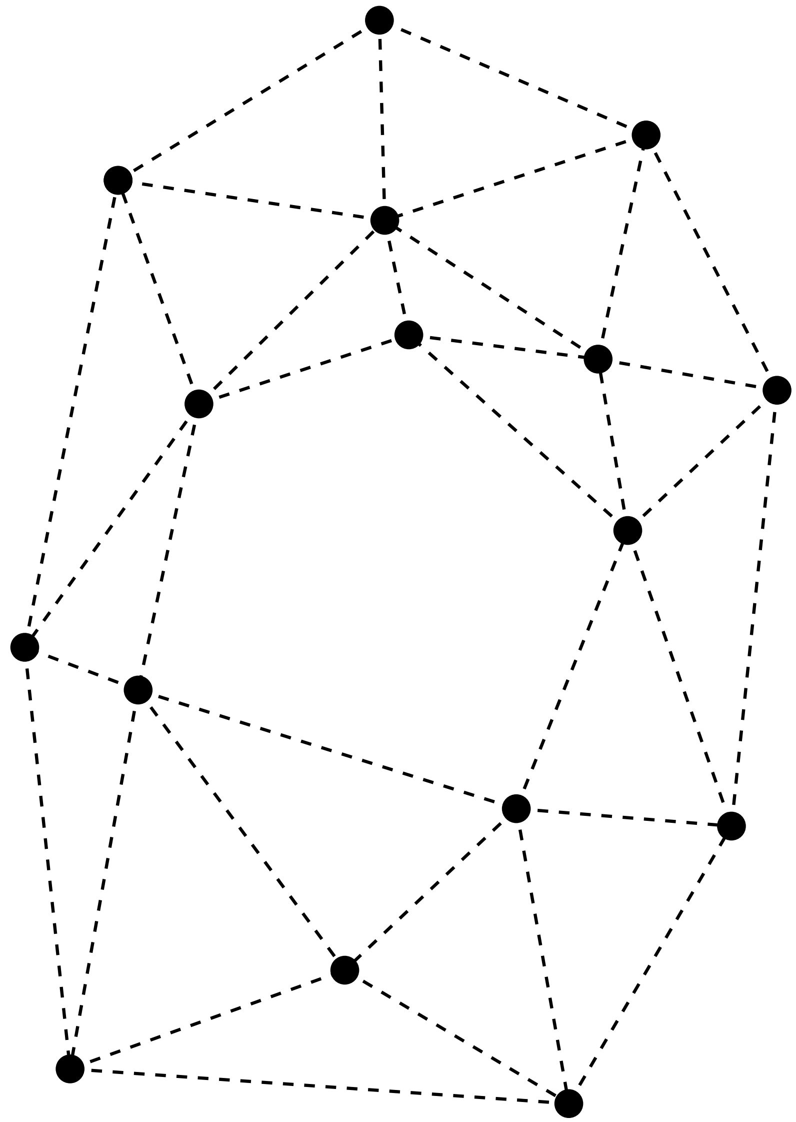
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

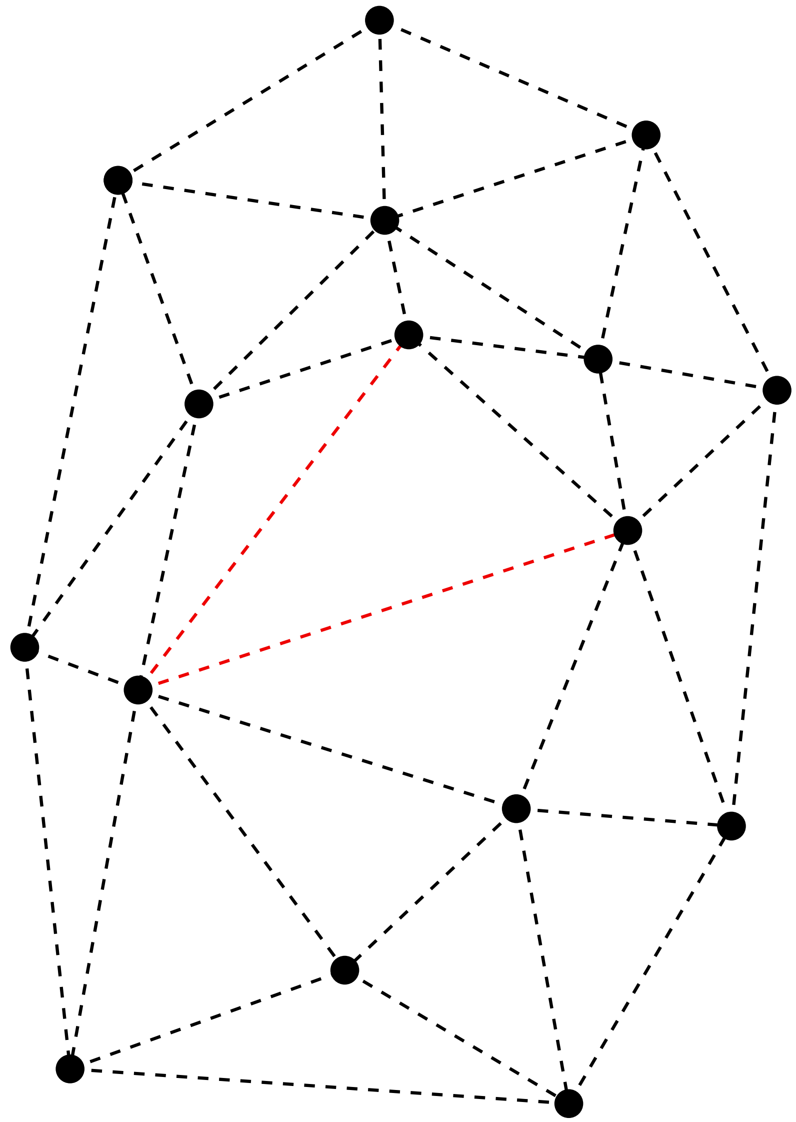
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

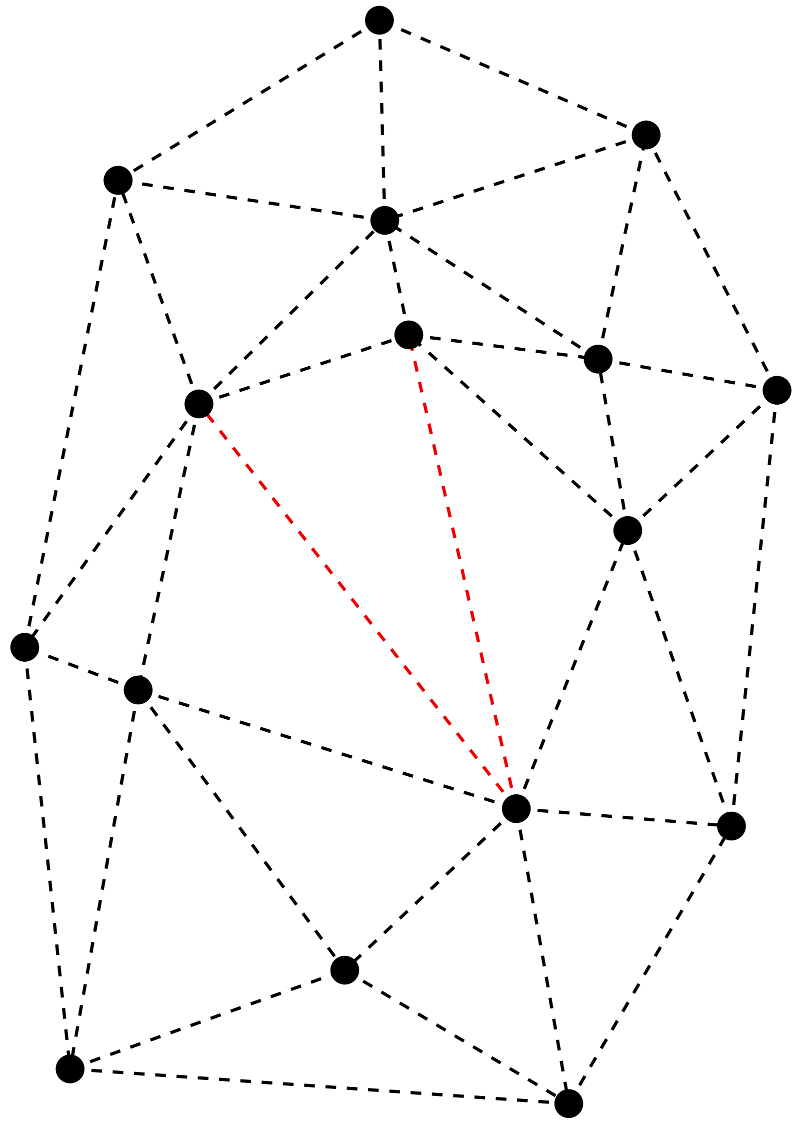
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

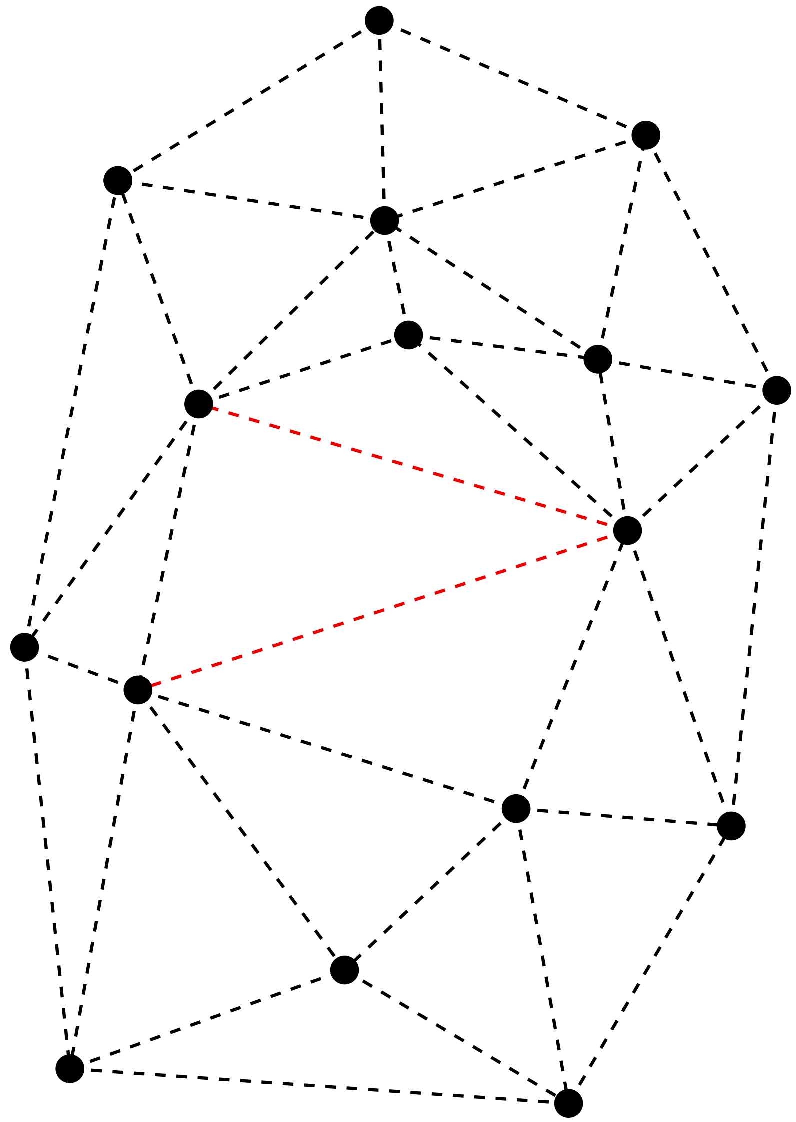
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

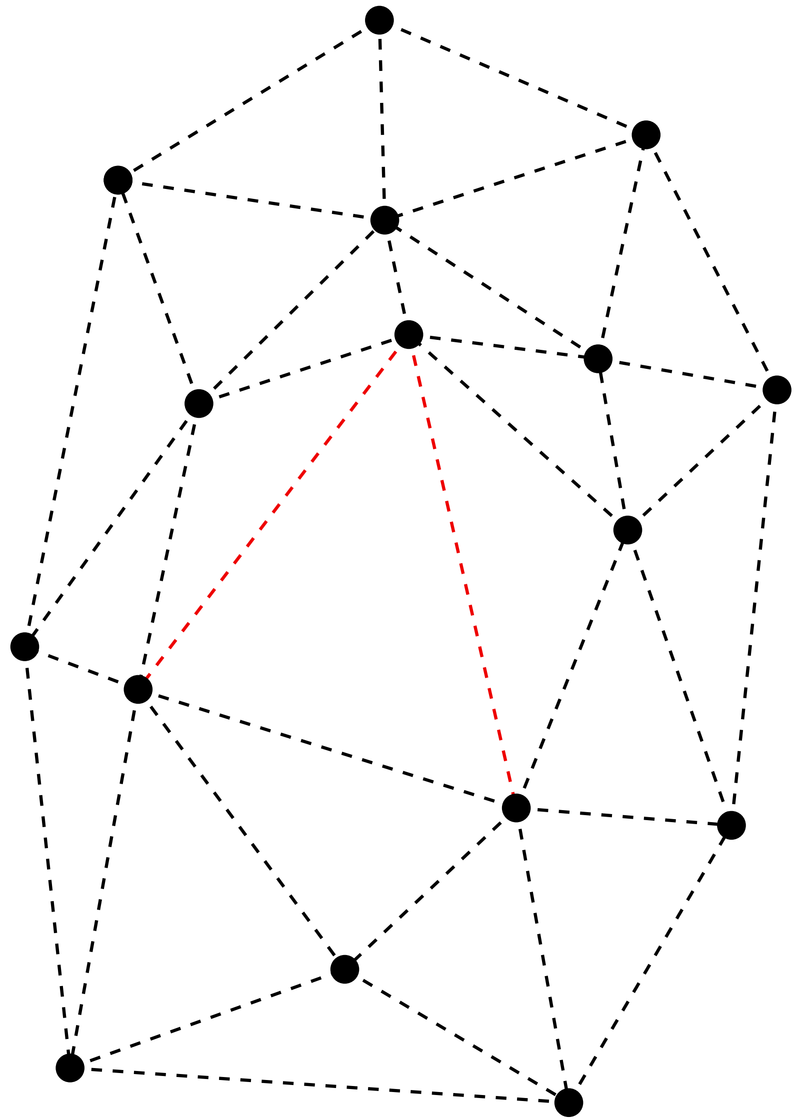
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be done trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION

DEFINITION AND PROPERTIES

Definition (triangle-based)

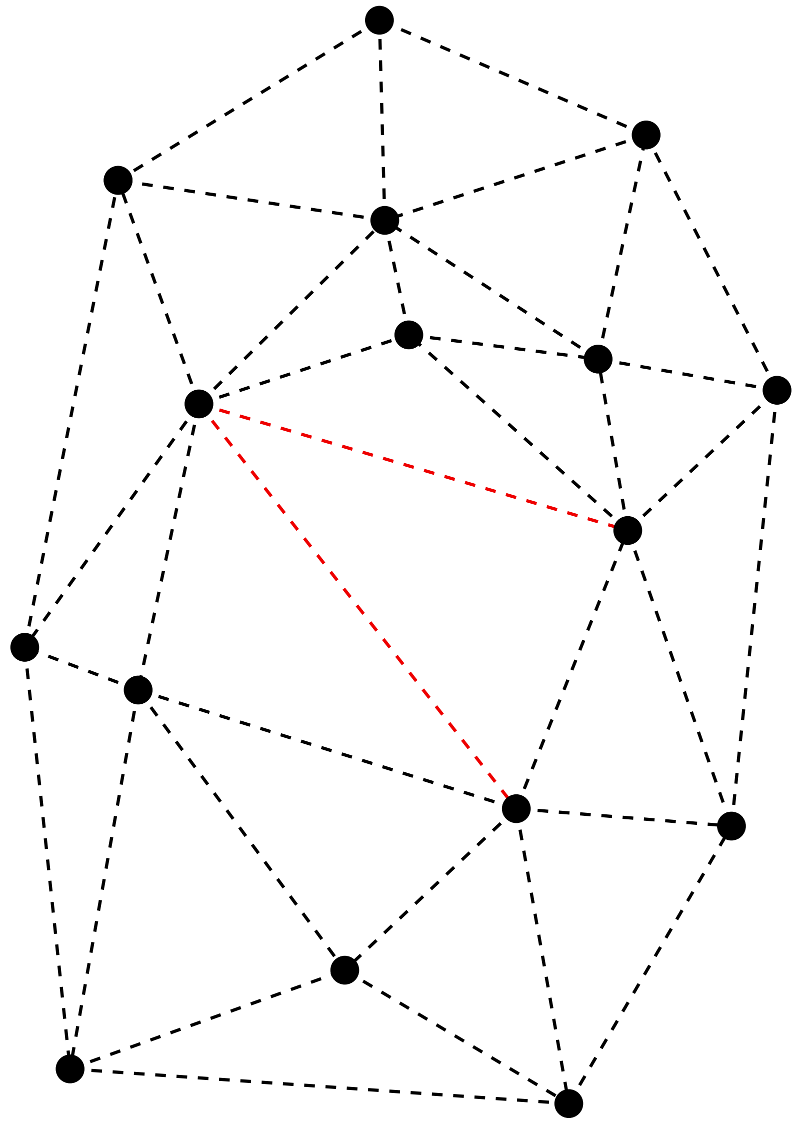
Given a set P of n points in the plane, the Delaunay triangulation of P, $Del(P)$, is a triangulation where all triangles are *Delaunay triangles*.

Properties of edge-based definition

$Del(P)$ is a plane graph

Property

$Del(P)$ is a triangulation of P, except when P has three or more concyclic points. In this case, it is a pre-triangulation which can be trivally completed (although this can be done in several different ways).



