Finite Element Method: MATLAB-based Implementation

Aarosh Dahal[1], Group: 04, GT ID: 903914858

School of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

1. Introduction

The laws of nature are often explained using differential equations. While analytical solutions to differential equations do give accurate results, complex systems are virtually impossible to solve analytically. This is where numerical methods come into play, especially with the progress in computation these methods have dominated the field of scientific modeling. The finite element method is a variational numerical method, the history of which dates back to the 1940's. One of the first primitive forms of the finite element method was delivered by Courant [1] in a lecture at the American Mathematical Society held in Washington D.C. Since then the finite element method has seen significant progress and ubiquitous implementation, from engineering to research. As such, the purpose of this project is to develop a basic finite element program capable of solving simple statics problems.

2. Theoretical Background

Let us consider an isotropic linear elastic body occupying an open bounded domain $\Omega_0 \subset \mathbb{R}^3$, with Dirichlet boundary $\partial\Omega_0^{\mathrm{D}}$ and Neumann boundary $\partial\Omega_0^{\mathrm{N}}$. Let us consider at any time t, the following symbols represent:
$\mathbf{u}$ = Displacement field, $\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)$ = Strain, $\boldsymbol{\sigma}$ = Stress, $\mathbf{b}$ = Body force, $\mathbf{t}$ = Traction,
which are all functions of the coordinates $\mathbf{x}$.

2.1. Principle of Minimum Potential Energy

The total potential energy of the system is given by:

$$\Pi(\mathbf{u}) = \text{Internal energy} - \text{External work done}$$

$$= \frac{1}{2}\int_{\Omega_0} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}\, dV - \int_{\Omega_0} \mathbf{b}\cdot\mathbf{u}\, dV - \int_{\partial\Omega_0^N} \mathbf{t}\cdot\mathbf{u}\, dS$$

The Principal of Minimal Potential Energy states that the system maintains a $\mathbf{u}$ that minimizes $\Pi(\mathbf{u})$.

Let $\delta\Pi$ be variation of $\Pi(\mathbf{u})$ w.r.t $\mathbf{u}$ in direction of $\delta\mathbf{u}$.

Then,

$$\delta\Pi = \left.\frac{\partial\Pi(\mathbf{u}+\eta\delta\mathbf{u})}{\partial\eta}\right|_{\eta\to 0} = 0$$

$$\frac{\partial}{\partial\eta}\left[\int_{\Omega_0}\frac{1}{2}\left(\boldsymbol{\sigma} : \boldsymbol{\varepsilon}(\mathbf{u}+\eta\delta\mathbf{u})\right)dV - \int_{\Omega_0}\mathbf{b}\cdot(\mathbf{u}+\eta\delta\mathbf{u})\, dV - \int_{\partial\Omega_0^N}\mathbf{t}\cdot(\mathbf{u}+\eta\delta\mathbf{u})\, dS\right|_{\eta\to 0}\Bigg] = 0$$

Email address: adahal8@gatech.edu (GT ID: 903914858)

where $\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon}$, $\mathbb{C}$ is modulus of elasticity tensor.

Thus,

$$\int_{\Omega_0} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\delta\mathbf{u})\, dV - \int_{\Omega_0} \mathbf{b} \cdot \delta\mathbf{u}\, dV - \int_{\partial\Omega_0^N} \mathbf{t} \cdot \delta\mathbf{u}\, dS = 0 \tag{A}$$

This is the **weak form**.

To get strong form, we apply integration by parts and then divergence theorem to the first term:

$$\int_{\Omega_0} \boldsymbol{\sigma} : \nabla\delta\mathbf{u}\, dV = \int_{\Omega_0} \nabla \cdot (\boldsymbol{\sigma} \cdot \delta\mathbf{u})\, dV - \int_{\Omega_0} (\nabla \cdot \boldsymbol{\sigma}) \cdot \delta\mathbf{u}\, dV = \int_{\partial\Omega_0^N} (\boldsymbol{\sigma} \cdot \delta\mathbf{u}) \cdot \mathbf{n}\, dS - \int_{\Omega_0} (\nabla \cdot \boldsymbol{\sigma}) \cdot \delta\mathbf{u}\, dV$$

