Implementation of two-scale programming for a 2D Quadrilateral FEA Element

Technische Universität Bergakademie Freiberg

Personal Programming Project

By Harikeshava Ranganathan
Matriculation Number: 63941
Department: Computational Materials Science
email: harikeshava.ranganathan @ student.tu-freiberg.de

5th April,2020

Contents

List of Figures

Chapter 1

Introduction

The Heterogeneous (Composite) material has gained huge attention in the field of manufacturing. Almost all the material used in the present world is heterogeneous. The heterogeneous material in general consists of some impurities like micro-voids, inclusions,micro-cracks, etc., which will drastically impact the material's mechanical properties. The application of heterogeneous material can be seen in industries like Aerospace, Automotive, sports equipment, etc.

The objective of this project is to implement the FE^2 method. The behavior of the macro scale is analyzed by solving a nested Boundary value problem(BVP). First, The code for 2 different scales is developed separately. The developed code for two different scales is coupled to analyze the nested BVP. The macro-scale consists of cantilever plate with the representative volume element (RVE), arranged in the periodic pattern along the plate's length and height, at each gauss point of the macro quadrilateral element(as shown in figure 1.1). The RVE element is assumed to be, an element with a void in the center of the geometry. The Macro scale consists of a 2D quadrilateral FEM element and the system is solved using the FE^2 technique.The accuracy of the FE^2 method is better than a normal FEM analysis.

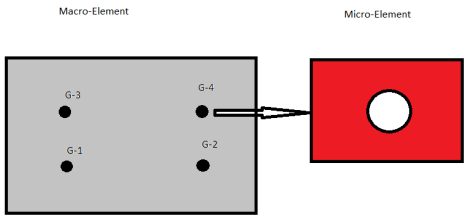

Figure 1.1: Macro and micro element

Chapter 2

Theory

2.1 Theoretical background

The Multi-scale FEM analysis is a technique in which we have a nested Boundary valued problem(BVP) to be solved [2], for the analysis of the system's behaviour under external factors. The macro and the micro scale are dependent on each other because of this nested format. The macro has a separate BVP problem to be solved, on the other hand, the micro has a separate BVP problem to be solved. The coupling of the system is the challenging part of this technique. The macro scale is a force-driven system and the microscale is a strain driven system.

The microprogram plays the role of the material routine in the coupled program[2]. As mentioned earlier the problem is a nested BVP problem, where at each gauss point of the macro element the micro-scale program is called. Therefore the input for the micro-scale is the Macro scale's strain and the output from the microscale is the tangent stiffness matrix and the stress at that gauss point's of the macro scale. The results of the macro scale are obtained after the convergences of Newton Raphson's at both the scales.

The classical FEM analysis [3] is applied to the system for analysis on each scale. The inelastic behavior of the system is also considered. The assumptions made for the system under interest are:
1. The system is force driven in macro-level and the system is strain driven in the micro-scale.
2. The system is assumed to be analyzed using plain strain.
3. The system is assumed to be deformed due to isotropic hardening(plastic flow).
4. The Von Mises yield criteria are used for the analysis of plastic flow.
5. The problem is solved using Newton-Raphsons method.
6. The number of gauss points per direction is 2 per element. In total there are 4 gauss points in an element.

The brief theory [4, 5, 1]behind the assumptions are mentioned below:
Plane strain: The problem is defined by having strain in the XY plane which is called in-plane strains and the system's strain component does not vary in the direction perpendicular to the plane XY.This means that we have uniform stress in the 3rd direction, that is z.

$$\epsilon_{33} = \epsilon_{32} = \epsilon_{13} = 0, \sigma_{33} = fn(\sigma_{11}, \sigma_{22})$$

Von Mises Yield Criteria: The criteria have a function ϕ, which speaks about the point or the boundary from which the deformation of material begins. The system reaches the yield boundary when the principal stresses of the system satisfy the below-mentioned equation,

$$\phi(\sigma, \beta) = \mid dev(\sigma) \mid -\sqrt{\frac{2}{3}}(\sigma_y + \beta)$$

Isotropic Hardening: The deformation of material causes the defects to accumulate along the grain boundaries in the micro-scale,due to which the material shows a hardening behavior. The isotropic hardening is the phenomena in which the yield boundary expands or widens isotropically, that is the stretch in all directions is the same. The shape of the yield surface changes in the hardening region. The picture below defines the phenomena.

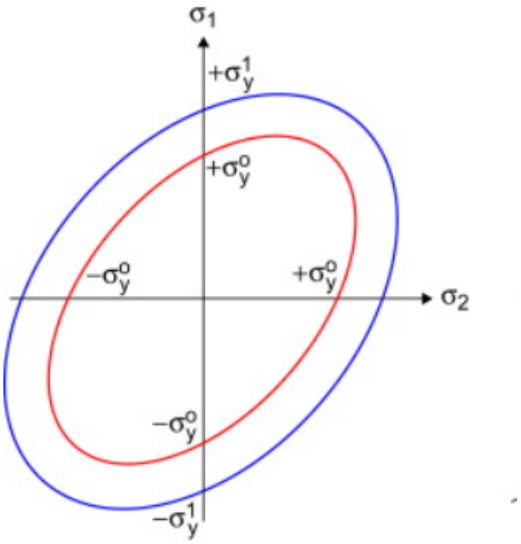

Figure 2.1: Isotropic Hardening,[1]

2.2 Constitutive Equations

2.2.1 Macro-Scale

The quadrilateral (Rectangular) bi-linear element is used in the Macro scale BVP. The equations involved in the first set of BVP are:
The PDE equation[3] of the strong form is:

$$\sigma_{ij,j} + f_j - \rho\ddot{u}_j = 0$$

Weak Form: The representation of the differential equation of the underlying problem in terms of integral function is called a weak form of the given differential equation. This

is done using mathematical manipulation by multiplying a virtual variable to the system and solving it by using the integration concept. The weak form of the problem of interest is:

$$\int_V \sigma_{ij}\delta\epsilon \; dV - \int_{\partial S} t_j \delta u_j \; ds - \int_V f_j \delta u_j \; dV$$

Boundary Condition(BC): The additional constraints using which BVP is solved is called as the Boundary condition. There are 2 types of Boundary condition in classical FEM:
1. Natural BC: They are represented in terms of Forces or pressures acting in the system. The natural BC can be in terms of distributive loading along the surface or it can be a concentrated load acting in the node positions. The natural BC is given by the F_{ext} term in the weak form.
2. Essential BC: They are represented in terms of the displacement values of the nodes of the assumed element.
In our case the system is a cantilever plate, so the nodes along the left end of the plate are restricted to move or rotate along any direction, that is its degree of freedom is zero. so the essential boundary condition is:

$$u_j = 0$$

The system is now divided into n elements, in the element domain. Later, the elements are mapped from element domain to unit domain for ease of calculation. The two domains are related by the Jacobian matrix, which maps the 2 co-ordinate domains of the elements. The equations related to the unit domain and respective matrix are mentioned below:
1. Shape function and its matrix's representation:

$$N_I = \frac{1}{4}(1 \pm \xi_1)(1 \pm \xi_2)$$

$$\underline{\underline{N}} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix}$$

2. Strain-displacement matrix's representation:

$$B_I = \begin{bmatrix} N_{I,1} & 0 \\ 0 & N_{I,2} \\ N_{I,2} & N_{I,1} \end{bmatrix}$$

$$\underline{\underline{B}} = \begin{bmatrix} B_1 & B_2 & B_3 & .... & B_n \end{bmatrix}$$

3. Stiffness matrix's K:

$$\underline{\underline{K}} = \int_V B^T C_t B dV$$

4. Displacement Matrix's:

$$u_I = \begin{bmatrix} u_{Ix} \\ u_{Iy} \end{bmatrix}$$

$$\underline{\underline{U}} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_I \end{bmatrix}$$

5. Co-ordinate matrix's of node points:

$$X_e = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}$$

6. Jacobin Matrix's:

$$\frac{\partial N}{\partial \xi} = \begin{bmatrix} \frac{\partial N_1}{\partial \xi_1} & \frac{\partial N_2}{\partial \xi_1} & \frac{\partial N_3}{\partial \xi_1} & \frac{\partial N_4}{\partial \xi_1} \\ \frac{\partial N_1}{\partial \xi_2} & \frac{\partial N_2}{\partial \xi_2} & \frac{\partial N_3}{\partial \xi_2} & \frac{\partial N_4}{\partial \xi_2} \end{bmatrix}$$

$$\underline{\underline{J}} = \frac{\partial N}{\partial \xi} \cdot X_e$$

7. Force Matrix's:

$$\underline{\underline{F_{int}}} = \int_V B^T \sigma \; dV$$

$$\underline{\underline{F_{ext}}} = \int_{\partial S} t_j \delta u_j \; ds - \int_V f_j \delta u_j \; dV$$

2.2.2 Micro-Scale

The micro scale is initiated at every gauss point present on the macro scale. The micro-scale also follows the same procedure of classical FEM as mentioned above in the macro section, with some additional equations and concepts. The Representative volume element is considered in this analysis. The RVE is treated as a pure heterogeneous system, so the quadrilateral meshing technique is adopted here. The RVE element consists of a plate with a void present in its center.