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

DELAUNAY TRIANGULATION AND 3D CONVEX HULL

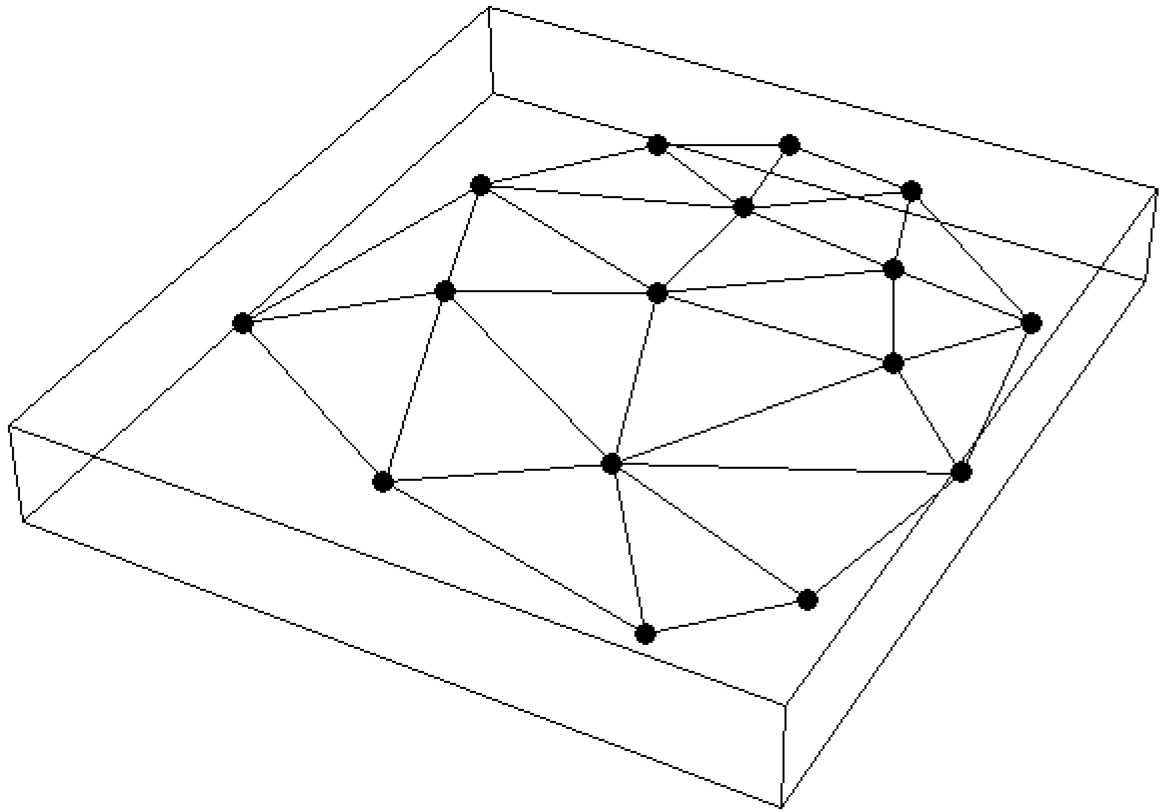
Theorem

Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

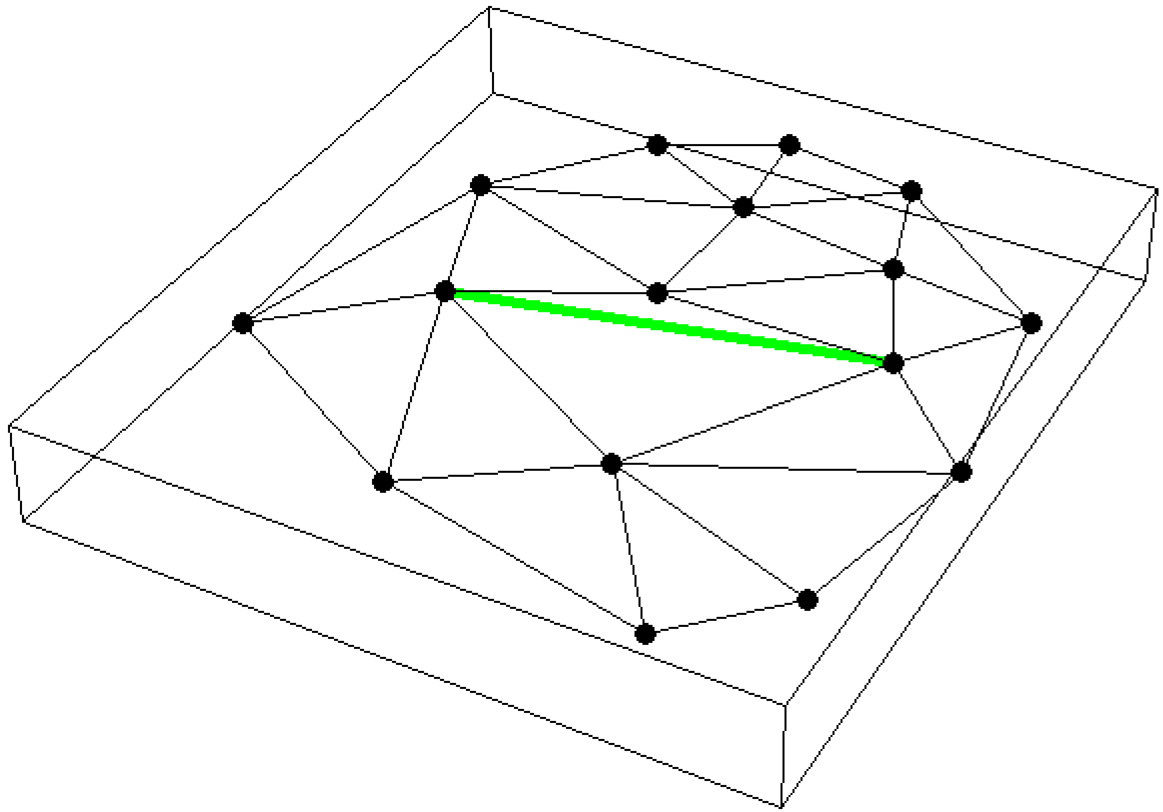
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

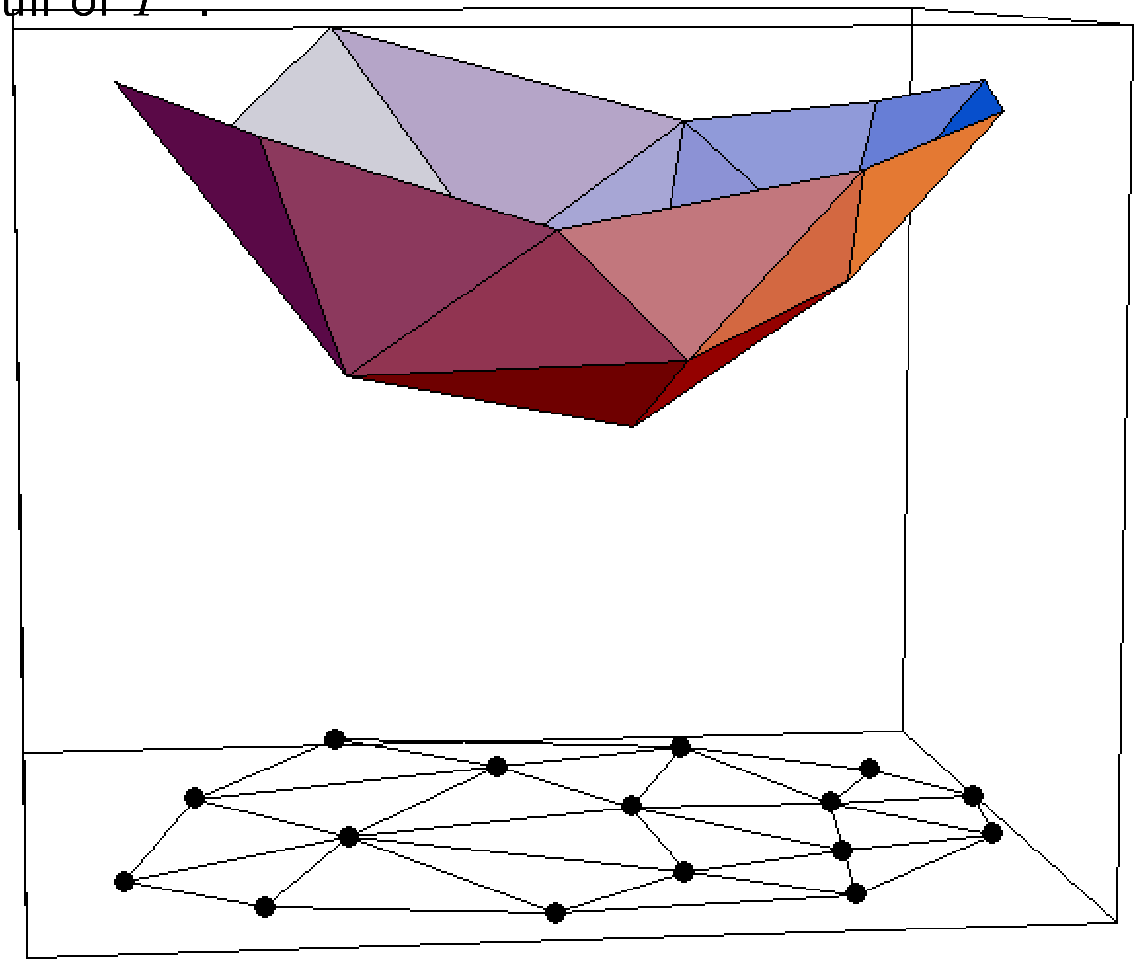
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

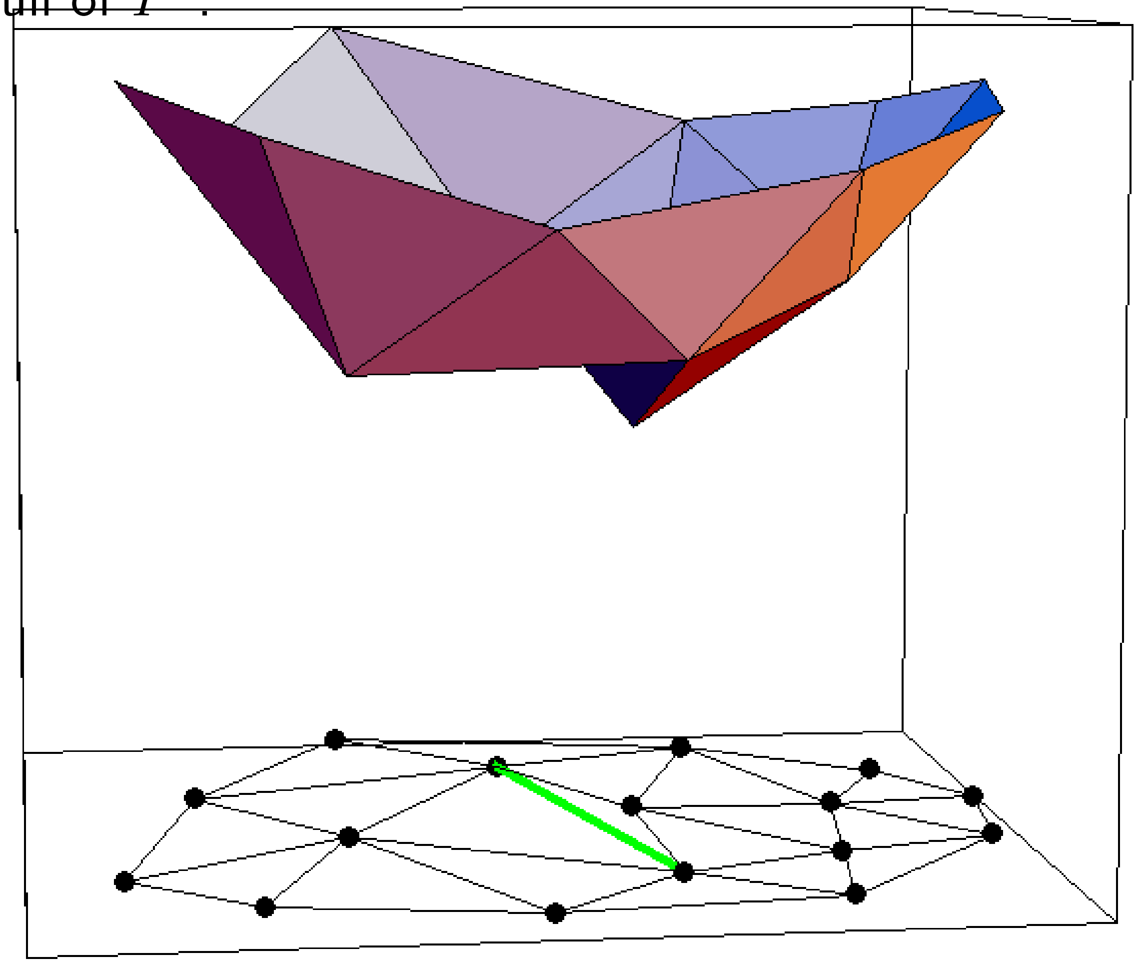
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

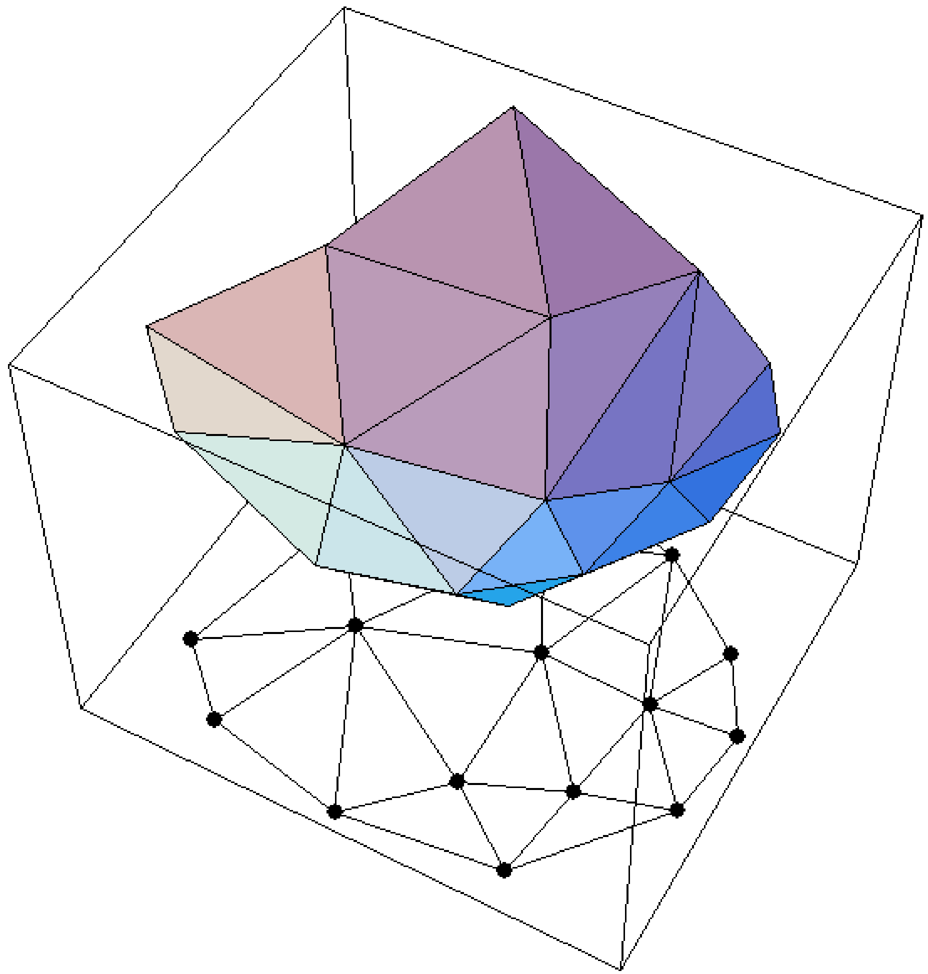
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

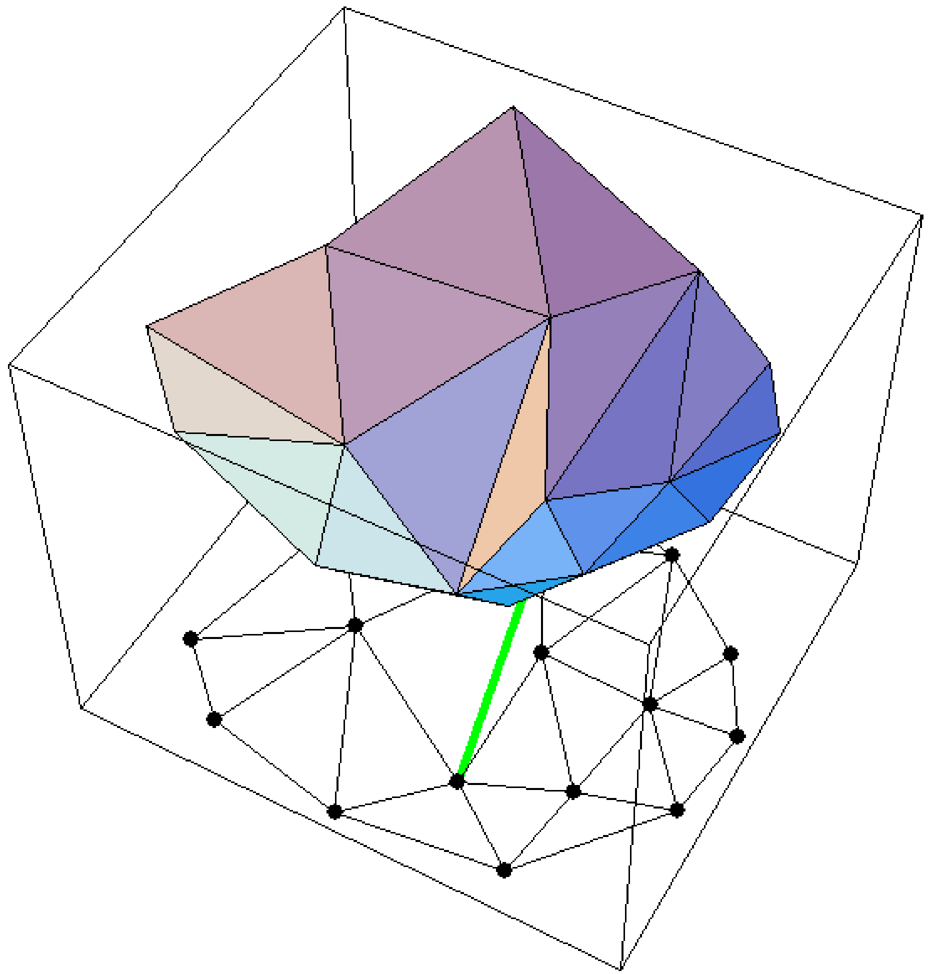
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

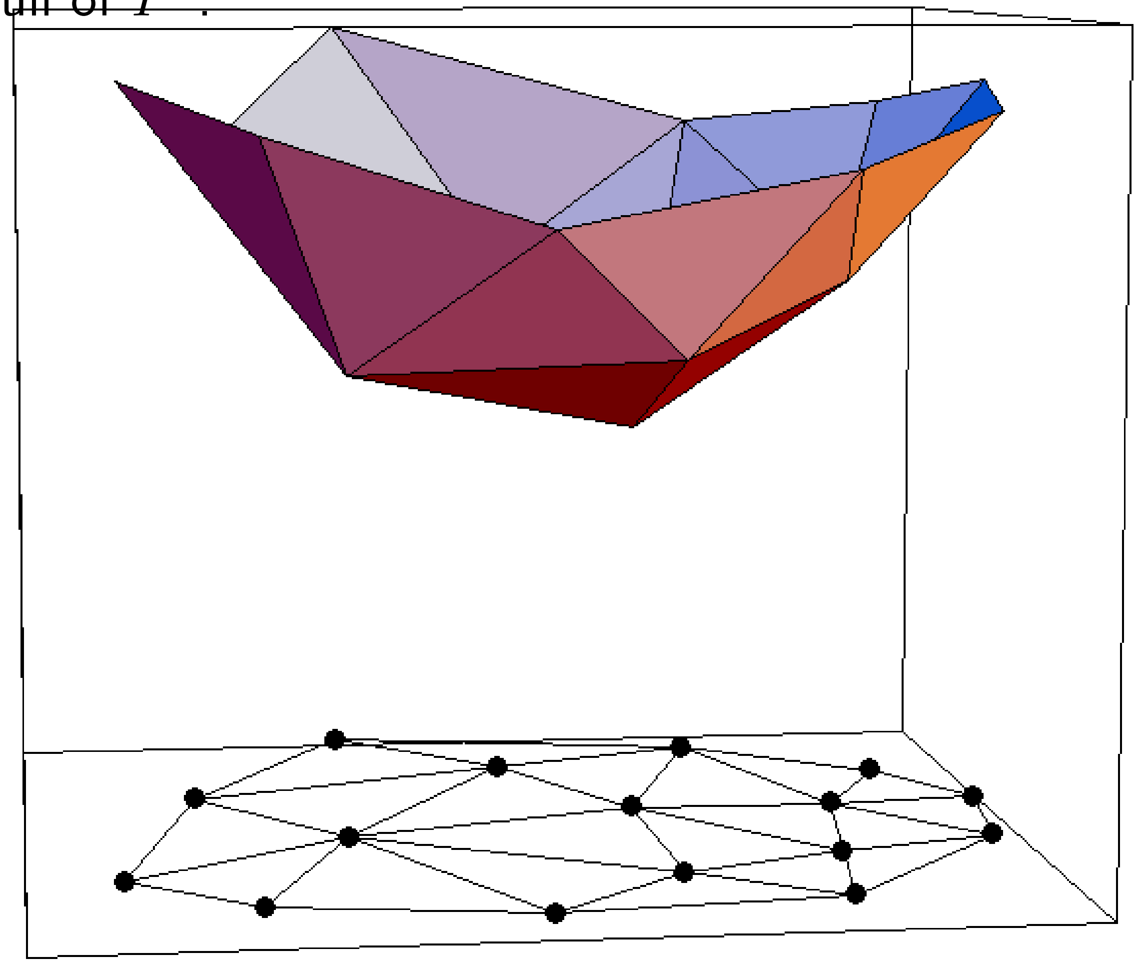
Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