Thus,

$$\int_{\Omega_0} (\nabla \cdot \boldsymbol{\sigma}) \cdot \delta\mathbf{u}\, dV - \int_{\Omega_0} \mathbf{b} \cdot \delta\mathbf{u}\, dV + \int_{\partial\Omega_0^N} (\boldsymbol{\sigma} \cdot \mathbf{n}) \cdot \delta\mathbf{u}\, dS - \int_{\partial\Omega_0^N} \mathbf{t} \cdot \delta\mathbf{u}\, dS = 0$$

Since $\delta\mathbf{u}$ is arbitrary:

$$\nabla \cdot \boldsymbol{\sigma} - \mathbf{b} = 0 \quad \text{at} \quad \Omega_0 \tag{B}$$

$$\boldsymbol{\sigma} \cdot \mathbf{n} - \mathbf{t} = 0 \quad \text{at} \quad \partial\Omega_0^N$$

This is the **strong form** of the system which is also the equilibrium equations.

2.2. Galerkin Variational Approach

Notice that in (A), we have $\delta\mathbf{u}$ (the test function), which is allowed to take any form. If we choose $\delta\mathbf{u}$ such that it is in the same function space as the approximate displacement field $\mathbf{u}^h(x)$, satisfying $\delta\mathbf{u} = 0$ at $\partial\Omega_0^D$, we obtain the Galerkin form.

In short,

$$\mathbf{u}^h \in \{\mathbf{u} \in H^1 \,|\, \mathbf{u} = \mathbf{u}_0 \text{ on } \partial\Omega_0^D\}$$

then,

$$\delta\mathbf{u} \in \{\mathbf{v} \in H^1 \,|\, \delta\mathbf{u} = 0 \text{ on } \partial\Omega_0^D\}$$

where H^1 represents Hilbert space with square integrable derivatives (needed to maintain C^0 continuity).

2.3. Shape Functions

Now, the crux of FEM is in approximating the function $\mathbf{u}$ using approximation over each element, i.e.,

$$\mathbf{u}^h(x) = \sum_i N_i^{(j)} u_i^{(j)}$$

where j represents the element, i represents the nodes in the element, $u_i^{(j)}$ represents the value of ith d.o.f. in jth element.

Then,

$$\mathbf{u}^h = \sum_j \sum_i N_i^{(j)} u_i^{(j)}$$

This concept of relating field to nodal values can be explained as follows.
Say,

$$u(x) = a_1x_1 + a_2x_2^2 + a_3x_3^3$$

Then,

$$u_1 = a_1x_1 + a_2x_1^2 + a_3x_1^3$$
$$u_2 = a_1x_2 + a_2x_2^2 + a_3x_2^3$$
$$u_3 = a_1x_3 + a_2x_3^2 + a_3x_3^3$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} x_1 & x_1^2 & x_1^3 \\ x_2 & x_2^2 & x_2^3 \\ x_3 & x_3^2 & x_3^3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

$$\Rightarrow \quad A = M^{-1}\{u_i\}$$

Thus,

$$u(x) = (x \quad x^2 \quad x^3)\, A = (x \quad x^2 \quad x^3)\, M^{-1}\, \{u_i\} = N\{u_i\}$$

where N represents shape function vector.

Properties of N:

i) Compact Support: $N_i(x_j) = \delta_{ij}$

ii) Ability to reproduce constant fields exactly: $\sum_i N_i = 1$

iii) Ability to reproduce fields with constant strain exactly: $\sum_i N_i x_i = x$

2.4. Stiffness Matrix

We know, $\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon}$, where $\mathbb{C}$ is the modulus of elasticity tensor of 4th order.

$$\Rightarrow \sigma_{ij} = C_{ijkl}\varepsilon_{kl}$$

In 2D problems, we can reduce the order of $\mathbb{C}$ by repackaging $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ as vectors.
Thus, for plane stress:

$$\begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{pmatrix} = \frac{E}{1-\nu^2} \begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{pmatrix} \begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \gamma_{12} \end{pmatrix}$$

and for plane strain:

$$\begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{pmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{pmatrix} \begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \gamma_{12} \end{pmatrix}$$

where $\gamma_{12} = (\varepsilon_{12} + \varepsilon_{21})$ is the engineering shear strain.

