5. Delaunay triangulation

Files: The accompanying files for this assignment are the ones contained in zip file `geoc_lab5.zip`.
Delivery: upload all files required to run your program to the Racó.

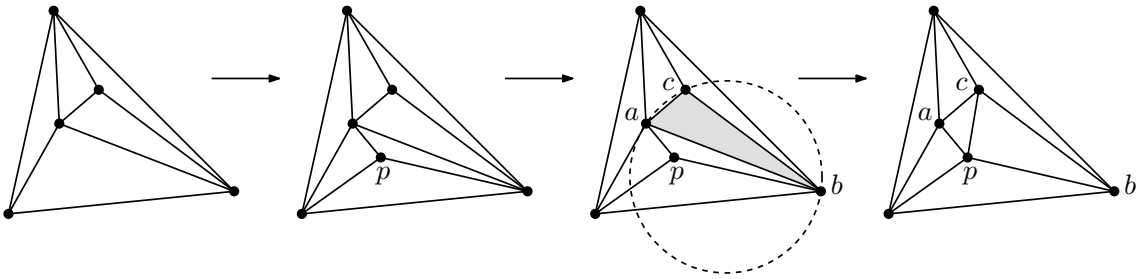
Write a program to incrementally construct a Delaunay triangulation of a set of points in the plane. Moreover, apply it to the Lanzarote data set, and use the boundary information to prune the triangles that do not belong to the triangulation of the given point set.

Step 1: Delaunay triangulation algorithm

Your algorithm is expected to be an extension of the one of Assignment 4.

The difference should lie in the fact that the Delaunay property must hold at each step of the algorithm. In other words, each time that one point p is inserted in the triangulation, the algorithm should check, for each triangle t incident to p, whether or not p lies in the circumcircle of the triangle adjacent to t through the edge opposite to p. If it does, the common edge of the two triangles should be replaced by the edge connecting p with the opposite vertex in the other triangle.

For example, in the figure below, p is interior to the circumcircle of triangle $\triangle abc$: in this case, edge ab should be replaced by edge pc. This test should be repeated for all the triangles incident to p that appear after each edge flip.



Your code should be written in the file `triangulation.js`, and run through the HTML file `geoc_lab4-5.html`.

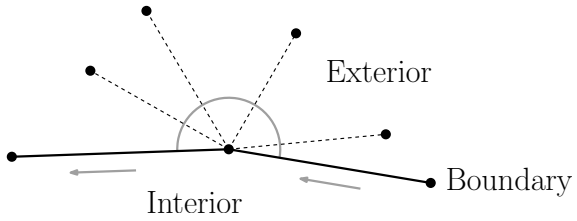
Step 2: Lanzarote input file: dealing with the boundary

The test file corresponding to Lanzarote comes with additional boundary information. In addition to the terrain points, the JSON file contains the list of point indices

that form the boundary of the different connected components (islands), given in counter-clockwise order along the boundary of each island. See attribute `boundaries` in JSON files. This allows, once the Delaunay triangulation has been obtained, to eliminate the edges and triangles lying in the ocean, which do not correspond to the triangulation of the original point set.

In order to eliminate the undesired edges, two cases can appear:

- The connected component is one single point: Eliminate all the incident edges (as well as all the corresponding triangles).
- The connected component has more than one point. Each point has two incident boundary edges, and the location of the interior and the exterior of the island is known. Eliminate all the external incident edges (dotted lines in the figure below), as well as all the corresponding triangles.



Step 3: Terrain visualization

The final step is to visualize the terrain in 3D. The file `geoc_lab5_3dview.html` invokes the triangulation method from `triangulation.js` and applies it to the Lanzarote data set, rendering the resulting triangles in 3D. Make sure that you appropriately orient all triangles for their visualization.

It is interesting to compare the resulting terrain with the one that can be obtained using the triangulation of Assignment 4.

Since the rendering of the whole Lanzarote data set can be rather slow, we also provide a smaller version, `lanzarote-small.json.js`, which has a subset of about 5,000 points, and should render much faster.

5. Delaunay triangulation

Goals of the assignment

The full grade will be obtained when successfully achieving all of the following four goals:

1. Constructing a Delaunay triangulation for points in general position (i.e., assuming the input points do not include any three collinear points, or any 4 concyclic points). To try this out, you can use the file `lanzarote.json.js`.

2. Constructing a Delaunay triangulation for points in any position (that is, handling degeneracies such as collinear and concyclic points). To try this out, you can use the file `lanzarote-degen.json.js` (notice that this file does not contain boundary information).

3. Pruning the boundary for the Lanzarote data set from file `lanzarote.json.js`, producing a triangulation that does not contain triangles lying in the ocean (that is, triangles that do not correspond to the triangulation of the original point set).

4. Correctly visualizing the resulting terrain. This should at least work correctly when applied to the file `lanzarote-small.json.js`.