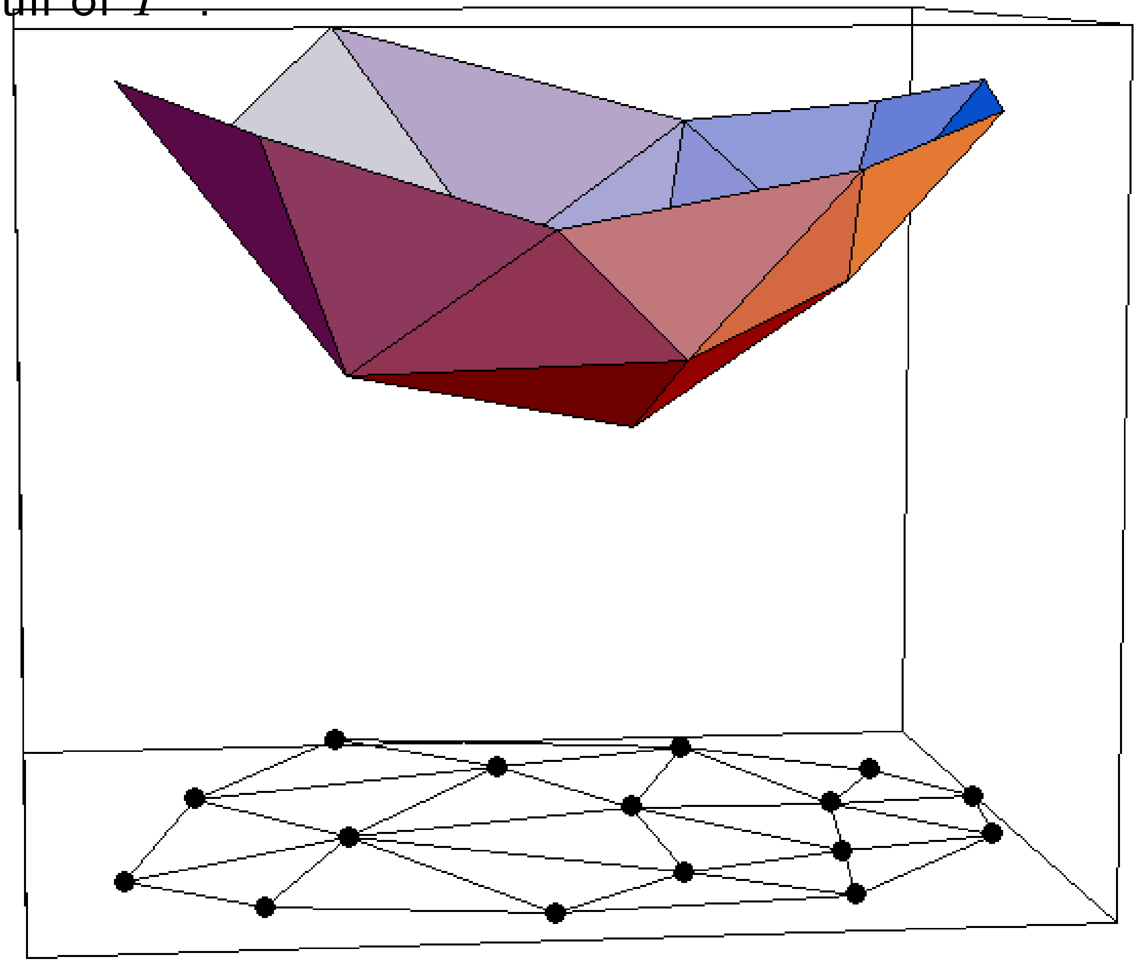


DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

p_i^*, p_j^*, p_k^* form a (triangular) face of the lower convex hull of P^*



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

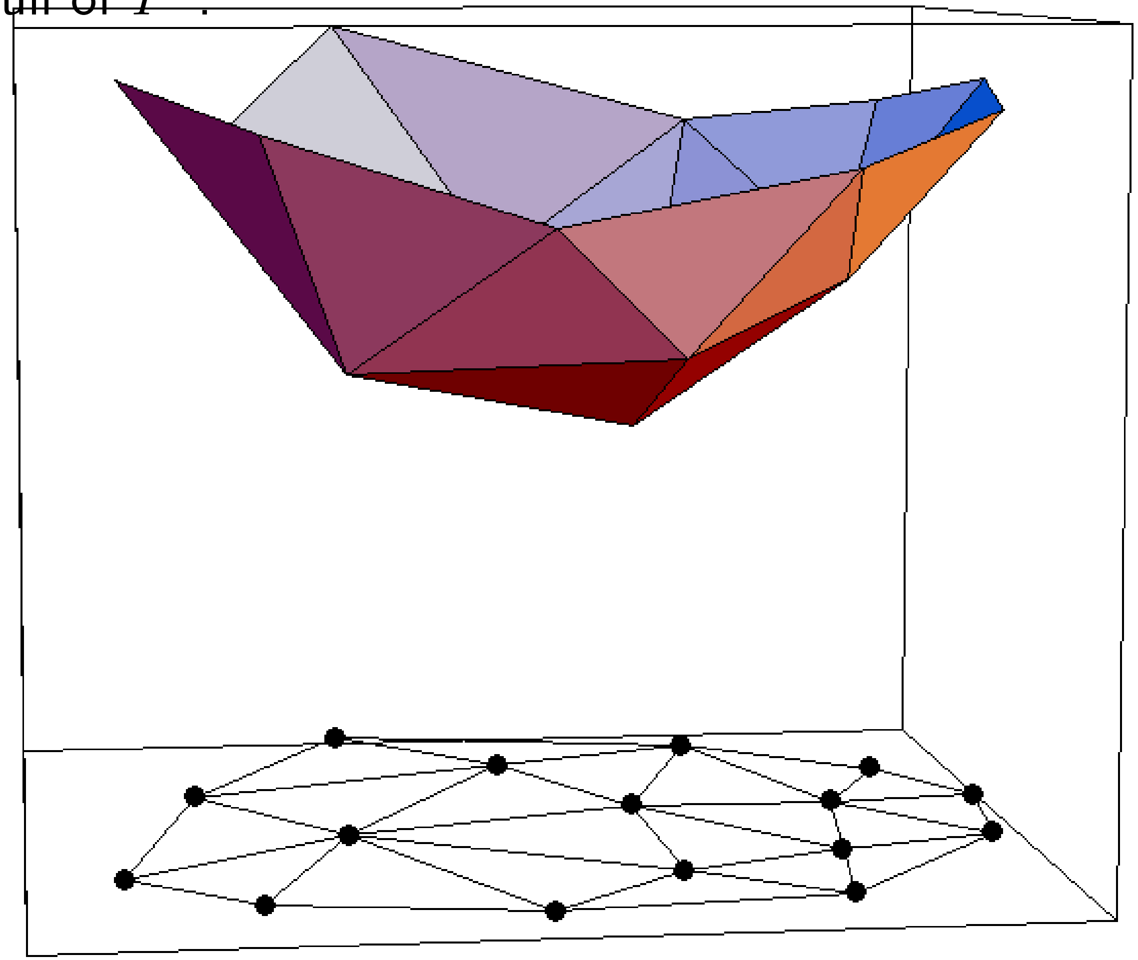
Theorem

Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

p_i^*, p_j^*, p_k^* form a (triangular) face of the lower convex hull of P^*

$$\Updownarrow$$

The plane through p_i^*, p_j^*, p_k^* leaves all the remaining points of P^* above it



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

Let $P = \{p_1, \ldots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

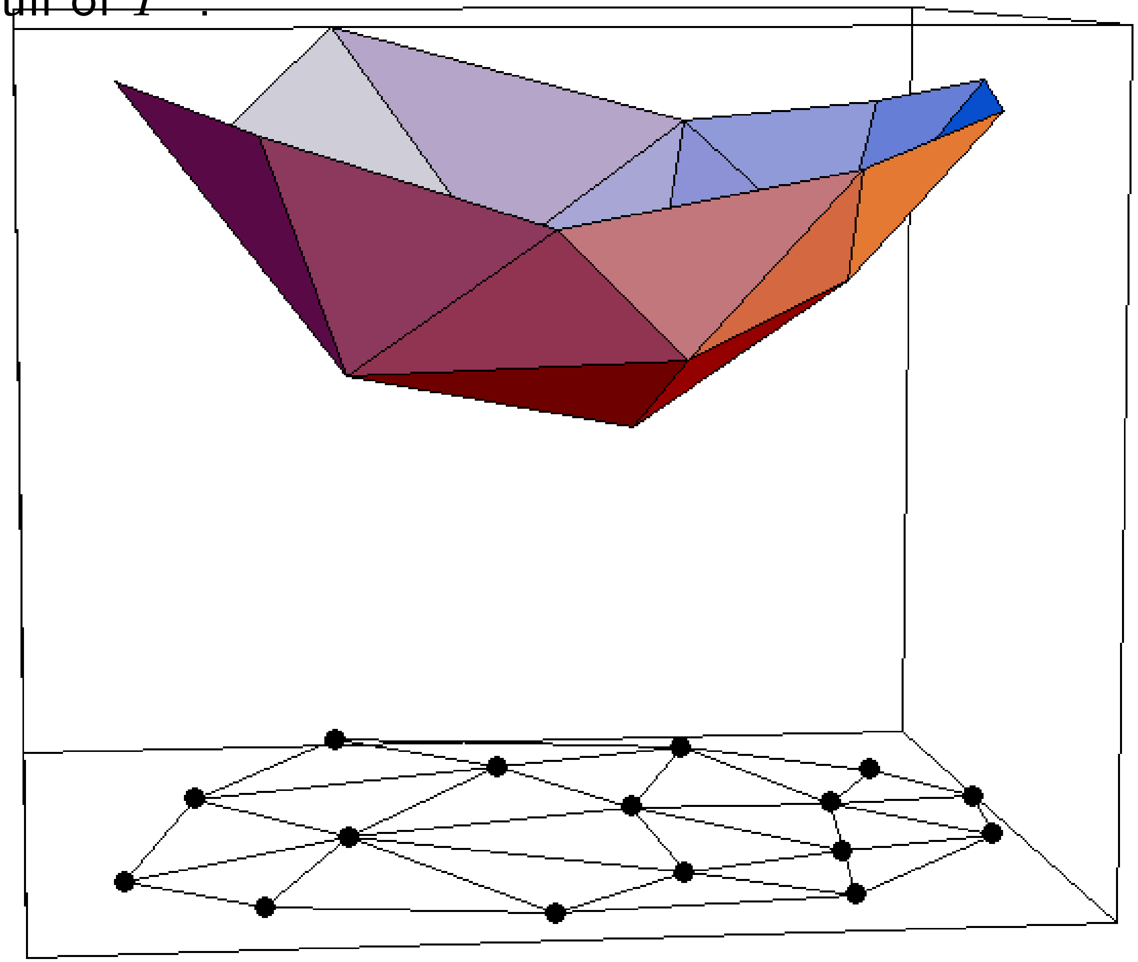
p_i^*, p_j^*, p_k^* form a (triangular) face of the lower convex hull of P^*

$$\Updownarrow$$

The plane through p_i^*, p_j^*, p_k^* leaves all the remaining points of P^* above it

$$\Updownarrow$$

The circle through p_i, p_j, p_k leaves all the remaining points of P in its exterior



DELAUNAY TRIANGULATION AND 3D CONVEX HULL

Theorem

Let $P = \{p_1, \dots, p_n\}$ with $p_i = (a_i, b_i, 0)$. Let $p_i^* = (a_i, b_i, a_i^2 + b_i^2)$ be the vertical projection of each point p_i onto the paraboloid $z = x^2 + y^2$. Then $Del(P)$ is the orthogonal projection onto the plane $z = 0$ of the lower convex hull of P^*.

p_i^*, p_j^*, p_k^* form a (triangular) face of the lower convex hull of P^*

$\Updownarrow$

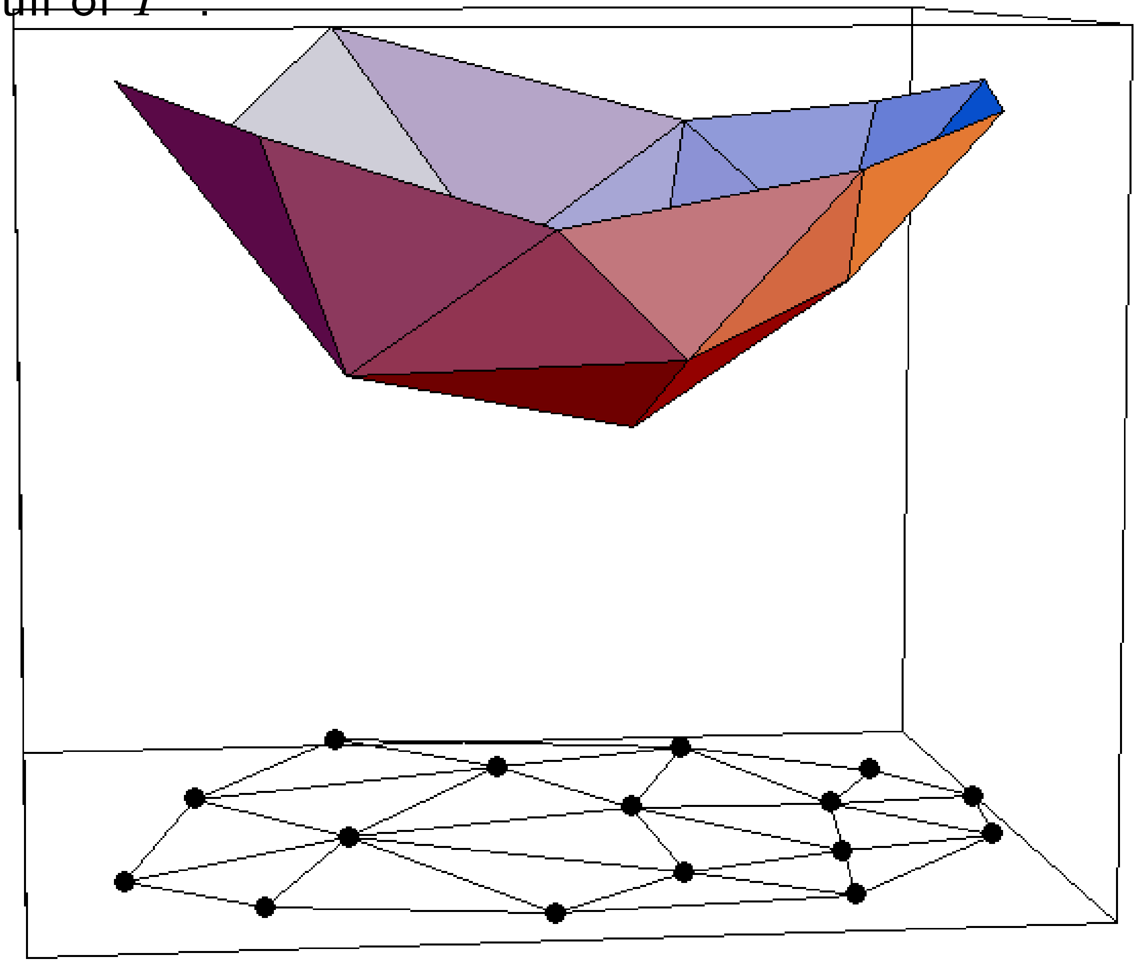
The plane through p_i^*, p_j^*, p_k^* leaves all the remaining points of P^* above it

$\Updownarrow$

The circle through p_i, p_j, p_k leaves all the remaining points of P in its exterior

$\Updownarrow$

p_i, p_j, p_k form a triangle of $Del(P)$



DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

LOCAL CHARACTERIZATION

Definition

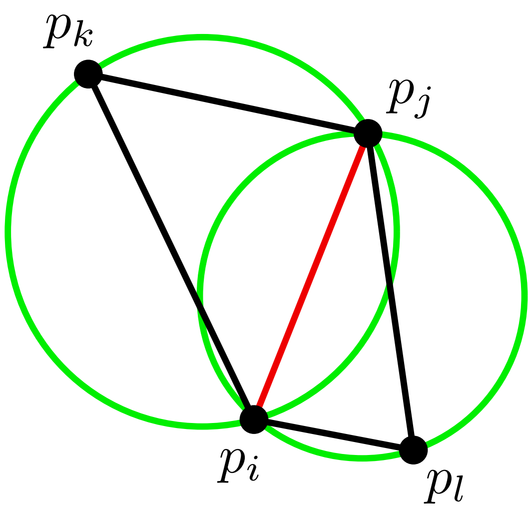
A triangulation $T(P)$ is locally Delaunay if each pair of triangles $p_ip_jp_k$ and $p_ip_jp_l$ sharing an edge p_ip_j satisfies $p_l \notin C_{ijk}$ and $p_k \notin C_{ijl}$.

DELAUNAY TRIANGULATION

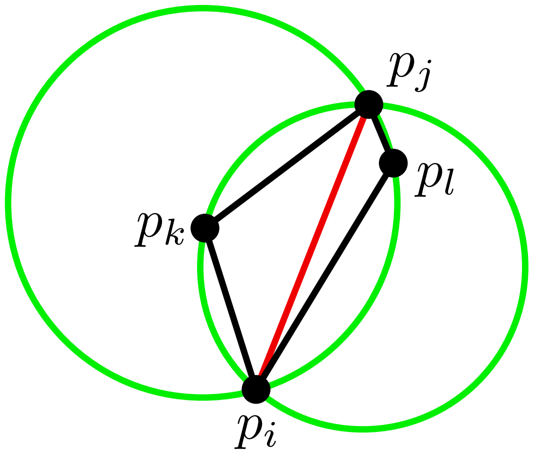
LOCAL CHARACTERIZATION

Definition

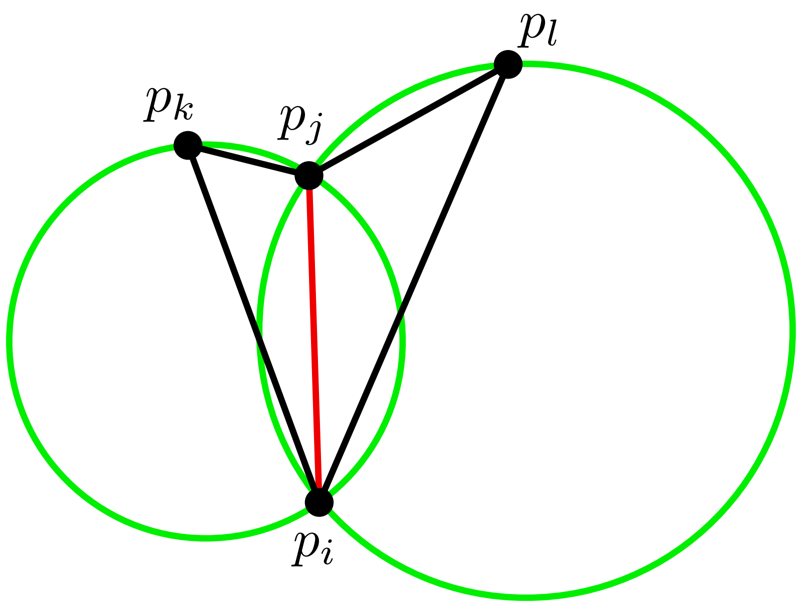
A triangulation $T(P)$ is locally Delaunay if each pair of triangles $p_i p_j p_k$ and $p_i p_j p_l$ sharing an edge $p_i p_j$ satisfies $p_l \notin C_{ijk}$ and $p_k \notin C_{ijl}$.



The edge $p_i p_j$ is locally Delaunay



The edge $p_i p_j$ is not locally Delaunay



The edge $p_i p_j$ is locally Delaunay

Note that the quadrilateral $p_i p_l p_j p_k$ is not convex

LOCAL CHARACTERIZATION

Theorem

A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

Theorem

A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

DELAUNAY TRIANGULATION

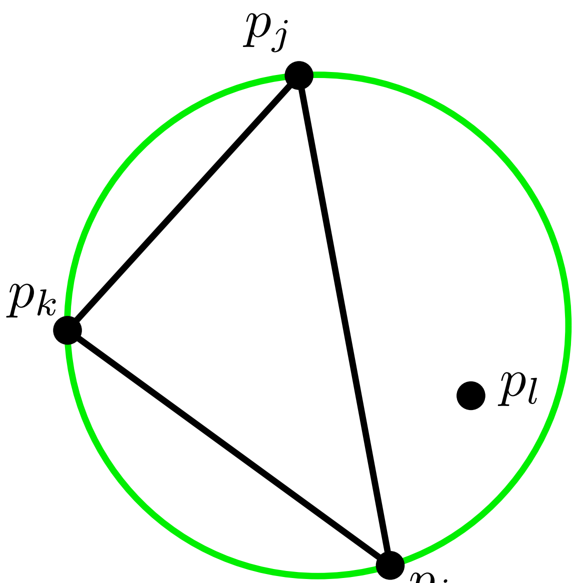
LOCAL CHARACTERIZATION

Theorem

A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.



DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

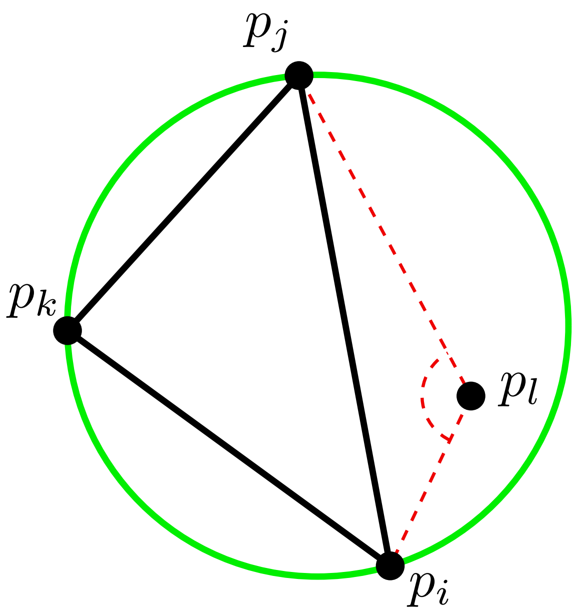
Theorem

A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.

Let $\overline{p_i p_j}$ be the edge of T_{ijk} separating p_l from T_{ijk}. Among all 4-tuples in this situation, let $ijkl$ maximize the angle $p_i p_l p_j$.



DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

Theorem

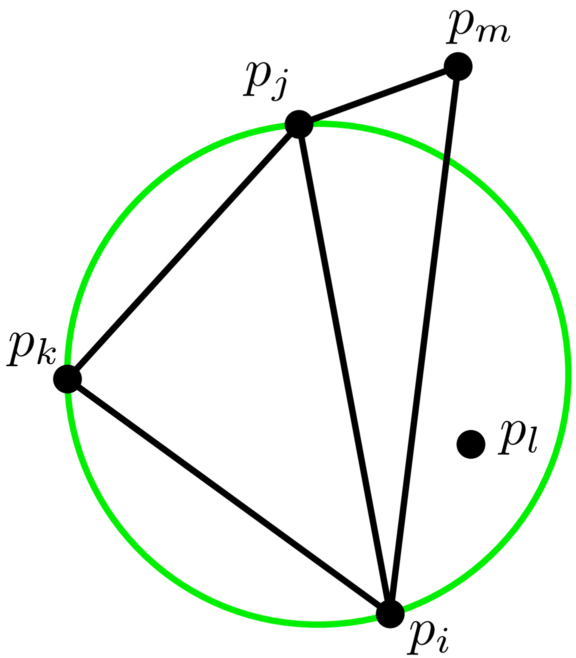
A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.

Let $\overline{p_i p_j}$ be the edge of T_{ijk} separating p_l from T_{ijk}. Among all 4-tuples in this situation, let $ijkl$ maximize the angle $p_i p_l p_j$.

Let T_{ijm} be the triangle adjacent to T_{ijk} through the edge $\overline{p_i p_j}$.



DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

Theorem

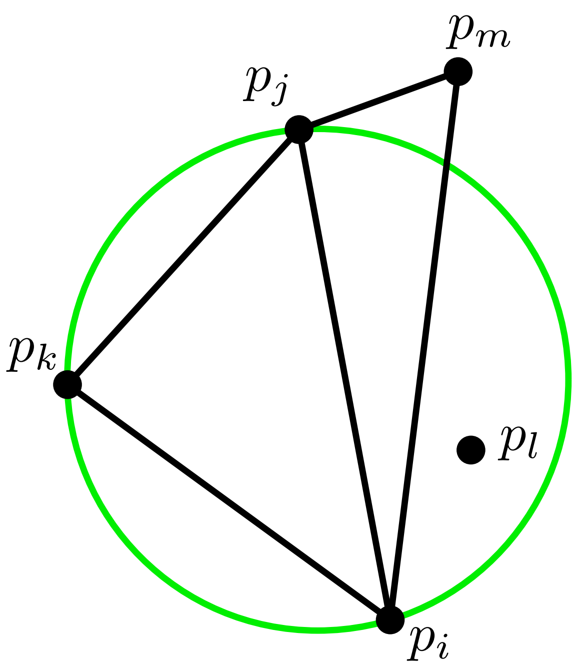
A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.

Let $\overline{p_i p_j}$ be the edge of T_{ijk} separating p_l from T_{ijk}. Among all 4-tuples in this situation, let $ijkl$ maximize the angle $p_i p_l p_j$.

Let T_{ijm} be the triangle adjacent to T_{ijk} through the edge $\overline{p_i p_j}$.



As $T(P)$ is locally Delaunay, $m \neq l$.

DELAUNAY TRIANGULATION

LOCAL CHARACTERIZATION

Theorem

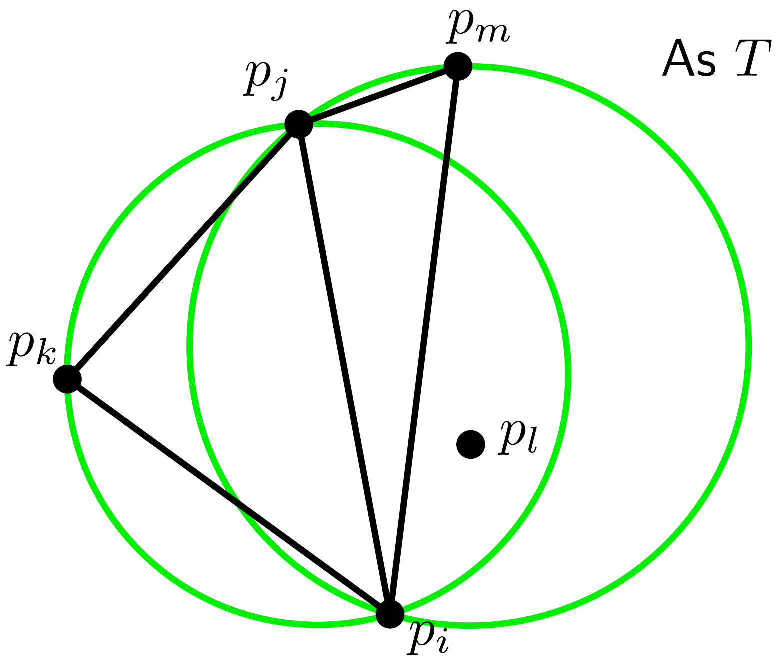
A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.

Let $\overline{p_i p_j}$ be the edge of T_{ijk} separating p_l from T_{ijk}. Among all 4-tuples in this situation, let $ijkl$ maximize the angle $p_i p_l p_j$.

Let T_{ijm} be the triangle adjacent to T_{ijk} through the edge $\overline{p_i p_j}$.



As $T(P)$ is locally Delaunay, $m \neq l$.

Then $p_l \in C_{ijm}$.

LOCAL CHARACTERIZATION

Theorem

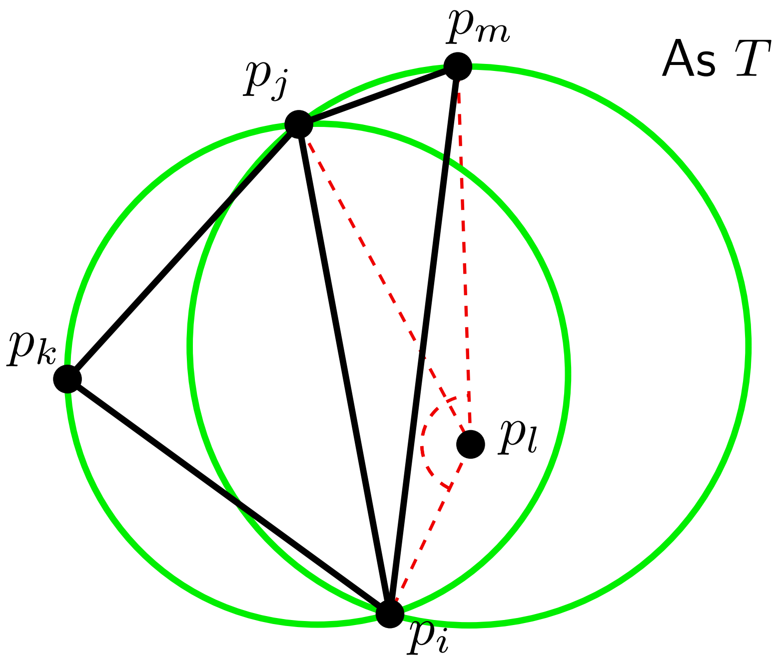
A triangulation $T(P)$ is a Delaunay triangulation if and only if it is locally Delaunay.

Suppose that $T(P)$ was locally Delaunay without being a Delaunay triangulation.

There would exist a triangle $T_{ijk} = p_i p_j p_k$ and a point p_l such that $p_l \in int(C_{ijk})$.

Let $\overline{p_i p_j}$ be the edge of T_{ijk} separating p_l from T_{ijk}. Among all 4-tuples in this situation, let $ijkl$ maximize the angle $p_i p_l p_j$.

Let T_{ijm} be the triangle adjacent to T_{ijk} through the edge $\overline{p_i p_j}$.



As $T(P)$ is locally Delaunay, $m \neq l$.

Then $p_l \in C_{ijm}$.

Hence, one of the angles $p_i p_l p_m$ or $p_j p_l p_m$ would be greater than $p_i p_l p_j$.

DELAUNAY FLIPS

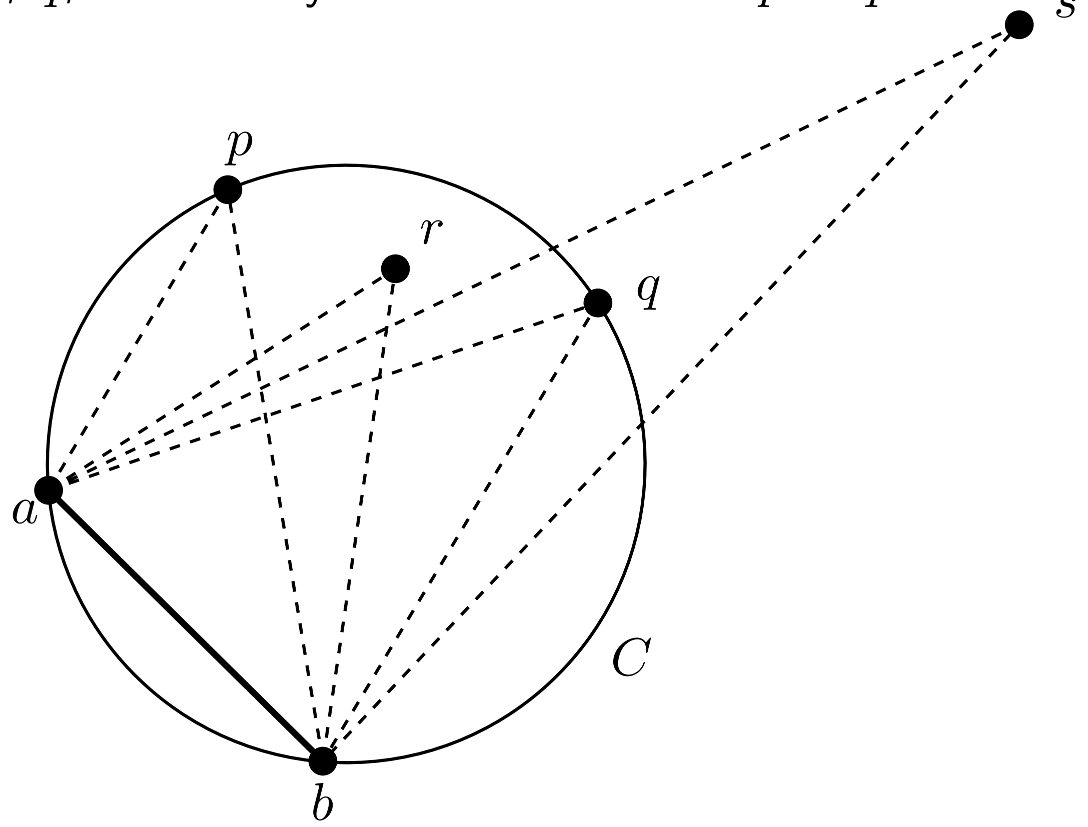
DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 1. Let C be a circle, $\overline{ab}$ a chord of C, and p, q, r and s four points lying to the same side of the line $\overline{ab}$. If r is internal to C, p and q lie in C, and s is external to C, then the following relations hold between the angles formed at p, q, r and s by the chord $\overline{ab}$: $\widehat{s} < \widehat{p} = \widehat{q} < \widehat{r}$.

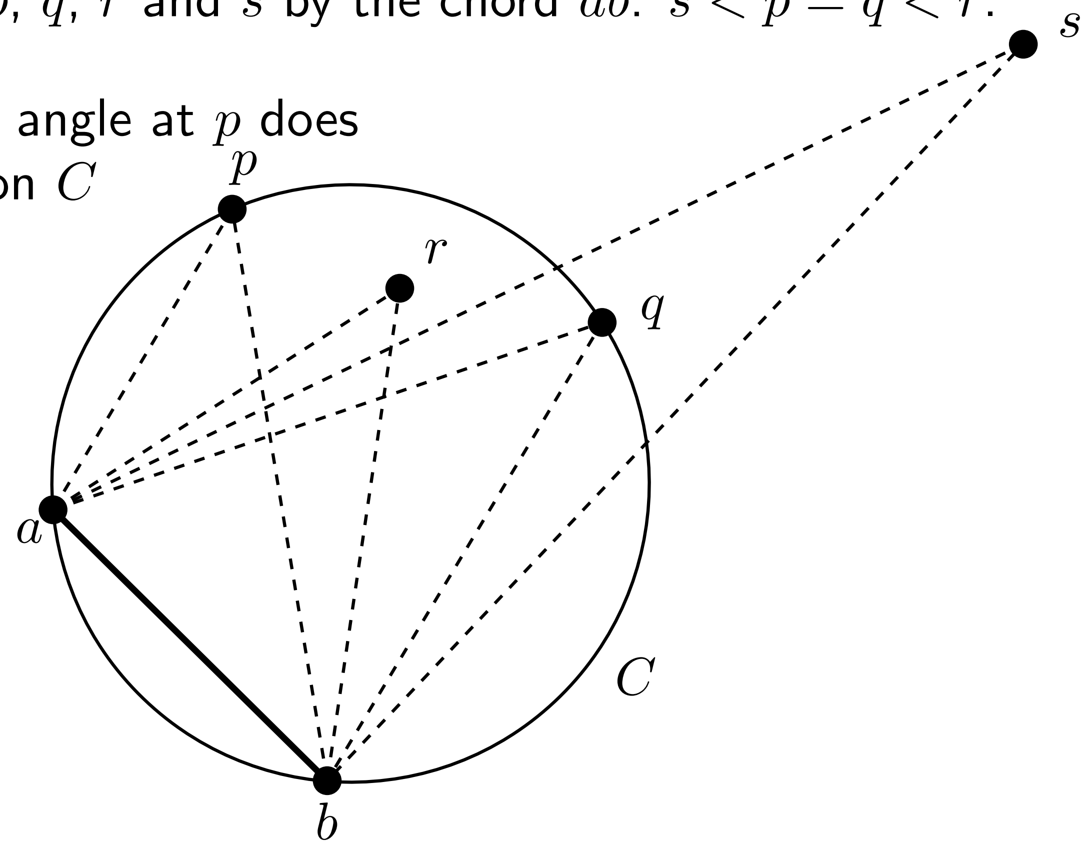


DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 1. Let C be a circle, $\overline{ab}$ a chord of C, and p, q, r and s four points lying to the same side of the line $\overline{ab}$. If r is internal to C, p and q lie in C, and s is external to C, then the following relations hold between the angles formed at p, q, r and s by the chord $\overline{ab}$: $\widehat{s} < \widehat{p} = \widehat{q} < \widehat{r}$.

Let us prove that $\widehat{p} = \widehat{q}$, by showing that the angle at p does not depend on its position, as long as it lies on C



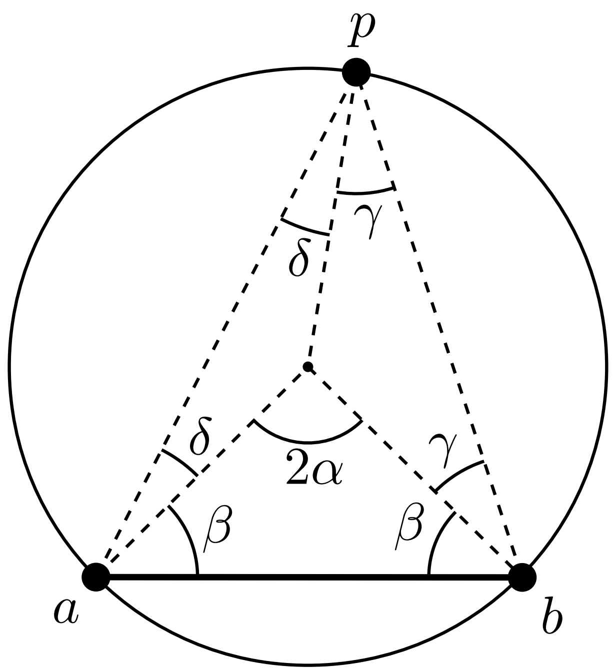
DELAUNAY TRIANGULATION

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 1. Let C be a circle, $\overline{ab}$ a chord of C, and p, q, r and s four points lying to the same side of the line $\overline{ab}$. If r is internal to C, p and q lie in C, and s is external to C, then the following relations hold between the angles formed at p, q, r and s by the chord $\overline{ab}$: $\widehat{s} < \widehat{p} = \widehat{q} < \widehat{r}$.

Let us prove that $\widehat{p} = \widehat{q}$, by showing that the angle at p does not depend on its position, as long as it lies on C



First case:

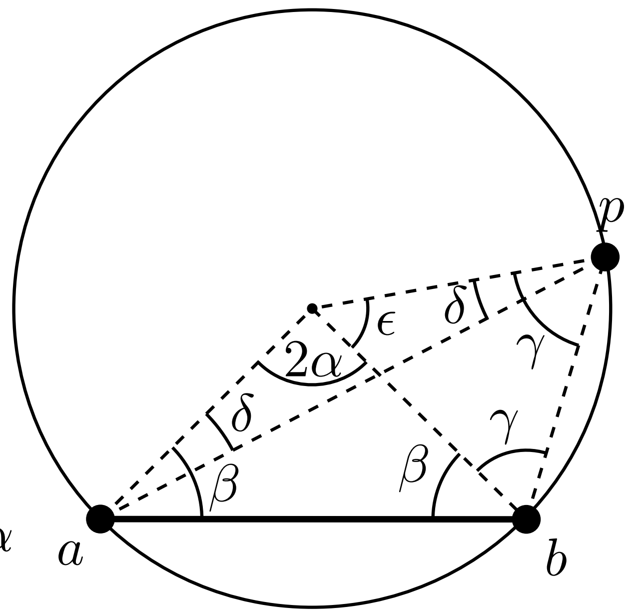
$$\left.\begin{aligned} 2\delta + 2\gamma + 2\beta &= \pi \\ 2\alpha + 2\beta &= \pi \end{aligned}\right\} \Rightarrow 2\alpha = 2\gamma + 2\delta \Rightarrow \alpha = \gamma + \delta \Rightarrow \widehat{p} = \widehat{q} = \alpha$$

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 1. Let C be a circle, $\overline{ab}$ a chord of C, and p, q, r and s four points lying to the same side of the line $\overline{ab}$. If r is internal to C, p and q lie in C, and s is external to C, then the following relations hold between the angles formed at p, q, r and s by the chord $\overline{ab}$: $\widehat{s} < \widehat{p} = \widehat{q} < \widehat{r}$.

