FINITE ELEMENT METHODS

ARDEN RASMUSSEN

Abstract. Finite Element Methods (FEM) is extremely useful across many different fields of research, due to the versatility that it is able to supply.

1. Introduction

The Finite Element Method(FEM) is a numerical solution for solving partial differential equations with complex boundary conditions. The finite element method generates a system of algebraic solutions approximating the solution to the partial differential equation problem. The general premise of the method is to divide the domain into a set of finite elements. The problem is then solved for each of the finite elements, then the global solution can be found by combining the solutions to all the finite elements.

2. Generation of Finite Mesh

The first stage in the computational process of FEM, is to generate the finite mesh over the domain of the problem. There are several different methods that can be used to generate the mesh, the method that this paper utilizes is Delaunay triangulation. An alternative that is less utilized is the method of polygon clipping. The scope of this paper is restricted to two dimensional domains. However, the method for mesh generation is easily generalizable to higher dimensions.

The generation of a mesh using Delaunay triangulation has several stages.

(1) Delaunay triangulation
(2) Applying boundaries
(3) Mesh refinement

2.1. **Delaunay Triangulation.** A Delaunay triangulation for a set of points P is a triangulation $DP(P)$ such that no point in P is in the circumcircle of any triangle in the triangulation $DP(P)$. This form of triangulation maximizes the minimum angle of all triangles in the triangulation, thus avoiding sliver triangles. See Fig 1 for an example of Delaunay Triangulation.

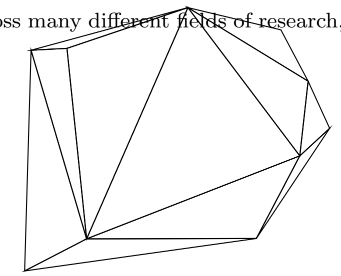

Figure 1. Example of Delaunay Triangulation

The generation of the Delaunay triangulation has several different algorithms, and is constantly advancing in efficiency. There are three main algorithms that are documented for the generation of the triangulation.

(1) Edge flipping
(2) Incremental
(3) Divide and conquer

It is important to note that each algorithm will produce the same resulting triangulation. This is due to the fact that the Delaunay triangulation associated with a set of points is unique, with one exceptions of the points on a square.

2.1.1. *Edge Flipping.* This method for generating the Delaunay triangulation, is a very simple algorithm, and not very efficient.

2.1.2. *Incremental.* This method for generating the Delaunay triangulation is better than the 1.1.1 algorithms, but can still be improved upon.

The basis of this method, is to incrementaly add the points from P to the triangulation, and regenerate the triangles that that point is within the circumcircle of.

The steps of the algorithm are detailed below. which is then followed by psudocode.

(1) Generate a bounding triangle, such that all points in P are within the triangle. Append the points of this bounding triangle to the end of P.
(2) For every point in P do 3-6

Date: September 30, 2018.

(3) Select p from P
(4) For every triangle in $DT(P)$ if p is within the circumcircle of that triangle add that triangle to a list of bad triangles BT and remove that triangle from $DT(P)$.
(5) For every triangle in BT, add every edge of the triangle to a polygon $Poly$, only if that edge is not shared by any other triangles in BT.
(6) For every edge in $Poly$ create a new triangle by connecting that edge to p.
(7) Remove all triangles from $DT(P)$ if one of the verticies is one of the three generated in step 1.

2.1.3. *Divide and Conquer.* This algorithm for constructing the Delaunay triangulation is currently shown to be the fastest, and allows for optimization in computational execution that the previous algorithms lacked.

This algorithm allows for easy parallelism in the computation of the triangulation, thus allowing computers to calculate the triangulation much more efficiently than the other algorithms permit.

2.2. **Applying Boundaries.** Now that the Delaunay triangulation has been generated, the next stage is to apply the constraints of the boundary conditions to the triangulation.

2.3. **Mesh Refinement.** The final stage is to refine the mesh, because the restriction of the boundary conditions are able to cause non-delaunay triangles. This stage also applies larger restrictions to the triangles, resulting is “nicer” triangles with restricted minimum angles.