MESH GENERATION

Num2Ing TU Berlin

with Ali and Dirk (WS 2019/20)

Preparation

Get Julia/Python/**MATLAB** to work.

Plot a single triangle using triplot.

Download & compile triangle.

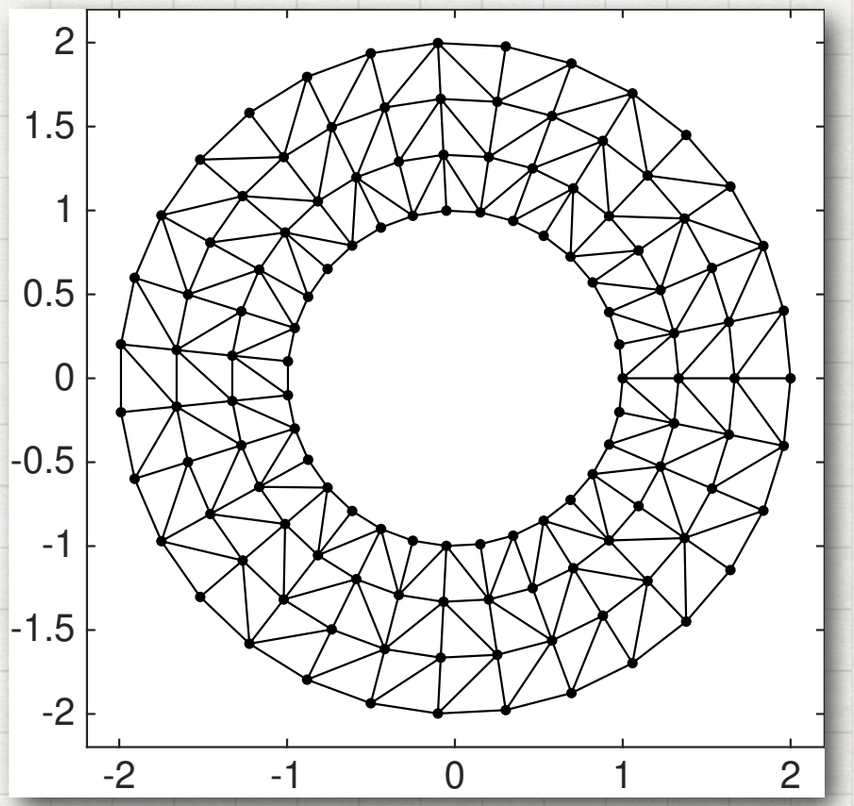
```python
import matplotlib.pyplot as plt
import numpy as np

tri = np.array([[0,1,2]],dtype=np.int64)
x   = np.array([0,1,0],dtype=float)
y   = np.array([0,0,1],dtype=float)

plt.figure()
plt.triplot(x,y,triangles=tri)
plt.show()
```

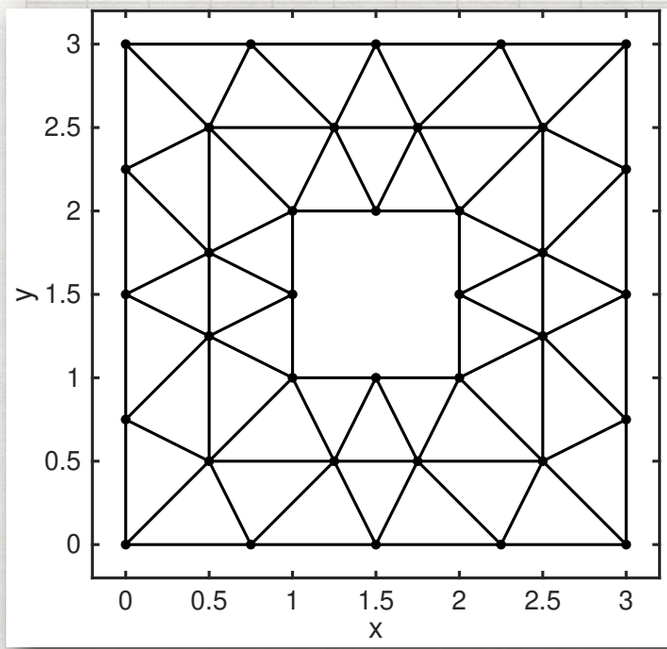
```
>> x=[0 1 0];
>> y=[0 0 1];
>> tri=[1 2 3];
>> triplot(tri,x,y)
>>
```

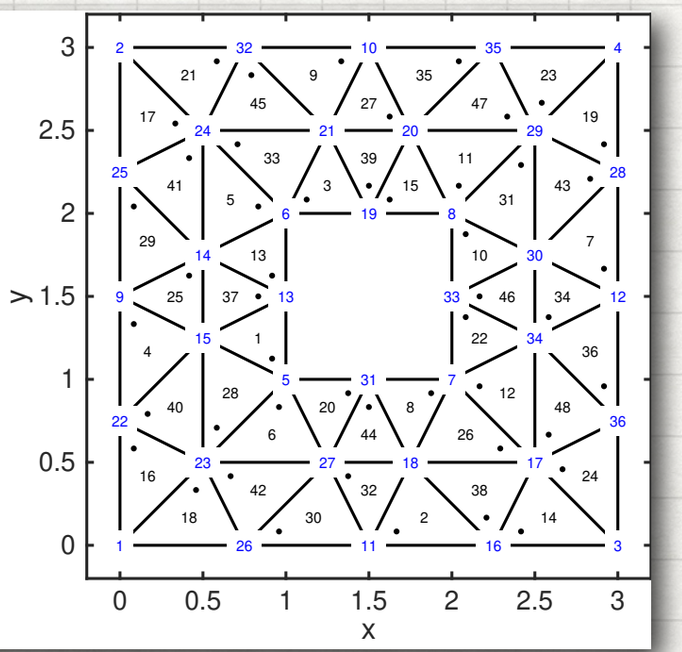
- Gmsh (mesh generation) http://gmsh.info/
- Triangle (2D mesh generation) https://www.cs.cmu.edu/~quake/triangle
- TetGen (3D mesh generation) http://www.tetgen.org/