Let us prove that $\widehat{p} = \widehat{q}$, by showing that the angle at p does not depend on its position, as long as it lies on C



First case:

$$\left.\begin{aligned} 2\delta + 2\gamma + 2\beta &= \pi \\ 2\alpha + 2\beta &= \pi \end{aligned}\right\} \Rightarrow 2\alpha = 2\gamma + 2\delta \Rightarrow \alpha = \gamma + \delta \Rightarrow \widehat{p} = \widehat{q} = \alpha$$

Second case:

$$\left.\begin{aligned} 2\alpha + \epsilon + 2\delta &= \pi \\ 2\gamma + \epsilon &= \pi \end{aligned}\right\} \Rightarrow 2\alpha + 2\delta - 2\gamma = 0 \Rightarrow \alpha = \gamma - \delta \Rightarrow \widehat{p} = \widehat{q} = \alpha$$

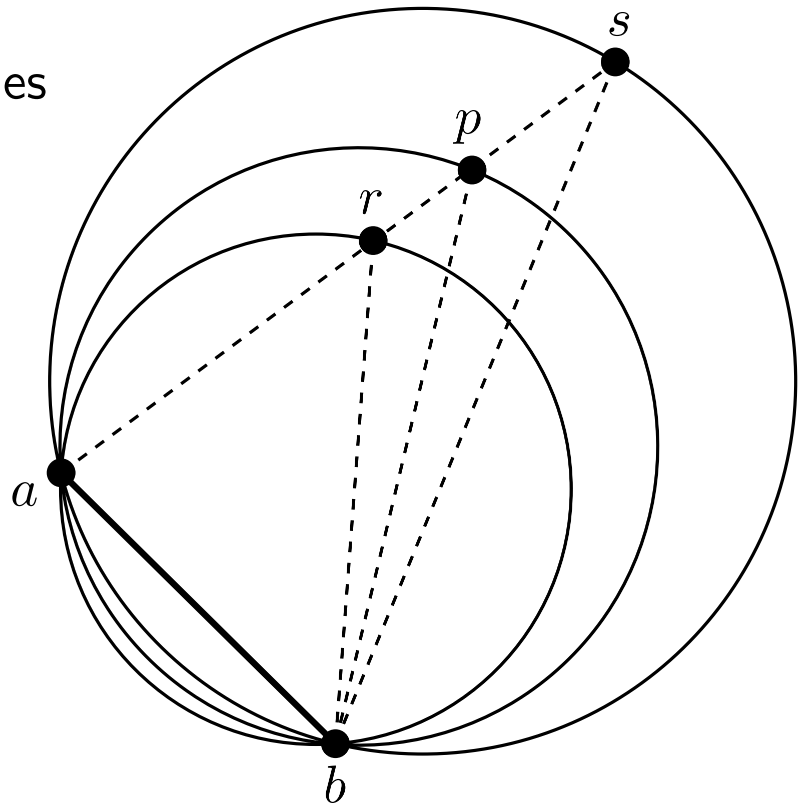
DELAUNAY TRIANGULATION

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 1. Let C be a circle, $\overline{ab}$ a chord of C, and p, q, r and s four points lying to the same side of the line $\overline{ab}$. If r is internal to C, p and q lie in C, and s is external to C, then the following relations hold between the angles formed at p, q, r and s by the chord $\overline{ab}$: $\widehat{s} < \widehat{p} = \widehat{q} < \widehat{r}$.

Let us prove that $\widehat{p} = \widehat{q}$, by showing that the angle at p does not depend on its position, as long as it lies on C



The remaining relations follow immediately.

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

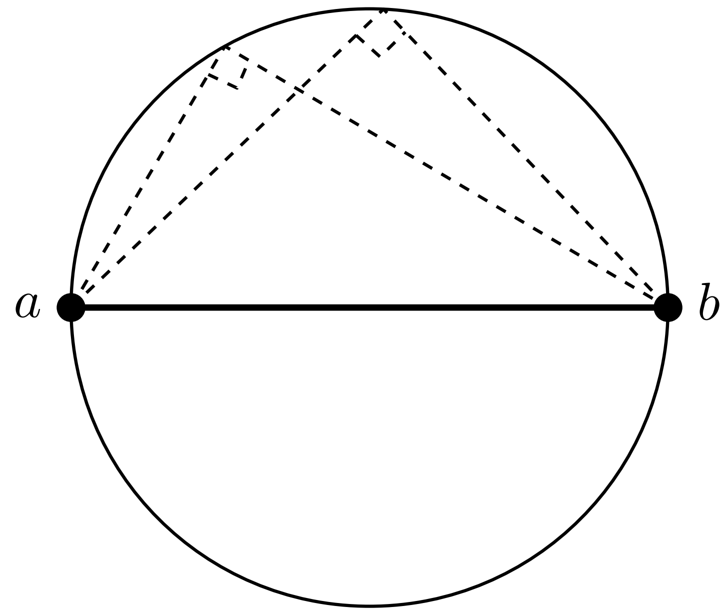
Lemma 2. When the chord $\overline{ab}$ is a diameter of C, the angle $\widehat{p}$ for any $p \in C$ is $\pi/2$.

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 2. When the chord $\overline{ab}$ is a diameter of C, the angle $\widehat{p}$ for any $p \in C$ is $\pi/2$.

Since in this case $2\alpha = \pi$.



DELAUNAY FLIPS

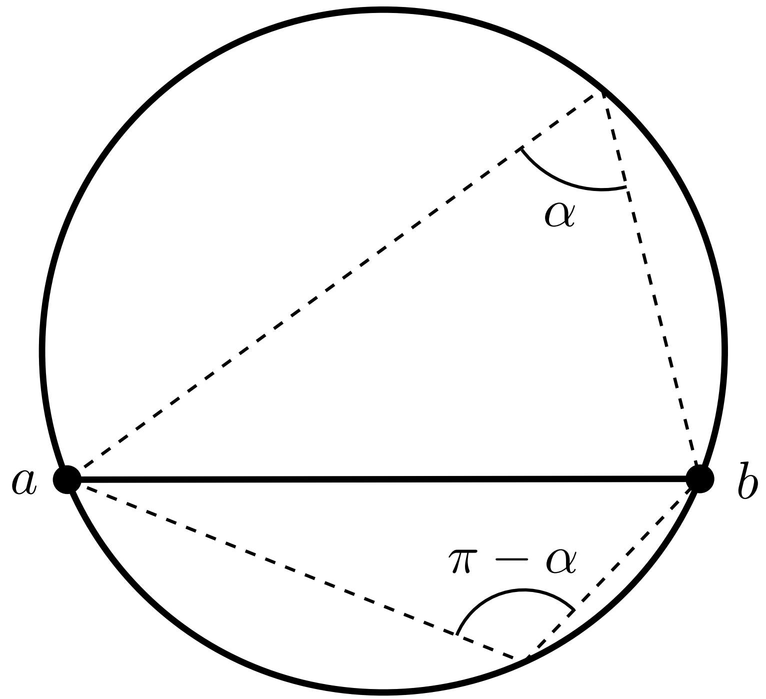
We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 3. Given any chord $\overline{ab}$ in a circle C, if one of the arcs corresponds to α, then the other one corresponds to $\pi - \alpha$.

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

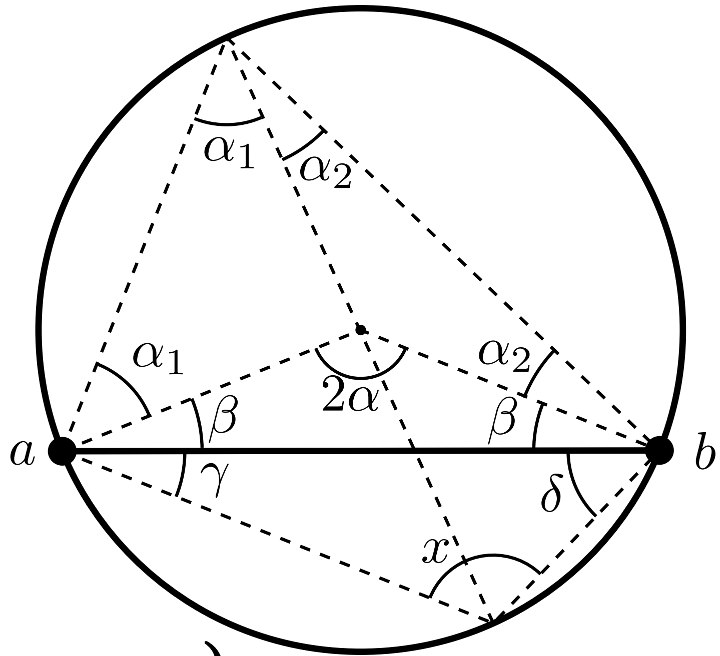
Lemma 3. Given any chord $\overline{ab}$ in a circle C, if one of the arcs corresponds to α, then the other one corresponds to $\pi - \alpha$.



DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 3. Given any chord $\overline{ab}$ in a circle C, if one of the arcs corresponds to α, then the other one corresponds to $\pi - \alpha$.



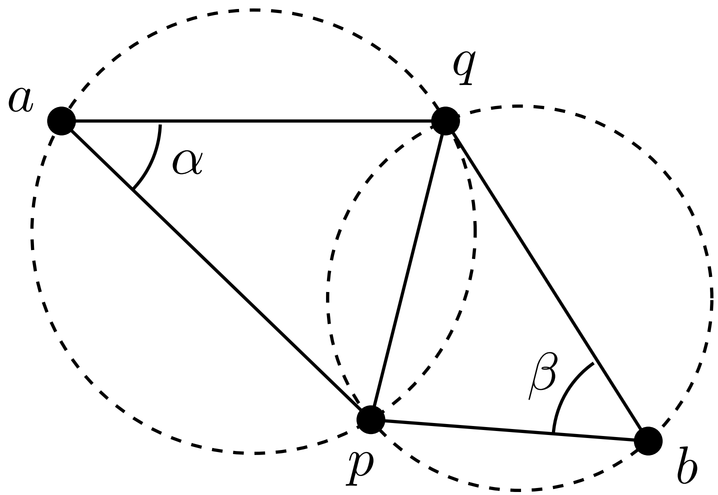
$$\left.\begin{array}{r}\left.\begin{array}{r}\left.\begin{array}{r}\alpha_1+\beta+\gamma=\frac{\pi}{2}\\ \alpha_2+\beta+\delta=\frac{\pi}{2}\end{array}\right\}\Rightarrow \alpha+2\beta+\gamma+\delta=\pi\\ x+\gamma+\delta=\pi\end{array}\right\}\Rightarrow x=\alpha+2\beta\\ 2\alpha+2\beta=\pi\end{array}\right\}\Rightarrow x=\pi-\alpha$$

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 4. Let $\overline{pq}$ be the common edge of the triangles pqa and pqb, forming a convex quatrilateral. Then:

$$a \in ext(C_{pqb}) \iff b \in ext(C_{pqa})$$

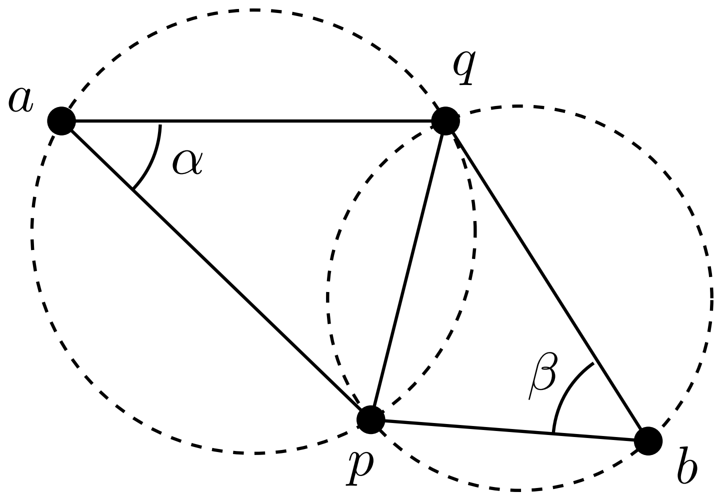


DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 4. Let $\overline{pq}$ be the common edge of the triangles pqa and pqb, forming a convex quatrilateral. Then:

$$a \in ext(C_{pqb}) \iff b \in ext(C_{pqa})$$



$$a \in ext(C_{pqb}) \iff \alpha < \pi - \beta \iff \beta < \pi - \alpha \iff b \in ext(C_{pqa})$$

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 5. Consider a convex quadrilateral with diagonals $\overline{ab}$ and $\overline{pq}$. Then:

$$\overline{ab} \text{ is not locally Delaunay} \Longleftrightarrow \overline{pq} \text{ is locally Delaunay}$$

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Lemma 5. Consider a convex quadrilateral with diagonals $\overline{ab}$ and $\overline{pq}$. Then:

$$\overline{ab} \text{ is not locally Delaunay} \iff \overline{pq} \text{ is locally Delaunay}$$

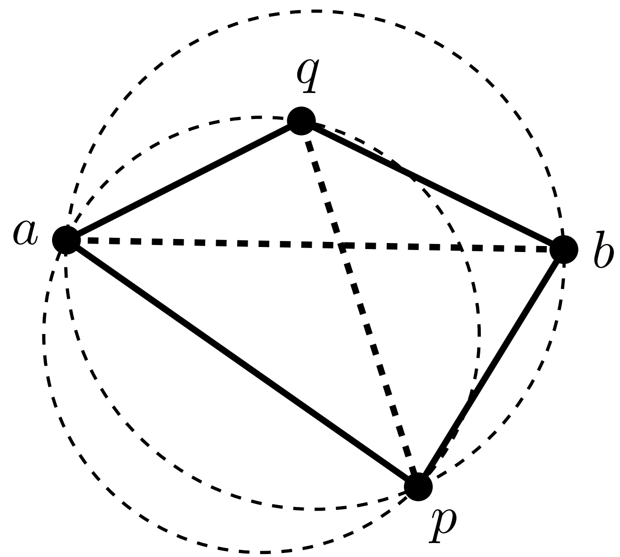
$\overline{ab}$ is not locally Delaunay

$\iff q \in int(C_{abp})$

$\iff \widehat{aqp} > \widehat{abp}$

$\iff b \in ext(C_{apq})$

$\iff \overline{pq}$ is locally Delaunay



DELAUNAY FLIPS

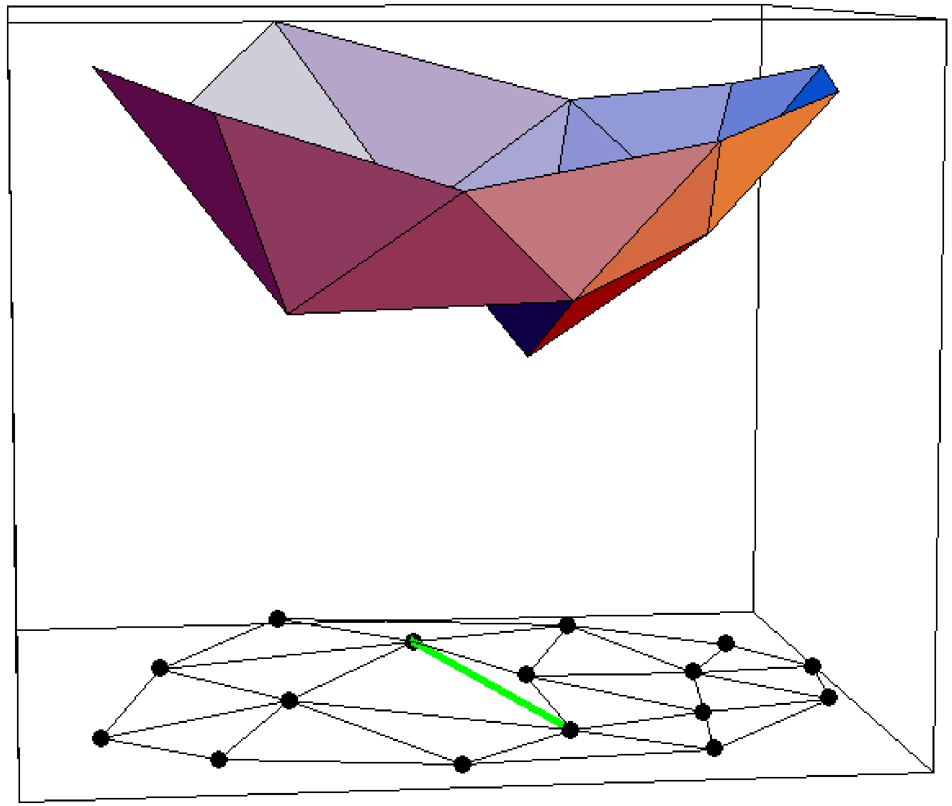
We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Observation. Let P be a set of points $p_i = (x_i, y_i, 0)$ in the plane, and let P^* be the set of their vertical projections $p^* = (x_i, y_i, x_i^2 + y_i^2)$ onto the unit paraboloid. Producing a Delaunay flip in a triangulation of P corresponds to "sticking" a tetrahedron from below to the corresponding polyhedrization of P^*.

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Observation. Let P be a set of points $p_i = (x_i, y_i, 0)$ in the plane, and let P^* be the set of their vertical projections $p^* = (x_i, y_i, x_i^2 + y_i^2)$ onto the unit paraboloid. Producing a Delaunay flip in a triangulation of P corresponds to “sticking” a tetrahedron from below to the corresponding polyhedrization of P^*.

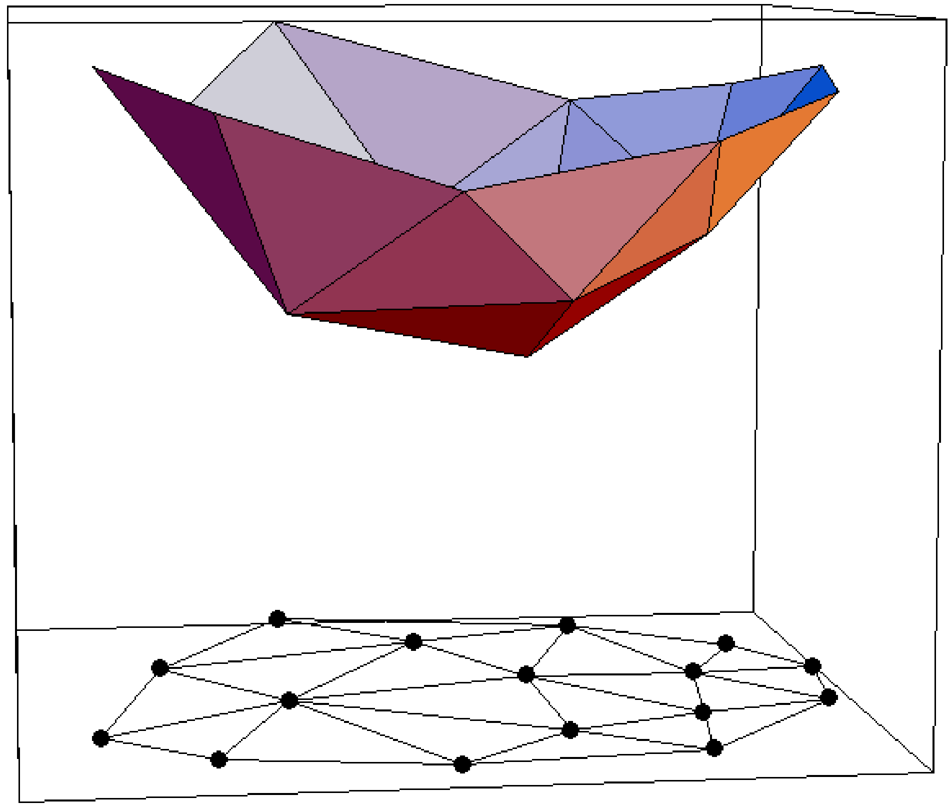


DELAUNAY TRIANGULATION

DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Observation. Let P be a set of points $p_i = (x_i, y_i, 0)$ in the plane, and let P^* be the set of their vertical projections $p^* = (x_i, y_i, x_i^2 + y_i^2)$ onto the unit paraboloid. Producing a Delaunay flip in a triangulation of P corresponds to "sticking" a tetrahedron from below to the corresponding polyhedrization of P^*.