The partitioning techniques is used here to separate the nodes of the system into in-

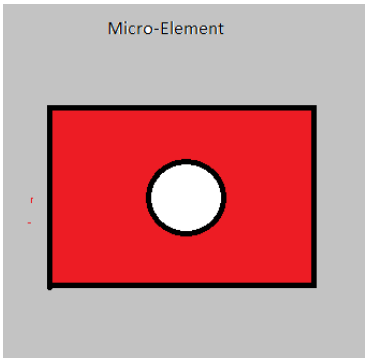

Figure 2.2: Micro Element

dependent and dependent nodes[2]. The global stiffness matrix and the internal force matrix are partitioned into the combinations of the independent and dependent nodal representations.The macro-scales strain is used to find the initial displacement of the the dependent nodes, using which the program kick starts.The plasticity models and equations are obtained from [4] and from plasticity lecture notes.

Plasticity Model:

$$\underline{\underline{\sigma}} = \kappa tr(\underline{\underline{\epsilon}})\underline{\underline{I}} + 2\mu dev(\underline{\underline{\epsilon_e}})$$

$$\epsilon = \epsilon_e + \epsilon_p$$

$$\phi(\sigma, \beta) = \mid dev(\sigma) \mid - \sqrt{\frac{2}{3}}(\sigma_y + \beta)$$

$$\beta = h\alpha$$

$$\xi = dev(\sigma)$$

$$\dot{\epsilon}_p = \dot{\lambda}\frac{\partial \phi}{\partial \xi}$$

$$\dot{\alpha} = \sqrt{\frac{2}{3}}\dot{\lambda}$$

Boundary condition:
1.The displacement of node 9 in the interior is fixed(as mentioned in the below equation).
2. The boundary nodes are paired to apply the periodic boundary condition(as mentioned in the below equation).

$$U_9 = \underline{0}$$

$$u^- = u^+ + \epsilon\ \delta x$$

The Classical FEM included with the partitioning techniques together formulates the micro-scale code. The same FEM equations were used as mentioned above. The additional partitioning equations [6, 7] used are:

$$K_{Total} = K_{ii} + K_{id}A + AK_{di} + A^T K_{dd} A$$

$$F_{internal} = F_{ii} + A^T F_{dd}$$

The flow and the algorithm of the program is mentioned in the next chapter.

Chapter 3

Numerical methods and Implementation details

3.1 Numerical methods

Newton-Raphsons Method: The Newton-Raphson method [3] is a mathematical method to solve the non-linear system of equations for finding the root of a real-valued function f(x)=0. It uses the idea that a continuous and differentiable function can be approximated by a straight-line tangent to it. Newton's method provides a better approximation for the root and is given by,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

3.1.1 Constitutive Newton-Raphson Equations

$$\underline{\underline{G}} = F_{int} - F_{ext}$$

$$\underline{\underline{K}}_{Total} = A^T K_I A$$

$$\underline{\underline{F}}_{internal} = A^T F_I$$

$$\triangle \underline{\underline{U}} = -\underline{\underline{K}}^{-1} \cdot \underline{\underline{G}}$$

$$\underline{\underline{U}} = \underline{\underline{U}} + \triangle \underline{\underline{U}}$$

Newton Raphson equation and the convergence criteria are mentioned in the above equation.

Gaussian Quadrature: The Gaussian quadrature is a technique using which a definite integral of a function can be represented in terms of the weighted sum of the function at some points in the domain of the integration itself.

α	1	2	3	4
$\xi_1(\alpha)$	-0.57735	0.57735	-0.57735	0.57735
$\xi_2(\alpha)$	-0.57735	-0.57735	0.57735	0.57735

3.2 Implementation

The program is split into 3 parts:
1. Pre-processing
2. Analysis
3. Post-processing

Pre-processing: The pre-processing is the stage where the constructed model is divided into N number of discrete subregions or elements. The elements are connected at the intersection points called nodes. The subregions in our analysis represent a geometry of quadrilateral in shape, these subregions are also called as meshes. Now, this developed system's data are used for the FEM analysis, as an input parameter.

Analysis: The data-set obtained from the part of pre-processing is used as the input for our FEM analysis. The code developed will solve the system of equation to find the unknowns, which are the external parameters. The system has 3 main variables the Total stiffness matrix, total displacement matrix(the unknowns) and the values of the known forces of each node(external parameter).
The code consists of 4 functions:
(the actual names of the functions are mention in the respective sections)

1. Displacement arrangement function: This function is used for assembling the displacement from global to local variables and vice versa. The dictionary mapping technique is used here for doing the above function.

2. Assignment matrix Function: This function is used for assembling the assignment matrix for the element coming in as the local element. The assignment matrix is used for mapping the element stiffness matrix and global stiffness matrix.

3. Material routine: This function is used to calculate the solution-dependent state variables of the constitutive mechanical models used for the analysis. The stress, strains and internal state variables of the model assumed are computed here. The materials current state is kept as the key idea for computing the model's parameters. Here we use a predictor-corrector analysis for predicting the state of the material, that is whether it is in the elastic state or plastic state. The von-mises yield criteria are used for the plasticity model.

4. Element routine: This function is used to compute the stiffness matrix and the internal forces acting on the element. This function is called for each element that is coming in for the analysis. The material routine is called in this function for the calculation purpose, for obtaining the material tangent stiffness matrix and elements stress update(for the calculation of the internal forces).

Post-processing:
Post-processing is the step where the results are visualized and the inferences are made using the obtained results.

3.2.1 Macro-Scale

The macro-scale has the above 4 functions for the FEM analysis. The macro scale is a force-driven BVP problem. The Macroscale consists of the following attributes and

variables, as mentioned below:
1. **Macro-Geometry:**
macro_length = 100 cm
macro_height = 50 cm
thickness_plate = 500 cm
2. **Meshing parameter:**
N = 1 (number of elements along x-direction)
M = 1 (number of elements along y-direction)
Xe- array of $4 * 2$, holds the Global co-ordinates of the element.
x_disp - array that is used for the access of the next element in the same row (along the length).
y_disp - array that is used for the access of the next row elements(along the height).
3. **Material Properties:**
Youngs_modulus = 70 GPa
Poissons_ratio = 0.3
Yield_stress = 95 MPa
h= 200 MPa (Hardness)
4. **Global coordinate variables:**
Global_F_external - The array holds the nodal external forces.
Global_F_external - The array holds the nodal internal forces.
Global_displacement - The array holds the nodal displacement.
Gauss_point_plastic_ strain - The n-dimensional array holds the plastic strain values at each gauss point of the element, that is the first position consists of the total micro elements plastic strain of first gauss point of the macro scale.
Gauss_point_alpha - The n-dimensional array holds the drag strain values at each gauss point of the element, that is the first position consists of the total micro elements drag strain of first gauss point of the macro scale.
Global_stiffness_ matrixs - The array holds the total system stiffness values.
5. **Local coordinate variables:**
u_ element - The array holds the element's displacement values.
plastic_strain - The array holds the element's plastic strain values.
Assignment_ matrix - The array is used for mapping the local and global variables. The array consists of the values 0 or 1.
Element_stiffness_matrixs - The array consists of the element's stiffness values.
6.**Temporary variables:** The temporary variables are used in many parts of the program for the assigning and calculation purpose.
7.**Newton-Raphsons Variables:** All the variables starting with R or Reduced are the reduced form of the variables used for newton raphsons scheme.
8.**Techniques Used:**
1.data: The variable, data is used to store the global positions of the nodes available in the system.The dictionary mapping technique was used to assemble the global values of the node in the local variables and vice verse,using this variable.
2. Node_Numbering: The variable holds the node numbers in the $n*n$ matrix.The square patterning technique is used to group the nodes to represent a element.
The output parameters are the displacement of the nodes. The techniques and the role of the function is also mentioned in the code.
9.**Post-processing:** The U_x and U_y are variables used to plot a contourf plot to visualise the behaviour of the system along x and y direction due to the applied force in the

system.
The strain_x,strain_y and strainxy are the parameters used plotting the countourf plot of the strain in the system.On the other hand the stress_x,stress_y and stress_xy are used to plot the stress of the system.