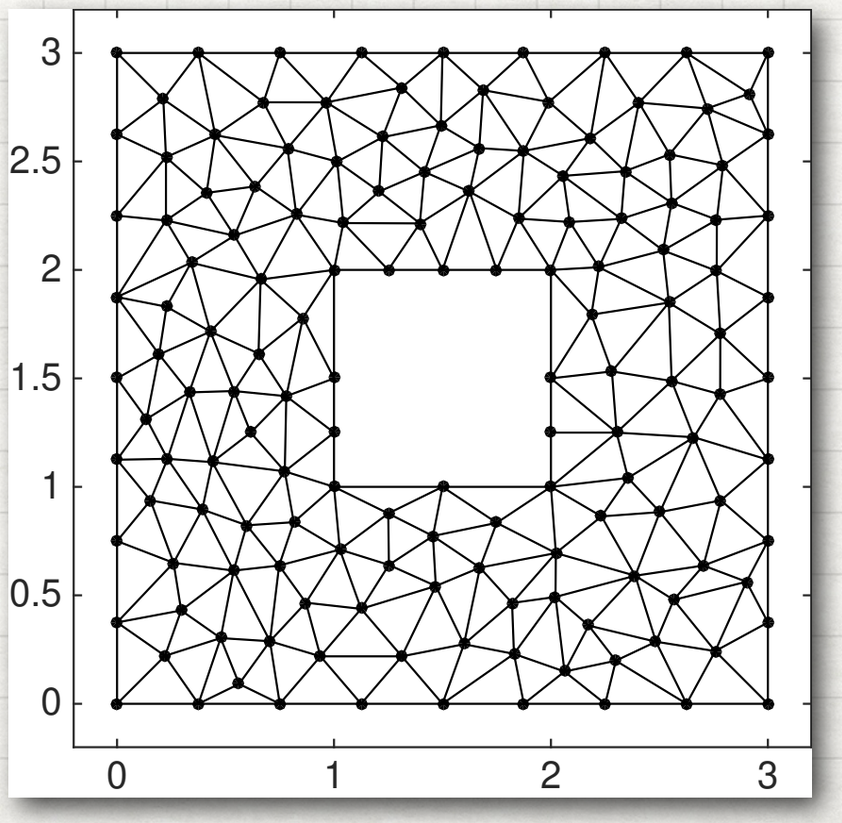
MATLAB functions

- triplot, trisurf (Plotting)
- delaunay (Mesh generation)
- readtria (Read triangle file)
- meshgrid (Point arrays)