DELAUNAY TRIANGULATION

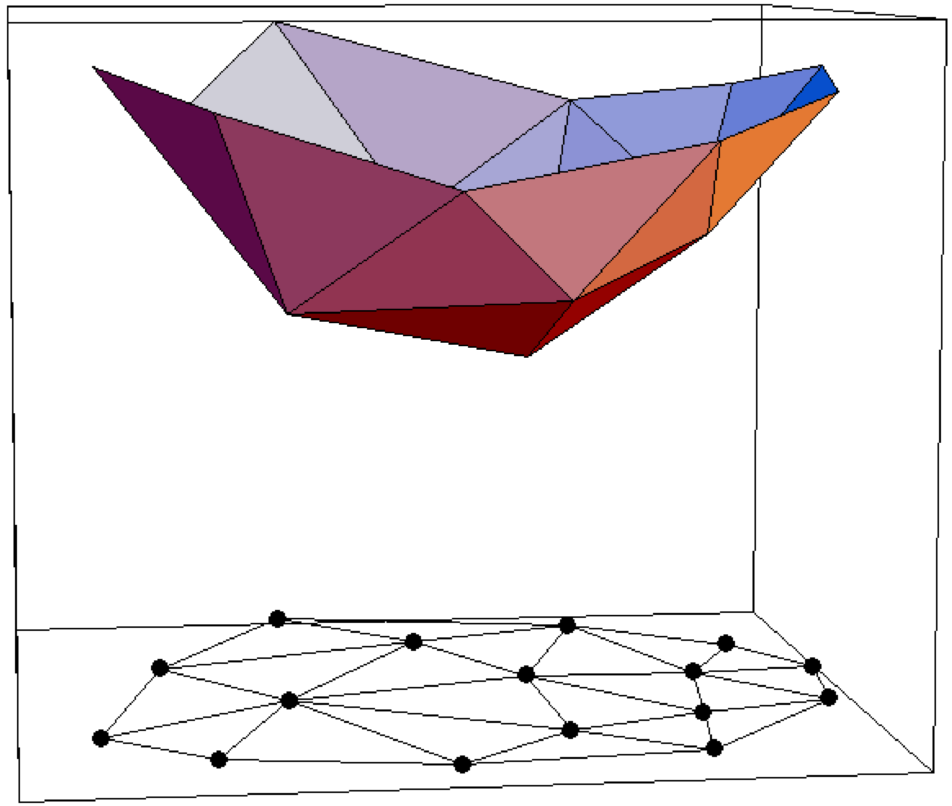
DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Observation. Let P be a set of points $p_i = (x_i, y_i, 0)$ in the plane, and let P^* be the set of their vertical projections $p^* = (x_i, y_i, x_i^2 + y_i^2)$ onto the unit paraboloid. Producing a Delaunay flip in a triangulation of P corresponds to "sticking" a tetrahedron from below to the corresponding polyhedrization of P^*.

Once flipped, the quatrilateral is locally Delaunay: the fourth point lies in the exterior of the circumcircle of the triangle.

In the paraboloid, this means that the fourth point lies above the triangular face of the polyhedrization.



DELAUNAY FLIPS

We intend to prove that $Del(P)$ can be obtained from any triangulation of P by Delaunay flips, which consist in deleting the diagonal of a convex quadrilateral if it is not locally Delaunay, and replacing it by the other diagonal of the quadrilateral.

Corollary. Given any triangulation of P, performing locally Delaunay flips is a procedure converging to $Del(P)$.

ALGORITHMS

ALGORITHMS

1. Project the points onto the paraboloid, compute the 3D convex hull by any of the known methods, and project it back onto the plane.

DELAUNAY TRIANGULATION

ALGORITHMS

1. Project the points onto the paraboloid, compute the 3D convex hull by any of the known methods, and project it back onto the plane.

2. Compute a triangulation, by any of the known methods, and apply Delaunay flips.

DELAUNAY TRIANGULATION

ALGORITHMS

1. Project the points onto the paraboloid, compute the 3D convex hull by any of the known methods, and project it back onto the plane.

2. Compute a triangulation, by any of the known methods, and apply Delaunay flips.

3. Incremental algorithm

Compute an enclosing triangle for $\{p_1, \ldots, p_n\}$

Compute $Del(p_1, \ldots, p_{i+1})$ from $Del(p_1, \ldots, p_i)$

INCREMENTAL ALGORITHM

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

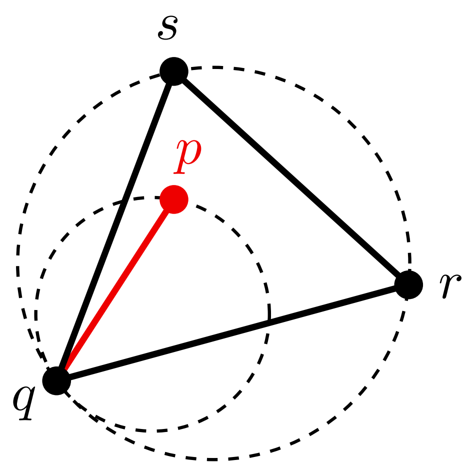
DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

As C_{qrs} is empty, there exist empty circles C_{pq}, such as the circle through p and q tangent to C_{qrs} in q. Similarly for r and s.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

DELAUNAY TRIANGULATION

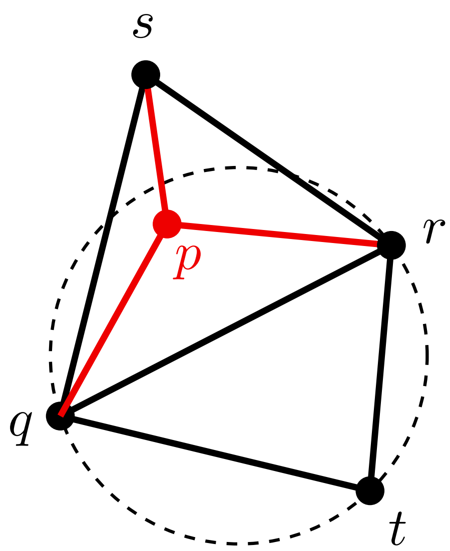
INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Since p may lie in the interior of C_{qrt}.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

> This holds because the property is local: it affects only quadrilaterals formed by two triangles sharing an edge.

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

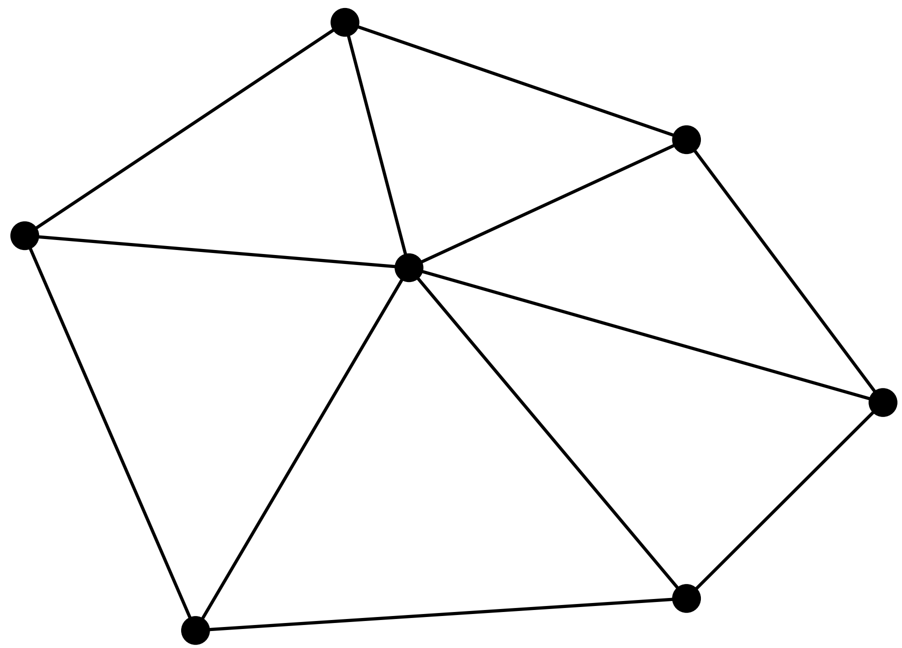
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

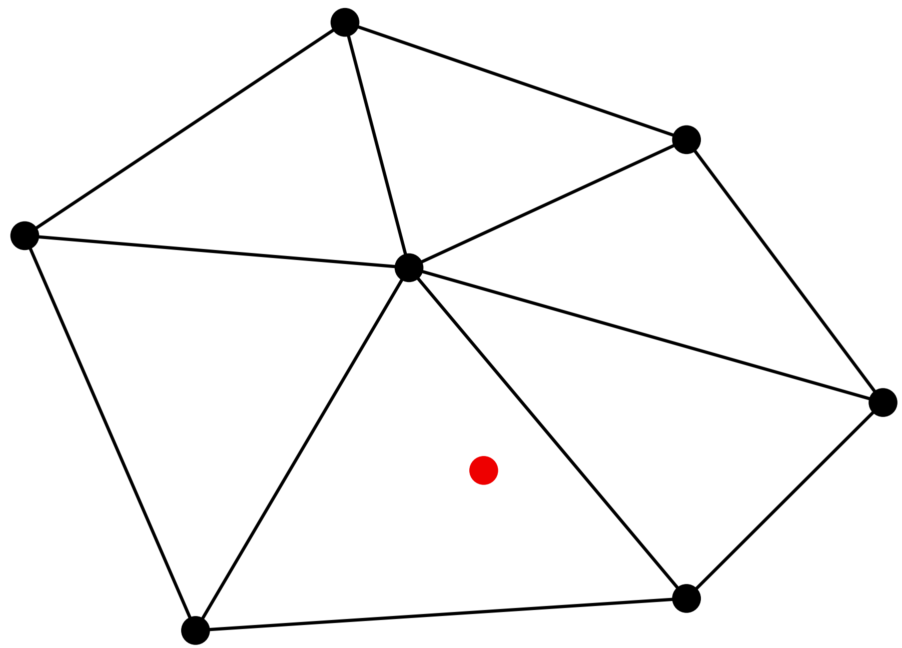
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

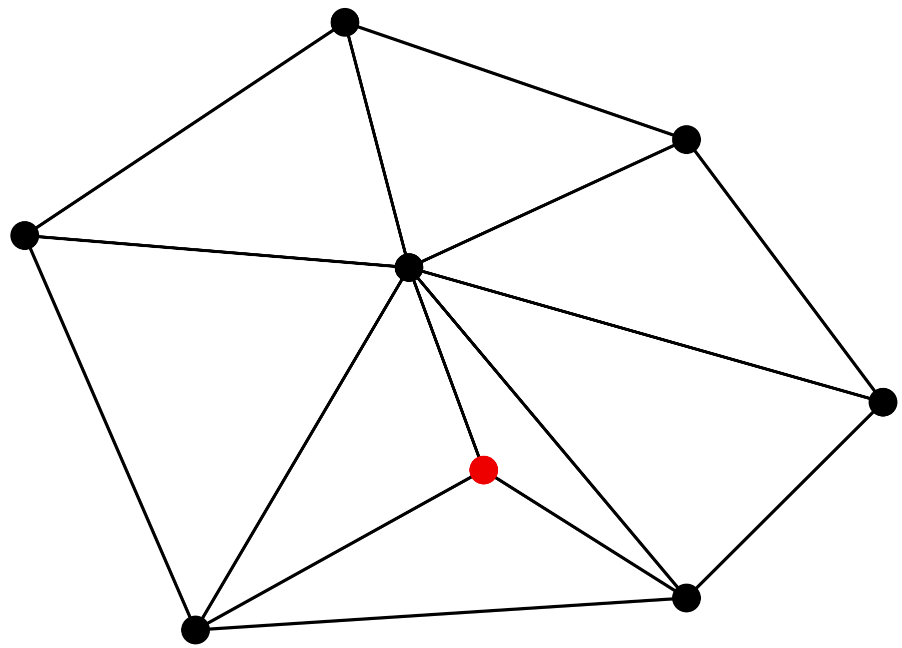
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

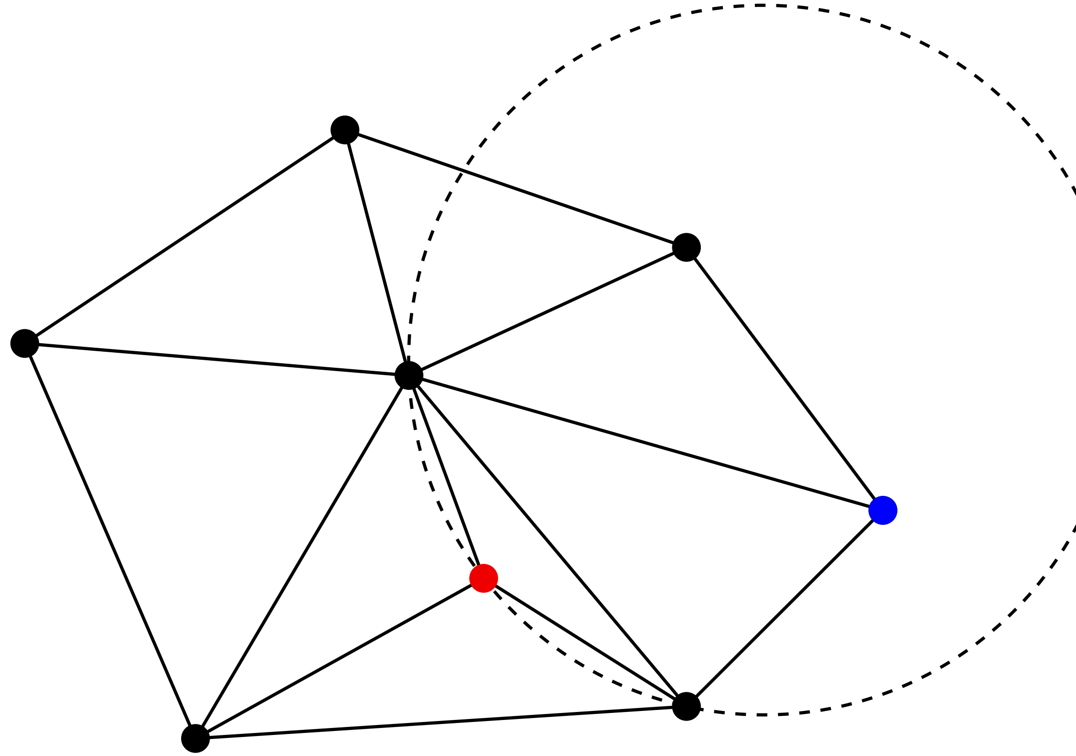
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

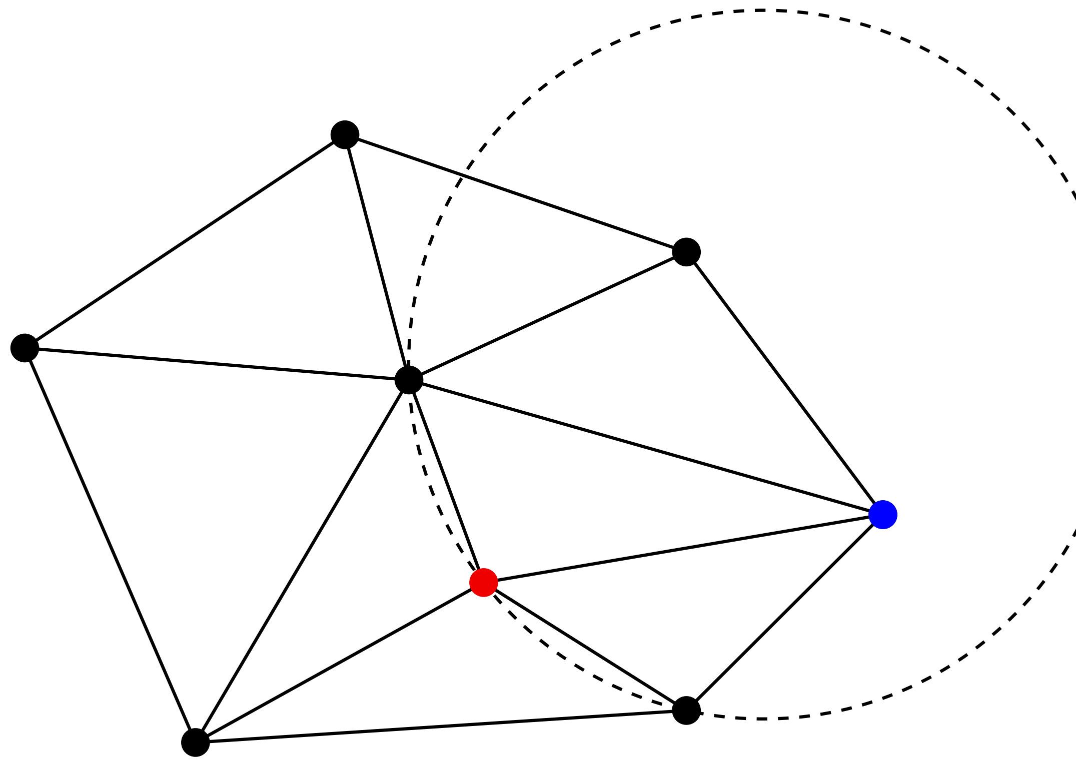
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

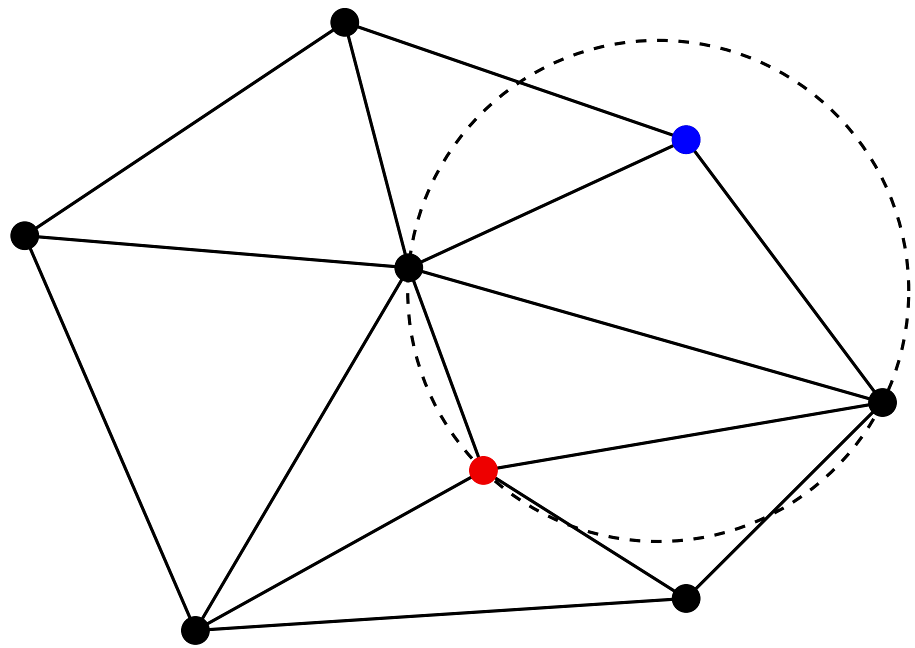
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

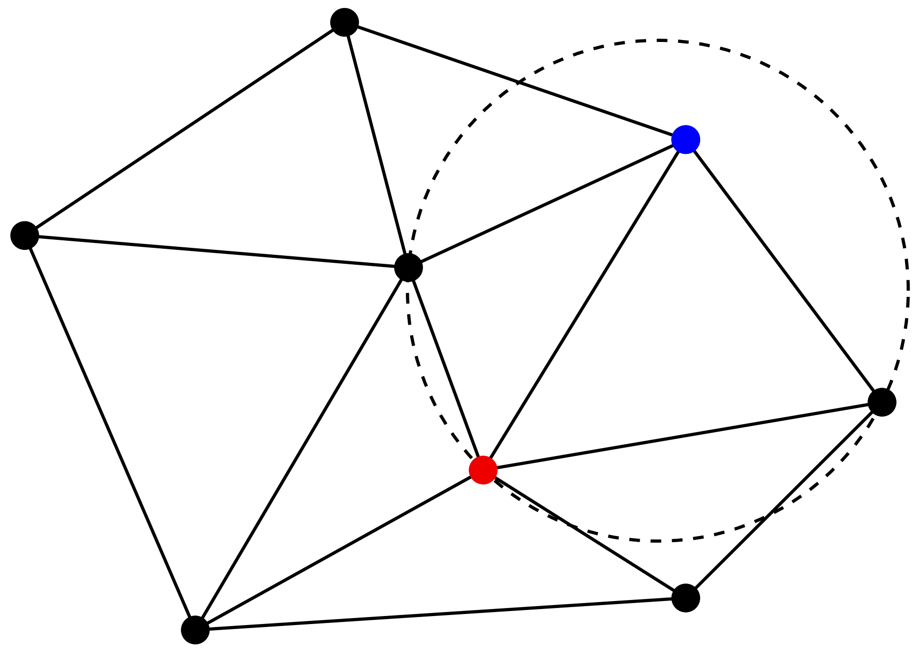
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

