Computational Materials Science

Faculty of Mechanical, Process and Energy Engineering

Fakultät IV

Leipziger Str. 30, EG

09599 Freiberg

Personal Programming Project (2022-23)

Title : Programming an XFEM routine to investigate the effect of hole and inclusion in a plate(2D)

Nandha Gopal Mariappan

Matriculation Number: 66994

19.05.2023

Supervised by

Dr.-Ing. Stefan Prüger

Incharge for Personal Programming Project :

Dr. Arun Prakash and Dr.-Ing. Stefan Prüger

Contents

1. Introduction

1.1 Abstract

The main abstract of the program is to build an XFEM routine to investigate the effect of holes and inclusions in a plate(2D) plate.XFEM has been widely used to detect weak discontinuities and the results are coming better than the Finite Element Method.

1.2 Extended Finite Element Method

In the last decade, the extended Finite Element Method (X-FEM) has gained popularity for its ability to replicate the displacement field discontinuities throughout the crack surface and singularity at the crack front without re-meshing. The X-FEM accurately approximates fields having jumps, kinks, singularities, and other non-smooth elements.Belytschko and Black (1999) pioneered X-FEM crack propagation ,based on the idea of the partition of unity approach (Melenk and Babuška 1996).
(Sukumar, Chopp, Moës, & Belytschko, 2001) employed the Level Set Method for modeling holes and inclusions where the level set function was used to represent the local enrichment for material interfaces.The XFEM approximation is made up of standard finite elements that are used in the majority of the domain and enriched elements that are used in the enriched sub-domain to capture specific solution features like discontinuities and singularities. The purpose of using enriched elements in the standard XFEM is to enlarge the standard Finite Element Method (FEM) approximation function space so that the enriched approximation can cover or come close to the precise solution.To calculate strong and weak discontinuities, different approximation functions are used. The level set function will be used to calculate weak discontinuities such as voids and inclusions, while the Heaviside enrichment function will be used to calculate strong discontinuities such as cracks.As a result, the XFEM is an effective tool for simulating discontinuous problems.

1.3 Comparison between XFEM and FEM

The XFEM follows the same procedure as standard FEM, but when it comes to calculating discontinuities like voids, inclusions, or cracks, the former has better precision in the final solutions because additional enrichment functions are used for elements that are cut by the interface.The finite element method uses ”shape functions” to create an approximation space so that a vector can represent the answer. In the standard FEM, these shape functions are polynomials; however, in the XFEM, additional ”enrichment” functions, in addition to the polynomial shape functions, are employed to estimate the solution. These enrichment functions are chosen to have properties that are known to be followed by the solution.

1.4 Overview of the project

The primary goal of this personal programming project (PPP) is to write an XFEM method (in Python 3.8.10) to examine the effect of holes and inclusions in a 2D plate (Linear Isotropic material) by applying appropriate boundary conditions using the plane-strain condition. The level set method is utilised in this project to discover the enhanced elements of the plate that are cut by the interface among the standard elements that are not cut by the interface. Following the separation of the standard and enriched elements, the rectangular sub-grids method methodology was applied to the enriched element. The enriched element is subdivided into 3x3 rectangular sub-grids with four Gauss points at each sub-quadrilateral using this procedure. The global stiffness matrix will be assembled after calculating the local stiffness matrix for both enriched and standard elements. The set of linear systems of equations will be solved after providing uniform body forces for each element (considering gravity forces) and traction forces at the relevant edges with appropriate constraints. The final findings will be displacement and stress extraction at nodal points in the x and y directions, which will be visualised in a contour map.

2. Extended Finite Element Method (XFEM) Theory

2.1 Governing equations

2.1.1 Strong form

Consider a two-dimensional body (Ω) with a weak discontinuity (void or inclusion) having boundary which is partitioned into displacement (Γ_u), Traction (Γ_t) and Traction-free (Γ_c) boundaries. From the article (Sukumar et al., 2001),the equilibrium conditions are given as :

$$\nabla \cdot \boldsymbol{\sigma} + \mathbf{b} = 0 \quad \text{in } \Omega, \tag{2.1}$$

$$\boldsymbol{\sigma} = \mathbf{C} : \boldsymbol{\varepsilon}, \tag{2.2}$$

$$\boldsymbol{\varepsilon} = \boldsymbol{\nabla}_s u \tag{2.3}$$

where ∇_u is the symmetric gradient operator, $\mathbf{C}$ is the tensor of elastic moduli for a homegenous isotropic material and $\mathbf{b}$ is the body force vector From the article (Sukumar et al., 2001), the essential and boundary conditions are :

$$\mathbf{u} = \bar{\mathbf{u}} \quad \text{on } \Gamma_u, \tag{2.4}$$

$$\boldsymbol{\sigma} \cdot \mathbf{n} = \bar{\mathbf{t}} \quad \text{on } \Gamma_t, \tag{2.5}$$

$$\boldsymbol{\sigma} \cdot \mathbf{n_h} = 0 \quad \text{on } \Gamma_h \tag{2.6}$$

where $\mathbf{n}$ is the unit outward to $\boldsymbol{\Omega}$, $\bar{\mathbf{u}}$ and $\bar{\mathbf{t}}$ are the prescribed displacements and tractions in respective manner and $\mathrm{n_h}$ is the unit normal vector to the hole

2.1.2 Weak form

From the article (Sukumar et al., 2001),a weak form of the equilibrium equation (2.1) with associated boundary conditions can be written as

$$\int_{\Omega} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v})\, \mathbf{d}\Omega = \int_{\Omega} \mathbf{b} \cdot \mathbf{v}\, \mathbf{d}\Omega + \int_{\Gamma_t} \bar{\mathbf{t}} \cdot \mathbf{v}\, \mathbf{d}\Gamma \tag{2.7}$$

The following discrete equations(2.8) are obtained by substituting the trial and test function in equation (2.7) and utilising the arbitrariness of the nodal variations.

$$\mathbf{K}\,\mathbf{d} = \mathbf{f} \tag{2.8}$$

where $\mathbf{K}$ is the Global stiffness Tensor, $\mathbf{d}$ is the vector of unknown nodal displacements and $\mathbf{f}$ is the external force vector.

2.2 Shape functions

The shape functions used for this project are 2x2 Gauss quadrature, which is taken from the book. (Fish & Belytschko, 2007)

$$\mathbf{N_1}(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 - \eta) \tag{2.9}$$

$$\mathbf{N_2}(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 - \eta) \tag{2.10}$$

$$\mathbf{N_3}(\xi, \eta) = \frac{1}{4}(1 + \xi)(1 + \eta) \tag{2.11}$$

$$\mathbf{N_4}(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 + \eta) \tag{2.12}$$

2.3 Gauss Integration

In this project, 2 Gauss quadrature points are chosen in η and ξ direction respectively which is shown as diagram 2.1.In the Gaussian quadrature algorithm, the integration point locations and weights are determined so that a polynomial of the highest degree possible can be precisely integrated.

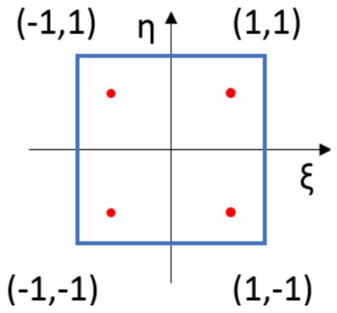


Figure 2.1: Unit Domain having 2x2 Gauss quadrature points. Image taken from (MinhNguyen, 2021)

In this project, 2 Gauss quadrature points are chosen in η and ξ direction respectively which is shown as diagram (2.1).In the Gaussian quadrature algorithm, the integration point locations and weights are determined so that a polynomial of the highest degree possible can be precisely integrated.

