DREXEL UNIVERSITY

The Finite Element Method Applied to Quantum Mechanics

by

William Paul Czaja

in the

Department of Physics

May 2010

Contents

Chapter 1

Background Theory

1.1 Eigenvalue problems

Resonating systems display distinct patterns known has eigenmodes. Every piece of these oscillating patterns vibrates at the same frequency λ, also known as an eigenvalue. In classical mechanical applications λ defines the normal mode frequencies, and in quantum mechanical applications λ defines the energy levels of bound states in a potential. To understand the structure of the eigenmode patterns we must take advantage of eigenvectors. Eigenvectors maintain their direction regardless of the linear transformation the system undergoes. Examples of linear transformations include reflection, rotation, stretching, compression, and shear. The general expression which describes the eigenmodes is as follows:

$$Ax = \lambda x \tag{1.1}$$

Where A is a matrix which describes the linear transformation, x is an eigenvector, and λ is an eigenvalue.

This can be solved using the following analytic algorithm:

1. Compute the determinant of $A - \lambda I$, where I is the identity matrix.
2. Set this determinant equal to zero and solve for its roots. This is known as the characteristic equation. Each root is an eigenvalue λ.
3. For each of these eigenvalues solve $(A - \lambda I)x = 0$, where x is an eigenvector.

1.2 The Finite Element Method

There is a very limited set of analytic solutions that can solve eigenvalue problems. It is exceedingly difficult to analytically examine nonlinear shapes so it's best to use a numerical approach. Additionaly, creating computer models of resonating systems enables testing different initial conditions efficiently without the need of experimental physical trials. A numerical technique known as the finite element method (FEM) is used to make the problem more discrete by using a linear piecewise approximation. Finite element methods solve partial differential equations for highly irregular geometries which may require the crunching of extremely large matrices. A finite dimensional subspace consisting of piecewise linear functions is used to approximate an infinite dimensional problem. The FEM process begins with creating a tessellation of polygons over the geometry. Boundary conditions are defined. Then basis functions are then chosen to treat the piecewise approach. There are robust software packages currently available which can handle FEMs. Software that I've found which can do this includes MATLAB, ANSYS, Solid Works, and Pro-Engineer. In particular I have choosen to use MATLAB packages to accomplish this task.

Chapter 2

The Finite Element Method Applied to Quantum Mechanics

2.1 Engineers Know Best

All of us are taught in our quantum mechanics courses how to analytically solve the ubiquitous Schrodinger's equation for various potentials, but our professors are not telling us a dirty little secret. Whereas it may be beneficial to understand the underlining theory and math behind solutions to the Schrodinger equation, the vast majority of real-world quantum mechanical applications of the Schrodinger equation are out of our ability to solve analytically. This is because the shapes of real-world potentials are usually highly irregular. As responsible scientists we must sacrifice our egos and lean on methods developed by the humble yet ever-resourceful mechanical engineers. They never hesitate to use approximate solutions if the accuracy suffices the application and it gets the job done. Just like finite element methods can be used to solve complex resonance problems for classically mechanical structures it can be applied to quantum mechanical problems using the wave mechanics of the Schrodinger equation.

2.2 Mapping Differential Equations to Matrix Mechanics

The first step in applying the finite element method to quantum mechanics is to map wave mechanics into matrix mechanics. We start by considering the variational version of Schrodinger's equation which expresses the wavefunction ψ of a particle, the potential

V it moves in, and its energy E:

$$\delta \int \left\{ \frac{\hbar^2}{2m} \nabla\psi^*(x) \cdot \nabla\psi(x) + \nabla\psi(x) + \psi^*(x) V(x) \psi(x) - \psi^*(x) E \psi(x) \right\} d^3x = 0 \quad (2.1)$$

It is now our task to quantize the wave mechanics at a discrete set of points in configuration space. To do this we define a basis set of real functions to approximate the wavefunction and potential:

$$\psi^*(x) = \sum_i \phi_i f_i(x) \qquad V(x) = \sum_j V_j f_j(x) \qquad \psi(x) = \sum_k \psi_k f_k(x) \quad (2.2)$$

Plugging in this basis set of functions into the variational version of Schrodinger's equation and converting to summation convention produces our eigenvalue equation:

$$\delta \left\{ \frac{\hbar^2}{2m} \phi_i (\nabla f_i \nabla f_k) \psi_k + \phi_i V_j (f_i f_j f_k) \psi_k - \phi_i E (f_i f_k) \psi_k \right\} = 0 \quad (2.3)$$

We can interpret this now as a matrix equation:

$$\begin{aligned} \mathbf{M_{ik}} \psi_\mathbf{k} &= 0 && (2.4) \\ \mathbf{M_{ik}} &= \mathbf{K_{ik}} + \mathbf{V_{ik}} - \mathbf{S_{ik}} \mathbf{E} && (2.5) \\ \mathbf{K_{ik}} &= (\nabla f_i \nabla f_j) && (2.6) \\ \mathbf{V_{ik}} &= V_j (f_i f_j f_k) && (2.7) \\ \mathbf{S_{ik}} &= (f_i f_k) && (2.8) \end{aligned}$$

These are known as the Kinetic Energy Matrix, the Potential Energy Matrix, and the Overlap Matrix, (**K**,**V**,**S**) respectively. The matrix **M** can be diagonalized and solved for the eigenvalues, and eigenvectors.

2.3 Tesselation

In order to establish a linear piecewise approximation for a particular geometry it needs to be divided into elements to create a finite dimentional subspace. This can be accomplished in various ways depending on the dimensions of the geometry and its irregularity. The elements are commonly known as primitives. Some meshing algorithms segment the edges of the geometry into equal divisions and then intelligently scatter nodes in the subspace. The nodes are then connected to create many ploygonal primitives which are usually choosen to be triangular. For simple configuration spaces like rectangles and triangles the sides are evenly segmented and then the nodes of the segments are cross

connected through the subspace. As the refinment of the tesselation increases so will the accuracy of the approximated eigenfunctions.