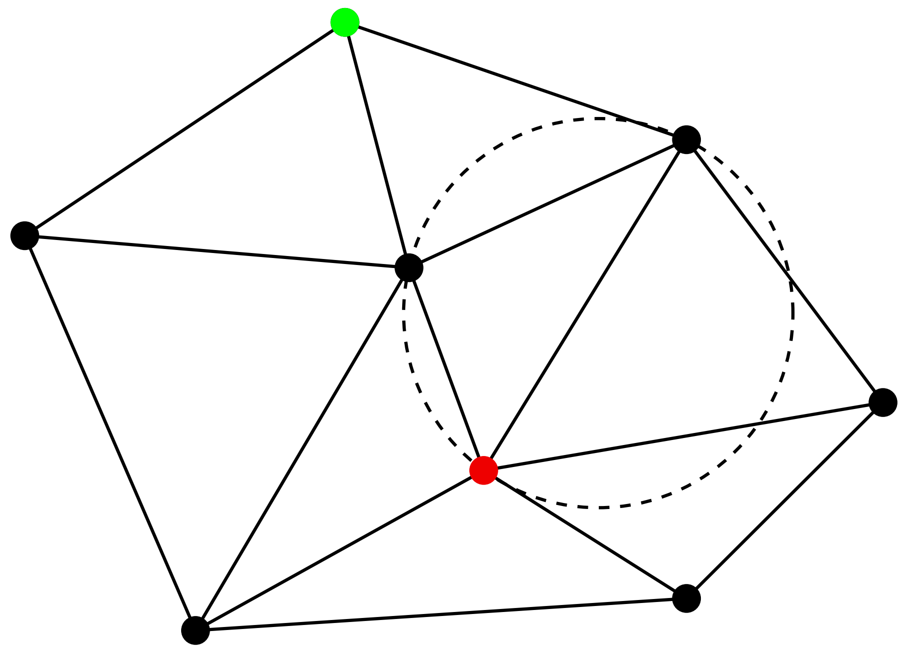
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

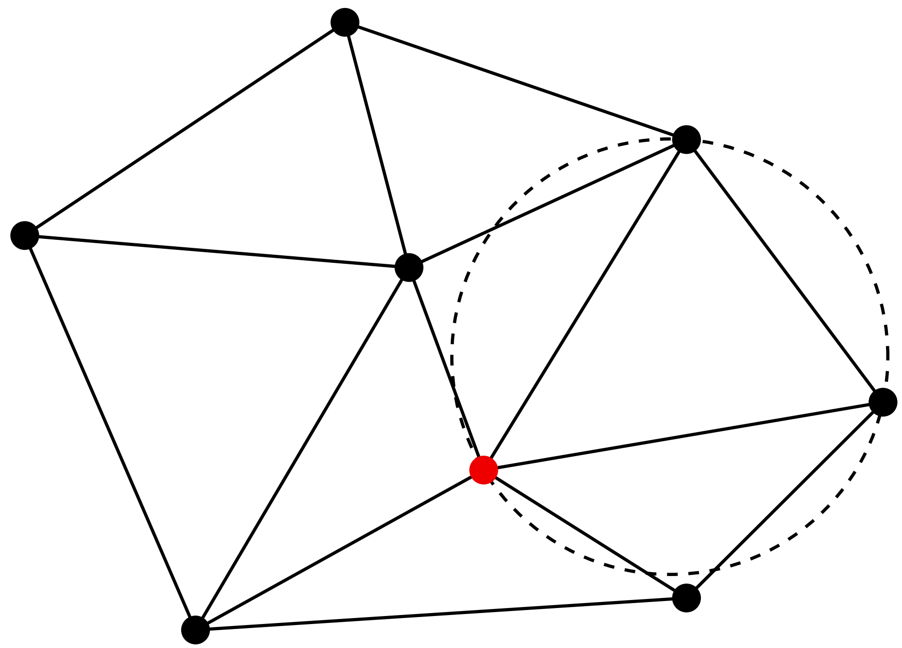
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

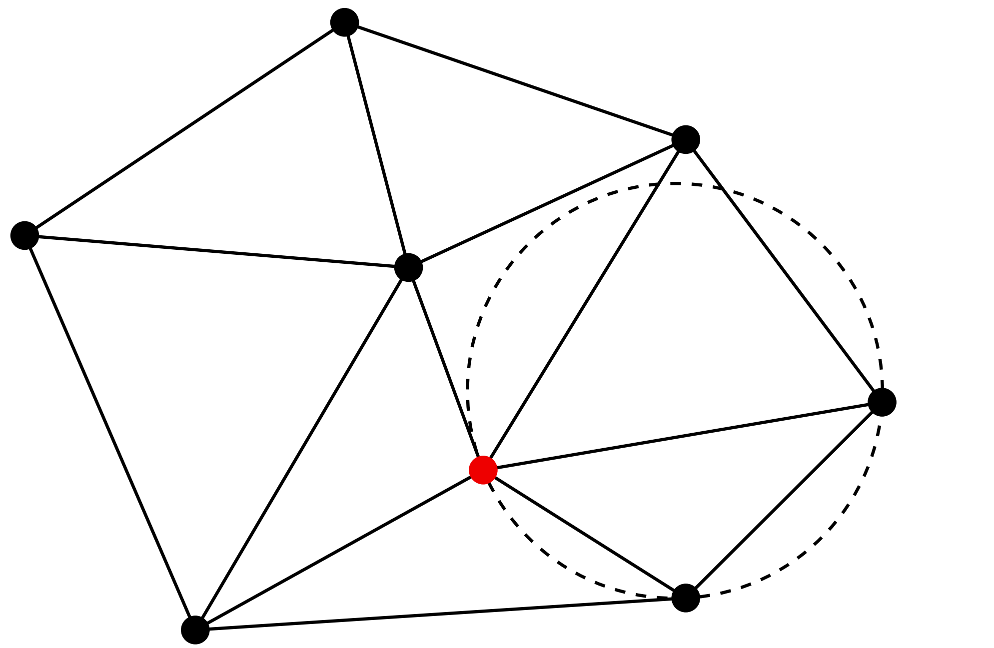
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

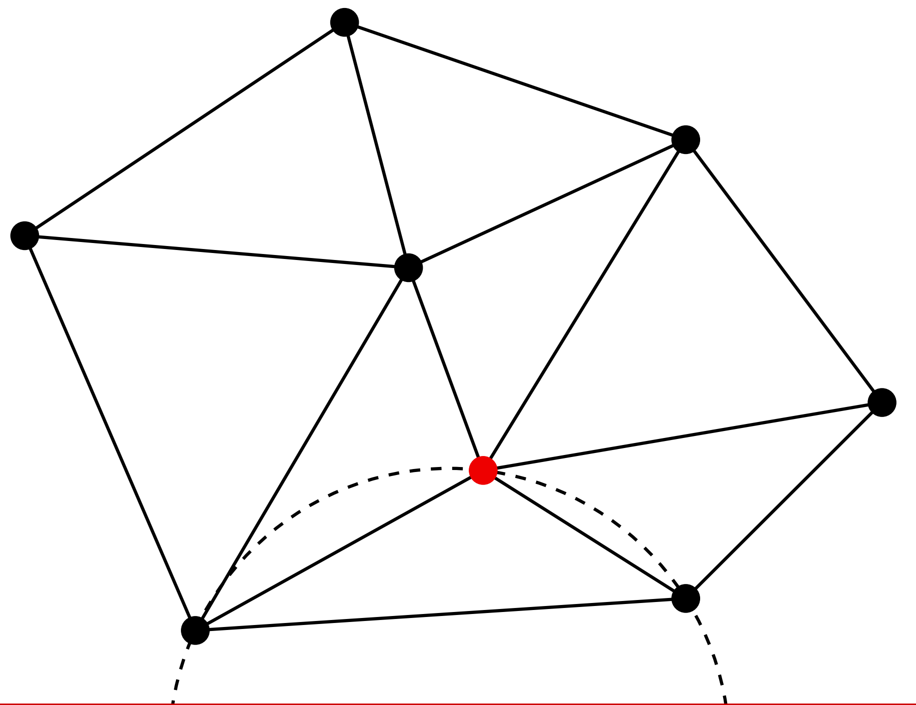
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

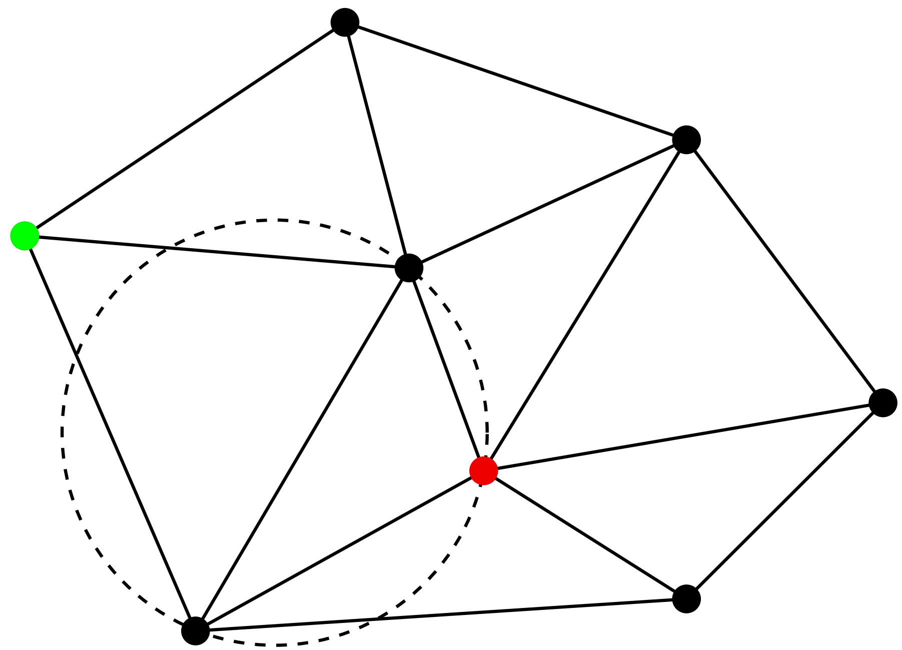
Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.



DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.

Added running time

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.

Added running time

The added running time of performing the flips when adding p_i is

$$O(\text{degree of } p_i \text{ in } D_i) = O(n).$$

DELAUNAY TRIANGULATION

INCREMENTAL ALGORITHM

Let $D_i = Del(p_1, \ldots, p_i)$ and $p = p_{i+1}$.

Observation 1. If qrs is the triangle of D_i containing p, then $\overline{pq}$, $\overline{pr}$ and $\overline{ps}$ are edges of D_{i+1}.

Observation 2. Let pqr be a triangle incident to p. The edge $\overline{qr}$ may not be a Delaunay edge.

Observation 3. The insertion of the point p can only violate the Delaunay property of the triangles incident to p.

Algorithm

Each time a new point is added to the triangulation, and before adding the next point, the following routine is executed:

Flips

While there are still edges of triangles incident to p that are non-locally Delaunay, flip them.

Added running time

The added running time of performing the flips when adding p_i is

$$O(\text{degree of } p_i \text{ in } D_i) = O(n).$$

Worst case: $O(n^2)$ time. However, as average degree is smaller than 6, the **expected** running time, if points are added in random order, is $O(n \log n)$.

DELAUNAY TRIANGULATION AND EQUIANGULARITY

DELAUNAY TRIANGULATION

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Among all the triangulations of P, the Delaunay triangulation maximizes the minimum angle (the angles of $Del(P)$ are less acute).

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Among all the triangulations of P, the Delaunay triangulation maximizes the minimum angle (the angles of $Del(P)$ are less acute).

Let us be more precise:

If $\mathcal{T} = \{T_1, \ldots, T_t\}$ is a triangulation of P, the "fineness" of $\mathcal{T}$ is the increasingly sorted list of the angles of all the triangles T_i of $\mathcal{T}$: $F(\mathcal{T}) = (\alpha_1, \ldots, \alpha_{3t})$.

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Among all the triangulations of P, the Delaunay triangulation maximizes the minimum angle (the angles of $Del(P)$ are less acute).

Let us be more precise:

If $\mathcal{T} = \{T_1, \ldots, T_t\}$ is a triangulation of P, the "fineness" of $\mathcal{T}$ is the increasingly sorted list of the angles of all the triangles T_i of $\mathcal{T}$: $F(\mathcal{T}) = (\alpha_1, \ldots, \alpha_{3t})$.

Since every triangulation of P has $t = 2n - h - 2$ triangles, these $3t$-tuples can be compared and lexicographically sorted.

DELAUNAY TRIANGULATION

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Among all the triangulations of P, the Delaunay triangulation maximizes the minimum angle (the angles of $Del(P)$ are less acute).

Let us be more precise:

If $\mathcal{T} = \{T_1, \ldots, T_t\}$ is a triangulation of P, the "fineness" of $\mathcal{T}$ is the increasingly sorted list of the angles of all the triangles T_i of $\mathcal{T}$: $F(\mathcal{T}) = (\alpha_1, \ldots, \alpha_{3t})$.

Since every triangulation of P has $t = 2n - h - 2$ triangles, these $3t$-tuples can be compared and lexicographically sorted.

The Delaunay triangulation maximizes the "fineness":

$$F(Del(P)) \geq F(\mathcal{T}), \quad \forall \mathcal{T} \text{ triangulation of } P.$$

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Among all the triangulations of P, the Delaunay triangulation maximizes the minimum angle (the angles of $Del(P)$ are less acute).

Let us be more precise:

If $\mathcal{T} = \{T_1, \ldots, T_t\}$ is a triangulation of P, the "fineness" of $\mathcal{T}$ is the increasingly sorted list of the angles of all the triangles T_i of $\mathcal{T}$: $F(\mathcal{T}) = (\alpha_1, \ldots, \alpha_{3t})$.

Since every triangulation of P has $t = 2n - h - 2$ triangles, these $3t$-tuples can be compared and lexicographically sorted.

The Delaunay triangulation maximizes the "fineness":

$$F(Del(P)) \geq F(\mathcal{T}), \quad \forall \mathcal{T} \text{ triangulation of } P.$$

The proof of this statement requires a last lemma.

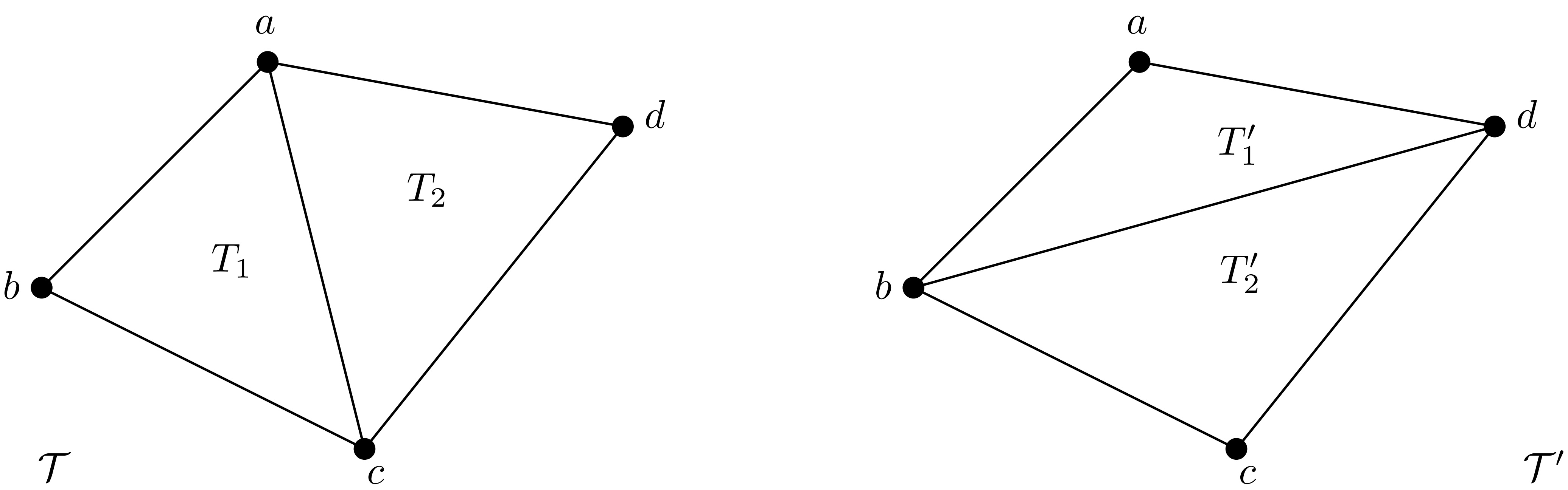
DELAUNAY TRIANGULATION AND EQUIANGULARITY

Lemma 7. Let a, b, c and d be four points forming a convex quadrilateral, in counterclockwise order. Let $\mathcal{T}$ and $\mathcal{T}'$ be the two possible triangulations of the quadrilateral: $\mathcal{T}$ uses the diagonal $\overline{ac}$ and $\mathcal{T}'$ uses $\overline{bd}$. Let ϵ and ϵ' respectively be the minimum angles of $\mathcal{T}$ and $\mathcal{T}'$. Then:

$$\epsilon > \epsilon' \iff d \in ext(C_{abc})$$

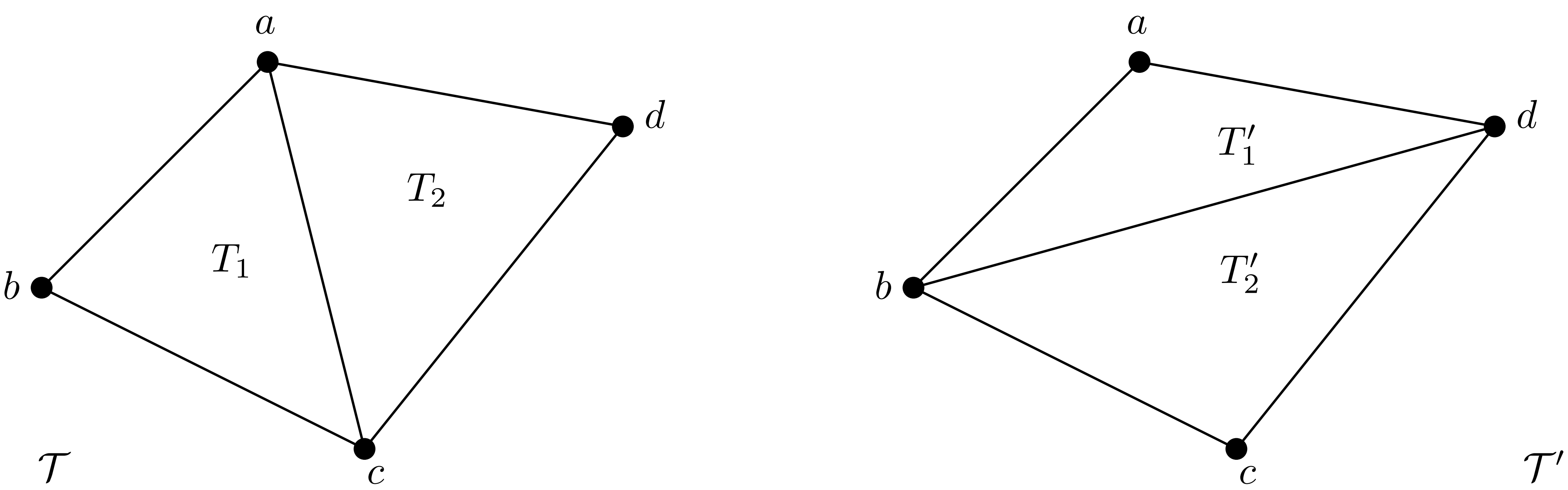
$$\epsilon = \epsilon' \iff d \in \partial(C_{abc})$$

$$\epsilon < \epsilon' \iff d \in int(C_{abc})$$



DELAUNAY TRIANGULATION AND EQUIANGULARITY

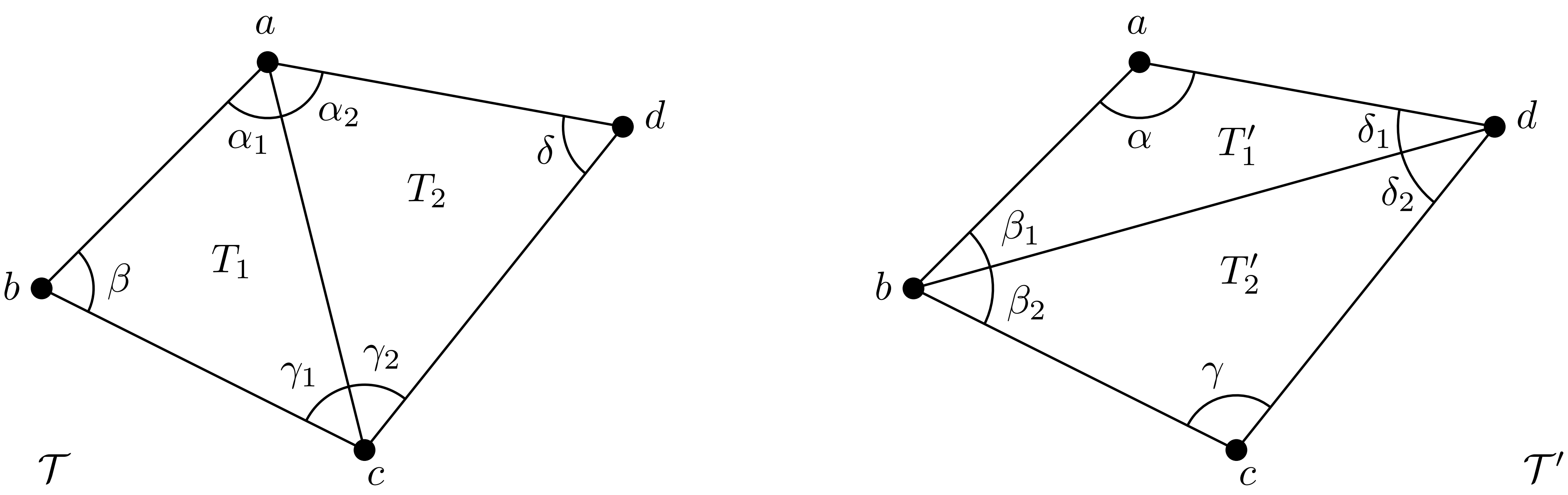
Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$