Consider a typical integral defined over the domain of a quadrilateral element which is taken from the book (Fish & Belytschko, 2007):

$$\mathbf{I} = \int_{\Omega^e} \mathrm{f}(\xi, \eta)\, d\Omega \tag{2.13}$$

By evaluating the infinitesimal area $d\Omega$ in terms of $d\eta$ and $d\xi$,we get

$$\mathbf{I} = \int_{\eta=-1}^{1} \int_{\xi=-1}^{1} |\mathbf{J}^e(\xi, \eta)| \mathrm{f}(\xi, \eta)\, d\xi\, d\eta \tag{2.14}$$

By carrying out Gauss Integration (2 Gauss points) over ξ and η,which yields

$$\mathbf{I} = \sum_{i=1}^{2} \sum_{j=1}^{2} \mathrm{W}_i \mathrm{W}_j |\mathbf{J}^e(\xi, \eta)| \mathrm{f}(\xi, \eta)\, d\xi\, d\eta \tag{2.15}$$

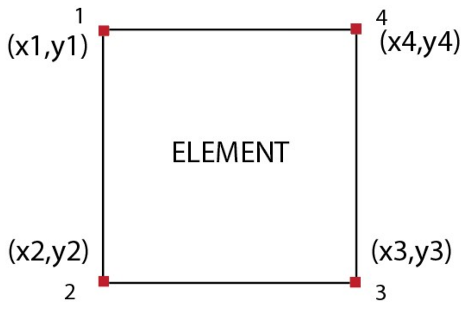
By substituting the coordinates and weights in the shape functions,which is taken from the table 2.1, we can able to evaluate the Stiffness matrix for both Standard and enriched elements.

Points	ξ	η	Weights($W_i\, and\, W_j$)
1	$-\frac{1}{\sqrt{3}}$	$-\frac{1}{\sqrt{3}}$	1
2	$-\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$	1
3	$\frac{1}{\sqrt{3}}$	$-\frac{1}{\sqrt{3}}$	1
4	$\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$	1

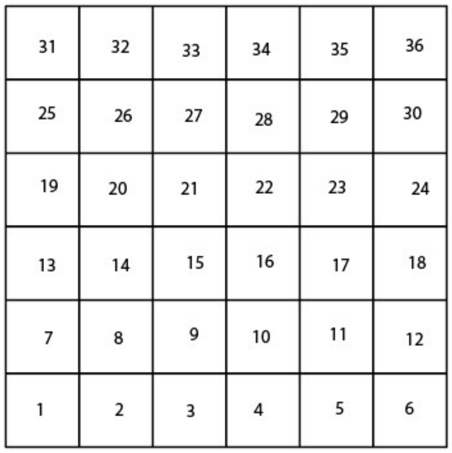
Table 2.1: Coordinates and weights of 2x2 Gauss quadrature taken from (Fish & Belytschko, 2007)

2.4 Node numbering and Element Numbering

In this project,Node numbering and element numbering in a plate will be carried out according to these figures2.2a and 2.2brespectively.



(a) Node Numbering in an element



(b) Element numbering in a plate

2.5 Level Set Method

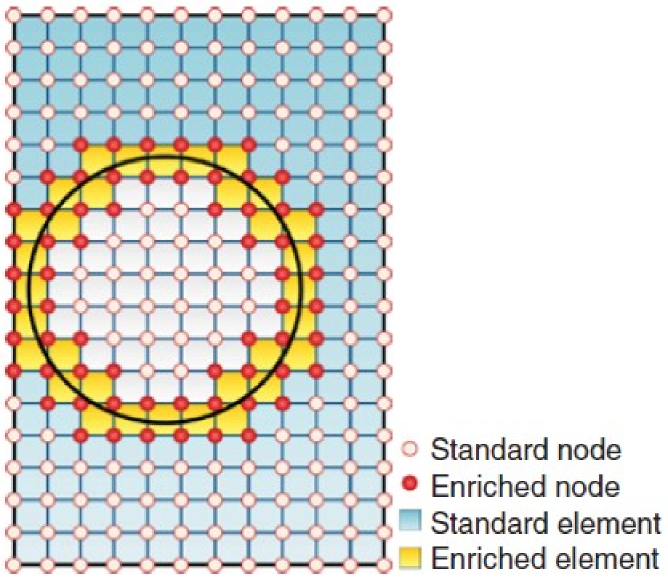


Figure 2.3: Plate highlighted with standard elements and enriched elements.Image taken from (Khoei, 2014)

The level set method will be used in this project to distinguish between items that are cut by an Interface (Holes/Inclusions) and those that are not. If the element is cut by an interface, it is regarded an enhanced element, however if the element is not cut by an interface, it is considered a standard element. Standard and enhanced nodes can be identified at the same time by locating standard and enriched elements.

$$\boldsymbol{\phi} = \sqrt{(\mathbf{x} - \mathbf{x_1}) + (\mathbf{y} - \mathbf{y_1})} - \mathbf{r} \tag{2.16}$$

where $\mathbf{x}, \mathbf{y}$ represent the coordinate of the node,$\mathbf{x_1}\, and \mathbf{y_1}$ represent the coordinate of the center of the circle and r represents the radius of the interface (voids or inclusions). We can use this equation to determine if a node is inside the interface, outside the interface, or on the interface.After replacing the values in the preceding equation (2.16), if $\boldsymbol{\phi}$ is larger than 0, the coordinate is outside the discontinuity (voids or inclusions).If $\boldsymbol{\phi} = 0$, the coordinate lies on the discontinuity (voids or inclusions) ; otherwise, the coordinate is inside the discontinuities (voids or inclusions) whose ϕ value will be less than one.

2.6 The Rectangular sub-grids method

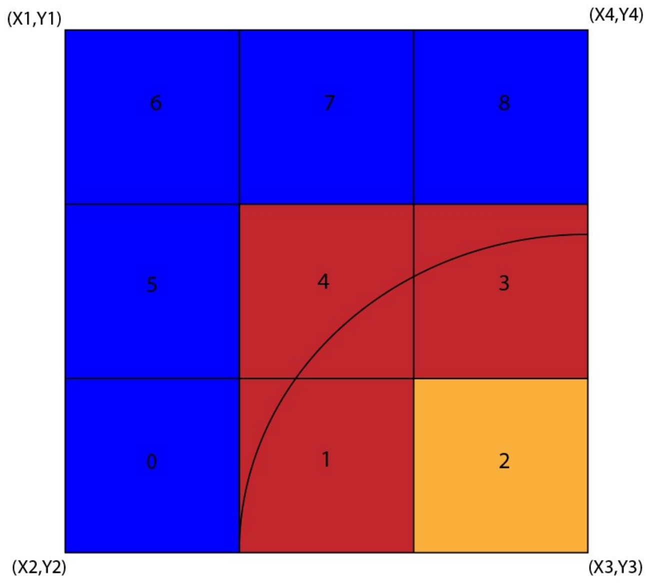


Figure 2.4: Enriched element with 9 sub-grids.Sub-grids with number (0,5,6,7,8) = outside the interface. Sub-grids with number (1,3,4) = Interface passing through the sub-grid, Sub-grid with number (2) = Inside the Interface

Because of the presence of a weak or strong discontinuity within the element, the enrichment functions in the X-FEM may not be smooth over an enriched element. As a result, if the element is crossed by a discontinuity, the usual Gauss quadrature procedure cannot be utilised, and the element quadrature points must be modified to appropriately evaluate the contribution to the weak form on both sides of the interface.

To get the smooth enrichment functions in the XFEM,the rectangular sub-grids method is used in this project.The enriched element is further divided into 3x3 sub-grids (9 sub-grids) so that the enrichment function will be smooth.Again by using the Level set method,the ϕ values will be calculated for four nodal points in each sub-grid.

2.7 Jacobi matrix

The Jacobian matrix is an integral part of the formulation of isoparametric element formulation. It is often expressed as [J] and represents a "scaling" factor between the natural and local coordinate systems' derivatives.From the book (Khoei, 2014),the Jacobian matrix **J** for the current configurations is defined as

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \sum_{I=1}^{4} \dfrac{\partial \mathbf{N_I}}{\partial \xi} \mathrm{x}_I & \sum_{I=1}^{4} \dfrac{\partial \mathbf{N_I}}{\partial \xi} \mathrm{y}_I \\ \\ \sum_{I=1}^{4} \dfrac{\partial \mathbf{N_I}}{\partial \eta} \mathrm{x}_I & \sum_{I=1}^{4} \dfrac{\partial \mathbf{N_I}}{\partial \eta} \mathrm{y}_I \end{bmatrix} \tag{2.17}$$

in which the Jacobian matrix between the parent and current elements is defined.