2.4 Choosing the Right Basis Functions

A basis on the subspace must be found to complete the discretization. Altering the tesselation method can improve the basis. Advanced tesselation algorithms are self adaptive and will change the placement of the nodes, the refinement of the elements, and the order of the functions to insure the best fit possible. The basis functions (f_i, f_j, f_k) at a particular vertex are found through a process of unioning the basis functions of all other elements connected to that vertex. The basis functions and their gradients are then integrated to construct the matrix elements for the Kinetic Energy Matix $\mathbf{K}$ and the Overlap Matrix $\mathbf{S}$. A connectivity matrix $\mathbf{T}$ is used as a matrix template to fit these elements. $\mathbf{T}$ describes how the coordinates of the vertices are interconnected.

Chapter 3

Tools

3.1 MATLAB's Partial Differential Equation Toolbox

MATLAB includes a powerful package called the Partial Differential Equation Toolbox, PDETool for short. PDETool can be used to numerically solve PDE problems. In it you can define 2-dimensional geometries, generate tessellations (meshes), output matrices, discretize equations, produce visualizations of the normal modes, and produce an approximate solution to the problem. The process of treating a particular problem in PDETool can be done in two ways, through the built in graphical interface, and through commands in a MATLAB script. PDETool uses the following partial differential equation for calculating eigenvalue problems:

$$-\nabla\cdot(c\nabla u) + au = \lambda du \tag{3.1}$$

where λ is an unknown complex eigenvalue, and a is a potential well. This problem is treated similarly to the quantum mechanics problem given earlier. The equation is discretized and solved algebraically. This forms the generalized eigenvalue equation:

$$\mathbf{KU} = \lambda\mathbf{MU} \tag{3.2}$$

where $\mathbf{M}$ is known as the mass matrix, and $\mathbf{K}$ is known as the stiffness matrix. This eigenvalue equation is then run through an Arnoldi algorithm until all eigenvalues are found on the specified interval. The Arnoldi algorithm runs through Gram-Schmidt iterations to find orthonormal vectors of Krylov subspaces.

3.2 DistMesh

DistMesh is a mesh generator package for MATLAB. It was created by Per-Olof Persson and Gilbert Strang of MIT. It can handle intelligent tesselation on very complex geometries in 2D and 3D. The tesselation is done by using the Delaunay method.

Chapter 4

PDE Tool Results and Interpretation

4.1 Geometry Definition

PDETool is started by entering 'pdetool' at the MATLAB command line. To begin an arbitrary geometry is defined. The function pdepoly(x,y) where **x** and **y** are vectors containing the coordinates of the vertices, can define an arbitrary geometry. You can also draw any 2-D geometry by going to the 'Draw' menu and selecting 'Draw Mode'

