Grids

torsdag 12. oktober 2017 11:33

1. In these lectures, we consider construction and properties of polygonal grids, i.e. given a domain $\Omega \subset \mathbb{R}^2$,

(mostly study $\mathbb{R}^2$ in these lectures, but will briefly mention $\mathbb{R}^3$)

our triangulation is a set T_h of polygons K for which

1. $\mathring{K} \neq \phi \quad \forall K \in T_h$.

2. $\mathring{K}_i \cap \mathring{K}_j = \phi \quad \forall K_i, K_j \in T_h, \; i \neq j$

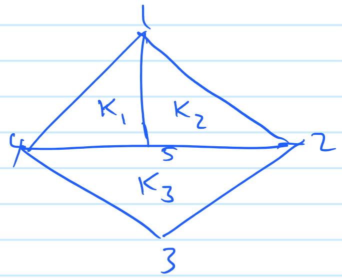
(here $\mathring{K}$ is the interior of K, i.e. $K \setminus \partial K$).

We often wish to restrict ourselves to grids that are:

3. Conforming: $K_i \cap K_j \neq \phi \quad (i \neq j)$

$\Rightarrow K_i \cap K_j =$ an edge or vertex

(i.e., elements cannot intersect along half edges, e.g.



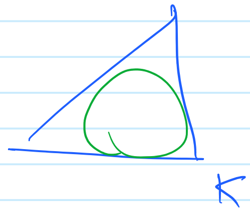
is nonconforming: K_1 and K_3 intersect along the line 45, which is only part of the edge 42 of K_3.)

1. Regular: we call this means

$$\frac{h_K}{\rho_K} \le \delta \qquad \forall K$$

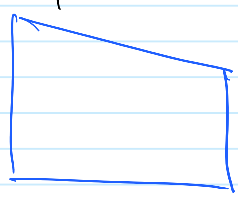
diameter of K

sphericity of K: diameter of largest inscribed circle.



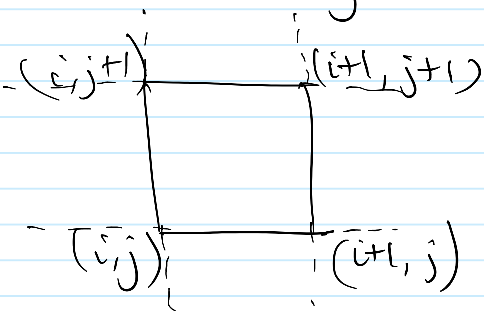
2. Typically polygons (in $\mathbb{R}^2$) are either triangular or quadrilateral.

eg.



sometimes, we can employ a structured grid with quadrangular elements

$\Rightarrow$ the connectivity matrix is trivial:



(all edges may be numbered like this).

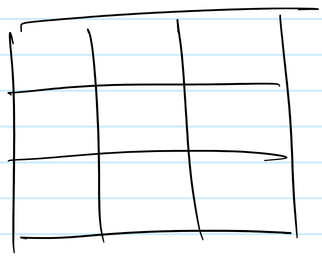
i.e. converting to a single index, by eg. $k = i + N_i(j-1)$ etc., have simple rule for connectivity.

a) generation of structured grid: typically, we construct a nice mapping

$[0,1] \times [0,1] \to \Omega$

$$\underbrace{[0,1]\times[0,1]}_{\hat{\Omega},\ \text{reference square}} \longrightarrow \Omega$$

and tile the reference square as a uniform grid

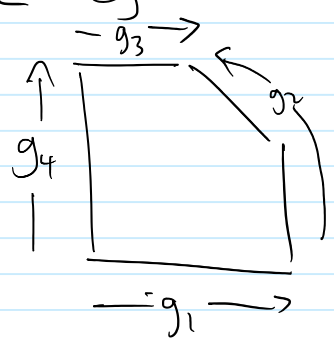


etc.

b) Methods for constructing such a mapping

i) Let $u: \hat{\underline{x}} \mapsto \underline{x}$ be harmonic, with boundary conditions of the form $u(\hat{x}) = \psi(x)$, where ψ parametrizes $\partial\Omega$.

Ω = eg



then
$$u(\hat{x}, 0) = g_1(\hat{x})$$
$$u(\hat{x}, 1) = g_3(\hat{x})$$
$$u(0, \hat{y}) = g_2(\hat{y})$$
$$u(1, \hat{y}) = g_4(\hat{y})$$

Careful: for nonconvex domains, we can fall outside the desired element!

ii) transfinite interpolation

(transfinite as we interpolate on infinite number of points. Don't discuss here).