2.8 B-matrix construction

From the book (Khoei, 2014),the standard FE approximation of displacement can be obtained as $\mathbf{u} = \mathbf{N}^{std}\bar{u}$ where

$$\mathbf{N}^{std}_{2\times 8} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \tag{2.18}$$

$$\bar{\mathbf{u}}^T_{8\times 1} = \begin{bmatrix} u_{x_1} & u_{y_1} & u_{x_2} & u_{y_2} & u_{x_3} & u_{y_3} & u_{x_4} & u_{y_4} \end{bmatrix} \tag{2.19}$$

The enhanced FE approximation of displacement field, like the ordinary FE approximation, can be generated based on standard and enriched parts using $\mathbf{u} = \mathbf{N}^{std}\bar{\mathbf{U}}$ where $\mathbf{N^{enh}} = [\mathbf{N^{std}}, \mathbf{N^{enr}}]$ and the vector of unknown nodal displacements $\mathbf{\bar{U}^T} = [\mathbf{\bar{u}^T}, \mathbf{\bar{a}^T}]$

are defined as ((Khoei, 2014))

$$\mathbf{N}^{enh}_{2\times16} = \begin{bmatrix} N_1 & 0 & N_1\bar{\psi}_1 & 0 & \cdots & N_4 & 0 & N_4\bar{\psi}_4 & 0 \\ 0 & N_1 & 0 & N_1\bar{\psi}_1 & \cdots & 0 & N_4 & 0 & N_4\bar{\psi}_4 \end{bmatrix} \tag{2.20}$$

$$\bar{\mathbf{U}}^{\mathbf{T}}_{16\times1} = \begin{bmatrix} u_{x_1} & u_{y_1} & u_{a_1} & u_{b_1} & \cdots & u_{x_4} & u_{y_4} & u_{a_4} & u_{b_4} \end{bmatrix} \tag{2.21}$$

As a result, the strain vector corresponding to the standard displacement field approximation can be calculated as $\boldsymbol{\varepsilon} = \mathbf{B}^{std}\,\bar{\mathbf{u}}$ in which $\mathbf{B}^{std}$ is the standard discretized gradient operator defined as ((Khoei, 2014))

$$\mathbf{B}^{std}_{3\times8} = \begin{bmatrix} N_{1,x} & 0 & N_{2,x} & 0 & N_{3,x} & 0 & N_{4,x} & 0 \\ 0 & N_{1,y} & 0 & N_{2,y} & 0 & N_{3,y} & 0 & N_{4,y} \\ N_{1,y} & N_{1,x} & N_{2,y} & N_{2,x} & N_{3,y} & N_{3,x} & N_{4,y} & N_{4,x} \end{bmatrix} \tag{2.22}$$

Similar to the standard strain element, the strain vector corresponding to the standard and enriched parts of the enhanced displacement field can be calculated as $\boldsymbol{\varepsilon} = \mathbf{B}^{enh}\,\bar{\mathbf{U}}$ in which $\mathbf{B}^{enh}$ is the enhanced discretized gradient operator $\mathbf{B}^{enh} = [\mathbf{B}^{std}, \mathbf{B}^{enr}]$ defined as ((Khoei, 2014))

$$\mathbf{B}^{enh}_{3\times16} = \begin{bmatrix} N_{1,x} & 0 & (N_1\bar{\psi}_1)_{,x} & 0 & \cdots & N_{4,x} & 0 & (N_{4,x}\bar{\psi}_4)_{,x} & 0 \\ 0 & N_{1,y} & 0 & (N_1\bar{\psi}_1)_{,y} & \cdots & 0 & N_{4,y} & 0 & (N_4\bar{\psi}_4)_{,y} \\ N_{1,y} & N_{1,x} & (N_1\bar{\psi}_1)_{,y} & (N_1\bar{\psi}_1)_{,x} & \cdots & N_{4,y} & N_{4,x} & (N_4\bar{\psi}_4)_{,y} & (N_4\bar{\psi}_4)_{,x} \end{bmatrix} \tag{2.23}$$

Note : The above equation (2.23) has been modified from the book(Khoei, 2014) for the ease of convenience at assembly of K-matrix which is implemented in the project successfully.

2.9 Stress/Strain or Constitutive [D] matrix

From the book(Fish & Belytschko, 2007),the constitutive matrix [D] (2.24) will be used in which $\sigma_Z = \tau_{xz} = \tau_{yz}$ will be zero for plane stress condition.

$$\mathbf{D}_{3\times 3} = \frac{\mathrm{E}}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{2.24}$$

where E is the modulus of elasticity and ν is Poisson's ratio.

For plane strain condition,the constitutive matrix [D] (2.25) will be used in which $\varepsilon_Z = \gamma_{xz} = \gamma_{yz}$ will be zero.((Fish & Belytschko, 2007))

$$\mathbf{D}_{3\times 3} = \frac{\mathrm{E}}{(1+\nu)(1-2\,\nu)}\begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\,\nu}{2} \end{bmatrix} \tag{2.25}$$

where E is the modulus of elasticity and ν is Poisson's ratio.

2.10 Stiffness matrix

The general equation form of stiffness matrix is given in equation(2.26) which is taken from the (Khoei, 2014).

$$\boldsymbol{K} = \int_{-1}^{1}\int_{-1}^{1} \boldsymbol{B}^{e^T}\,\boldsymbol{D}^e\,\boldsymbol{B}^e|\boldsymbol{J^e}|\,d\boldsymbol{\xi}\,d\boldsymbol{\eta} \tag{2.26}$$

By applying Gauss integration in equation (2.27),it has been transformed to equation (2.27), which will be applied for the standard element having 2 degrees of freedom at each node.((Khoei, 2014)

$$\boldsymbol{K}^{std}_{8\times8} = \sum_{i=1}^{2}\sum_{j=2}^{2}\boldsymbol{W}_i\,\boldsymbol{W}_j\,\boldsymbol{B}^{e^T}(\xi_i,\eta_j)\,\boldsymbol{D}^e\,\boldsymbol{B}^e(\xi_i,\eta_j)\,|\boldsymbol{J^e}(\xi_i,\eta_j)| \tag{2.27}$$

Below is the stiffness matrix equation (2.28) for enriched elements in which the enriched element will have 4 degrees of freedom at each node.

$$\boldsymbol{K}^{enh}_{16\times16} = \sum_{k=1}^{9}\sum_{i=1}^{2}\sum_{j=2}^{2}\boldsymbol{W}_i\,\boldsymbol{W}_j\,\boldsymbol{B}^{e^T}(\xi_i,\eta_j)\,\boldsymbol{D}^e\,\boldsymbol{B}^e(\xi_i,\eta_j)\,|\boldsymbol{J^{sub}}(\xi_i,\eta_j)|\,|\boldsymbol{J^e}(\xi_i,\eta_j)| \tag{2.28}$$

2.11 Body force and Traction force

The body force vector for an element will be calculated by using the below equation.2.29 (Fish & Belytschko, 2007)

$$\mathbf{f_b}_{8\times1} = \int_{-1}^{1}\int_{-1}^{1}[\mathbf{N}]^{\mathbf{T}}\,\mathbf{b}\,\mathbf{t}\,|\mathbf{J}|\,d\boldsymbol{\xi}\,d\boldsymbol{\eta} \tag{2.29}$$

where **N** represents the shape functions, **b** represents the body forces acting on an element in x,y-directions, **t** represents the thickness of the plate. After finding individual body force vector for all the elements,it will be assembled in the global body force vector.

The Traction force vector for a edge of the element will be calculated by using the below equation.2.30 (Fish & Belytschko, 2007)

$$\mathbf{f_S}_{4\times1} = \int_{-1}^{1}[\mathbf{N}]^{\mathbf{T}}\,\mathbf{T}\,\mathbf{t}\,|\mathbf{J}|\,d\boldsymbol{\xi} \tag{2.30}$$

where **N** represents the shape functions, **b** represents the Traction forces acting on the edge of the element in x,y-directions, **t** represents the thickness of the plate. After finding individual Traction forces for all the edges,it will be assembled in the global Traction force vector.