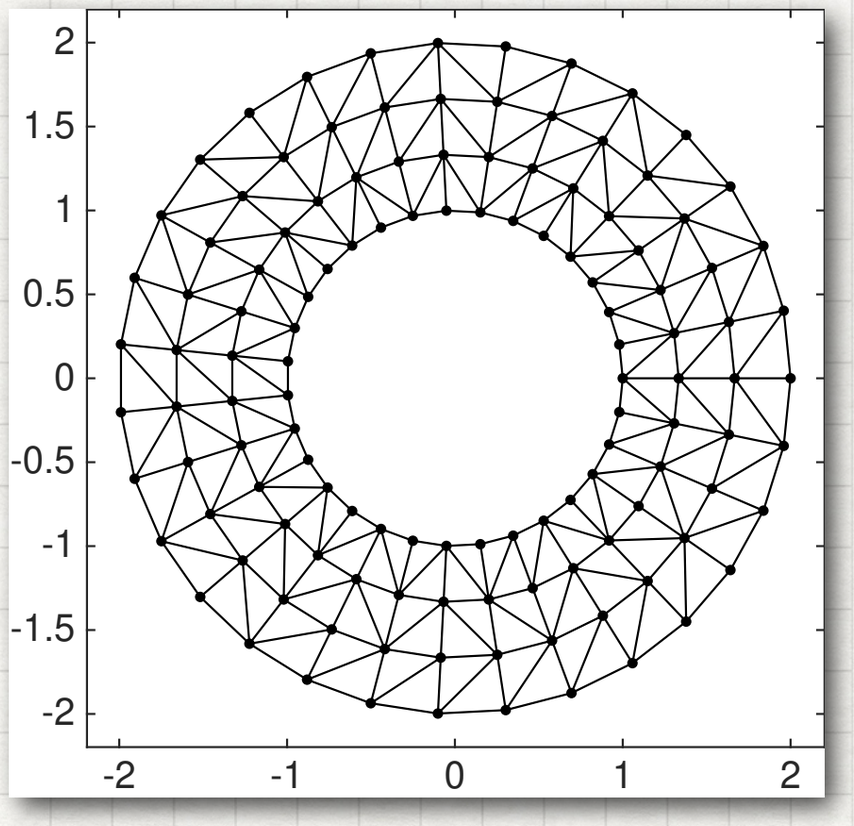


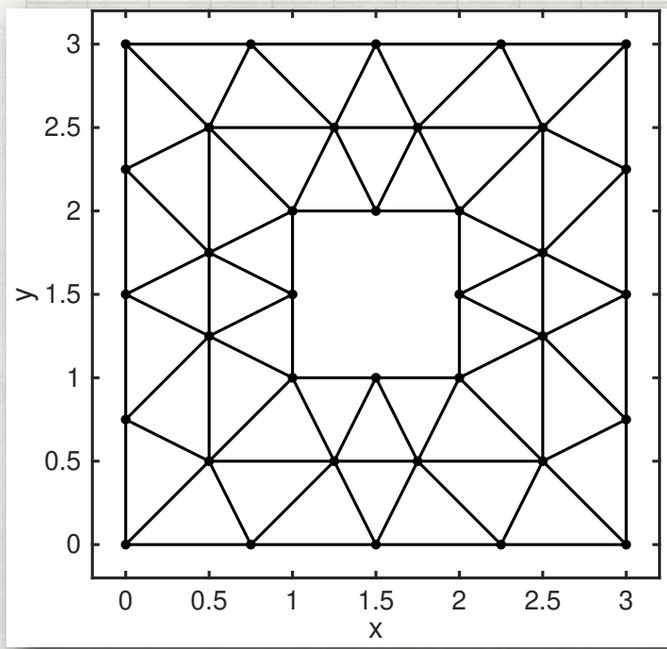


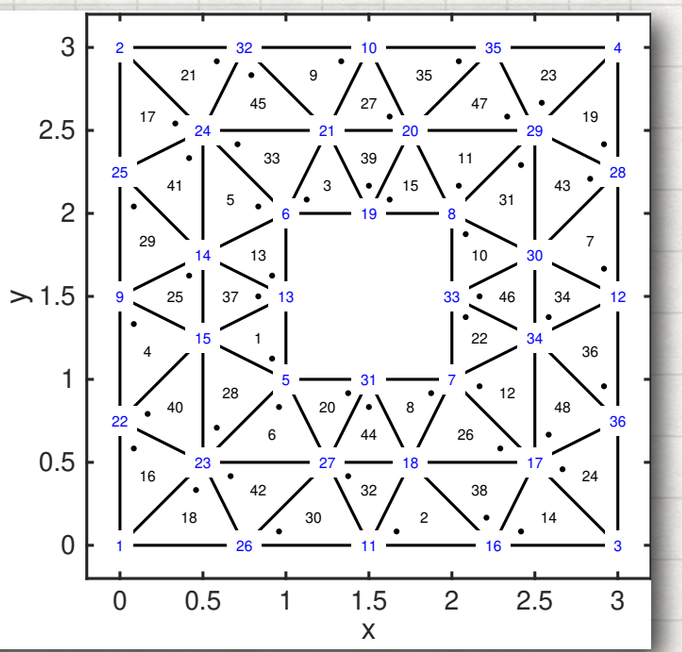
Python functions

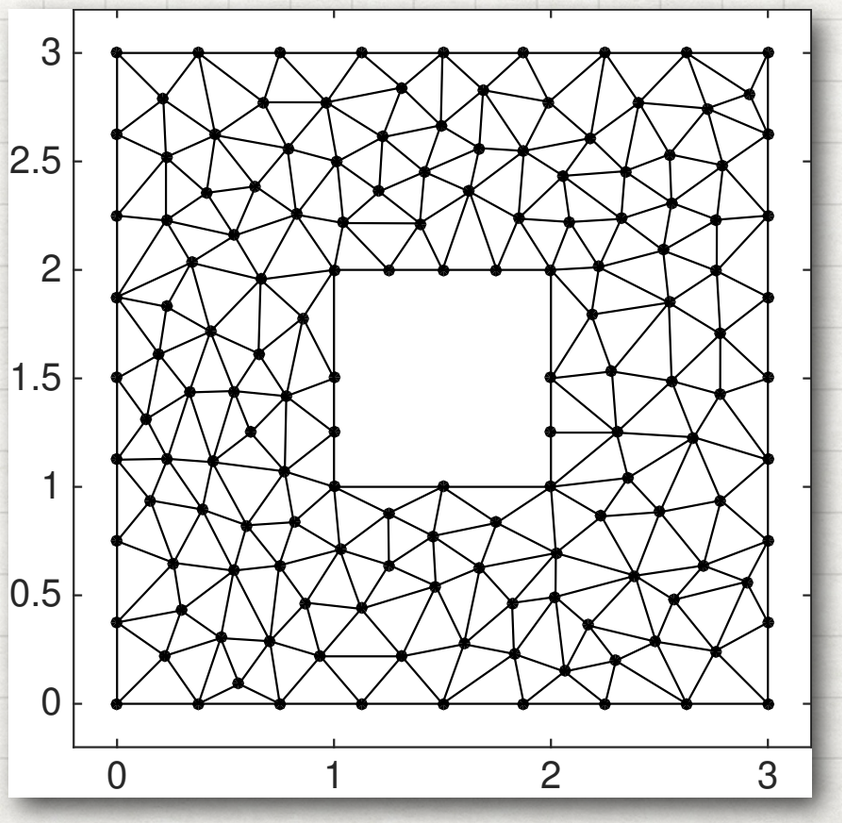
```
import matplotlib.pyplot as plt
import matplotlib.tri as tri
import numpy as np
```

- triplot, trisurf (Plotting)
- delaunay (Mesh generation)
- readtria (Read triangle file)
- meshgrid (Point arrays)









Definition III.27 (k-simplex)**:** Let $x_k \in \mathbb{R}^n$ be a set of $k+1$ distinct points so that $x_r - x_0$ for $r = 1, ..., k$ are linearly independent. Then the convex hull of points

$$C(x_0, ..., x_k) = \{\xi_0 x_0 + \xi_1 x_1 + ... + \xi_k x_k : \sum_{i=0}^{k} \xi_i = 1 \text{ and } \xi_i \geq 0\} \subset \mathbb{R}^n, \qquad \text{(III.15)}$$