3.2.2 Micro-Scale

The micro-scale also has 4 functions, the micro program plays the role of material routine for the macro program. The number of element is fixed here(Equal to 8). The meshing pattern for the micro scale was also developed, but failed to merge it with the micro program due to few internal complexity. The attributes and the variable used are mentioned below:

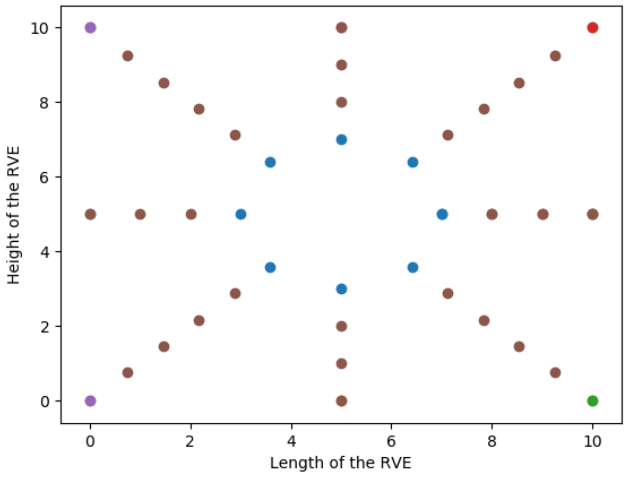

Figure 3.1: Meshing pattern for a plate with a hole

1. **Macro-Geometry:**

micro_length = 10 μm
micro_height = 10 μm
micro_thickness_plate = 500 cm
void_radius= 2 μm

2. **Meshing parameter:**

Micro_coordinate - holds the global coordinates of the element.
n- number of elements present in the RVE element.

3. **Material Properties:**

Youngs_modulus = 70 GPa
Poissons_ratio = 0.3
Yield_stress = 95 MPa
h= 200 MPa (Hardness)

4. **Global coordinate variables:**

global_k_ii - holds the global stiffness values of the independent -independent combinations of nodes.
global_k_id - holds the global stiffness values of the independent-dependent combination

of the nodes.
global_k_di - holds the global values of the dependent-independent combinations of nodes.
global_k_dd - holds the global stiffness values of the dependent-dependent combinations of nodes.
gauss_point_plastic_strain - holds the plastic strain of the RVE elements for the incoming gauss point.
gauss_point_alpha - holds the drag strain values of the RVE element for the incoming gauss point.
u_independent - holds the displacement value of the independent nodes,that is the interior and the positive surface boundary nodes.
F_total - holds the global internal forces values of the total system.
K_total - holds the global stiffness values of the total system.
5. **Local coordinate variables:**
Element_stiffness_micro - holds the stiffness values of the element of the RVE coming in for analysis.
F_int_ii - The variable holds the internal forces of the independent-independent node combinations, of the element coming in for analysis.
F_int_dd - The variable holds the internal forces of the dependent-dependent node combinations, of the element coming in.
k_ii - holds the stiffness values of the independent-independent values of the nodes of the element coming in for analysis.
k_id - holds the stiffness values of the independent-dependent values of the nodes of the element coming in for analysis.
k_di - holds the stiffness values of the dependent-independent values of the nodes of the element coming in for analysis.
k_dd - holds the stiffness values of the dependent-dependent values of the nodes of the element coming in for analysis.
6.**Temporary variables:** The temporary variables are used in many parts of the program for assigning and the calculation purpose.
7.**Newton-Raphsons Variables:** All the variables starting with R or Reduced are the reduced form of the variables used for newton raphsons scheme.
8.**Techniques Used:**
The dictionary mapping technique is used in majority part of the code.The global positions of the system is saved in the separate variables. The systems nodes are partitioned into independent and dependent nodes. So, the system consists of 2 different sets of parameters for each. The final results of the stiffness matrix and the internal forces are used to find the macro-scales material parameters. The output parameters are the macro-scale's material parameter. The techniques and the role of the function is also mentioned in the code.

3.2.3 Work flow of the coupled code

coupled code file name: **PPPSS19_63941_micro_macro_ep_coupling.py**

Micro part

function MICRO_ DISPLACEMENT_ ARRANGEMENT(*u_element...*)

The function to assemble local displacement variables w.r.t Global displacement variables and vice versa.
return $u\_element$
end function
function MICRO_MATERIAL_ROUTINE($MU, lamda....yield\_stress$)
The function returns the material properties of the micro element.
The trial values are found here.

if $plastic\_fn < tol$ **then**
The elastic equations are used
$deviatoric\_stress \leftarrow deviatoric\_stress\_trial$
$deviatoric\_tangent\_stiffness \leftarrow 2 * MU * P\_sym$
$plastic\_strain \leftarrow plastic\_strain$
$alpha \leftarrow alpha$
else
The plastic equations are used, The fn is the function which represents the correction factor due to plasticity.
$deviatoric\_stress \leftarrow deviatoric\_stress\_trial + fn(plasticcorrector, normal\_vector)$
$deviatoric\_tangent\_stiffness \leftarrow 2 * MU * P\_sym + fn(plasticcorrector, P\_sym)$
$plastic\_strain \leftarrow plastic\_strain + fn(plasticcorrector, normal\_vector)$
$alpha \leftarrow alpha + fn(plasticcorrector)$
end if
return $C\_tangential, stress\_element, B\_ps, C\_al$
end function
function MICRO_ELEMENT_ROUTINE($x\_element, Element\_stiffness\_micro, F\_int\_Element...$)
The function will return the element stiffness and the internal force of the element.

for $i = 1...4$ **do**
C_tangential,stress_element,B_ps,C_al = Micro_Material_routine(MU,lamda....yield_stress)
$Element\_stiffness\_micro \leftarrow (Element\_stiffness\_micro) + (B^T * C\_tangential * B)$
$F\_int\_Element \leftarrow (F\_int\_Element) + (B^T * stress\_element)$
end for
return $Element\_stiffness\_micro, F\_int\_Element$
end function

Macro part

function MACRO_MATERIAL_ROUTINE($MU, lamda....yield\_stress$)
The function consists of the micro's main part of the program.

while ($|\ \delta U\ | = |\ Global\_independent\ |$) **do**
for $i = 1...8$ **do**
$[K\_element, F_internal, B\_ps, C\_al] = Micro\_element\_routine(MU, lamda....yield\_stress)$
The matrix's coming as output is partitioned into different groups and are assembled into respective group variables.
end for

$K\_total \leftarrow (Global\_K\_ii)+(Global\_K\_id*A\_matrix)+((A\_matrix)^T*Global\_K\_di)+((A\_matrix)^T * Global\_K\_dd * A\_matrix)$

$F\_total \leftarrow (Global\_F\_ii) + ((A\_matrix)^T * Global\_F\_dd)$

end while
return $C\_tangential, stress\_element, B\_ps, C\_al$
end function
function U_ARRANGEMENT($u\_element...$)
The function to assemble local displacement variables w.r.t Global displacement variables and vice versa.
return $u\_element$
end function
function ASSEMBLE_FUNCTION($Assignment\_matrixs...$)
The function to assemble assignment matrix with 1 or 0 depending on the element coming into play.
return $Assignment\_matrixs$
end function
function MACRO_ELEMENT_ROUTINE($x\_element, Element\_stiffness\_micro, F\_int\_Element...$)
The function will return the element stiffness and the internal force of the element.

for $i = 1...4$ **do**
C_tangential,stress_element,B_ps,C_al = Macro_Material_routine(MU,lamda....yield_stress)
$Element\_stiffness\_matrixs \leftarrow (Element\_stiffness\_matrixs)+(B^T*C\_tangential*B)$
$F\_int\_Element \leftarrow (F\_int\_Element) + (B^T * stress\_element)$
end for
return $Element\_stiffness\_matrixs, F\_int\_Element$
end function

Main part of the program
The macro meshing and initialization of the variables for calculation is done here.
for $time\_step = 1...$ **do**
while ($|\ \delta U\ |=|\ Global\_displacement\ |$) **do**
for $i = 1...N$ **do**
for $j = 1...M$ **do**
$Assignment\_matrixs = assemble\_function(Assignment\_matrixs...)$
$u\_element = u\_arrangement(u\_element...)$
$Element\_stiffness\_micro, F\_int\_Element = Macro\_Element\_routine(x\_element.....)$
$Global\_stiffness\_matrixs \leftarrow (Global\_stiffness\_matrixs)+((Assignment\_matrixs)^T*Element\_stiffness * Assignment\_matrixs)$
$Global\_F\_int \leftarrow (Global\_F\_int)+((Assignment\_matrixs)^T*F\_int\_Element)$
end for
end for
$G = Global\_F\_int - Global\_F\_ext\ \delta U = -(Global\_stiffness\_matrixs)^-1 * G$
$Global\_displacement \leftarrow (Global\_displacement) + \delta U$
end while
end for

Chapter 4

Results of testing and Discussion

4.1 Results

4.1.1 Validation:

The test cases [8, 9] are done for both the macro and the micro-scale. The test case for the macro part is done using the macro program, which is a program for a 2D quadrilateral element. The system was considered homogeneous on the macro scale and are tested. The test results are mentioned below the procedure to run the code for the test cases is mentioned in the user manual section.