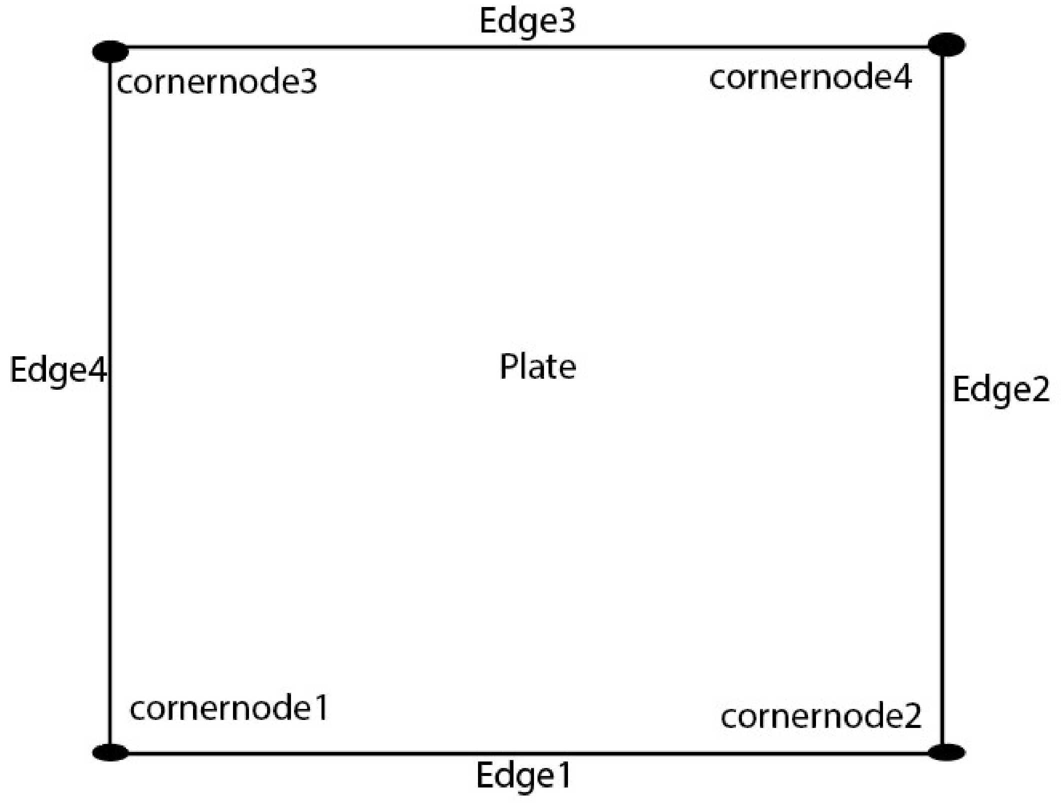


Figure 2.5: Edge numbering and corner numbering to apply traction force on the respective edges

From above figure 2.5,the respective edges for applying Traction forces can be known.

2.12 Solving Linear system of equations

Following the assembly of the local stiffness matrix into the global stiffness matrix **K**, the same approach is followed for body forces on elements and traction forces on the boundary, which are integrated into the global force vector **F**. The global stiffness matrix and the global force vector are now substituted in equation (2.8), and the Dirichlet boundary condition is applied to the equations.From above figure 2.5,the respective edges or the respective corner nodes for applying Dirichlet boundary conditions can be known.Thus, the solution to nodal displacements can be determined.

2.13 Extraction of results

Following the assembly of the local stiffness matrix into the global stiffness matrix **K**, the same approach is followed for body forces on elements and traction forces on the boundary, which are integrated into the global force vector **F**. The global stiffness matrix and the global force vector are now substituted in equation (2.8), and the Dirichlet boundary condition is applied to the equations. Thus, the solution to nodal displacements can be determined.

2.14 Flowchart of the project

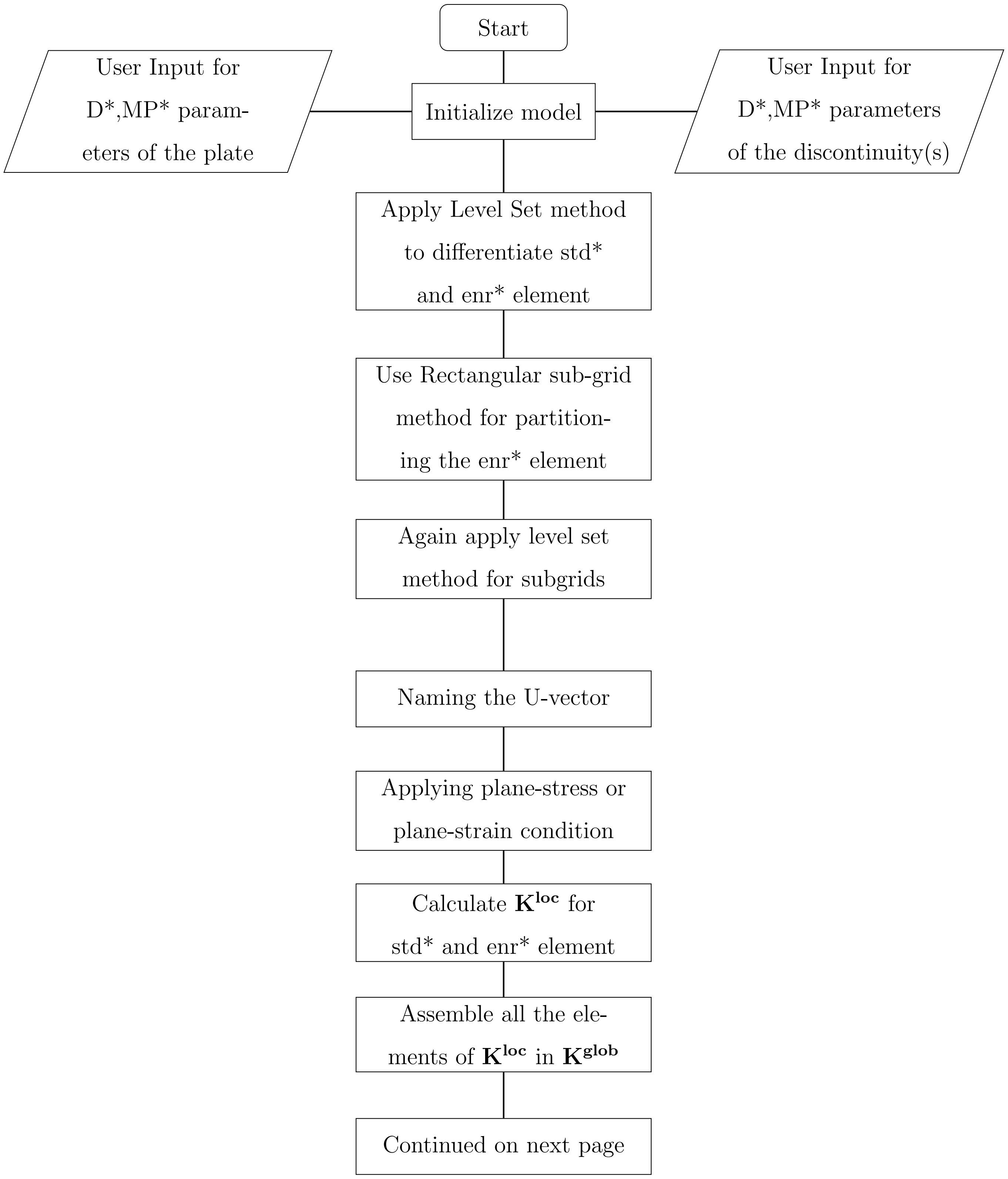

Start
User Input for D*,MP* parameters of the plate
Initialize model
User Input for D*,MP* parameters of the discontinuity(s)
Apply Level Set method to differentiate std* and enr* element
Use Rectangular sub-grid method for partitioning the enr* element
Again apply level set method for subgrids
Naming the U-vector
Applying plane-stress or plane-strain condition
Calculate $\mathbf{K}^{\mathbf{loc}}$ for std* and enr* element
Assemble all the elements of $\mathbf{K}^{\mathbf{loc}}$ in $\mathbf{K}^{\mathbf{glob}}$
Continued on next page

Figure 2.6: Flowchart for the XFEM routine to investigate the effect of holes and inclusions in a 2D plate (D*=Dimensions,MP*=Material parameters,std*=standard,enr*=enriched)