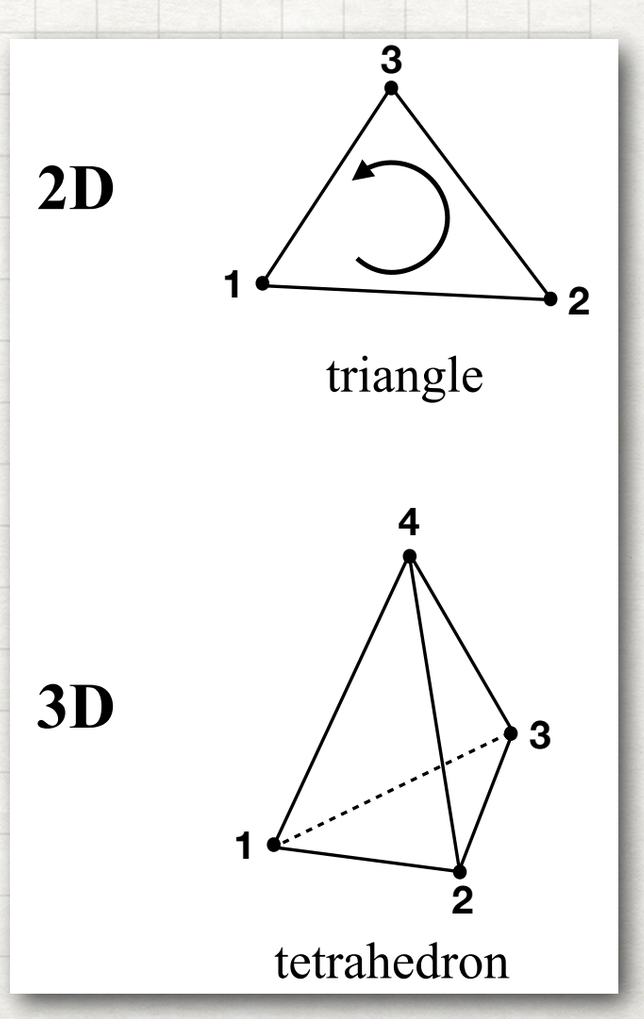
is a k-simplex. For a point $x \in C$ we call $(\xi_0, ...\xi_k)$ its barycentric coordinates. A 0-simplex is a vertex, a 1-simplex is a line, a 2-simplex is a triangle, a 3-simplex is a tetrahedron and linear independency requires $n \geq k$. By removing the ith vertex each k-simplex is composed of $k+1$ $(k-1)$-simplices $C(x_0, ..., \widehat{x_{i+1}}, ..., x_k)$ for $i = 0, ..., k$. E.g., each tetrahedron (3-simplex) is composed of 4 triangles (2-simplex), which are the faces/boundaries of the tetrahedron, each triangle (2-simplex) is composed of 3 lines (1-simplex), which are the edges/boundaries of the triangle. Each line (2-simplex) is composed of 2 vertices (0-simplex), which are the boundaries of the line.

Definition III.28 (Mesh representation with simplices)**:** Let $x_k \in \mathbb{R}^n$ for $k = 1, ..., n_p$ be a set of $n_p \in \mathbb{N}$ distinct points and $e_k \in \mathbb{N}^{n_e \times (k+1)}$ such that

$$\Omega_l = C(x_{e_k(l,1)}, ..., x_{e_k(l,k+1)}),$$

using (III.15) is a k-simplex for all $l = 1, ..., n_e$. We say n_p is the *number of points* and n_e is the *number of elements*. As before we require the set Ω_l to form an admissible decomposition of $\overline{\Omega} = \bigcup_{l=1}^{n_e} \overline{\Omega}_l$. The mesh is represented using the vertex positions $x_k \in \mathbb{R}^n$ and using the element description encoded in $e_k \in \mathbb{N}^{n_e \times (k+1)}$.

$$\bar{\Omega} = \bigcup_{l=1}^{n_e} \bar{\Omega}_l$$



Definition III.24 (Admissible decomposition)**:** Assume a given domain $\Omega \subset \mathbb{R}^d$ with polygonal boundary and let

$$\overline{\Omega} = \bigcup_{i=1}^{n+1} \overline{\Omega}_i,$$

be a decomposition of $\overline{\Omega}$ into elements $\{\overline{\Omega}_i\}_{i=1,..,n+1}$.

- $n = 1$: If $\Omega = (a, b) \subset \mathbb{R}$, then an admissible decomposition is given by a union of non-overlapping but adjacent intervals, i.e., $\Omega_i = (x_{i-1}, x_i)$, $i = 1, \ldots, n+1$ and $a = x_0 < x_1 < \ldots < x_n < x_{n+1} = b$.

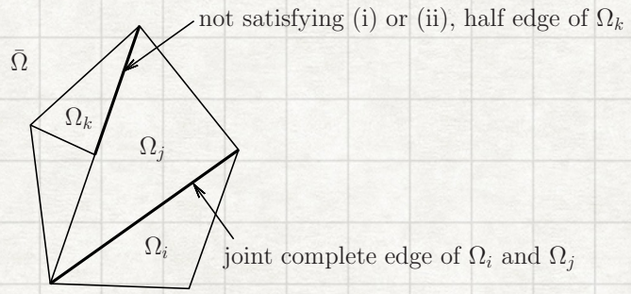
- $n = 2$: A decomposition of Ω into triangles (or convex quadrilaterals etc) $\{\Omega_i\}_{i=1}^n$ is admissible, if for each i and j exactly one of the following cases is true:

  i) $\Omega_i = \Omega_j$;

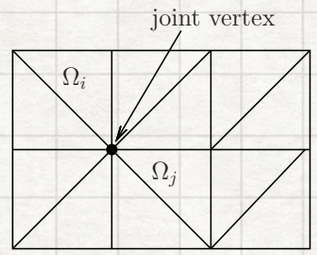
  ii) $\overline{\Omega}_i \cap \overline{\Omega}_j$ is a joint complete edge of both Ω_i and Ω_j;

  iii) $\overline{\Omega}_i \cap \overline{\Omega}_j$ is a joint vertex of Ω_i and Ω_j;

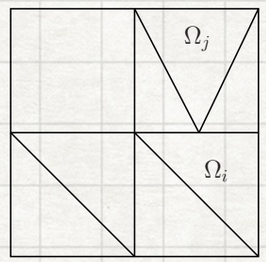
  iv) $\overline{\Omega}_i \cap \overline{\Omega}_j = \emptyset$.



not admissible



admissible



not admissible

- $n = 3$: A decomposition into tetrahedrons (or hexahedrons, pyramids, wedges, etc) is called admisslbe, if for each i and j exactly one of the following cases is true:

  i) $\Omega_i = \Omega_j$;

  ii) $\overline{\Omega}_i \cap \overline{\Omega}_j$ is a joint complete face of both Ω_i and Ω_j (triangle, quadrilateral, ...);