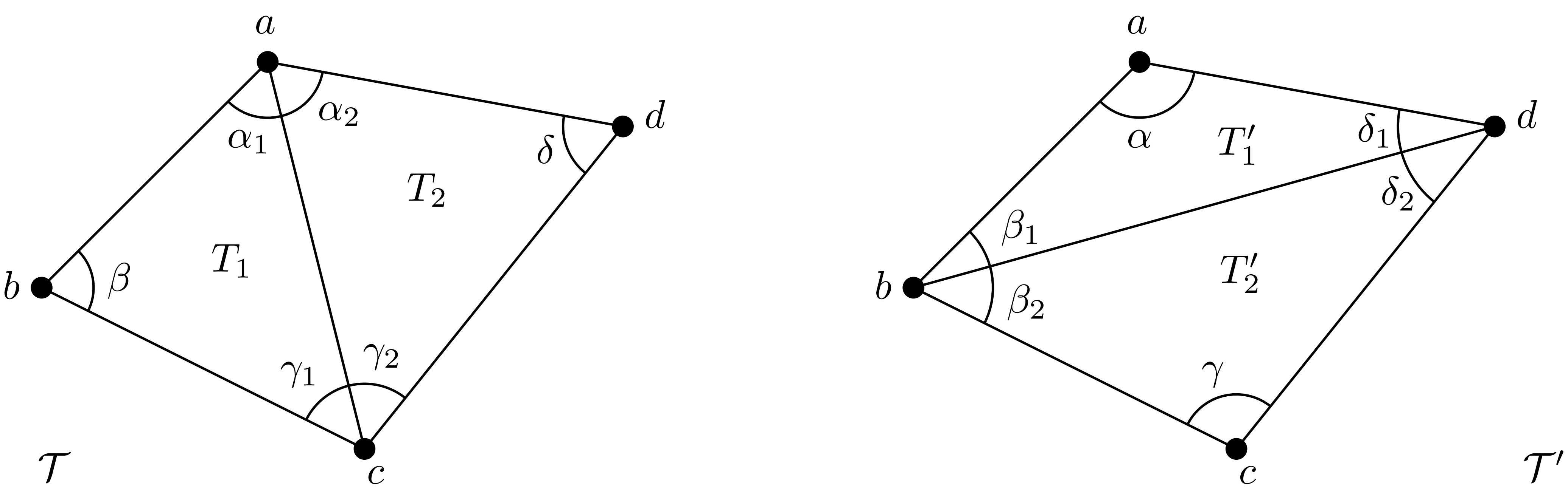
DELAUNAY TRIANGULATION

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

If $\epsilon > \epsilon'$, then ϵ' cannot be α, nor γ.



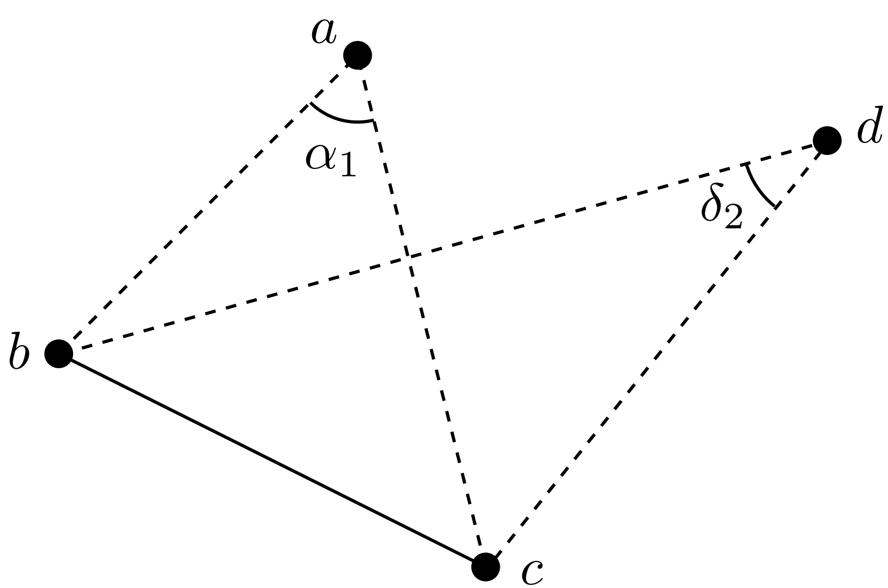
DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

If $\epsilon > \epsilon'$, then ϵ' cannot be α, nor γ.

If $\epsilon' = \delta_2$, then $\delta_2 = \epsilon' < \epsilon \leq \alpha_1$ and, therefore, $d \in ext(C_{abc})$.



DELAUNAY TRIANGULATION AND EQUIANGULARITY

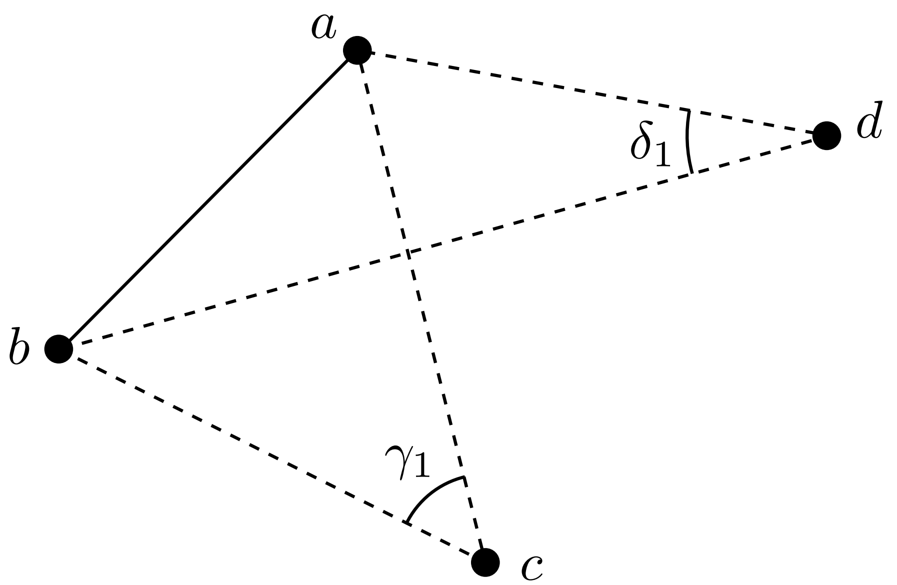
Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

If $\epsilon > \epsilon'$, then ϵ' cannot be α, nor γ.

If $\epsilon' = \delta_2$, then $\delta_2 = \epsilon' < \epsilon \leq \alpha_1$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \delta_1$, then $\delta_1 = \epsilon' < \epsilon \leq \gamma_1$ and, therefore, $d \in ext(C_{abc})$.



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

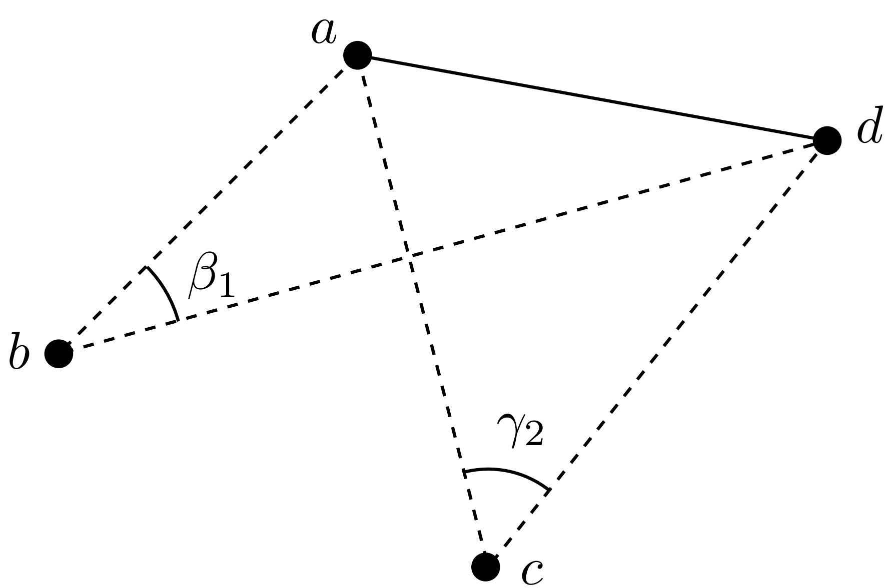
$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

If $\epsilon > \epsilon'$, then ϵ' cannot be α, nor γ.

If $\epsilon' = \delta_2$, then $\delta_2 = \epsilon' < \epsilon \leq \alpha_1$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \delta_1$, then $\delta_1 = \epsilon' < \epsilon \leq \gamma_1$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \beta_1$, then $\beta_1 = \epsilon' < \epsilon \leq \gamma_2$ and, therefore, $d \in ext(C_{abc})$.



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

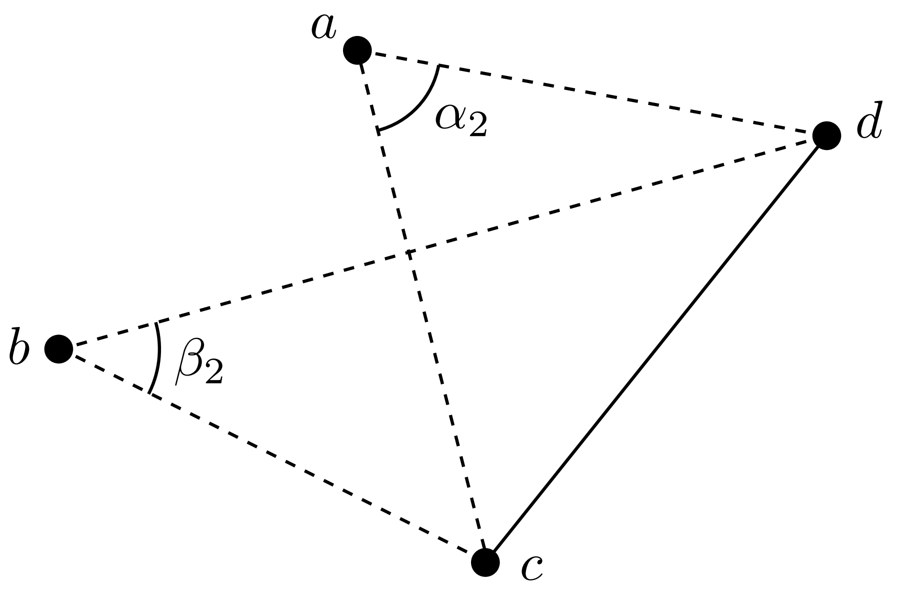
If $\epsilon > \epsilon'$, then ϵ' cannot be α, nor γ.

If $\epsilon' = \delta_2$, then $\delta_2 = \epsilon' < \epsilon \leq \alpha_1$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \delta_1$, then $\delta_1 = \epsilon' < \epsilon \leq \gamma_1$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \beta_1$, then $\beta_1 = \epsilon' < \epsilon \leq \gamma_2$ and, therefore, $d \in ext(C_{abc})$.

If $\epsilon' = \beta_2$, then $\beta_2 = \epsilon' < \epsilon \leq \alpha_2$ and, therefore, $d \in ext(C_{abc})$.



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

$\Leftarrow$) If $d \in ext(C_{abc})$, then $\epsilon > \epsilon'$

DELAUNAY TRIANGULATION AND EQUIANGULARITY

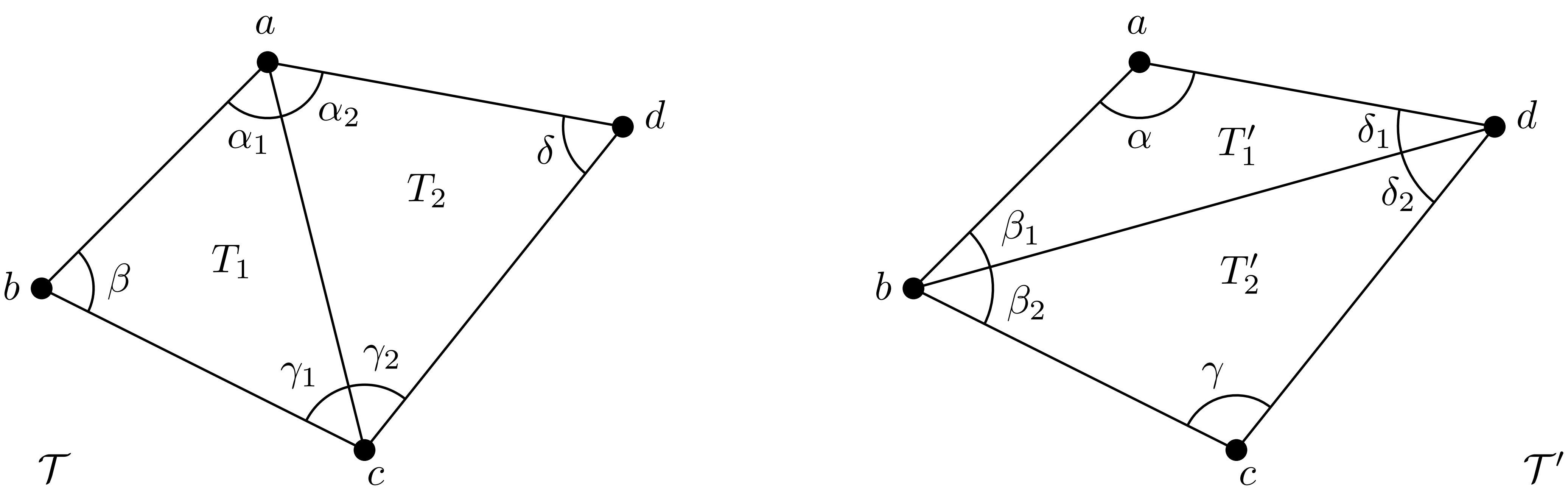
Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

$\Leftarrow$) If $d \in ext(C_{abc})$, then $\epsilon > \epsilon'$

If $\epsilon = \beta$, then $\epsilon = \beta > \beta_1 \geq \epsilon'$

If $\epsilon = \delta$, then $\epsilon = \delta > \delta_1 \geq \epsilon'$



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

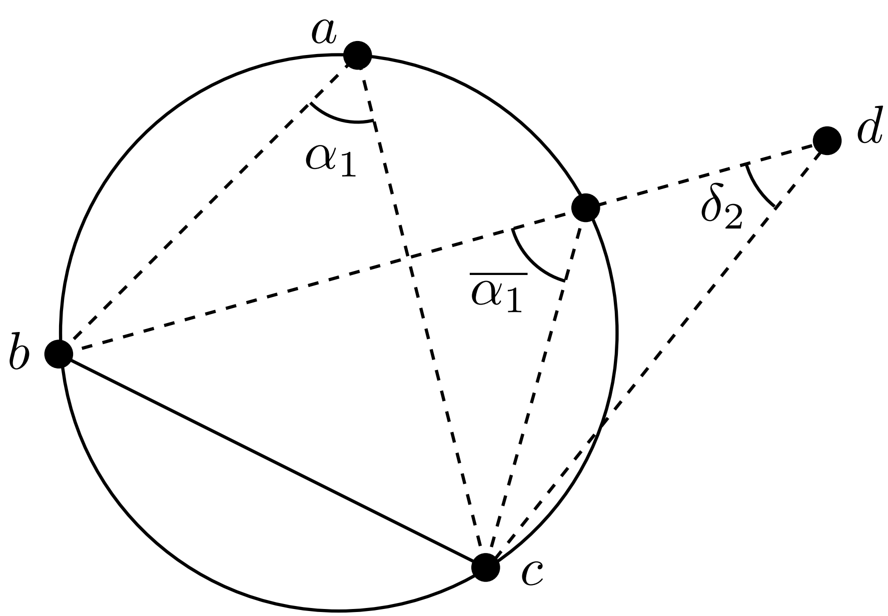
$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

$\Leftarrow$) If $d \in ext(C_{abc})$, then $\epsilon > \epsilon'$

If $\epsilon = \beta$, then $\epsilon = \beta > \beta_1 \geq \epsilon'$

If $\epsilon = \delta$, then $\epsilon = \delta > \delta_1 \geq \epsilon'$

If $\epsilon = \alpha_1$, then $\epsilon = \alpha_1 = \overline{\alpha_1} > \delta_2 \geq \epsilon'$



DELAUNAY TRIANGULATION AND EQUIANGULARITY

Due to the symmetry of the problem, we only need to prove that $\epsilon > \epsilon' \iff d \in ext(C_{abc})$.

$\Rightarrow$) If $\epsilon > \epsilon'$, then $d \in ext(C_{abc})$

$\Leftarrow$) If $d \in ext(C_{abc})$, then $\epsilon > \epsilon'$

If $\epsilon = \beta$, then $\epsilon = \beta > \beta_1 \geq \epsilon'$

If $\epsilon = \delta$, then $\epsilon = \delta > \delta_1 \geq \epsilon'$

If $\epsilon = \alpha_1$, then $\epsilon = \alpha_1 = \overline{\alpha_1} > \delta_2 \geq \epsilon'$

If $\epsilon = \alpha_2$, then $\epsilon = \alpha_2 = \overline{\alpha_2} > \beta_2 \geq \epsilon'$

If $\epsilon = \gamma_1$, then $\epsilon = \gamma_1 = \overline{\gamma_1} > \delta_1 \geq \epsilon'$

If $\epsilon = \gamma_2$, then $\epsilon = \gamma_2 = \overline{\gamma_2} > \beta_1 \geq \epsilon'$

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Corollary. The Delaunay triangulation is the most equiangular among all triangulations of a given set of points.

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Corollary. The Delaunay triangulation is the most equiangular among all triangulations of a given set of points.

If P does not contain four or more concyclic points, it follows from the previous lemma.

DELAUNAY TRIANGULATION AND EQUIANGULARITY

Corollary. The Delaunay triangulation is the most equiangular among all triangulations of a given set of points.

If P does not contain four or more concyclic points, it follows from the previous lemma.

If P contains four or more concyclic points, $Del(P)$ contains a polygon inscribed in a circle which can be triangulated in several ways. Nevertheless, Lemma 1 (on the geometrical locus of all the points from which a segment is seen under a given angle) guarantees that every triangulation of a polygon inscribed in a circle has the same fineness, since each edge of the polygon belongs to a triangle, and every possible triangle gives rise to the same angle.

DELAUNAY TRIANGULATION AND INTERPOLATION

In the context of terrain modeling, the Delaunay triangulation has another nice property: it minimizes the *roughness* of the terrain (i.e., the integral of the square of the L_2-norm of the terrain's gradient).

It is important to notice that this property is independent from the data, in other words, it is independent from the values of the z-coordinates of the input points.

DELAUNAY TRIANGULATION

TWO BOOKS WITH MUCH MORE INFORMATION

A. Okabe, B. Boots, K. Sugihara, S. N. Chiu
Spatial Tessellations
2nd ed., J. Wiley & Sons, 2000.

F. Aurenhammer, R. Klein, D.-T. Lee
Voronoi Diagrams and Delaunay Triangulations
World Scientific, 2013.