Macro scale Test Cases:

Macro code file name: **PPPSS19_63941_macro_scale_program.py**

Rigid rotation:

Aim: The rigid rotation is a test done to verify that our system does not strain for a normal coordinate transformation.

Expected results: The system should be in equilibrium,$\sum_{i=1}^{N} F_i^x = 0 and \sum_{i=1}^{N} F_i^y = 0$.

Obtained results: The $Internal force(F_{int})$ will resemble as shown in the fig.4.3 below.

```
180  ############################
181  ######## Testing Parameters ######
182  #For verifying the test cases, follow the instructions in the manual section (to run the program
183  tension_disp=np.array([[0,0],[-4.909*10**-9,-7.751*10**-10],[0,0],[-4.909*10**-9,7.751*10**-10]])
184  Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) #+ tension_disp
185  Global_displacement[2][0]= 4.909*10**-7
186  Global_displacement[3][0]= 7.748*10**-8
187  Global_displacement[4][0]= -4.909*10**-7
188  Global_displacement[5][0]= -7.748*10**-8
189  #### Boundary condition ####
190  #A=np.array([[2],[3],[4],[5]]) The variable A array should be uncommented in the line 293 of the
191  ############################
```

Figure 4.1: Rigid Body rotation parameters

Figure 4.2: Rigid body rotation diagram

```
Internal_force:

[[-9.09494702e-12]
 [ 1.81898940e-12]
 [ 1.11300923e+05]
 [-5.56504615e+04]
 [-1.11300923e+05]
 [ 5.56504615e+04]
 [ 0.00000000e+00]
 [-2.27373675e-12]]
[[0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]]
displacement:
 [[-2.58542857e-08]
 [-2.58045714e-08]
 [ 4.90900000e-07]
 [ 7.74800000e-08]
 [-4.90900000e-07]
 [-7.74800000e-08]
 [ 2.58542857e-08]
 [ 2.58045714e-08]]
Iteration number: 2
```

Figure 4.3: Rigid Body rotation

Rigid rotation with tension/compression/shearing:

1. Rigid rotation with tension:

Aim: The rigid rotation with tension, first the system is given a tension and then rotated. This again states that after rotation the system should not experience strain.

Expected Results: The equation $F_{int}(U_{rig}) = F_{int}(U_{rig+load\_applied}))$ must satisfy.

Obtained Results: fig 4.4 and fig 4.5

```
181  ######## Testing Parameters ######
182  #For verifying the test cases, follow the instructions in the manual section (to run the program for the test cases)
183  tension_disp=np.array([[4.909*10**-7,0],[4.909*10**-7,0],[4.909*10**-7,0],[4.909*10**-7,0]])
184  Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
185  Global_displacement[2][0]= 4.909*10**-7
186  Global_displacement[3][0]=  7.748*10**-8 #4.909*10**-7
187  Global_displacement[4][0]= -4.909*10**-7
188  Global_displacement[5][0]= -7.748*10**-8
```

Figure 4.4: Rigid Body rotation with tension parameters

```
Internal_force:

[[-9.09494702e-12]
 [ 1.81898940e-12]
 [ 1.11300923e+05]
 [-5.56504615e+04]
 [-1.11300923e+05]
 [ 5.56504615e+04]
 [ 0.00000000e+00]
 [-2.27373675e-12]]
displacement:
 [[-2.58542857e-08]
 [-2.58045714e-08]
 [ 4.90900000e-07]
 [ 7.74800000e-08]
 [-4.90900000e-07]
 [-7.74800000e-08]
 [ 2.58542857e-08]
 [ 2.58045714e-08]]
Iteration number: 2
```

Figure 4.5: Rigid Body rotation with tension

2. Rigid rotation with compression

Aim: The rigid rotation with compression, first the system is given a compression and then rotated. This again states that after rotation the system should not experience strain.

Expected Results: The equation $F_{int}(U_{rig}) = F_{int}(U_{rig+load\_applied}))$ must satisfy.

Obtained Results: fig 4.6 and fig 4.7

```
181  ######## Testing Parameters ######
182  #For verifying the test cases, follow the instructions in the manual section (to run the program for the test cases)
183  tension_disp=np.array([[-4.909*10**-7,0],[-4.909*10**-7,0],[-4.909*10**-7,0],[-4.909*10**-7,0]])
184  Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
185  Global_displacement[2][0]= 4.909*10**-7
186  Global_displacement[3][0]=  7.748*10**-8 #4.909*10**-7
187  Global_displacement[4][0]= -4.909*10**-7
188  Global_displacement[5][0]= -7.748*10**-8
```

Figure 4.6: Rigid Body rotation with compression parameters

```
Internal_force:

[[-9.09494702e-12]
 [ 1.81898940e-12]
 [ 1.11300923e+05]
 [-5.56504615e+04]
 [-1.11300923e+05]
 [ 5.56504615e+04]
 [ 0.00000000e+00]
 [-2.27373675e-12]]
displacement:
 [[-2.58542857e-08]
 [-2.58045714e-08]
 [ 4.90900000e-07]
 [ 7.74800000e-08]
 [-4.90900000e-07]
 [-7.74800000e-08]
 [ 2.58542857e-08]
 [ 2.58045714e-08]]
Iteration number: 2
```

Figure 4.7: Rigid Body rotation with compression

3. Rigid rotation with shear

Aim: The rigid rotation with shear, first the system is given a shear load and then rotated. This again states that after rotation the system should not experience strain.
Expected Results: The equation $F_{int}(U_{rig}) = F_{int}(U_{rig+load\_applied}))$ must satisfy.
Obtained Results: fig 4.8 and fig 4.9

```
181  ######## Testing Parameters ######
182  #For verifying the test cases, follow the instructions in the manual section (to run the program fo
183  tension_disp=np.array([[4.909*10**-7,0],[-3.27266*10**-8,0],[3.27266*10**-8,0],[-4.909*10**-7,0]])
184  Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
185  Global_displacement[2][0]= 4.909*10**-7
186  Global_displacement[3][0]=  7.748*10**-8  #-4.909*10**-7
187  Global_displacement[4][0]= -4.909*10**-7
188  Global_displacement[5][0]= -7.748*10**-8
```

Figure 4.8: Rigid Body rotation with shear

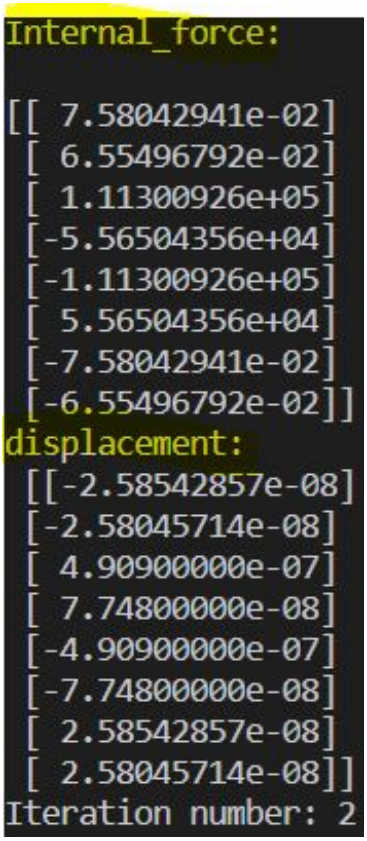

Figure 4.9: Rigid Body rotation with shear

The highlighted matrices in the figure are the Internal force matrix and the displacement matrix. The internal forces matrix shows that the above mentioned internal forces condition satisfy and the displacement matrix shows the direction of motion due to tension/compression/shear.

Patch Test:

Aim:The patch test is a test done to check the stability of the mesh. The element under analysis pass the patch test means the mesh is refined and stable. The patch test is done by considering an irregular mesh with at least 1 interior node present.

Expected Results: The external distributive load is applied to the system under consideration, for which the interior nodes are under constant stress state.

Obtained Results: Could not perform the patch test because, I could not figure out the parameter with respect to which the stress in the interior node is to be constant.

Micro Scale Validation:

Micro code file name: **PPPSS19_63941_micro_elasto_plastic_2.py**

The following test cases were also conducted in the micro program.

In all 4 cases of test mentioned above will give the same internal forces values(fig-4.10)(All the components are zero). The system as a whole is in equilibrium.