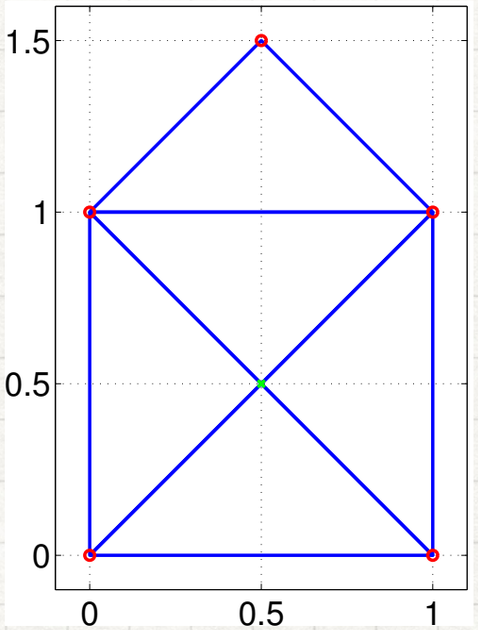
  iii) $\overline{\Omega}_i \cap \overline{\Omega}_j$ is a joint complete edge of both Ω_i and Ω_j;

  iv) $\overline{\Omega}_i \cap \overline{\Omega}_j$ is a joint vertex of Ω_i and Ω_j;

  v) $\overline{\Omega}_i \cap \overline{\Omega}_j = \emptyset$.

Structured mesh-Task 1: Generate and plot via triplot (by hand)

Example III.30 (House mesh)**:** The mesh shown below can be represented as follows:



$$x_1 = (0,0)^\top, \quad x_2 = (1,0)^\top, \quad x_3 = (0.5,0.5)^\top,$$
$$x_4 = (0,1)^\top, \quad x_5 = (1,1)^\top, \quad x_6 = (0.5,1.5)^\top,$$
$$e_2 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 4 \\ 2 & 5 & 3 \\ 3 & 5 & 4 \\ 4 & 5 & 6 \end{pmatrix} \in \mathbb{N}^{5\times 3}, \quad n_p = 6, \quad n_e = 5, \quad k = 2.$$

The domain consists of $n_p = 6$ vertices, 10 lines, $n_e = 5$ triangles.

Structured mesh-Task 2: Generate and plot via triplot

Listing III.1: MATLAB code tensor mesh generation

```
% generate points
k = 10; l = 20;
xx = linspace(0,1,k);
yy = linspace(0,1,l).^(1.3);
[x,y]=meshgrid(xx,yy);
x = x';y = y';

% lexicographical ordering
ix  = reshape(1:(k*l),[k l]);
ix1 = ix(1:k-1,1:l-1); ix1 = ix1(:);
ix2 = ix(2:k  ,1:l-1); ix2 = ix2(:);
ix3 = ix(1:k-1,2:l  ); ix3 = ix3(:);
ix4 = ix(2:k  ,2:l  ); ix4 = ix4(:);

% connectivity (2 triangles per quad)
e2 = [ix1 ix2 ix3;ix3 ix2 ix1];
% plot
triplot(e2,x(:),y(:),'k-')
```

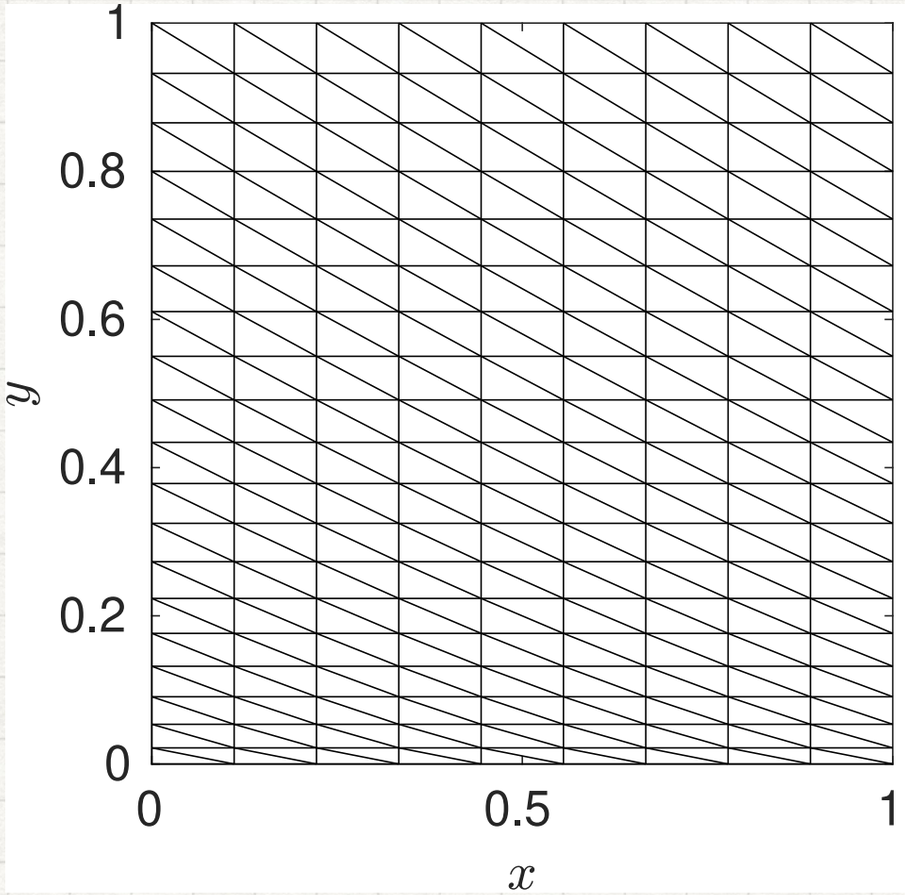


Figure III.5.: Structured tensor mesh.

```
# A box with eight vertices in 2D, no attributes, one boundary marker.
    8 2 0 1
     # Outer box has these vertices:
     1   0 0   0
     2   0 3   0
     3   3 0   0
     4   3 3   33     # A special marker for this vertex.
     # Inner square has these vertices:
     5   1 1   0
     6   1 2   0
     7   2 1   0
     8   2 2   0
    # Five segments with boundary markers.
    5 1
     1   1 2   5      # Left side of outer box.
     # Square hole has these segments:
     2   5 7   0
     3   7 8   0
     4   8 6   10
     5   6 5   0
    # One hole in the middle of the inner square.
    1
     1   1.5 1.5
```