Thus $\mathbb{C}$ becomes a second order tensor which is the material stiffness matrix.

Also, if we write shape functions as $\mathbf{N} = \begin{pmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & \cdots \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & \cdots \end{pmatrix}$, and

$$\mathbf{B} = \frac{\partial \mathbf{N}}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N_3}{\partial x} & 0 & \cdots \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N_3}{\partial y} & \cdots \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial x} & \cdots \end{pmatrix}$$

We can write the nodal displacements as:

$$\{\mathbf{u}\} = \begin{pmatrix} u_{1x} \\ u_{1y} \\ u_{2x} \\ u_{2y} \\ u_{3x} \\ u_{3y} \\ \vdots \end{pmatrix}$$

Then, from (A) and 2.3, we get:

$$\left[\int_{\Omega_0} \mathbf{B}^T \mathbb{C}\mathbf{B}\, dV\right] \{\mathbf{u}\} - \int_{\Omega_0} \mathbf{N}^T \mathbf{b}\, dV - \int_{\partial\Omega_0^N} \mathbf{N}^T \mathbf{t}\, dS = 0$$

$$\Rightarrow \mathbf{K}\mathbf{u} = \mathbf{f}$$

Here,

$$\mathbf{K} = \int_{\Omega_0} \mathbf{B}^T \mathbb{C}\mathbf{B}\, dV \quad \text{is the stiffness matrix}$$

Properties:

1) Symmetry: $K_{ij} = K_{ji}$
2) Positive semidefiniteness: $\mathbf{x}^T\mathbf{K}\mathbf{x} \geq 0$, where $\mathbf{x}$ is an arbitrary nonzero vector, i.e., eigenvalues of $\mathbf{K} \geq 0$
3) Singular: $\det(\mathbf{K}) = 0$
4) Sparse

2.5. Numerical Integration

Once the weak form is developed, computer has to perform discrete (numerical) integration. A robust method to do numerical integration is Gaussian quadrature. Gaussian quadrature can produce exact integration for polynomials.

$$\int f(x)\, dx = \sum_i w_i f(x_i) \quad \text{where } n \text{ points can produce exact integrals for polynomials of order up to } 2n-1.$$

In our case,

$$\int_{\Omega_0} \mathbf{B}^T \mathbb{C}\mathbf{B}\, dV = \sum_i w_i \mathbf{B}(x_i, y_i)^T \mathbb{C}\, \mathbf{B}(x_i, y_i)$$

Of course, $\int$ has to be in the range $[-1, 1]$, so we need to perform transformation of coordinates and use the Jacobian of transformation wherever needed.

2.6. Global Assembly and Boundaries

Now that element $\mathbf{K}_e$, $\mathbf{f}_e$ have been developed, we assemble them into a global system before solving. This can be done using two ways: using gather matrices and matching degrees of freedom. This project makes use of the latter. We keep track of the degrees of freedom (d.o.f.) associated with the components and sum up the matching elements.

The displacement boundaries are enforced in the $\mathbf{u}$ vector and traction boundaries make up the $\mathbf{f}$ vector after numerical integration.

2.7. Solution

Once the system $\mathbf{Ku} = \mathbf{f}$ is obtained, we make use of a method to solve the system, called the elimination method.