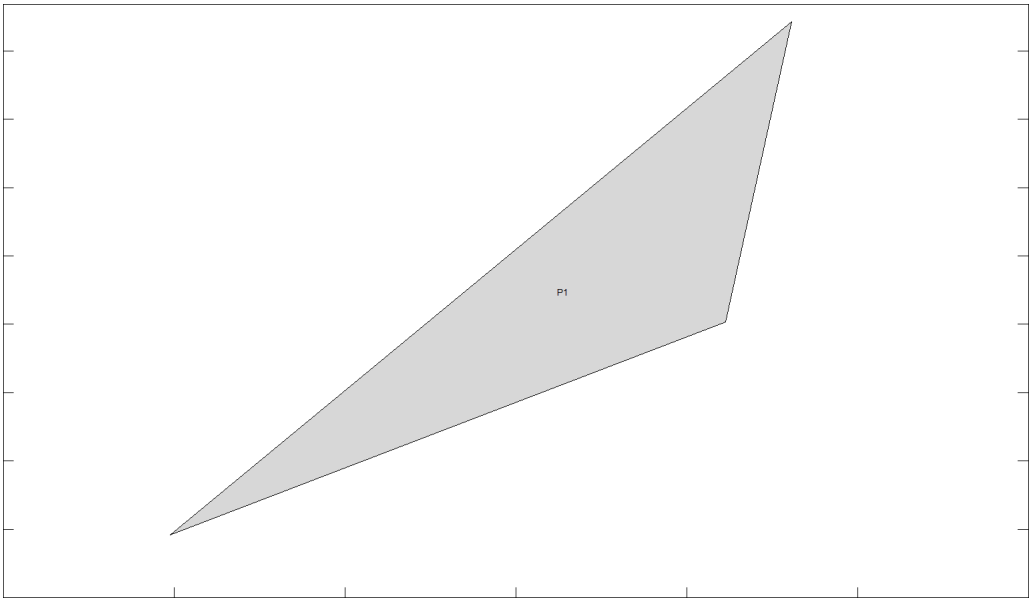

FIGURE 4.1: The Geometry definition for an arbitrary triangle

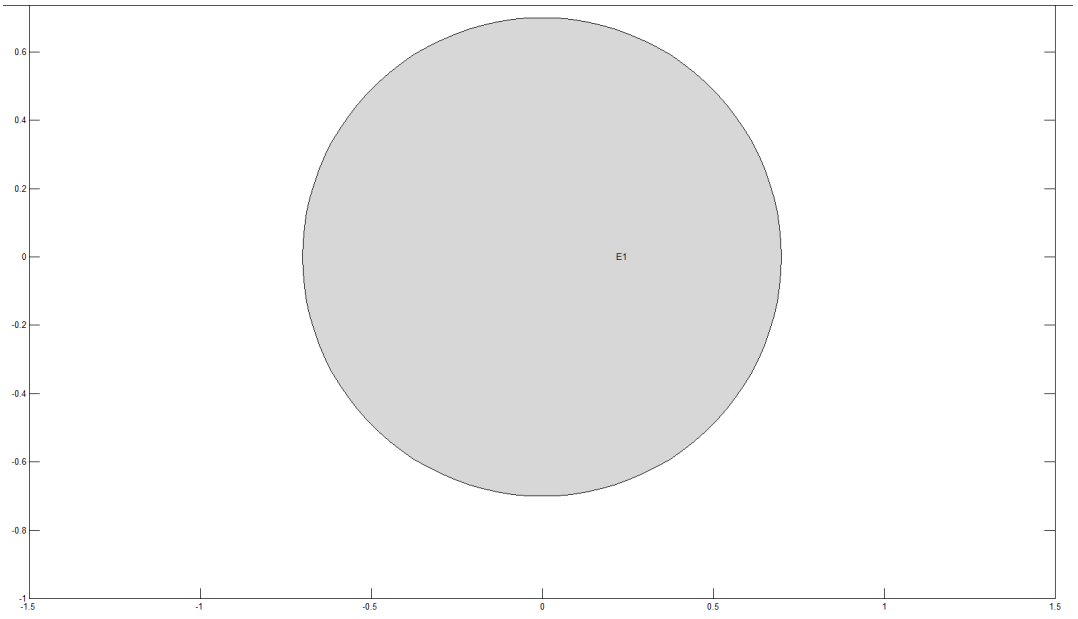

FIGURE 4.2: The Geometry definition for a circular disk centered at the origin

4.2 Tesselation

4.2.1 Initial Mesh

Next we generate a mesh on the geometry. We do this by going to the 'Mesh' menu and selecting 'Initialize Mesh'.

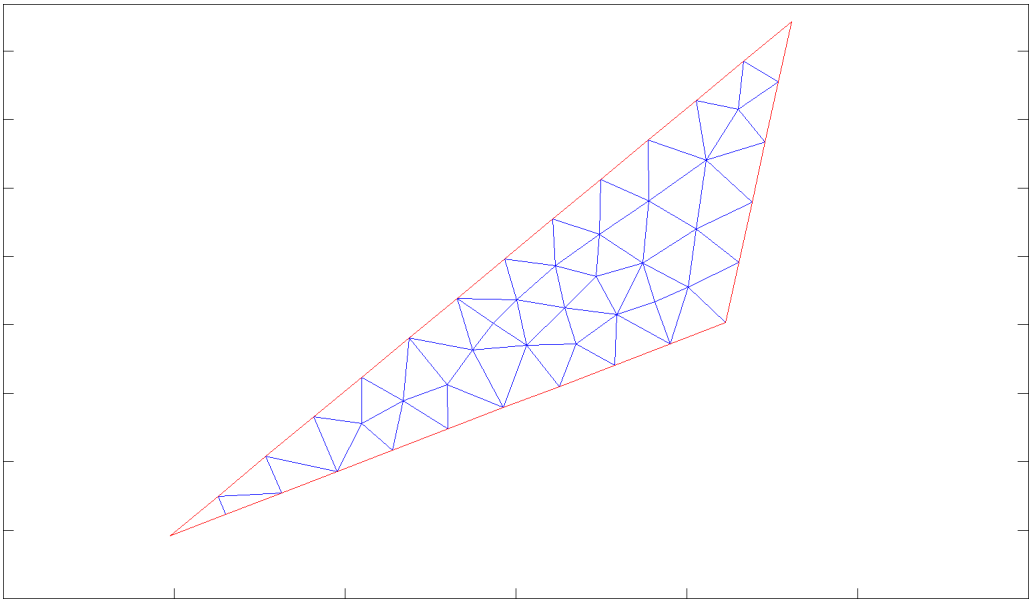

FIGURE 4.3: The initial meshing for an arbitrary triangle

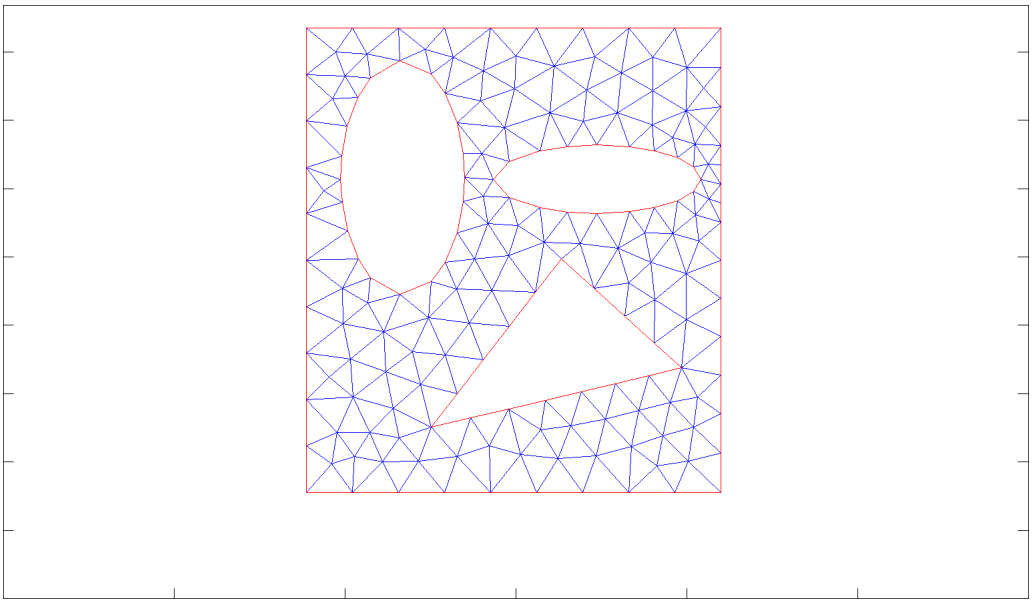

FIGURE 4.4: The initial meshing for an irregular geometry

4.2.2 Refined Mesh

In order to produce a more accurate result we need to refine the mesh. We do this by going to the 'Mesh' menu and selecting 'Refine Mesh'. This function doubles the edge vertices.