Triangle: box.poly example

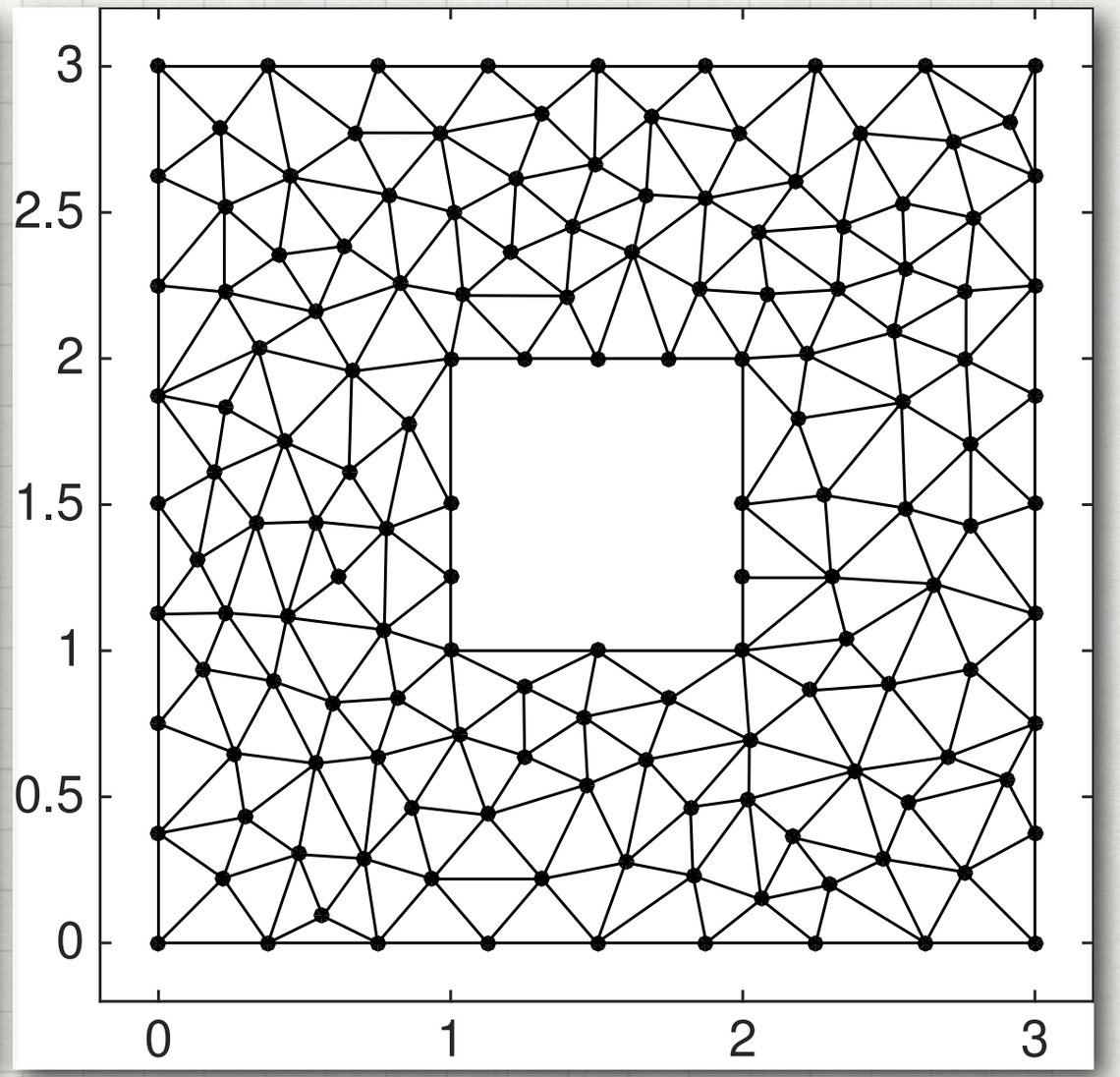
```
# A box with eight vertices in 2D, no attributes, one boundary marker.
    8 2 0 1
     # Outer box has these vertices:
     1   0 0   0
     2   0 3   0
     3   3 0   0
     4   3 3   33     # A special marker for this vertex.
     # Inner square has these vertices:
     5   1 1   0
     6   1 2   0
     7   2 1   0
     8   2 2   0
    # 8 segments with boundary markers.
    8 1
     1   1 2   0
     2   2 4   0
     3   4 3   0
     4   3 1   0
     5   5 6   0
     6   6 8   0
     7   8 7   0
     8   7 5   0
    # One hole in the middle of the inner square.
    1
     1   1.5 1.5
```

Triangle: box_v1.poly example

Unstructured mesh-Task 3: Generate using triangle and plot via triplot

```
# A box with eight vertices in 2D, no attributes, one boundary marker.
8 2 0 1
 # Outer box has these vertices:
 1   0 0   0
 2   0 3   0
 3   3 0   0
 4   3 3   33     # A special marker for this vertex.
 # Inner square has these vertices:
 5   1 1   0
 6   1 2   0
 7   2 1   0
 8   2 2   0
# 8 segments with boundary markers.
8 1
 1   1 2   0
 2   2 4   0
 3   4 3   0
 4   3 1   0
 5   5 6   0
 6   6 8   0
 7   8 7   0
 8   7 5   0
# One hole in the middle of the inner square.
1
 1   1.5 1.5
```

box_v1.poly




```
./triangle -qpca0.1 box_v1.poly
```

```
>> [x,y,npoint,nelement,tri,idp,ide] = readtria('box_v1.1');
read mesh file: box_v1.1
>> triplot(tri,x,y)
```