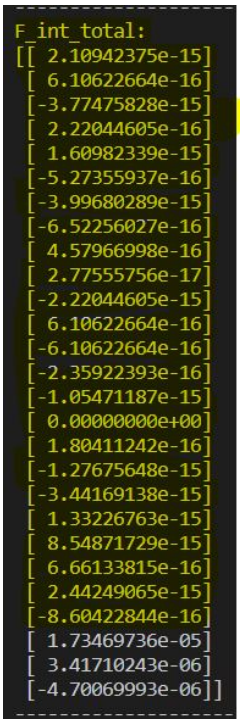

Figure 4.10: Rigid rotation with shear/tension/compression in micro scale

4.1.2 Verification:

The goal of the project is to implement a FE^2 method to bridge the micro and macro-scale analysis.The RVE element is assumed to have a hole in the center. As mentioned earlier, it is a nested BVP problem. The micro and macro scale program is developed separately, is coupled to bring in the FE^2 method. The test cases for both scales are implemented and are discussed in the validation section.
The material routine of the macro scale, that is the microprogram does not give the expected output(material properties, Tangent stiffness matrix, and the stress tensor at each gauss point). This will lead to an convergences error at the macro scale,as the input for the macro scale is not perfect.The reason behind the failure of the micro scale program are:
1. The micro program only works for the elastic case. There is a convergences error for the elasto-plastic case.
2. The concept behind the calculation of macro tangent stiffness matrix [6] (after the convergences of the microprogram) is not clear and there was very less information about that concept.
3. The static condensation technique [7] was utilized for the analysis in micro scale. The concept of getting the stress for each gauss point ,of the macro scale, was not understandable.
So,the microprogram was developed as per my understanding of the concept, by splitting it into independent and dependent nodes.

The results shown below are found using the homogeneous techniques[5],by replacing the RVE element by its homogeneous form. The homogeneous technique is implemented in the macro file named **PPPSS19_63941_macro_scale_program.py**.The

effective properties of the homogeneous system are found using the micro-mechanical technique. The effective properties of the plate with the hole is derived in [5]. The Micro program was replaced by using this technique, as there were difficulties to find the material parameters like tangential stiffness matrix, stresses and the internal state variables. The lemma's constant for the homogeneous system is found out using the effective properties.
The results are obtained by using the "**PPPSS19_63941_macro_scale_program.py**" code. The obtained results are displayed below.

Elastic state

The below figures 4.11 and 4.12 are the displacement graphs obtained using the interpolation technique of the classical FEM. The equation used is $U(x,y) = \sum_{i=1}^{N} N_i(x,y) * U_i$.

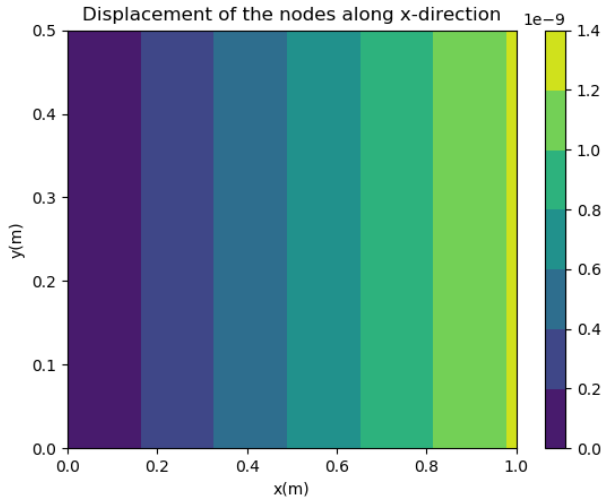

Figure 4.11: Displacement of the system along x-direction

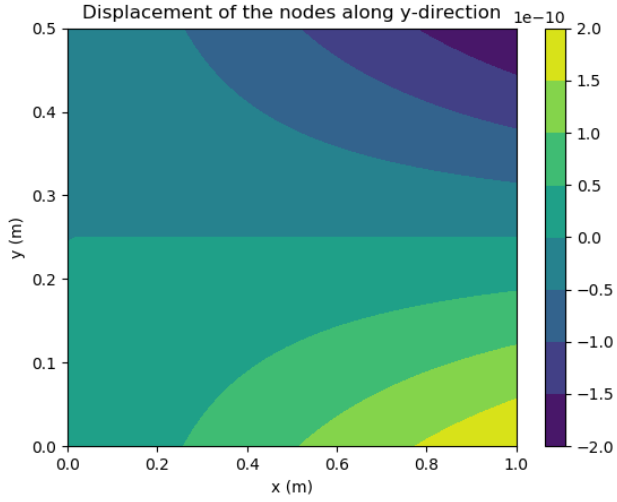

Figure 4.12: Displacement of the system along y direction

The strain(fig-4.13,4.14,4.15) and stress (fig-4.16,4.17,4.18) graphs of the system are displayed below.These results are obtained by using the interpolation technique. The equations used are $\epsilon_{ij} = \frac{\partial N_i}{\partial x_j} * u_i$ for strain and $\sigma_{ij} = C_{ijkl} * \epsilon_{kl}$ for stress.

