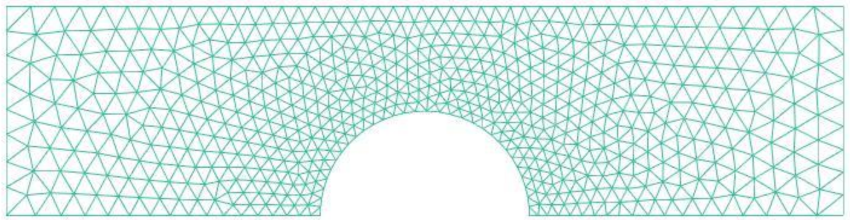
FEM2D

Malcolm D. Forbes, Je Park

Overview

- Introduction
- Algorithm Development
  - Meshing
  - Generating K^e and f^e
  - Assembling K and F
- Verification Problem
  - Analytical Solution
  - Numerical Results
- Conclusion
- Questions



Introduction

FEA approximates a PDE with a system of linear algebraic equations...

$$-\frac{\partial}{\partial x}\left(a_{11}\frac{\partial u}{\partial x}+a_{12}\frac{\partial u}{\partial y}\right)-\frac{\partial}{\partial y}\left(a_{21}\frac{\partial u}{\partial x}+a_{22}\frac{\partial u}{\partial y}\right)+a_{00}u-f=0$$

Governing equation (the PDE)

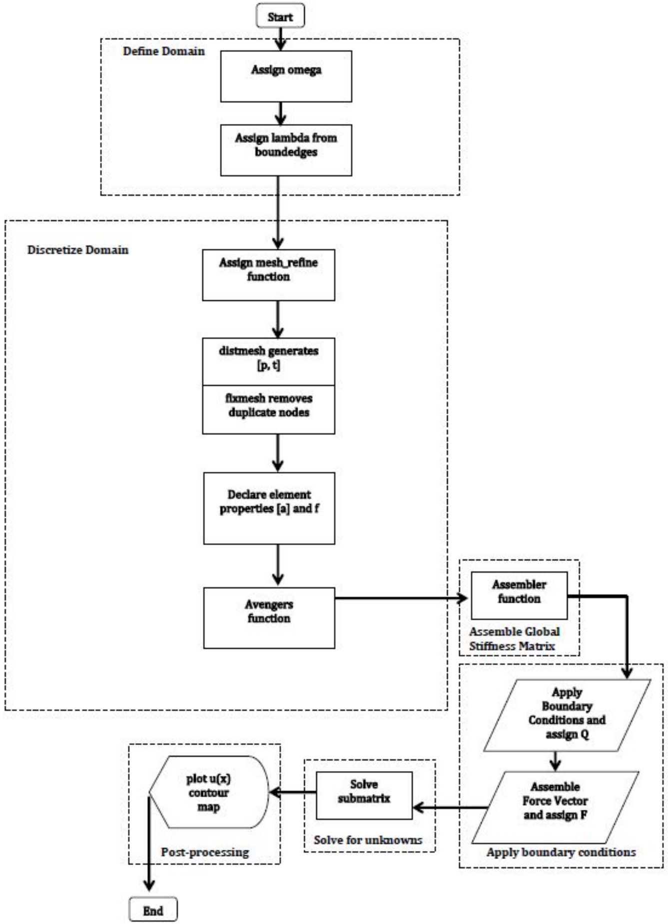
$$[K]\{u\} = \{f\} + \{Q\}$$

Linear System of Algebraic Equations (the model)

Algorithm Development

Algorithm Development

1. Define Domain
2. Discretize Domain
3. Create local stiffness matrices, force vectors
4. Assemble global stiffness matrix, force vector
5. Apply boundary conditions
6. Solve for nodal values, interpolate between nodes
7. Post-process results and visualize



Flowchart of Algorithm

Algorithm Development - Meshing

Algorithm: ‘distmesh2D.m’[1]

1. Create initial distribution in bounding box of equilateral triangles
2. Remove points outside the specified domain, applying a rejection method
3. Retriangulation using the Delaunay algorithm; save new positions
4. Create the set of edges of each triangular element as a set of 2 nodes
5. Graph it (not final result)
6. Move the elements around based on edge lengths -> get a better fit
7. Bring points that have drifted out of the domain back into it
8. Time-step until the all the nodes move less than specified tolerance distance