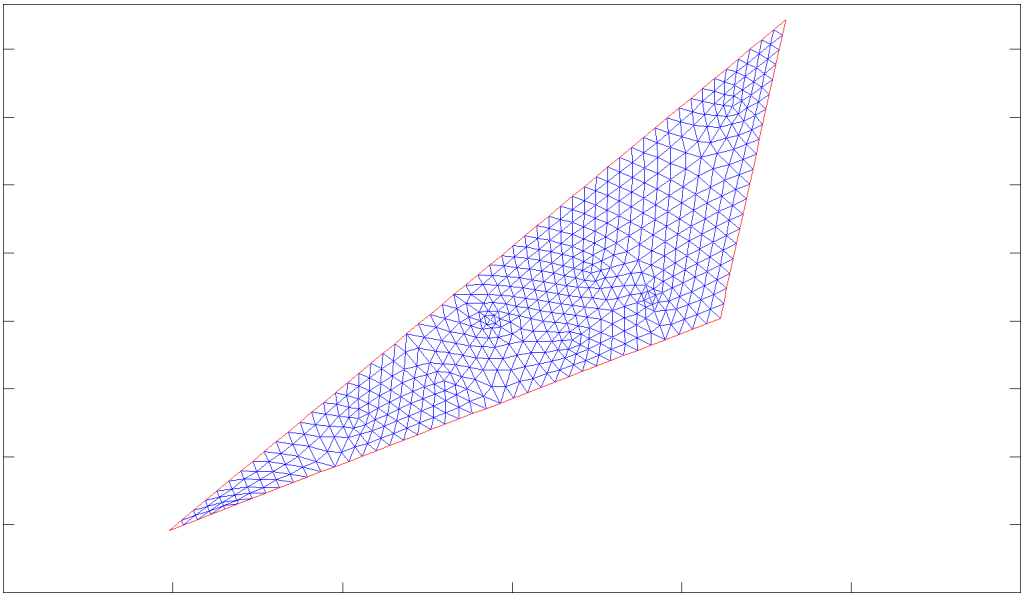

FIGURE 4.5: The refined meshing for an arbitrary triangle, refined twice

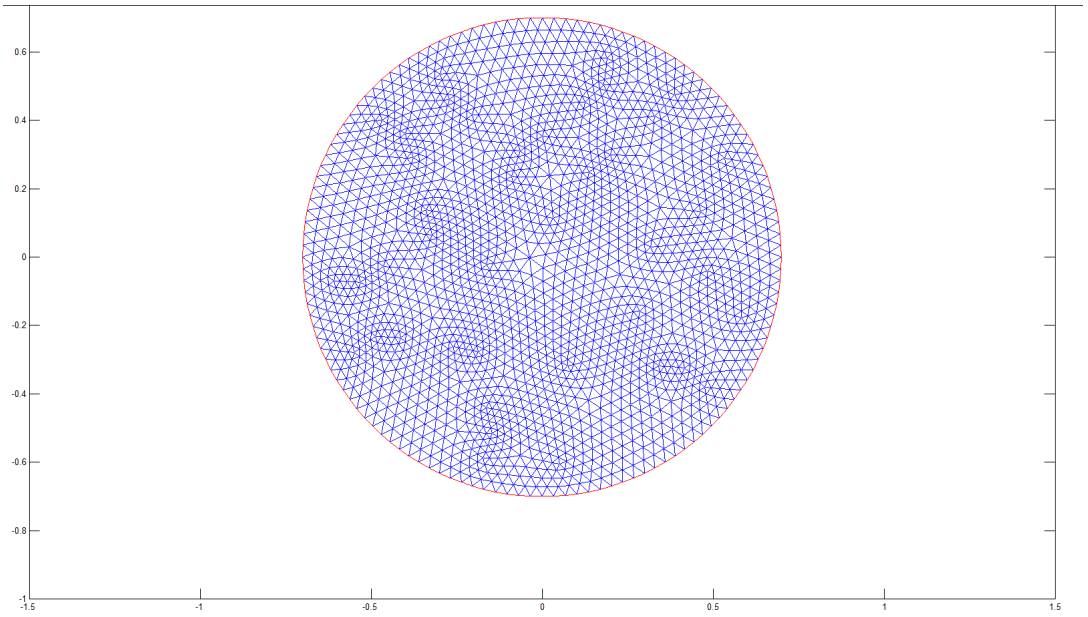

FIGURE 4.6: The refined meshing for a circular disk, refined twice

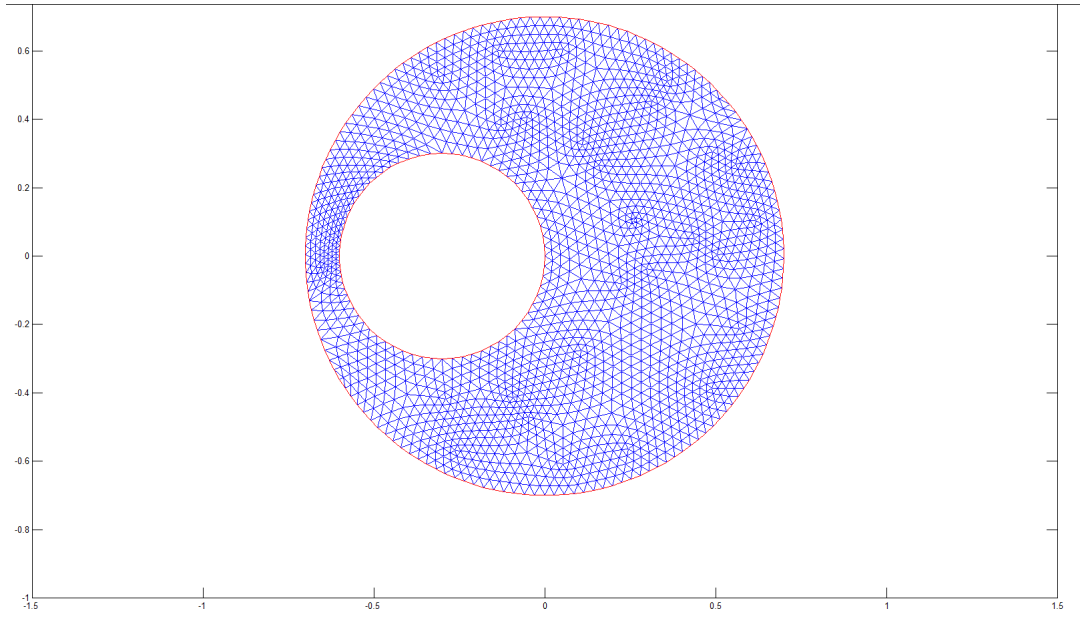

FIGURE 4.7: The refined meshing for an circular disk with offset hole, refined twice