Suppose,

$$\mathbf{Ku} = \mathbf{f} + \mathbf{r} \quad \text{(where } \mathbf{r} \text{ is the reaction vector)}$$

If $\bar{\mathbf{u}}_p$ (displacements on Dirichlet boundary) are enforced, then:

$$\mathbf{K}_f \mathbf{u}_f = \mathbf{f}_f + \mathbf{r}_f - \mathbf{K}_p \bar{\mathbf{u}}_p$$

where f represents free boundary dof's and p represents prescribed boundary dof's.
Thus,

$$\mathbf{u}_f = \mathbf{K}_f^{-1} \left(\mathbf{f}_f + \mathbf{r}_f - \mathbf{K}_p \bar{\mathbf{u}}_p \right)$$

This system is solved by MATLAB's linear solver to obtain the $\mathbf{u}_f$ vector. Also, the reaction forces are obtained by $\mathbf{r} = \mathbf{Ku} - \mathbf{f}$. Finally,

$$\therefore \mathbf{u}^h = \mathbf{N}\{\mathbf{u}\}$$

$$\boldsymbol{\varepsilon} = \mathbf{B}\{\mathbf{u}\} \quad \text{at integration points}$$

$$\boldsymbol{\sigma} = \mathbb{C}\,\boldsymbol{\varepsilon} \quad \text{at integration points}$$

3. The code

The Finite Element Analysis code of the project is modularly organized with the main script `main.m` as the head. This script lays down the complete procedure, which consists of:

- Opening the MATLAB environment.
- Reading the problem statement from a text-based input file using the given `inpFileReader.p` function, loading a structured variable named `Model` with all data that are needed (geometry, materials, loads, boundary conditions).
- Printing a summary of the input model using `printSummary.p`.
- Plotting the mesh, boundary conditions, and the loads using `plotMeshX.p`.
- Calling the principal analysis function `fem2D.m` with the `Model` structure.
- Retrieving results of the analysis within a `Results` struct from `fem2D.m`.
- Presenting the most important results (responses, displacements, stresses, strains) using a nicely formatted display with the help of `printOutput.m`.

The main computation routines exist within the `lib` directory, implemented by filling the provided function skeletons in the order determined by function dependencies according to the project requirement. The main development stages were as follows:

Application of Infinitesimal Theory of Displacement-Strain:

- **strainStressMatrix.m**
  Coded appropriate constitutive matrix $\mathbb{C}$ based on material constants (E, ν) and the type of analysis (plane stress or plane strain).

- **shapeFunctions.m**
  Formulated mathematical expressions of the shape functions N and its derivatives $\frac{dN}{d\xi}$ for the necessary 2D elements (quadrilaterals and triangles).

- **integrationPoints.m**
  Used to provide the coordinates (ξ) and the weight values (w) of Gauss-Legendre points for numerical integration within elements.

Development of Element-Level Computations:

- **elemStiff.m**
  Included logic to determine the element stiffness matrix $[k]$ by using numerical integration:

$$\mathbf{K}_e = t \int B^T \mathbb{C} B \det(J)\, dA$$

  including:

  - Looping through Gauss points (from `integrationPoints.m`).
  - Calculating the Jacobian J and the strain-displacement matrix B (from `shapeFunctions.m`).
  - Obtaining the material matrix D (from `strainStressMatrix.m`).
  - Summation of weighted contributions at each integration point.

- **elemEquivalentLoad.m**
  Used to determine the equivalent nodal force vector $\mathbf{f}_e$ with respect to traction loads $\mathbf{t}$ on an element edge by numerical integration:

$$\mathbf{f}_e = t \int [N]^T \mathbf{t} \det(J)\, d\xi$$

  by using shape functions and appropriate Gauss points.

Global System Assembly and Solution:

- **fem2D.m (Assembly Logic)**

  - Assembled the global stiffness matrix $\mathbf{K}$ by iterating over all elements, calling `elemStiff.m` for each one, and inserting the resulting $\mathbf{K}_e$ into the global matrix according to element connectivity.
  - Built the global force vector $\mathbf{f}$ by gathering contributions from:
    * Traction forces (calculated by `elemEquivalentLoad.m`).
    * Directly applied concentrated nodal forces.