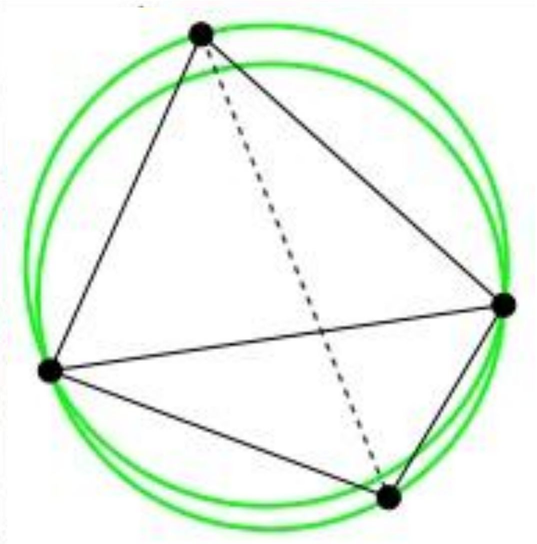
[1]: P.-O. Persson, **Mesh Generation for Implicit Geometries**.
Ph.D. thesis, Department of Mathematics, MIT, Dec 2004

Algorithm Development- Meshing

Delaunay Triangulation

$$\begin{vmatrix} A_x & A_y & A_x^2+A_y^2 & 1 \\ B_x & B_y & B_x^2+B_y^2 & 1 \\ C_x & C_y & C_x^2+C_y^2 & 1 \\ D_x & D_y & D_x^2+D_y^2 & 1 \end{vmatrix} = \begin{vmatrix} A_x-D_x & A_y-D_y & (A_x^2-D_x^2)+(A_y^2-D_y^2) \\ B_x-D_x & B_y-D_y & (B_x^2-D_x^2)+(B_y^2-D_y^2) \\ C_x-D_x & C_y-D_y & (C_x^2-D_x^2)+(C_y^2-D_y^2) \end{vmatrix}$$

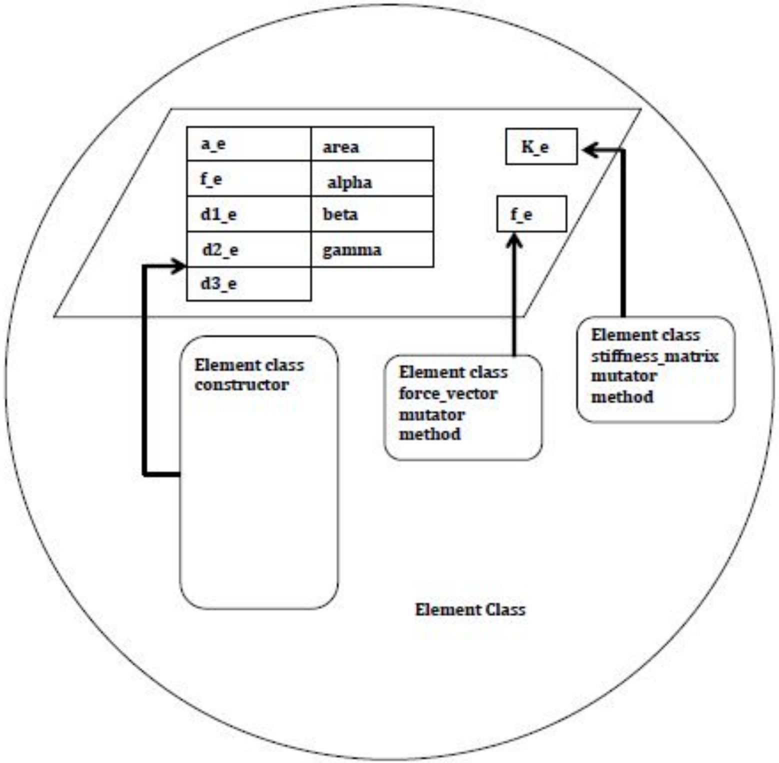
MATLAB function: `T = delaunayn(X)`



Algorithm Development - Generating K^e and f^e

- Object-Oriented approach using a class `Element2D`

1. Define element properties `[a]` and `f`
2. Create object-array of elements using `avengers`
   a. Create elements using `Element2D` constructor
   b. Build stiffness matrices for each element using `stiffness_matrix` method
   c. Build force vectors for each element using `force_vector` method

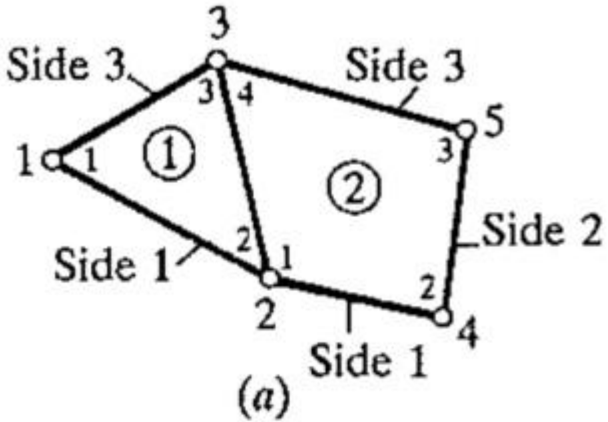


Object-Implementation Diagram of `Element2D`

Algorithm Development - Assembling K and F

1. Constructs cells from the element object-array to hold
2. Assembles stiffness matrix based on balance conditions
3. Assembles force vector based on balance conditions