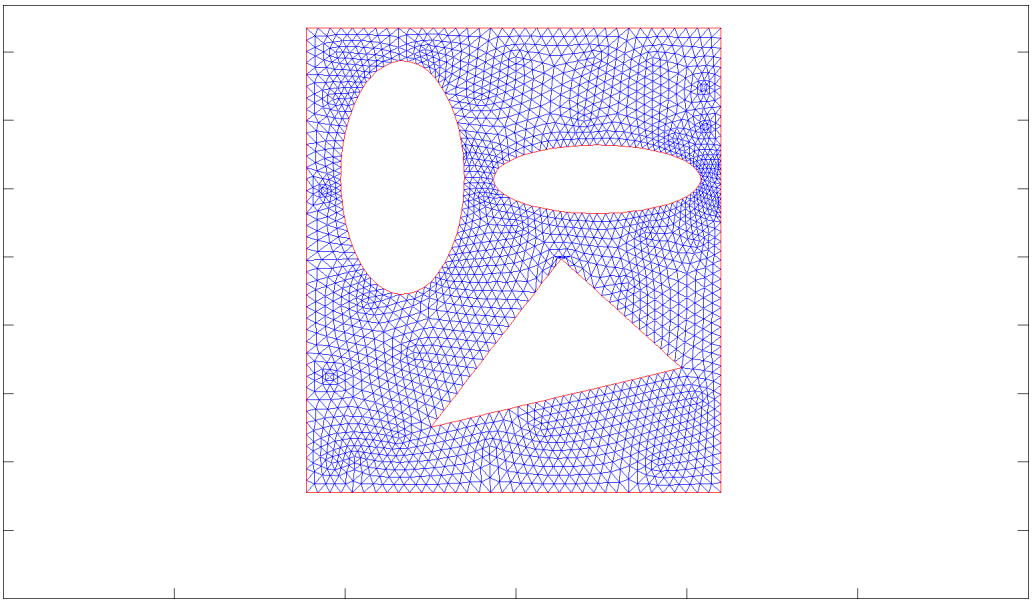

FIGURE 4.8: The refined meshing for an irregular shape, refined twice

4.3 PDE Specification

Now that we have defined our geometry and generated a mesh we can explore the eigenmodes. This can be achieved by hand by outputting the matrices of the mesh to the MATLAB workspace and solving the eigenvalue problem for the eigenvectors. PDETool can visualize the eigenmodes for you. You start by going to the 'PDE' menu and then selecting 'PDE Specification'. Choose 'Eigenmodes' from this window and then 'Ok'. With the PDE specified you can now go to the 'Solve' menu and then select 'Solve PDE'. This will produce a visualization of the first eigenmode.

4.4 Visualizations

To view other eigenmodes go to the 'Plot' menu and select 'Parameters'. Choose a different eigenvalue and click 'Plot'. A three dimensional visualization of the membrane can also be created through the plot parameters menu.