Examples

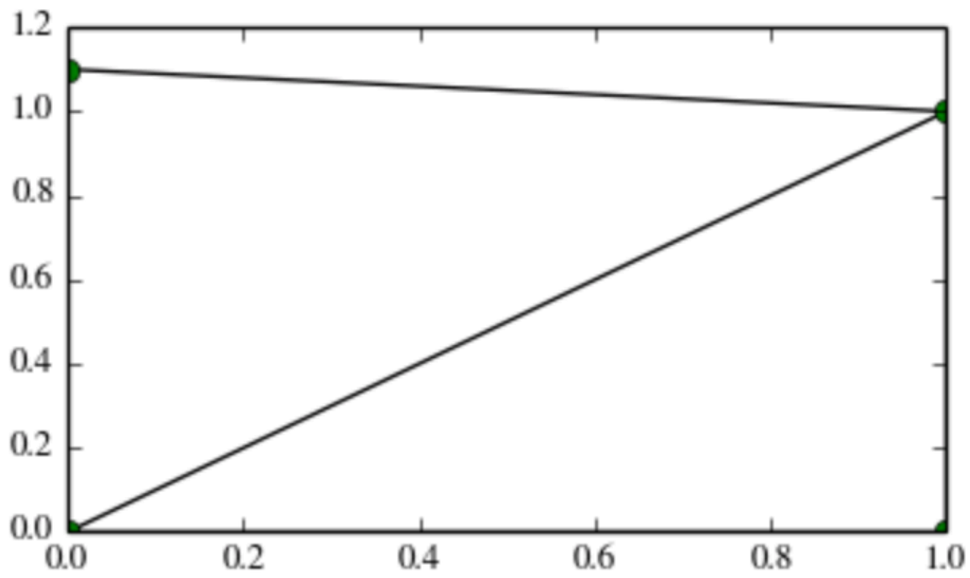
Triangulation of a set of points:

```
>>> points = np.array([[0, 0], [0, 1.1], [1, 0], [1, 1]])
>>> from scipy.spatial import Delaunay
>>> tri = Delaunay(points)
```

We can plot it:

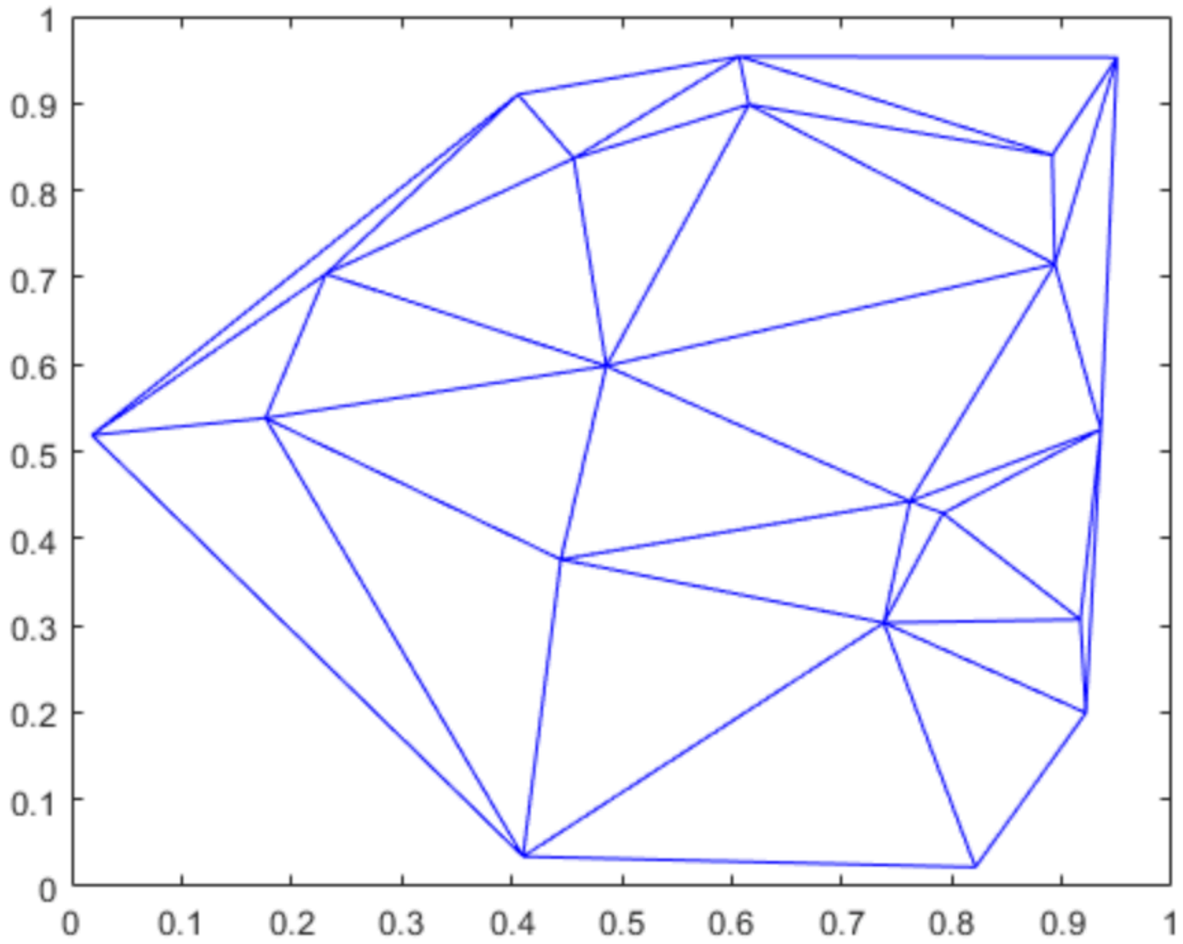
```
>>> import matplotlib.pyplot as plt
>>> plt.triplot(points[:,0], points[:,1], tri.simplices.copy())
>>> plt.plot(points[:,0], points[:,1], 'o')
>>> plt.show()
```