- **solution.m**
  Used to apply boundary conditions using the elimination method:

  - Simplified the system $\mathbf{Ku} = \mathbf{f}$ under specified displacement conditions.
  - Solved for unknown displacements using MATLAB's efficient backslash operator (\).
  - Calculated the reaction forces by:

$$\mathbf{r} = \mathbf{Ku} - \mathbf{f}$$

Post-Processing and Output:

- **fem2D.m (Stress/Strain Calculation)**

  - This section follows the computation of displacements with loops over each element and its integration points for the calculation of:

  - Strains: $\boldsymbol{\varepsilon} = \mathbf{B}\{\mathbf{u}\}$ at integration points
  - Stresses: $\boldsymbol{\sigma} = \mathbb{C}\,\boldsymbol{\varepsilon}$ at integration points

- **printOutput.m**
  Incorporate the `Results` structure with the output printed neatly in the command window including displacements, reaction forces, strains, and stresses.

4. Some Improvements

Although this code is aimed at developing a basic FEM code for 2D problems, it has a few shortcomings that could be fixed easily. Since the improved code involved some changes to the flow of the program, it was not uploaded to Canvas, instead it is publicly available here: `https://github.com/aaroshdahal/CSE-6504-FEM-Project`.

Mesh Complexity: Manually typing up nodes and elements for meshes with many elements is virtually impossible. To work around this issue, a simple script was written which converts .msh output from GMSH (a mesh generator) to the desired .txt format.

Boundaries: Again, applying boundaries to individual element edges or nodes becomes unscalable as the number of elements becomes large. To get around this issue, boundary conditions were coded in `fem2d.m` file such that one can define the coordinates criteria where the boundaries are to be applied. The code then searches all the nodes/edges that satisfy the criteria and applies those boundaries. Since the input text file needs boundary to be accepted by `inpFileReader.p`, dummy boundaries were typed up in the .txt file.

Body Force: The input file takes in nodal forces and traction, but there is no way to impose body force on the system through input file. Since the `inpFileReader.p` is an execute only file, body force had to be hard-coded into the `fem2d.m` file. The integration process to convert body force to nodal force is simple, and involves a process similar to that of traction.

Visualization: Printing output on MATLAB console is but a tedious way to visualize FEA outputs. MATLAB native plotter for image based output is better but nowhere as robust as some other visualization tools. As such, after FEA is completed, the mesh, displacement field, strains and stresses were exported to a .vtk file that can be read by Paraview, a finite element results visualizer.

5. Results

Some of the results from thus developed FEM code are presented in this section.
Fig. 1 and Fig. 2 show the console output on MATLAB for different input files. Fig. 1 is for input1.txt and Fig. 2 is for stress concentration problem shown in Fig. 5. As discussed in section 4, complicated models were analyzed using improved code. Fig. 3 and Fig. 4 show results from one such example where a plane strain tofu block (100 cm x 100 cm) is being squished under its own weight (body force).

Similarly, the famous stress concentration around a circular hole problem as shown in Fig. 5 was modeled. It is known from analytical solution that σ_{22} at the edge of circular hole is given by the relation:

$$\frac{\sigma_{yy}(x, y=0)}{\sigma^{\infty}} = \frac{\sigma_{\theta\theta}(\theta=0)}{\sigma^{\infty}} = \frac{1}{2}\left(2 + \frac{1}{r^2} + \frac{3}{r^4}\right) = 3, \text{ if } r = 1$$

As shown in Fig. 7, we obtained $\dfrac{\sigma_{yy}(x, y=0)}{\sigma^{\infty}}$ ratio of 2.92 which is 2.7% off from it's analytical value of 3.

```
================================================================
                    O U T P U T   S U M M A R Y
================================================================

                        Nodal Displacements
----------------------------------------------------------------
[ 1]         0.00000          0.00000
[ 2]         0.09615         -0.00000
[ 3]         0.09615         -0.02885
[ 4]         0.00000         -0.02885

                        Reaction Forces
----------------------------------------------------------------
[ 1]        -1.25000
[ 2]        -0.00000
[ 3]        -1.25000

                  Strains and Stresses at Gauss Points
----------------------------------------------------------------
                        Element Number = 1
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  0.3333,   0.3333)    0.0962   -0.0288   -0.0000    2.5000    0.0000   -0.0000
                        Element Number = 2
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  0.6667,   0.6667)    0.0962   -0.0288   -0.0000    2.5000    0.0000   -0.0000
================================================================================================================
>>
```