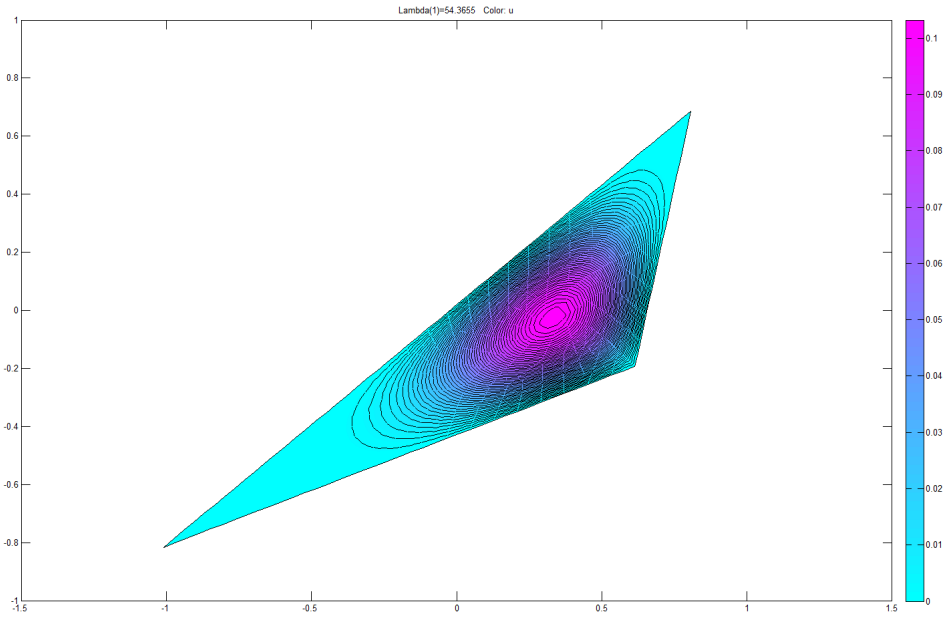

FIGURE 4.9: An arbitrary triangle

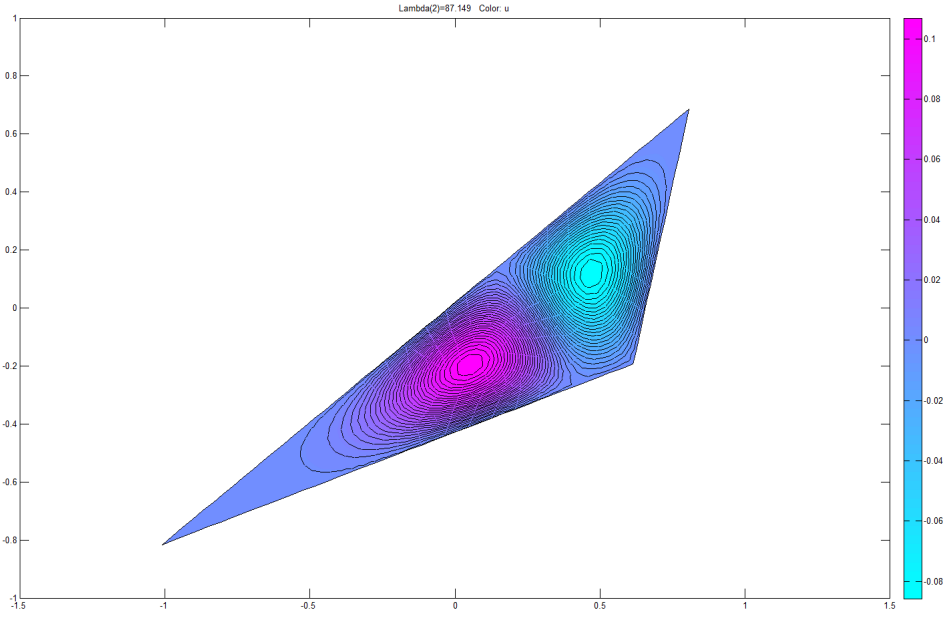

FIGURE 4.10: An arbitrary triangle

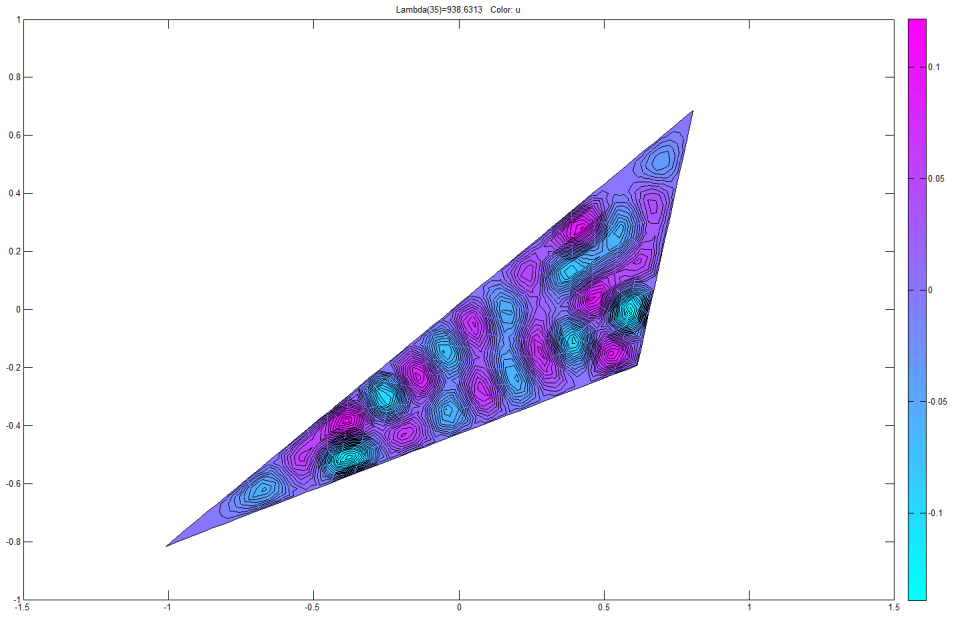

FIGURE 4.11: An arbitrary triangle

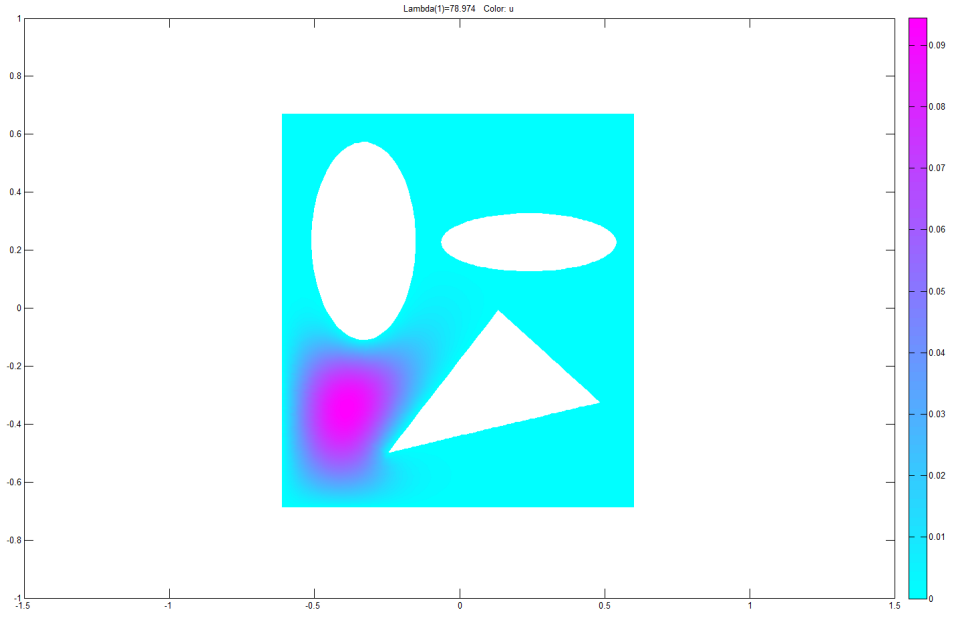

FIGURE 4.12: Eigenmode for an irregular shape

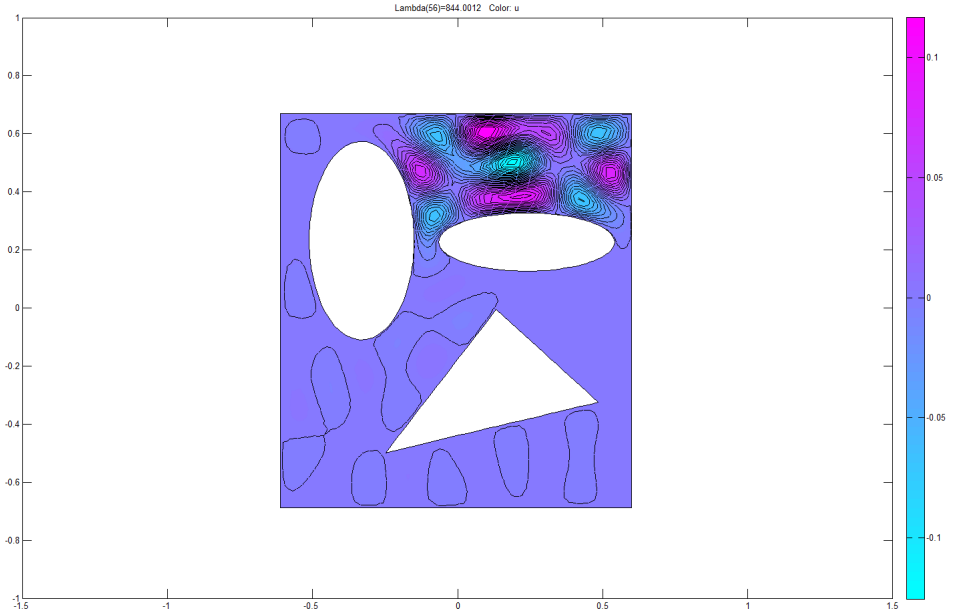

FIGURE 4.13: Eigenmode for an irregular shape

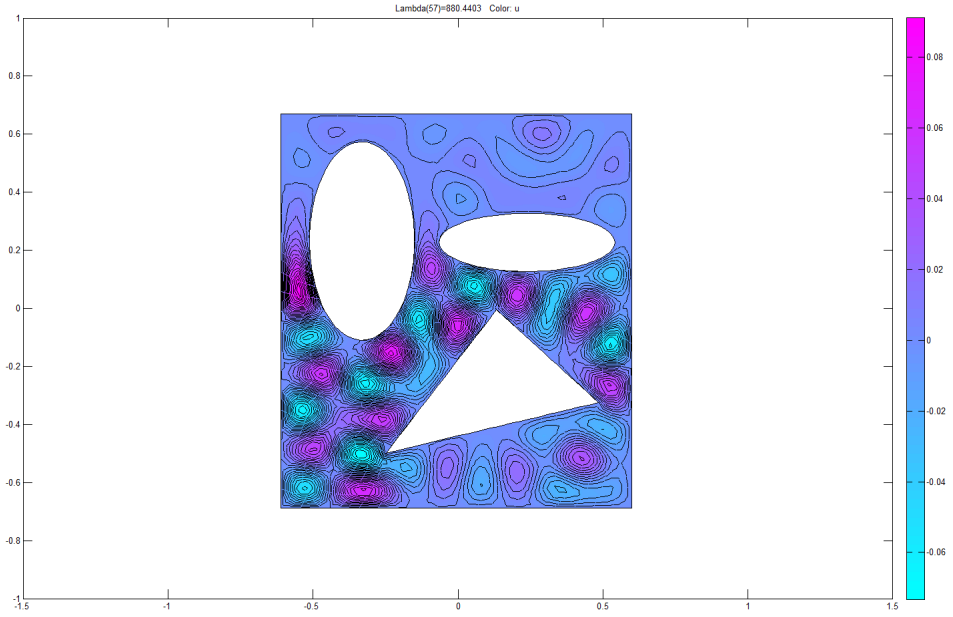

FIGURE 4.14: Eigenmode for an irregular shape

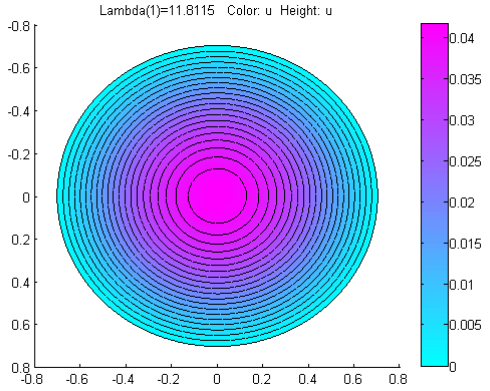