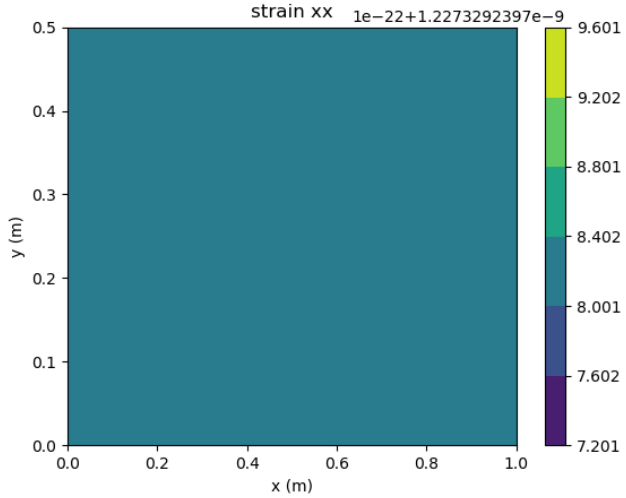

Figure 4.13: Strain xx

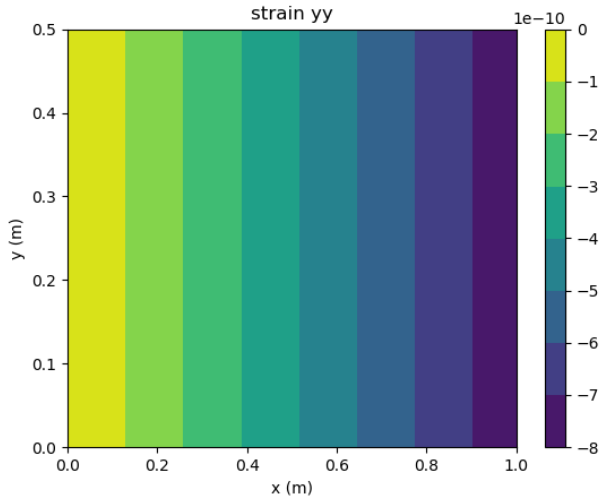

Figure 4.14: strain yy

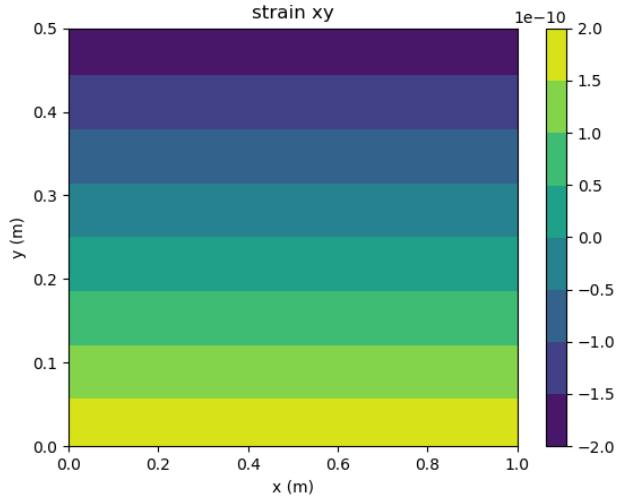

Figure 4.15: strain xy

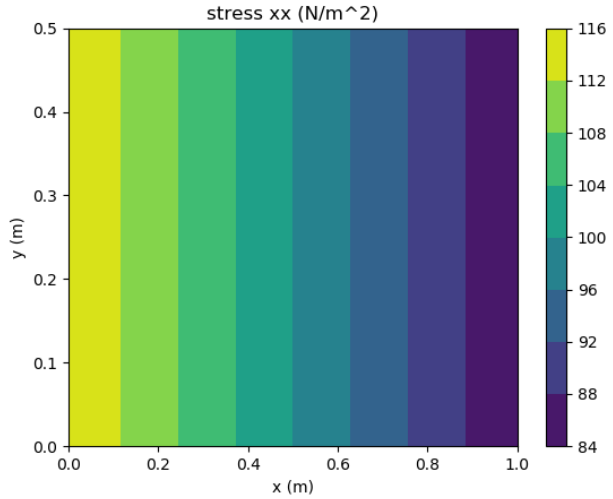

Figure 4.16: stress xx

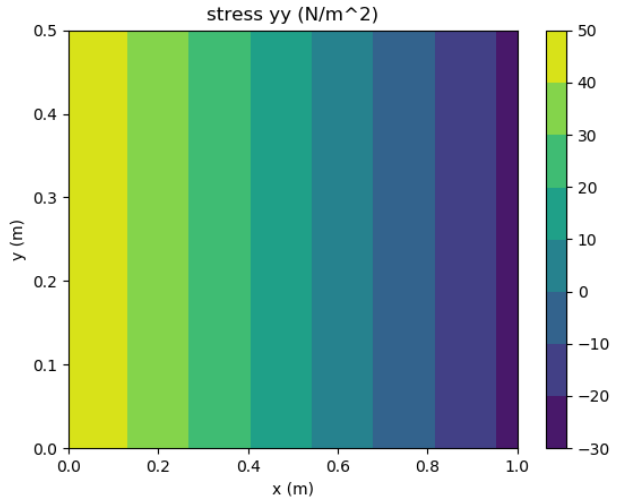

Figure 4.17: stress yy

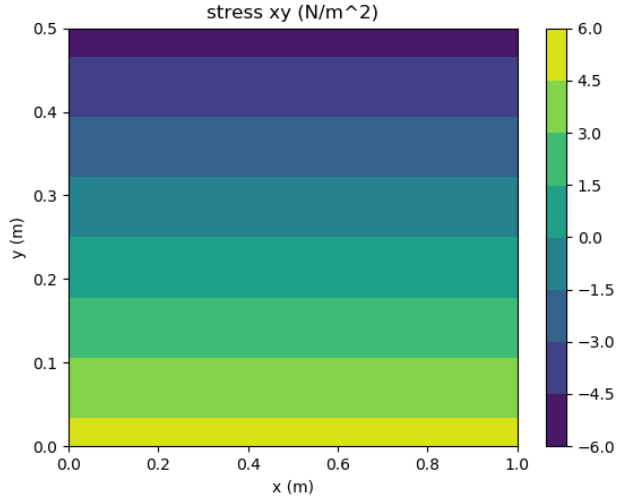

Figure 4.18: stress xy

Elasto-Plastic State

1. The figures(4.19 and 4.20) are the displacement graphs.
2. The figures(4.21,4.22 and 4.23) are the strain graphs.
3. The figure(4.24,4.25 and 4.26) are the stress graphs.