Figure 2.7: Continue...2.6

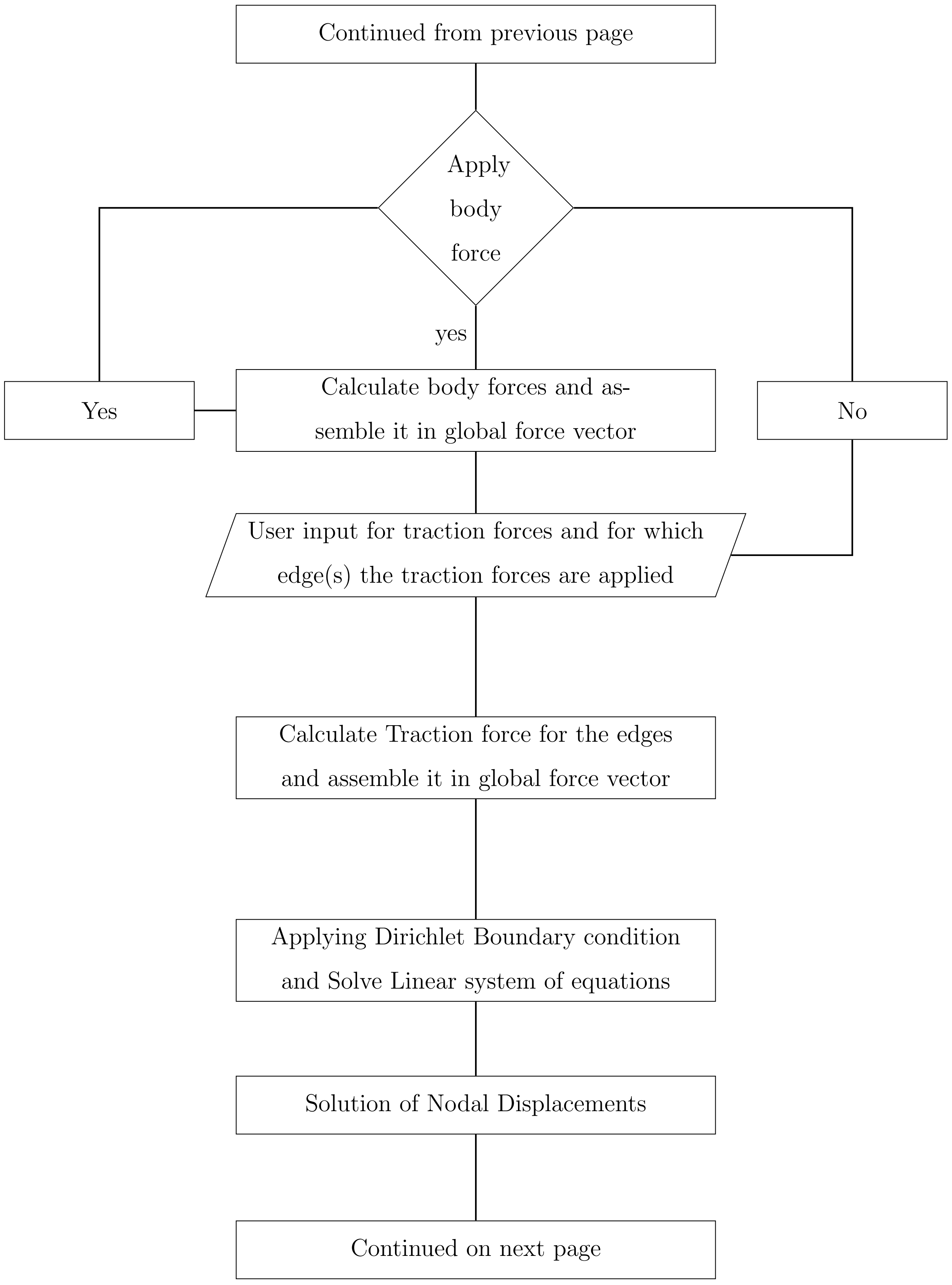
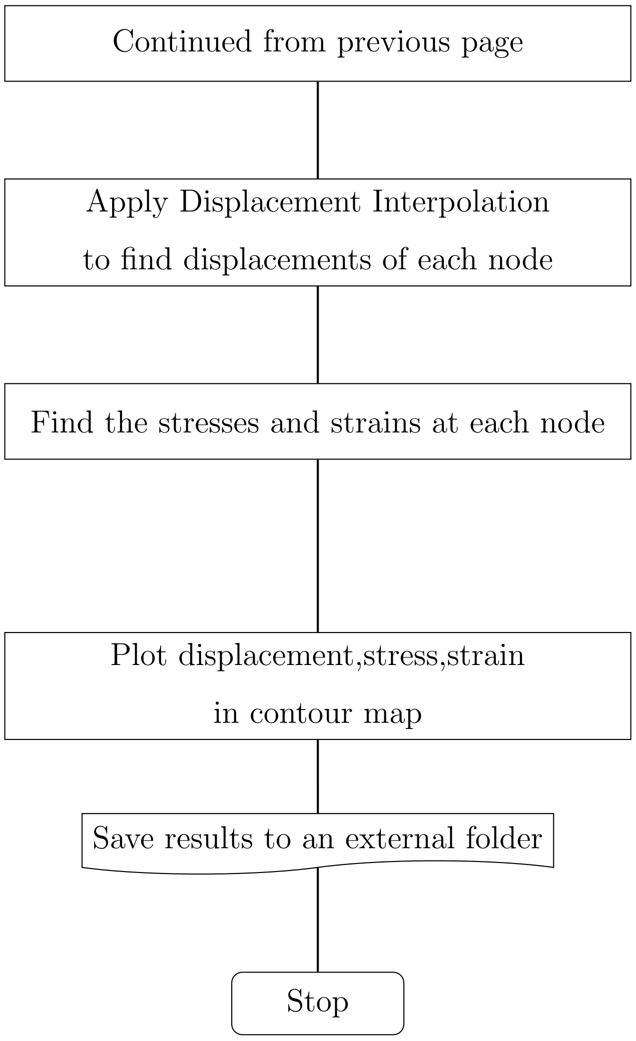

Continued from previous page
Apply body force
yes
Yes
Calculate body forces and assemble it in global force vector
No
User input for traction forces and for which edge(s) the traction forces are applied
Calculate Traction force for the edges and assemble it in global force vector
Applying Dirichlet Boundary condition and Solve Linear system of equations
Solution of Nodal Displacements
Continued on next page

Figure 2.8: Continue...2.7


Continued from previous page
Apply Displacement Interpolation
to find displacements of each node
Find the stresses and strains at each node
Plot displacement,stress,strain
in contour map
Save results to an external folder
Stop

3. Testing

3.1 Unit test

In this section,the different function files used to run the main program has been tested here.

3.1.1 test coordinates

This unit test case will test the generation of nodes for an element in a plate (2D).For that,mesh size and element number(in the form of i and j variable),need to be given in the test case.

Aim:-

To check the generation of nodal coordinates for an element.

Expected result:-

[1,1],[1,0],[2,0],[2,1] are the nodal coordinates which will be the expected result.

Obtained result:-

Same nodal coordinates as mentioned above in the expected result has been obtained.

Conclusion :

Test passed successfully.

Command used to run the program :

```
$ pytest -s test_unittest.py::test_coordinates
```

3.1.2 test levelsetmethod

This unit test case will determine whether or not the created elements are affected by the discontinuity. If the discontinuity goes through or cuts the element, it will be classified as an enriched element. Those that are not cut by a discontinuity will be classified as standard elements. Four random elements of a 10-by-10-meter plate were identified and placed within the test case. Input also included the discontinuity coordinates and radius.

Aim:-

To generate the phivalues for the given elements with respective nodal coordinates.

Expected result:-

[1,1,1,1],[1,1,1,-1],[-1,1,1,-1],[-1,-1,-1,-1] - These are the phivalues which will be coming as an output for the expected result.

Obtained result:-

Same phivalues as mentioned above in the expected result has been obtained.

Conclusion :

Test passed successfully.