(Source code)



https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.Delaunay.html

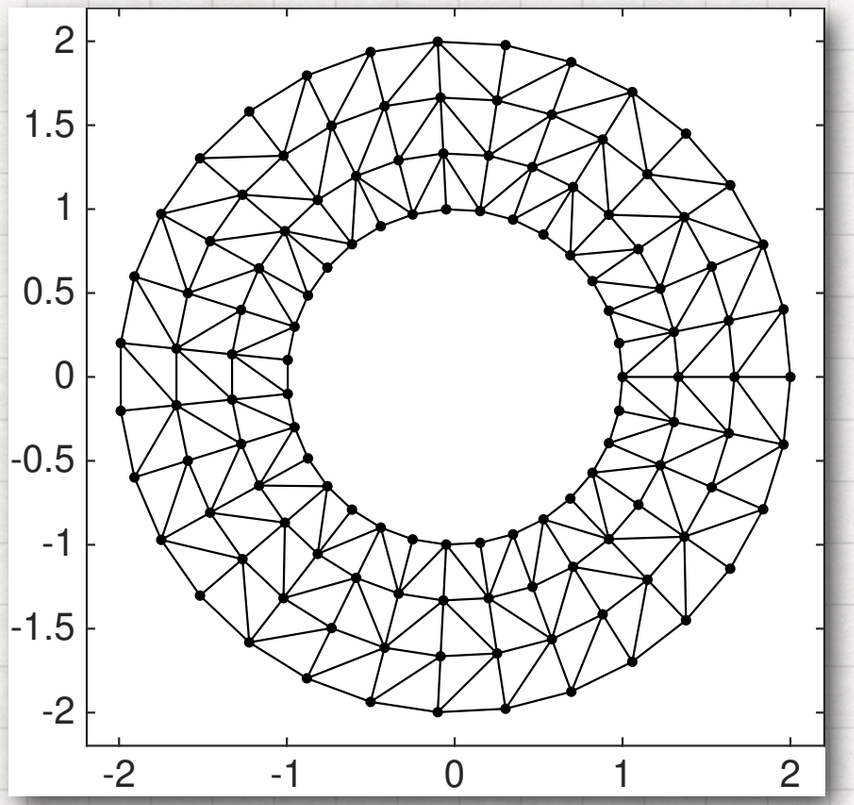
```
x = gallery('uniformdata',[20,1],0);
y = gallery('uniformdata',[20,1],1);
DT = delaunay(x,y);
triplot(DT,x,y);
```



https://www.mathworks.com/help/matlab/ref/delaunay.html

Structured or Unstructured Mesh-Task 4:

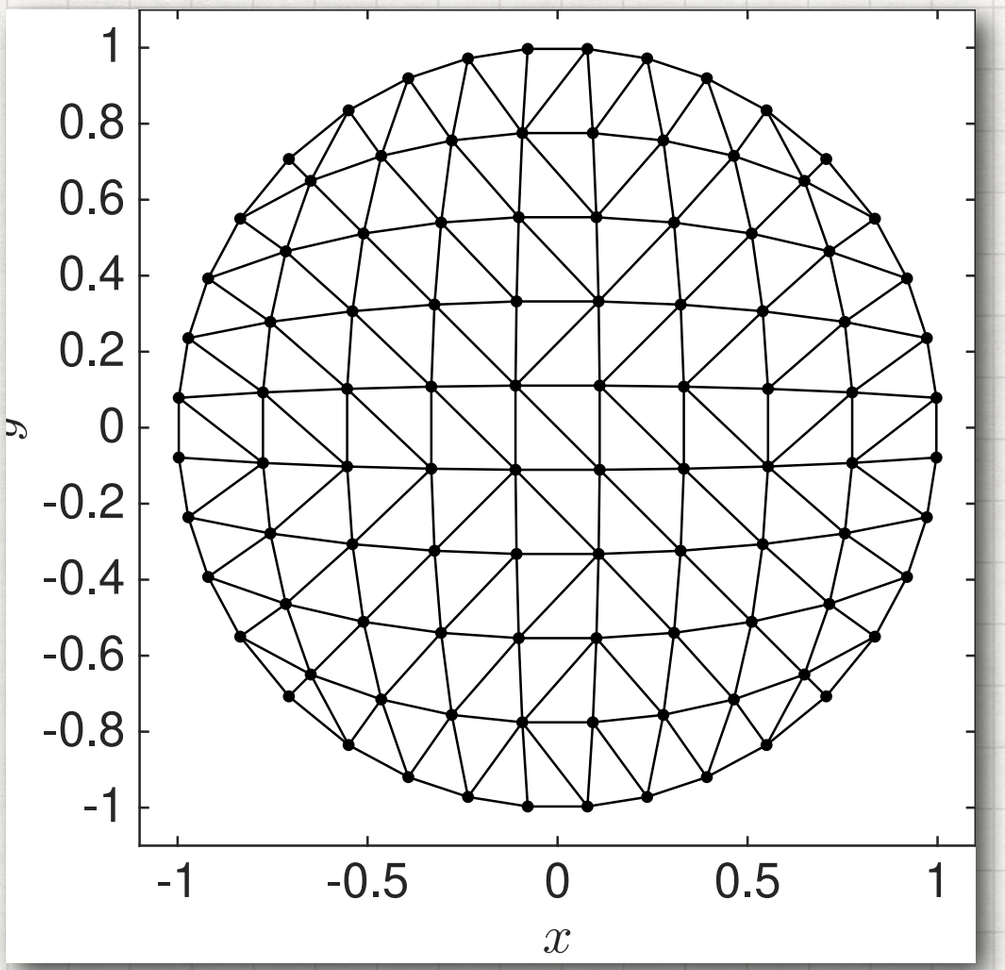
Generate using delaunay and/or triangle and plot via triplot



Generate domain where either (cumbersome) e2 takes care identification of vertices for phi=0 and 2pi using a deformed structured mesh or (better) from triangle using an inner/outer circular boundary.

Structured or Unstructured Mesh Task 5:

Generate using delaunay and/or triangle and plot via triplot



Generate disc domain using triangle or alternatively using Delaunay or ...