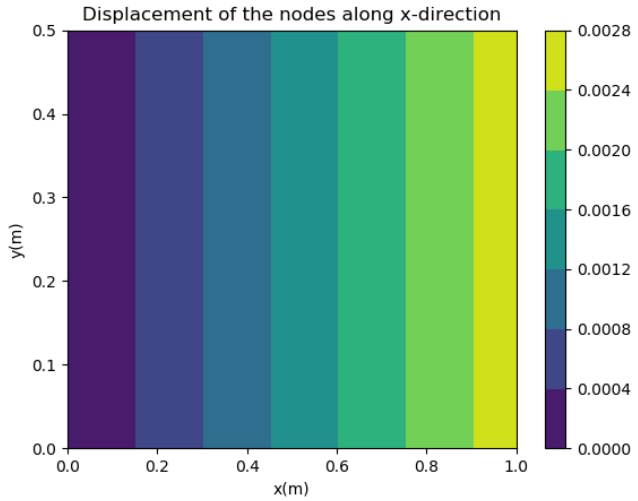

Figure 4.19: Displacement of the system along x-direction

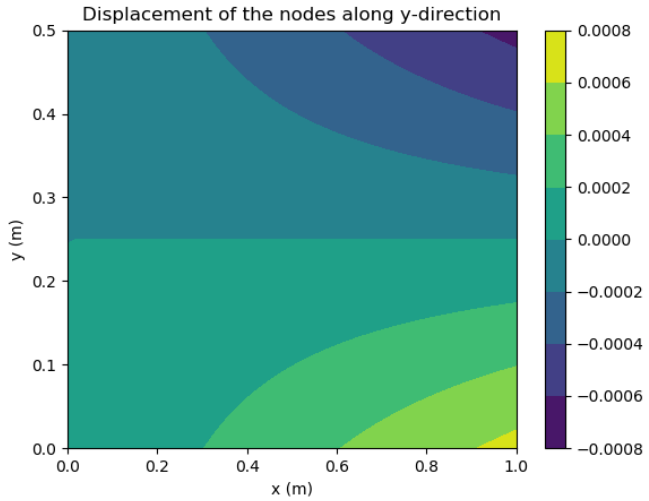

Figure 4.20: Displacement of the system along y direction

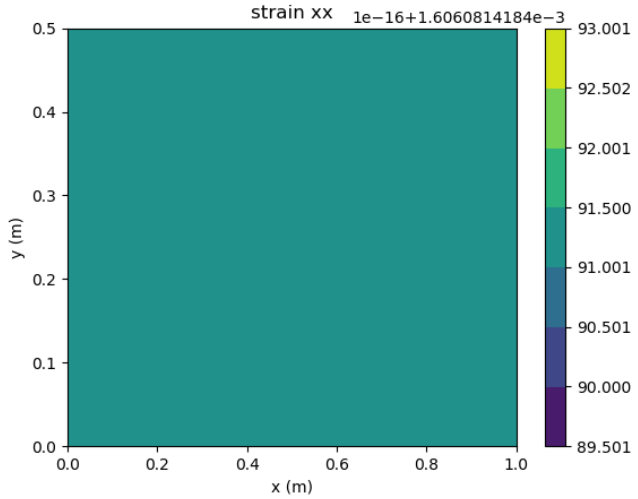

Figure 4.21: Strain xx

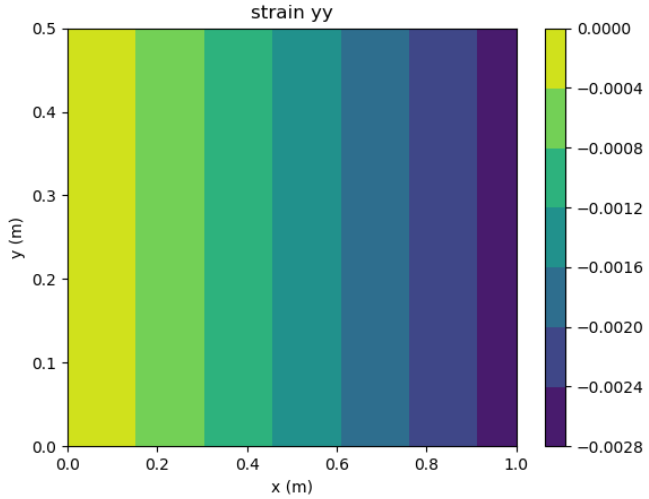

Figure 4.22: strain yy

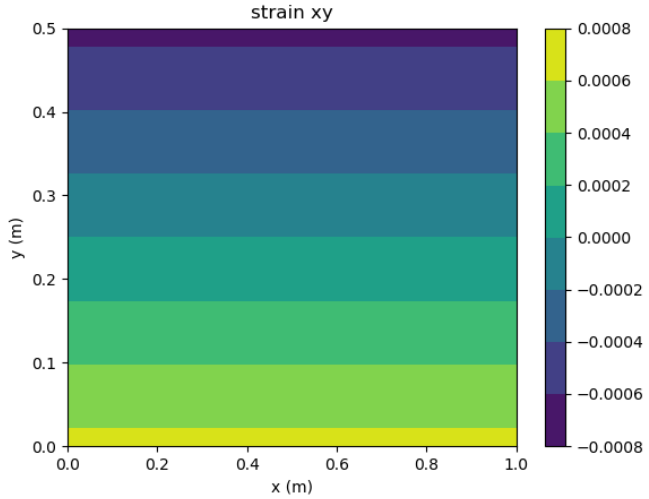

Figure 4.23: strain xy

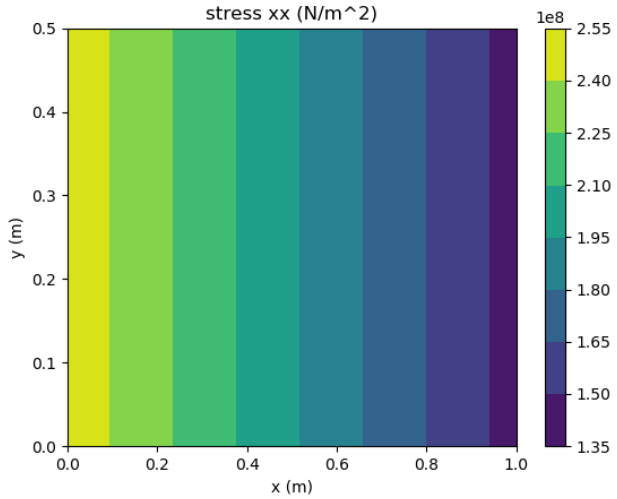

Figure 4.24: stress xx

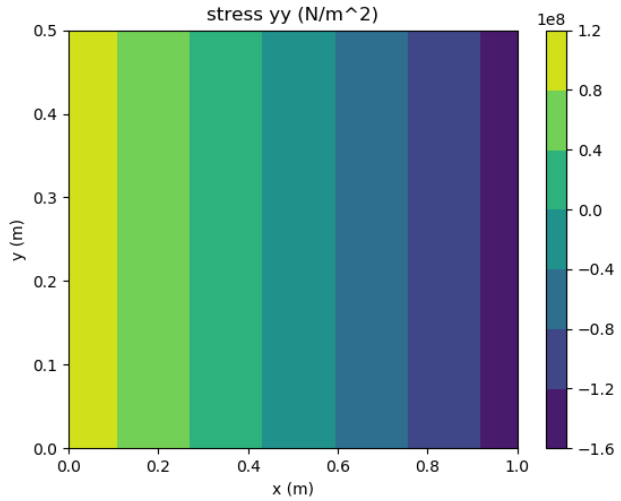

Figure 4.25: stress yy

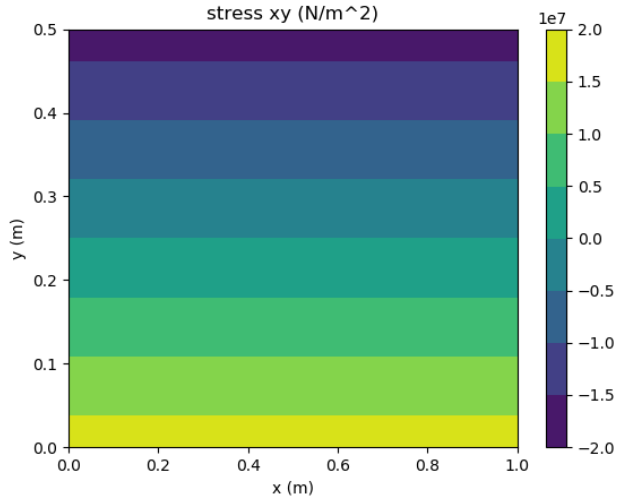

Figure 4.26: stress xy

Results Obtained using abaqus:

The results obtained from abaqus for the homogeneous case is shown below. The displacement obtained are in mm :

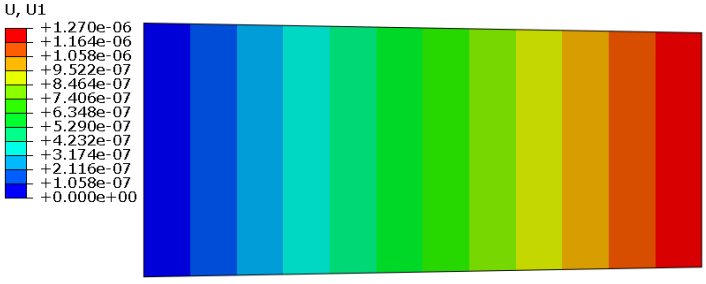

Figure 4.27: Displacement along x direction

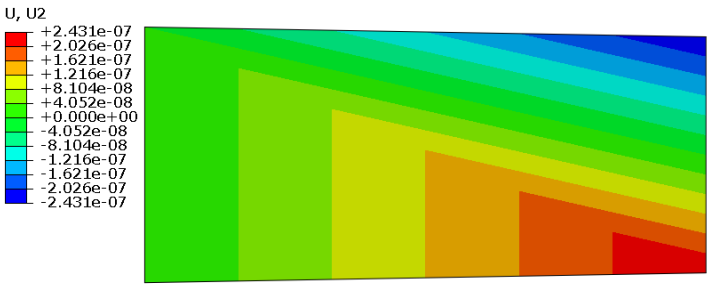

Figure 4.28: Displacement along y direction

The comparisons of the obtained results and the abaqus results are mentioned in the Discussion section.

4.2 Discussion

The cantilever plate assumed is introduced with a external distributed traction/stress in the free end as mentioned in the figure(4.27) below. The obtained results are mentioned above in the result section. The obtained inferences from the analysis are mentioned below:

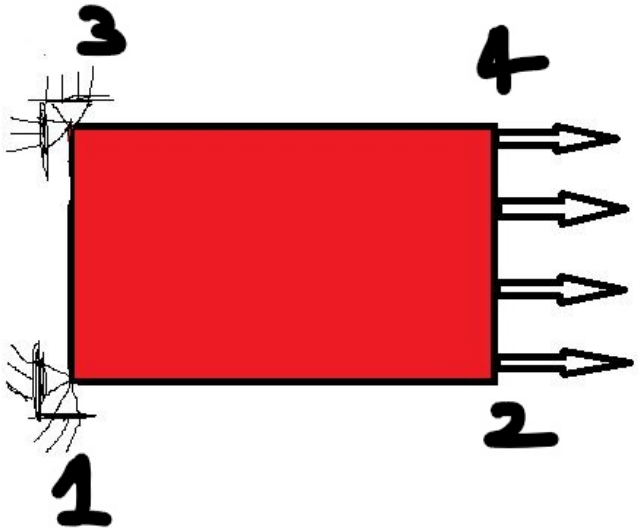

Figure 4.29: Element introduced with the external force

1. Displacement:
The node numbering is mentioned in the above figure (4.29). The displacement of the nodes 2 and 4 is uniform along the x-direction with the same positive magnitude. On the other hand, the displacement of the nodes in the y-direction is in the opposite direction. They move towards each other in the y-direction. The resultant trend of the motions of the nodes are, node 2 and 4 tend to come close to each other. The resultant system will be a tapered plate.

2. Strain
The strain_xx along the x-direction is uniform and is straining in the applied direction, that is the length is increasing, whereas the strain_yy along y-direction is varying along

the x-direction. The strain along the y-direction is higher close to the region where the external force is applied. The strain in the y-direction is decreasing the height of the plate.
The strain_xy is the shear strain acting in the system, which is due to the presence of the shear stress in the system. The neutral plane ,which is the centre plain along the height,does not undergo deformation, that is the shear strain is zero, which can be seen in figure 4.14. The region above the neutral plain experiences a negative(clockwise tilt) shear strain and the region below the neutral plain experiences a positive(anti-clockwise) shear. The tapered shape is obtained due to the presences of this shear strain.
3. Stress
The stress is higher in the fixed region of the plate, that is the stress is higher at the 3rd and 1st node. This can be seen in both stress_xx and stress_yy graph. This also inferences that the necking will happen close to the region which experiences high stress. On the other end, the shear stress is higher in the regions/plain perpendicular to height. The mid/neutral plain does not experience the shear effect, which can be seen in figure 4.17.
4.Comparison of the Verification Results:
The results obtained from the code and from the abaqus are almost the same. The trend obtained in both case are the same(tapered geometry). The displacement obtained in x direction is same in both the case,but the displacement obtained in the y direction has a slight deflection. There is a deflection of $0.43 * 10^{-}7(mm)$ along the y direction.

4.2.1 Conclusion

We can conclude that the FE^2 method is an efficient technique to bring the accuracy in the analysis. The technique also bridges the micro and macro scale. This is a useful technique for analysing the composite materials. The milestones achieved are mentioned below in the table.
The coupling of the program is done, but the expected results from the micro program could not be obtained due to the reasons mentioned in the verification technique.

Proposed	achieved
Formulated quadrilateral meshing technique in micro and macro	yes
Implementations of Numerical technique(NR method)	yes
Implementation of FEM code for Macro Scale	yes
Implementation of FEM code for Micro Scale with isotropic hardening model	yes
Coupling of the Micro and Macro scale program	No

Chapter 5

User Manual

To run the program the following steps should be followed as mentioned below:

5.0.1 Macro code:

The results are obtained using the macro file as mentioned earlier, where the homogeneous techniques were implemented. The below procedure is used to run the program:
step 1 - Open the file named "**PPPSS19_63941_macro_scale_program.py**".
step 2 - Initialize the parameter values of your choice in the Macro input column as shown in the figure-5.1. Enter the value as per the unit mention in the comment.
step 3 - Run the file name "PPPSS19_63941_macro_scale_program.py" in the terminal.

```
########################## Macro Input parameter ##############################
### units as of now in Meters ###
### Geomentrical variables
macro_length=100 * 10**-2 #(100cm)
macro_height=50 * 10**-2  #(50cm)
thickness_plate=500*10**-2  #(500cm)
### Material property variables
Youngs_modulus= 70E9 #N/meter^2
Poissons_ratio=0.30
yield_stress=95E6 #N/meter^2
h=200e6 #Hardening parameter
#Variables to be changed for analsysis
force_mag = 100 # N/m^2 # stress magnitude
```

Figure 5.1: Input parameters

step 4 - The program calculates the required results and will display the graphs as the output to visualize the behaviour of the system.