Command used to run the program :

```
$ pytest -s test_unittest.py::test_levelsetmethod
```

3.1.3 test partitionelement

This unit test case will generate the nodal coordinates for 9 sub-grids for an enriched element using rectangular sub-grids method.For this,an enriched element with their nodal coordinates will be given as input.

Aim:-

To generate nodal coordinates for 9 sub-grids for an enriched element.

Expected result:-

Nodal coordinates of 9 sub-grids have been given.

Obtained result:-

Nodal coordinates of 9 sub-grids which has been stored in a variable have been exactly matched with the actual output.

Conclusion :

Test passed successfully.

Command used to run the program :

```
$ pytest -s test_unittest.py::test_partitionelement
```

3.1.4 test globnodesandnumber

This unit test case used two function files in which the first function generate the global nodes numbering for the respective nodal coordinates and the second function will arrange the global nodes number for the respective element.The output from these function will be later used in the Stiffness matrix assembly of the program.

Aim:-

To generate global node and global numbering for the elements of the given input which is a 3*3 metre plate with mesh size of 1.

Expected result:-

Global nodes and global numbering for an element have been given as an array in the respective variables.

Obtained result:-

Expected results have been matched with the actual outputs

Conclusion :

Test passed successfully.

Command used to run the program :

```
$ pytest -s test_unittest.py::test_globnodesandnumber
```

3.2 Sanity test

3.2.1 Sanity test for sum of shape functions

Aim:-

To check the sum of the shape functions with respect to their Gauss points should be one as per the theory of Finite element method.

Expected result:-

Expected result to be one for the sum of shape functions with their respective Gauss points.

Obtained result:-

The obtained results will have an output value of 1 that corresponds to the expected results.

Conclusion :

Test passed successfully.

Command used to run the program :

```
$ pytest -s test_sanitytest.py::test_shapefunctions
```

3.2.2 Sanity test for sum of derivative of shape functions

Aim:-

To check the sum of derivative of shape functions with respect to their Gauss points should be zero.

Expected result:-

Expected result to be zero for the sum of derivative of shape functions with their respective Gauss points.

Obtained result:-

The obtained results will have an output value of 0 that corresponds to the expected results.

Conclusion :

Test passed successfully. **Command used to run the program :**

```
$ pytest -s test_sanitytest.py::test_derivativeshapefun
```

3.2.3 Sanity check for volume conserving nature of Jacobi matrix

Aim:-

To check the volume conserving nature of the Jacobi matrix by multiplying with the rotation matrix. Q = $\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$

Expected result:-

The determinant of the Jacobi matrix will have the value 0.25 (for the given input in the test case) which will be equal before and after rotation of Jacobi matrix calculated for all 4 Gauss points.

Obtained result:-

Obtained result of this test section after rotation of Jacobi matrix for the given input θ is 0.25 which is the same with all four Gauss points.

Conclusion :

Test passed successfully. **Command used to run the program :**

```
$ pytest -s test_sanitytest.py::test_Jacobimatrix
```

3.2.4 Sanity check for stiffness matrix

Unconstrained rigid body

In this test section, Eigen values of the stiffness matrix will be calculated for a 2*2 metre plate with mesh size of 1.Here there will be no traction,body forces and Dirichlet boundary conditions are not applied.

Aim:-

To check the Eigen values of unconstrained rigid body

Expected result:-

Expected of three eigen values to have the value of zero among the other eigen values.

Obtained result:-

The output variable in the test section will have number as 3(mentioning the number of zero eigen values) which proves that the nature of stiffness matrix is conserved.

Constrained rigid body

In this same test section, Eigen values of the stiffness matrix will be calculated for a 2*2 plate with mesh size of 1 with Dirichlet boundary conditions applied on one of the edges with constrained on both directions.

Aim:-

To check the Eigen values of unconstrained rigid body

Expected result:-

Expected of all eigen values to have the values not lesser than or equal to zero (i.e) All the Eigen values should be non-negative.

Obtained result:-

The output variable in the test section will have number as 0(mentioning the number of zero eigen values) which proves that the nature of stiffness matrix is conserved.

Conclusion :

Test passed successfully.

```
$ pytest -s test_sanitytest.py::test_eigenvaluetest
```

3.2.5 Rigid body displacement Test

In this test section,elements will be genrated for a 2*2mm plate with mesh size of 1mm.Rigid body displaced with the displacement of 2mm by adding it to the nodal vectoru(both directions) except the center node.

Aim:-

To check the translation of the rigid body for the given displacement

Expected result:-

The center node of the plate should also the have nodal displacements as 2mm in both directions.

Obtained result:-

The obtained result will have the center node with displacement 2mm on both directions

Conclusion :

Test passed successfully.

```
$ pytest -s test_sanitytest.py::test_rigidbodydisplacementtest
```

3.2.6 Rigid body Rotation Test

In this test section,elements will be generated for a 2*2 mm plate with mesh size of 1mm. Rigid body rotated with the angle of **theta** = 135 in this test section.

Aim:-

To check the rotation of the rigid body for the given angle **theta**

Expected result:-

The center node [1,1] of the plate should have the nodal displacements as [-1.084456,-0907719] in both directions.

Obtained result:-

The obtained result will have the center node with displacement of [-1.084456,-0907719] which is matched with the Expected result.

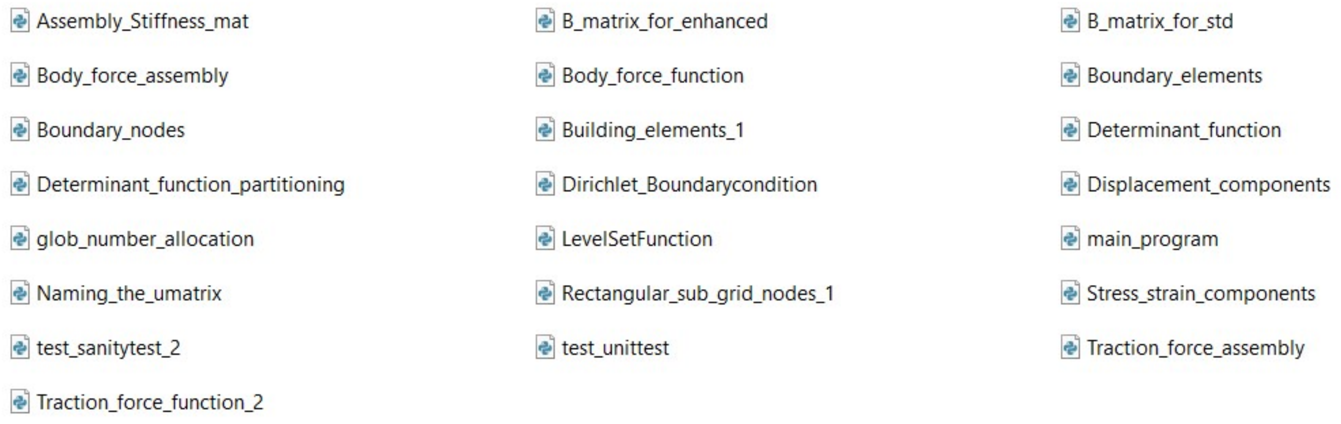
Conclusion :

Test passed successfully.

```
$ pytest -s test_sanitytest.py::test_rigidbodyrotationtest
```

4. User Manual

4.0.1 File contents and library used



Assembly_Stiffness_mat
B_matrix_for_enhanced
B_matrix_for_std
Body_force_assembly
Body_force_function
Boundary_elements
Boundary_nodes
Building_elements_1
Determinant_function
Determinant_function_partitioning
Dirichlet_Boundarycondition
Displacement_components
glob_number_allocation
LevelSetFunction
main_program
Naming_the_umatrix
Rectangular_sub_grid_nodes_1
Stress_strain_components
test_sanitytest_2
test_unittest
Traction_force_assembly
Traction_force_function_2

Figure 4.1: All python files in the folder

Before running the main program,the user should be aware of the files needed to run the main program.Above figure4.1 gives the total number of python files in the main folder.In order to run the main program(main$_{p}rogrampythonfile), 19files(savedasseparatefunctions)$

4.0.2 External Libraries used

The external libraries used here are numpy, sympy, matplotlib with their latest version.With the installation of the above packages in their respective system,the program can able to run without any problem.