Global →	Local
K_{11}	K_{11}^{1}
K_{12}	K_{12}^{1}
K_{22}	$K_{22}^{1} + K_{11}^{2}$
K_{14}	0
K_{15}	0
K_{23}	$K_{23}^{1} + K_{14}^{2}$



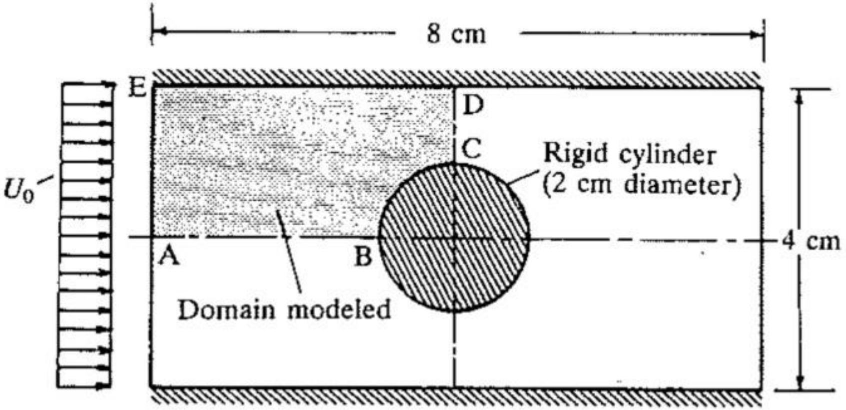
(reproduced from Reddy, for educational purposes only)

Verification Problem

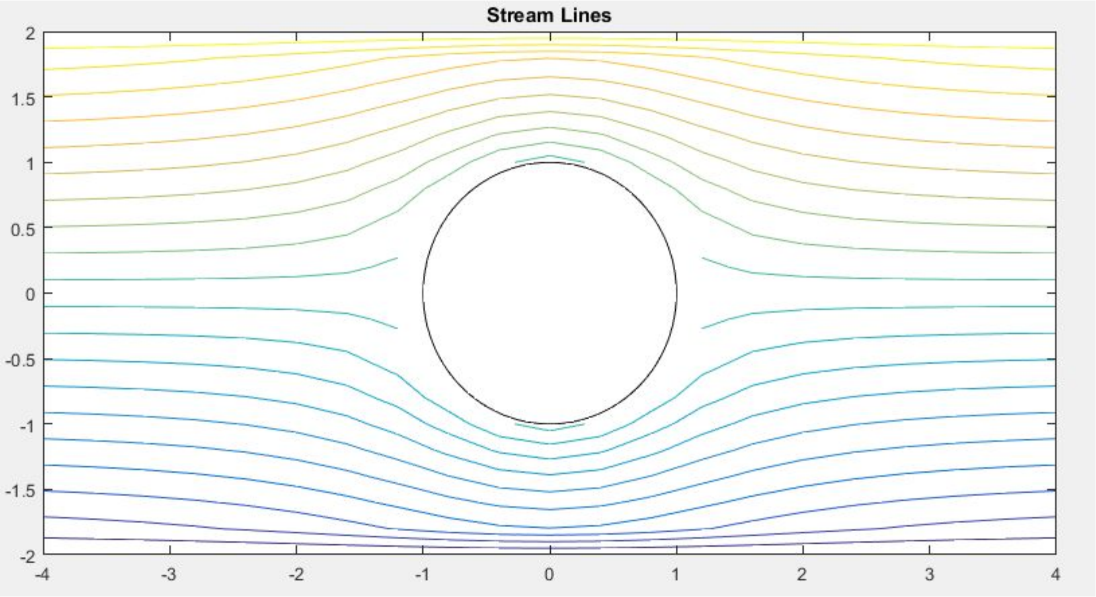
Verification Problem

Find the stream function of an inviscid, incompressible fluid over a cylinder between two plates. Assume irrotational flow.

$$\nabla^2\psi = 0$$



Verification Problem - Analytical Solution



$$\psi = U_{\infty} r \left(1 - \frac{a^2}{r^2}\right) \sin\theta$$

Verification Problem - Numerical Results

TBD

Conclusion

- Developed a finite element algorithm in 2D
  - Meshes
  - Generates K^e and f^e
  - Assembles K and f
  - User can define domain and apply boundary conditions to solve
- Currently verifying algorithm with a 2D fluids problem

Questions?