Figure 1: Console Output for input1.txt

```
                        Element Number = 13422
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  0.3110,   1.4708)  -100.9501  149.2090  -247.0675  -10460565.4551  29949737.0501  -19955453.3697
                        Element Number = 13423
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  1.9355,   1.1359)  -260.5002  534.1072   38.1257  -3278407.0787  121650478.3510  3079703.1598
                        Element Number = 13424
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  3.9163,   2.1708)  -202.4804  462.5907   26.9478  -1196288.9049  106240859.4667  2178165.7040
                        Element Number = 13425
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  1.4905,   1.4724)  -283.0225  532.8080  -130.6436  -19656537.6092  116331463.5911  -11190133.4602
                        Element Number = 13426
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  1.9762,   1.2371)  -244.5796  529.4766   23.7952  -4992633.1519  120047204.1660  1921923.4504
                        Element Number = 13427
----------------------------------------------------------------
Integration     Location          Location                 Strains                  Stresses
  Point         (Natural)         (Global)          e11       e22       e12       s11       s22       s12
----------------------------------------------------------------
  1         (  0.3333,   0.3333)  (  0.3461,   1.4903)  -106.5915  162.2090  -266.9538  -10593425.1170  32892815.2936  -21561650.9122
================================================================
Wrote VTK file: stress_conc.vtk  (open in ParaView)
```