4.0.3 Guidelines for executing the main program :

Command used to run the main program :

```
$ python3 ./main_program,py
```

1. After ensuring the file contents and library package,type the above respective command in the terminal.

2. Inputs for the model : The system will ask from the user has to give the dimensions,mesh size,material parameters of the plate.The system will ask about the coordinates,radius of the discontinuity(s).
3. Inputs for the model : The system will ask from the user about which conditions need to be applied to the plate(Plane-stress or Plane-strain)
4. Inputs for the model : The system will ask from the user about which conditions need to be applied to the plate(Plane-stress or Plane-strain)
5. After the system calculates and assembles the Global Stiffness matrix,the system will ask the user about the Traction forces needed to be applied on the plate.
6. Please type the Edge (for reference :(2.5) that need to be applied with Traction force.After that, enter the value in the x or y-direction or both directions.
7. After the Traction force,the system asks the user to apply Dirichlet boundary conditions on edges or corner nodes (2.5. After mentioning the edge or corner node ,kindly type the directions that need to be constrained.
8. After that the system solving the linear system of equations,it will go to the result section where the contour plot for displacement in x and y direction and stresses in x and y direction.
9. The results will be saved in the folder

4.0.4 Guidelines for executing the testing program :

1. The command used to run the unit test file which tests the functions implemented in the main program.Here is the command below :

   ```
   $ pytest -s test_unittest.py
   ```

2. The command used to run the sanity test file which tests the sanity of the implemented program to avoid violation of the theory.Here is the command below :

```
$ pytest -s test_sanitytest.py
```

5. Validation and results

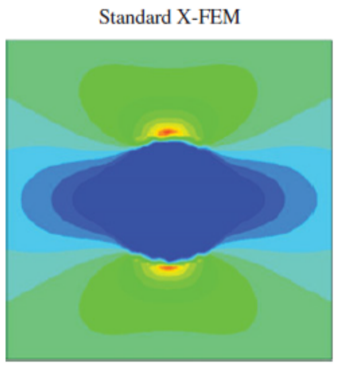
5.1 Validations:-

5.1.1 Validation 1

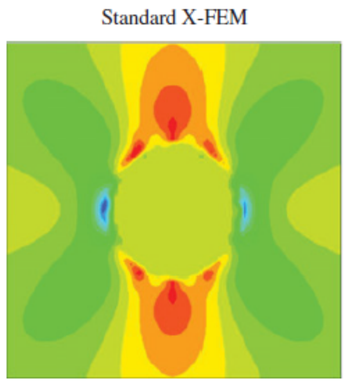
For validation of results from the main program,one result regarding stress contours in x-direction and y-direction from the book(Khoei, 2014) have been compared with the program's output.

Inputs for the validation(1) from the book :

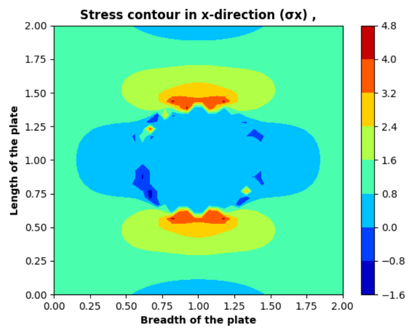
1. The author has taken a square of length 2cm.
2. Radius of the hole in the centre of the plate = 0.4cm
3. Uniaxial Tension along x and y-direction = 1 and -1 kg/cm^2
4. Young's modulus of plate and hole = 10^5 kg/cm^2 and 10^2 kg/cm^2
5. Poisson's ratio = 0.3 , Mesh size = 0.05cm
6. Dirichlet Boundary conditions are not mentioned by the author



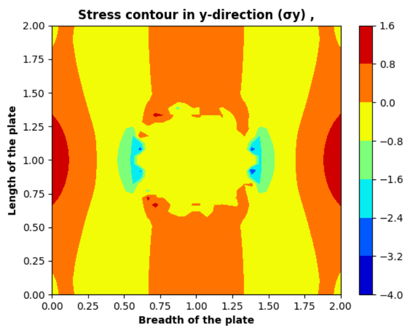
(a) Stress contour in x-direction



(b) Stress contour in y-direction



(a) Stress contour in x-direction



(b) Stress contour in y-direction

Results for the validation(1) from the project(main program) 5.4a 5.4b:

1. Plate of square of length 2cm.
2. Radius of the hole in the centre of the plate = 0.4cm
3. Uniaxial Tension along x and y-direction = 1 and -1 kg/cm^2
4. Young's modulus of plate and hole = 10^5 kg/cm^2 and 10^2 kg/cm^2
5. Poisson's ratio = 0.3 , Mesh size = 0.05cm
6. Dirichlet Boundary conditions are applied on the corner node 1(constrained in y-direction) and corner node 2(constrained in y-direction)

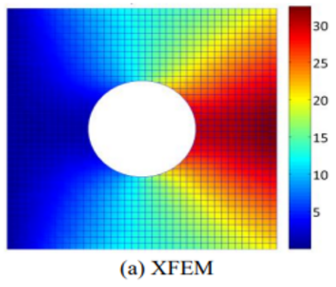
5.1.2 Validation 2

For validation of results from the main program,another result regarding displacement in x-direction and y-direction from the book(Jiang, Du, & Gu, 2014) have been compared with the program's output.

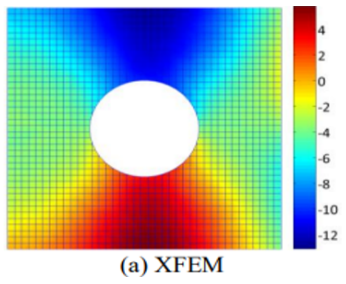
Inputs for the validation(2) from the journal :

1. The author has taken a square of length 2m.
2. Radius of the hole in the centre of the plate = 0.4m

3. Uniaxial Tension along x direction = 1000 Pa
4. Young's modulus of plate and hole = 10^5 kg/cm^2 and 10^2 kg/cm^2
5. Poisson's ratio = 0.3 , Mesh size = 39*39, Poisson ratio of hole = 0.3
6. Dirichlet Boundary conditions are not mentioned in the paper



(a) Stress contour in x-direction



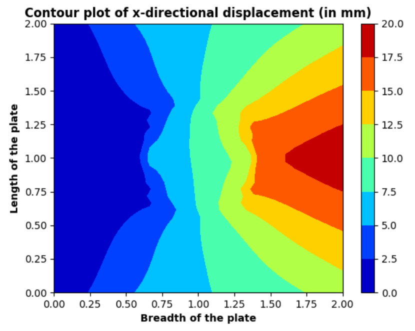
(b) Stress contour in y-direction

Results for the validation(2) from the project(main program):

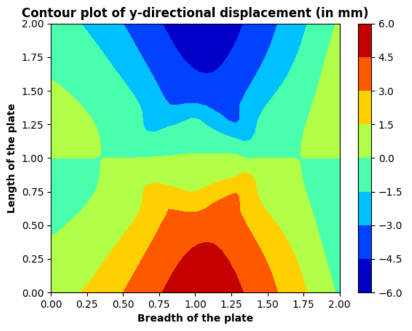
1. Plate of square of length 2m.
2. Radius of the hole in the centre of the plate = 0.4m
3. Uniaxial Tension along x direction = 1000 Pa
4. Young's modulus of plate and hole = 10^5 kg/cm^2 and 10^2 kg/cm^2
5. Poisson's ratio = 0.3 , Mesh size = 40*40
6. Dirichlet Boundary conditions are applied on the Edge 4 (Both edges are constrained)

Results from the project for Hole and inclusion(main program):

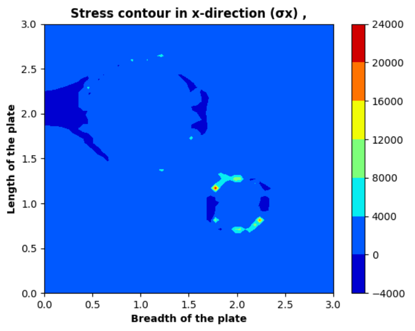
1. Plate of square of length 3*3m.
2. Radius of the hole and inclusion in the centre of the plate = 0.6 m and 0.4m