FIGURE 4.15: Eigenmode for a circular disk

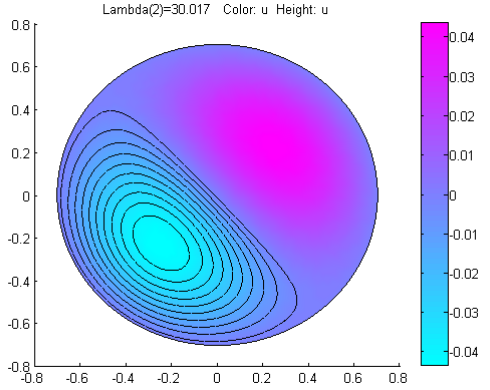

FIGURE 4.16: Eigenmode for a circular disk

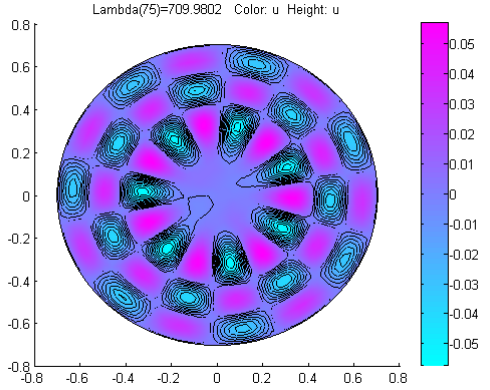

FIGURE 4.17: Eigenmode for a circular disk

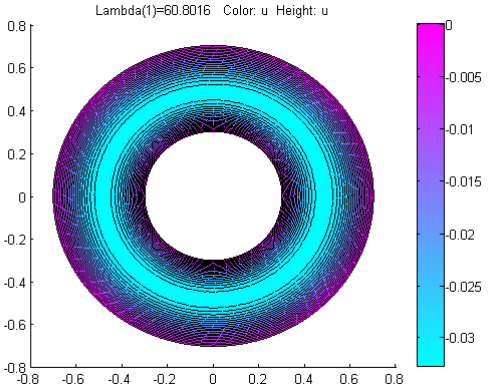

FIGURE 4.18: Eigenmode for a circular disk with a hole at the origin

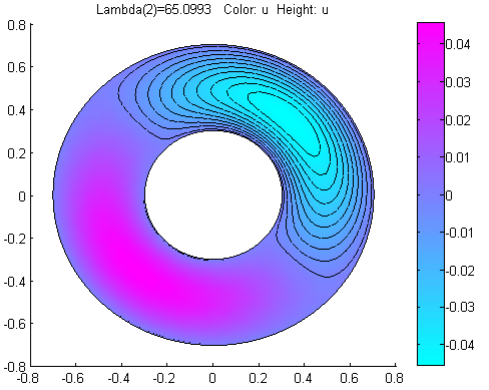

FIGURE 4.19: Eigenmode for a circular disk with a hole at the origin

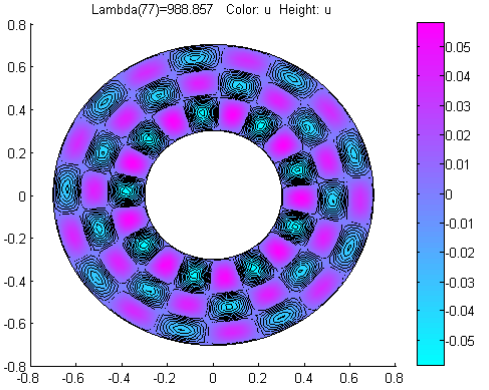

FIGURE 4.20: Eigenmode for a circular disk with a hole at the origin