Note that the image will in general have curved edges.

3\. Non-structured grids. Revert to our favourite triangles!

- Here we discuss the methods used to construct most grids you have been working with: <u>Delaunay triangulation</u>,

the unique (unless > 3 (d+1) points lie on a common circle (d-1)-sphere) triangulation of a set of given vertices maximizing the minimum angle of all the grid triangles.

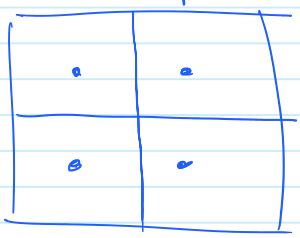
- <u>NB</u>: only applicable to convex domains (modifications to non-convex case exist - constrained Delaunay triangulations).

- other characterizations exist, eg. a more constructive form is:

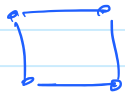
the Delaunay triangulation corresponds to the dual graph of the <u>Voronoi diagram</u> of the set of points.

essentially,

partitions a plane into cells according to the cells according to the nearest point, eg

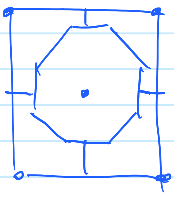
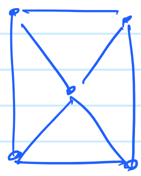


essentially, connect vertices whose Voronoi cells share an edge

would result in 

which is not a triangulation, reason: the 4 points all lie on a common circle.

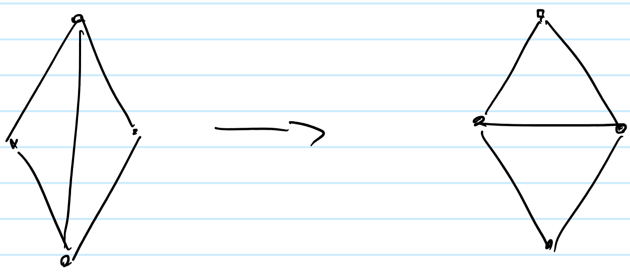
Example 2:

 =>  etc.

4. Delaunay algorithms: not like the above!

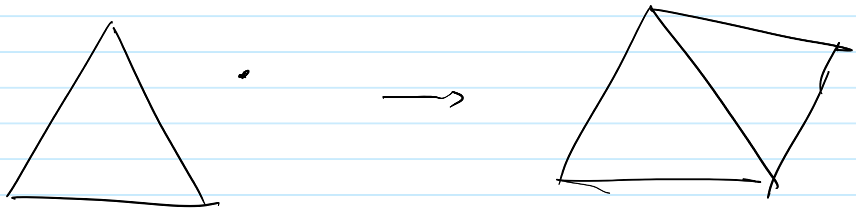
typically, progress one vertex at a time.

Key idea is edge swapping:



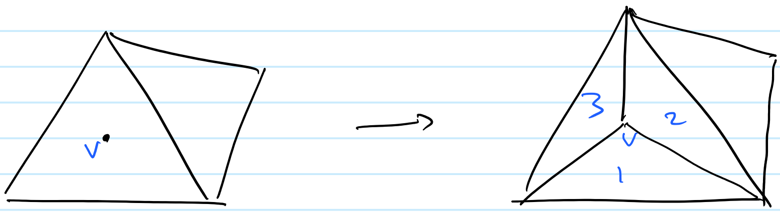
start with 3 points:

add a new point:

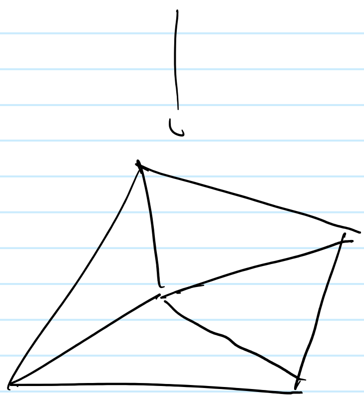


(no swap required)

another



Now test if any of 1, 2, 3 could benefit from swapping along edge not containing v. Here, answer is yes!



etc.

worst case complexity: $O(n^2)$

in practice, can make run in $O(n \log n)$ time.

→ Techniques based on merging have been shown to achieve $O(n \log \log n)$.