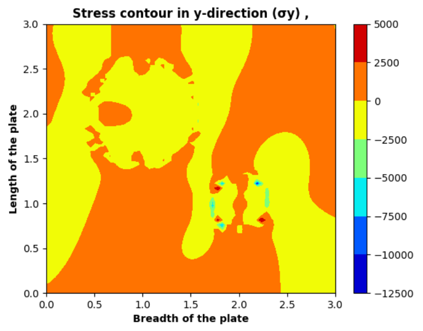
(a) Stress contour in x-direction



(b) Stress contour in y-direction



(a) Stress contour in x-direction



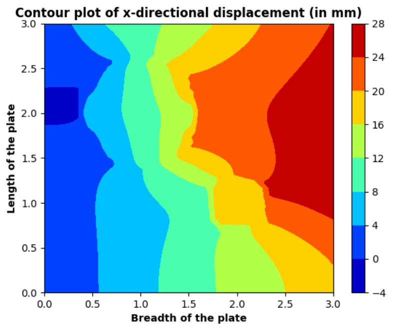
(b) Stress contour in y-direction

3. Uniaxial Tension along x - direction = 1000 Pa

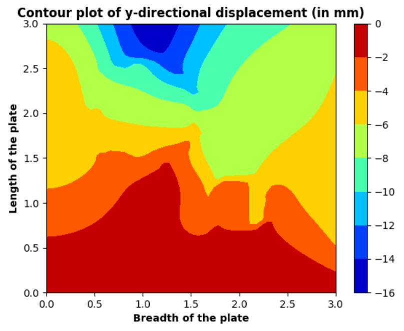
4. Young's modulus of plate and hole = 10^5 kg/cm^2 and 10^2 kg/cm^2

5. Poisson's ratio = 0.3 , Mesh size = 40*40

6. Dirichlet Boundary conditions are applied on the Edge 1(constrained in y-direction) and Edge 4 (constrained in y-direction),



(a) Displacement contour in x-direction



(b) Displacement contour in y-direction

6. Git log

Theses are the commits done for the program in github

```
commit 693d711d8062e4ddb36341a5db0b014b01033625 (HEAD -> main, origin/main, origin/HE
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Fri May 19 12:45:48 2023 +0200

    Rename Traction_force_function_1.py to Traction_force_function_2.py

commit 8bb315fda9204a5b4a50462ce6f6d60b7133013f
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Fri May 19 12:44:51 2023 +0200

    Update and rename test_1.py to test_sanitytest.py

    All sanity test has been done

commit 13b5e909e5b218fea9c8cc8dc27a7a525631674b
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Fri May 19 03:23:40 2023 +0200

    Update and rename test_XFEM.py to test_unittest.py

    Unit_test has been successfully completed

commit e99e301e8694378a6bc73bd03b7716166500e2a4
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Thu May 18 18:08:29 2023 +0200
```

```
    Adding Patch_test

commit 96621a0130cd4cd2182585e0fe0e9ac8d6fd52fc
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Thu May 18 06:48:39 2023 +0200

    Update Stress_strain_components.py

commit 2fa65284069d7dbce4f4c2515de1b9b7ecca1230
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Thu May 18 06:47:56 2023 +0200

    Update Main_program.py

    Converting stress_strain results to function file

commit f65de388ec3e8c50d4a6ab1073d057a6f209b670
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Thu May 18 01:50:59 2023 +0200

    Add files via upload

    Extended as function file from main program

commit 69dee670ee8142731b93713eb661d5b11e0489f8
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Thu May 18 01:49:57 2023 +0200

    Update Main_program.py
    Adding comments to the program

commit f7c937ceda5a7de36721dc2e50c6b981fad0938a
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Wed May 17 00:53:59 2023 +0200

    Update Dirichlet_Boundarycondition.py

    Updating for force vector

commit 77ba5893a9e0a92cf5a0738b1f284fa16b8ccf58
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Wed May 17 00:52:22 2023 +0200

    Update Body_force_function.py

commit 955dfb6e0cb047b0357e08603842d5e103ec8a13
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Wed May 17 00:51:39 2023 +0200

    Update Traction_force_function_1.py

    Rectification

commit 8fa424503f362b2a51269247fde98e5236c9dc34
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Wed May 17 00:50:39 2023 +0200

    Update and rename XFEM_python_34_TestingBlendingelements.py to Main_program.py
```

```
commit ac4b058be0b6977fda5ca5cd131515df3962b32c
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Tue May 9 04:40:52 2023 +0200

    Update Naming_the_umatrix.py

    Error rectification done in coordinates with 'float' datatype.

commit bc67d949e3164b55fdd3693273ee46031a99610d
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Tue May 9 04:38:14 2023 +0200

    Update Boundary_nodes.py

    Error rectification

commit c17cfca7fe97f71e16353b4420bc4cf43c8a1f80
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Tue May 9 04:37:19 2023 +0200

    Update Boundary_elements.py

    Error rectification

commit a773707858934f13ae05b0a1b0556f1275ede87d
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Tue May 9 04:36:13 2023 +0200

    Update LevelSetFunction.py
    Error rectification

commit f990e6f9079205bdaa4c2d3015b00690700d641d
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Tue May 9 04:35:14 2023 +0200

    Update XFEM_python_35

    Updated for coordinates having decimal points.

commit 8a7a48a5300c2faead1022ef58b7e70cc2a37cf9
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Mon May 8 03:52:56 2023 +0200

    Updating_stress value extraction file

commit 3f3abc4f7c3714198f3e33ab7d4898353a0d2c9b
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Mon May 8 03:45:04 2023 +0200

    Updating stress contour map

commit ef0568b2fb6af15a80b09ff4c261befd673e2205
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:43:54 2023 +0200

    Main_program

commit c46bcad3ba15ecf51ae58d055e09f3b132dd51a0
```

```
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:43:04 2023 +0200

    Update B_matrix_for_enhanced.py

commit 4845bf0358e86a06380706787fa2e7b5634cec11
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:40:44 2023 +0200

    Delete Building_elements.py

commit efbca598a6e03fdcc82afddce6925fd4a8854abc
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:38:37 2023 +0200

    Add files via upload

commit 3aca62b6acea3d38e5358285911912b1baaedb99
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:37:47 2023 +0200

    Adding further function files

commit cf531927adabf2ada370dc3be153e9a8c3565e88
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:36:11 2023 +0200

    Uploading B_matrix as Separate function files

commit bcf0f39156befe9fbc6f0c358459ef790bcad19c
```

```
Author: Nandha Gopal Mariappan <129469684+Nandha1911@users.noreply.github.com>
Date:   Sun May 7 02:34:33 2023 +0200

    Adding the function files

commit 0200a3f40c0e7212017e9660f941e6bdb3da2a30
Author: Nandha1911 <129469684+Nandha1911@users.noreply.github.com>
Date:   Mon Apr 24 00:54:27 2023 +0200

    Assembly_function for Stiffness matrix

commit 72664034d66756322b207c650fcc9ebd9ba17acd
Author: Nandha1911 <129469684+Nandha1911@users.noreply.github.com>
Date:   Sat Apr 8 04:54:31 2023 +0200

    Function files to run the test_XFEM.py

commit b9b3edc9207f496c93fd4ea9bcddc41615883aec
Author: Nandha1911 <129469684+Nandha1911@users.noreply.github.com>
Date:   Sat Apr 8 04:42:31 2023 +0200

    Testing file
```

Bibliography

Fish, J., & Belytschko, T. (2007). *A first course in finite elements* (Vol. 1). Wiley New York.

Jiang, S., Du, C., & Gu, C. (2014). An investigation into the effects of voids, inclusions and minor cracks on major crack propagation by using xfem. *Structural Engineering and Mechanics*, *49*(5), 597–618.

Khoei, A. R. (2014). *Extended finite element method: theory and applications.* John Wiley & Sons.

MinhNguyen. (2021). *2d gaussian quadrature.* Retrieved from `https://www.minhnguyencae.info/2d-gaussian-quadrature/`

Sukumar, N., Chopp, D. L., Moës, N., & Belytschko, T. (2001). Modeling holes and inclusions by level sets in the extended finite-element method. *Computer methods in applied mechanics and engineering*, *190*(46-47), 6183–6200.