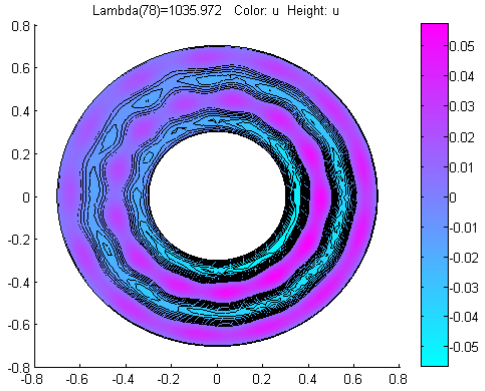

FIGURE 4.21: Eigenmode for a circular disk with a hole at the origin

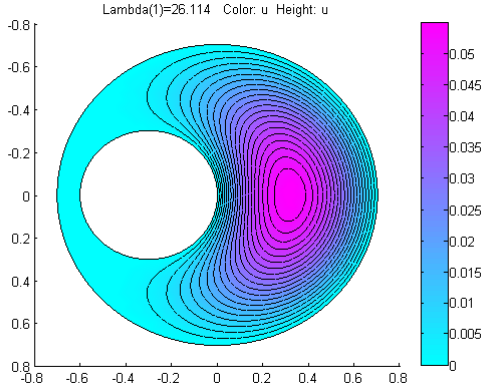

FIGURE 4.22: Eigenmode for a circular disk with offset hole

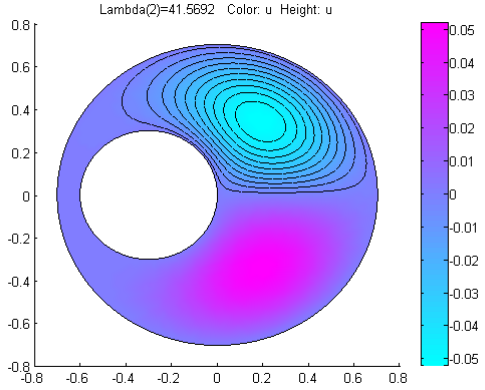

FIGURE 4.23: Eigenmode for a circular disk with offset hole

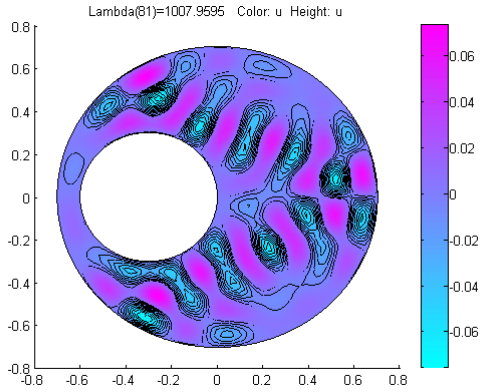

FIGURE 4.24: Eigenmode for a circular disk with offset hole

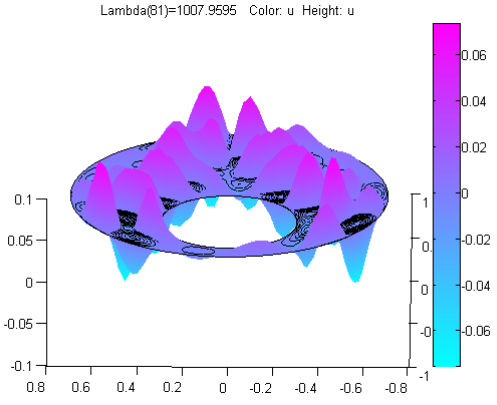

FIGURE 4.25: 3D eigenmode for a circular disk with offset hole

Bibliography