Figure 2: Console Output for Stress Concentration Model

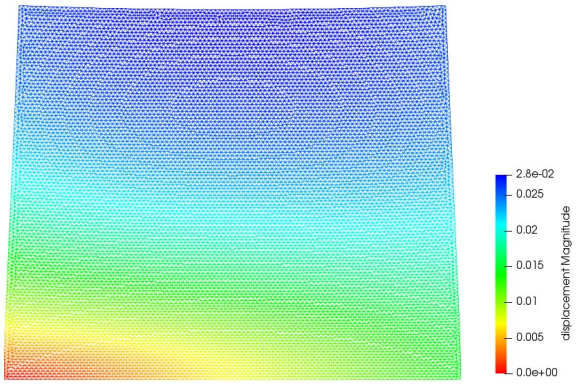

Figure 3: Mesh Plot of Deformed Model Showing Displacement

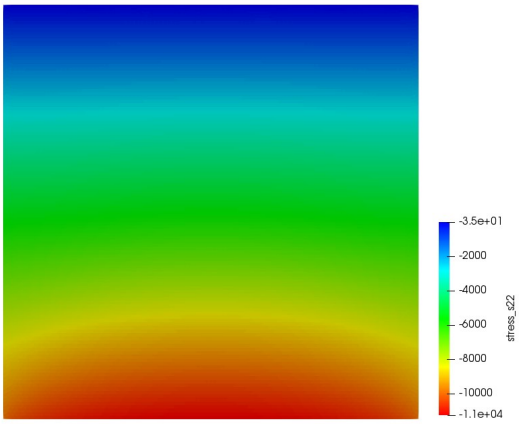

Figure 4: Surface Plot of Undeformed Model Showing σ_{22}

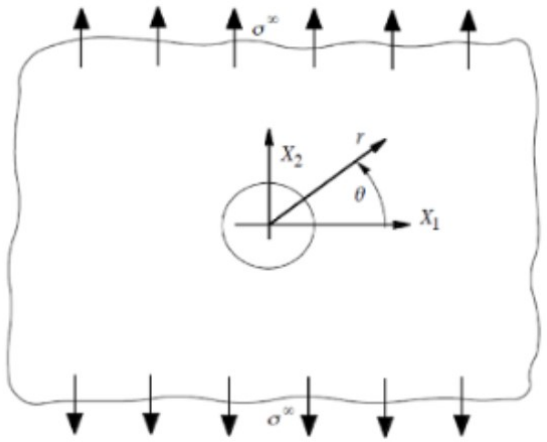

Figure 5: Problem Being Modeled

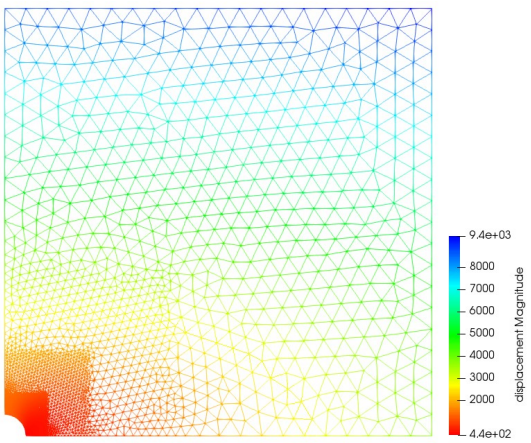

Figure 6: Mesh Plot Showing Displacement

Acknowledgements

The author A. Dahal would like to acknowledge the contribution of Dr Rafi Muhanna for his immense support, and instilling in us students the concepts of FEM.

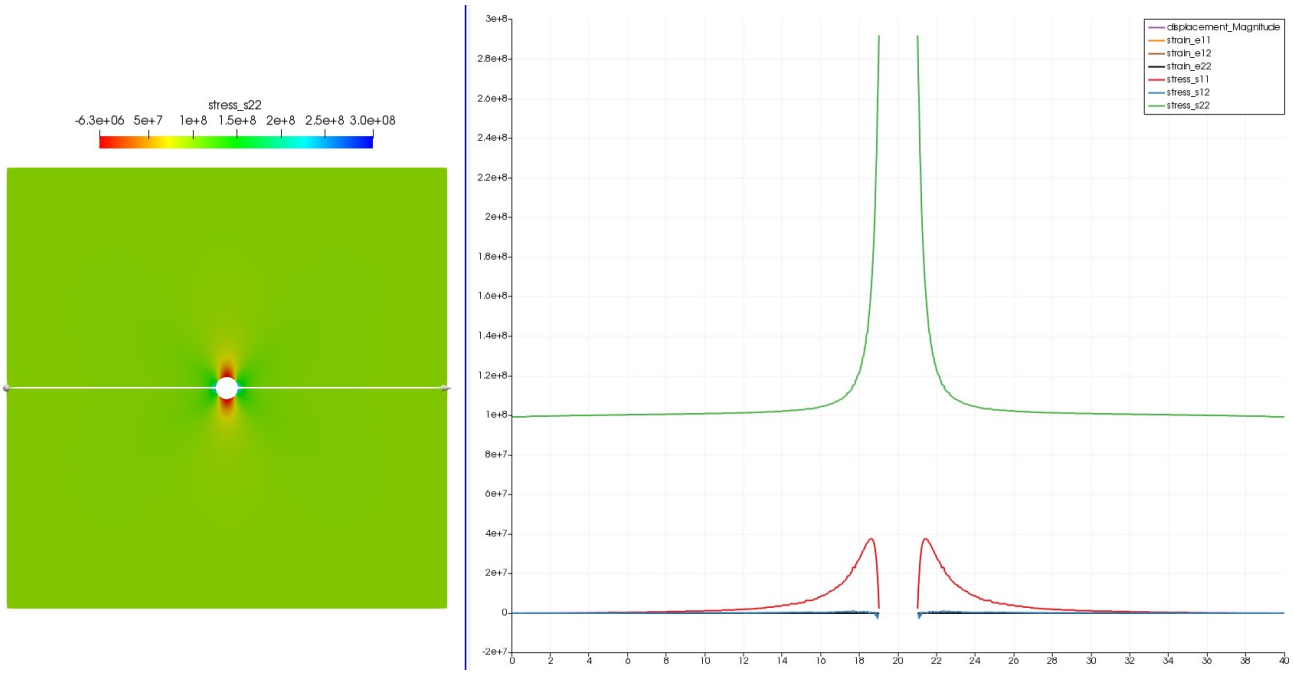

Figure 7: Plot Over Line

References

[1] R. Courant, et al., Variational methods for the solution of problems of equilibrium and vibrations, Lecture notes in pure and applied mathematics (1994) 1–1.