5.0.2 Micro code:

The strain is given as input to run the micro program.
step 1 - Open the file named "**PPPSS19_63941_micro_elasto_plastic_2.py**".
step 2 - Initialize the macro strain(obtained in the macro program) value as shown in figure-5.6. Enter the value as per the unit mention in the comment.
step 3 - Run the file name "PPPSS19_63941_micro_elasto_plastic_2.py" in the terminal.
step 4 - The program calculates the material parameters like tangent stiffness ,stress etc at each gauss point and prints it as output.
In the coupled case the calculated material parameters in the micro code is passed as

input to the macro program.
coupled code:
coupled code file name: **PPPSS19_63941_micro_macro_ep_coupling.py**
The coupled code does not run due to the convergences error.

5.0.3 Procedure to run the test cases:

Macro program

Macro file name:"**PPPSS19_63941_macro_scale_program.py**"
The test cases for the macro scale are implemented using the macro code by converting it into a displacement driven system. The parameter's value and the lines to be commented and uncommented during the test cases are mentioned below.
step 1:
Lines to be commented:
The following lines(in the code) are first to be commented :
1.line **139**(force_mag)
2. line **217** ($Global\_F\_external[i][0] = force\_mag$)
3. line **240** (Global_displacement)
4. line **241**(Global_F_external_reduced)
Lines to be uncommented:
1.The testing parameters are uncommented(lines **182 to 187**).
2.The boundary conditions for the test cases are uncommented(line **307**).
3.The range of the for loop is changed in line **237**(time_step for loop) - from range(5,105,5) to **range(1,2,1)**. That is the for loop is compiled for 1 iteration.
step 2:
The values of each parameters vary according to the test case performed. So, the parameter values are as mentioned in the file named "**README.pdf**" and can be cross verified from the images shown below.
rigid Body test:
1. The tension_disp is commented and not used for this test case and the parameters used are mention below.

```
180  ############################
181  ######## Testing Parameters ######
182  #For verifying the test cases, follow the instructions in the manual section (to run the program
183  tension_disp=np.array([[0,0],[-4.909*10**-9,-7.751*10**-10],[0,0],[-4.909*10**-9,7.751*10**-10]])
184  Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) #+ tension_disp
185  Global_displacement[2][0]= 4.909*10**-7
186  Global_displacement[3][0]= 7.748*10**-8
187  Global_displacement[4][0]= -4.909*10**-7
188  Global_displacement[5][0]= -7.748*10**-8
189  #### Boundary condition ####
190  #A=np.array([[2],[3],[4],[5]]) The variable A array should be uncommented in the line 293 of the
191  ############################
```

Figure 5.2: Rigid Body rotation parameters

rigid Body with tension/shear/compression:
1. The tension_disp is used for the analysing the rigid body with tension/shear/compression. The parameters used are mentioned below.

```
######## Testing Parameters ######
#For verifying the test cases, follow the instructions in the manual section (to run the program for the test cases)
tension_disp=np.array([[4.909*10**-7,0],[4.909*10**-7,0],[4.909*10**-7,0],[4.909*10**-7,0]])
Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
Global_displacement[2][0]= 4.909*10**-7
Global_displacement[3][0]=  7.748*10**-8 #4.909*10**-7
Global_displacement[4][0]= -4.909*10**-7
Global_displacement[5][0]= -7.748*10**-8
```

Figure 5.3: Rigid Body rotation with tension parameters

```
######## Testing Parameters ######
#For verifying the test cases, follow the instructions in the manual section (to run the program for the test cases)
tension_disp=np.array([[-4.909*10**-7,0],[-4.909*10**-7,0],[-4.909*10**-7,0],[-4.909*10**-7,0]])
Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
Global_displacement[2][0]= 4.909*10**-7
Global_displacement[3][0]=  7.748*10**-8 #4.909*10**-7
Global_displacement[4][0]= -4.909*10**-7
Global_displacement[5][0]= -7.748*10**-8
```

Figure 5.4: Rigid Body rotation with compression parameters

```
######## Testing Parameters ######
#For verifying the test cases, follow the instructions in the manual section (to run the program fo
tension_disp=np.array([[4.909*10**-7,0],[-3.27266*10**-8,0],[3.27266*10**-8,0],[-4.909*10**-7,0]])
Xe=np.array([[0,0],[Le,0],[0,He],[Le,He]]) + tension_disp
Global_displacement[2][0]= 4.909*10**-7
Global_displacement[3][0]=  7.748*10**-8  #-4.909*10**-7
Global_displacement[4][0]= -4.909*10**-7
Global_displacement[5][0]= -7.748*10**-8
```

Figure 5.5: Rigid Body rotation with shear

After setting up the testing parameters the steps of the macro program is used to run the files.

Micro program:

Micro file name: **PPPSS19_63941_micro_elasto_plastic_2.py**
The macro strain obtained during test cases of the macro code,is given as the input parameter in the micro code. The parameter's values are mentioned below. The strain values for all the test cases are the same.

```
############################## Input parameter ######################################
### units as of now in Meters
### Geomentrical variables
micro_length=10*10**-6
micro_height=10*10**-6
thickness_plate=500*10**-2
void_radius=2*10**-6
### Material Properties ###
Youngs_modulus=70E9 #N/meters
Poissons_ratio=0.30
yield_stress=95E6
h=200e6 #Hardening parameter
Strain_macro=np.array([[5.16754*10**-7],[-1.0335*10**-7],[-4.268*10**-7]])
#####################################################################################
```

Figure 5.6: Micro parameters for test cases

Bibliography

[1] Yung-Li Lee, Mark E Barkey, and Hong-Tae Kang. *Metal fatigue analysis handbook: practical problem-solving techniques for computer-aided engineering*. Elsevier, 2011.

[2] C Miehe and A Koch. Computational micro-to-macro transitions of discretized microstructures undergoing small strains. *Archive of Applied Mechanics*, 72(4-5):300–317, 2002.

[3] Olek C Zienkiewicz, Robert L Taylor, and Jian Z Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2005.

[4] Juan C Simo and Thomas JR Hughes. *Computational inelasticity*, volume 7. Springer Science & Business Media, 2006.

[5] Siavouche Nemat-Nasser and Muneo Hori. *Micromechanics: overall properties of heterogeneous materials*. Elsevier, 2013.

[6] V Kouznetsova, WAM Brekelmans, and FPT Baaijens. An approach to micro-macro modeling of heterogeneous materials. *Computational mechanics*, 27(1):37–48, 2001.

[7] Zu-Qing Qu. *Model Order Reduction Techniques with Applications in Finite Element Analysis: With Applications in Finite Element Analysis*. Springer Science & Business Media, 2004.

[8] K Mallikarjuna Rao and U Shrinivasa. A set of pathological tests to validate new finite elements. *Sadhana*, 26(6):549–590, 2001.

[9] Yeong-Bin Yang and Hwa-Thong Chiou. Rigid body motion test for nonlinear analysis with beam elements. *Journal of engineering mechanics*, 113(9):1404–1419, 1987.

Chapter 6

Git log

6.1 Commit messages

account link: https://github.com/Harikeshava/PPP_files/commits/master
commit $e7ba7380231972f7d643e3ecd11b74a04a9feee5$
Merge: 82837402411b81
Author: $hari < harikeshava089@gmail.comm >$
Date: $WedApr102 : 29 : 352020 + 0530$

Final submission
commit $2411b81024f49c35411bf35a76420571fc36515a$
Author: $Harikeshava < 52134291 + Harikeshava@users.noreply.github.com >$
Date: $SatMar2823 : 56 : 052020 + 0530$

Update $README.md$
commit $fd8faa8faf6ffaade9f4811a250eca73910b9575$
Author: $hari < harikeshava089@gmail.comm >$
Date: $WedMar2502 : 08 : 082020 + 0530$

cleaning the code
commit $68bcc83cb5430e2a22d13c6a6196641493fe584e$
Author: $hari < harikeshava089@gmail.comm >$
Date: $WedFeb2616 : 51 : 512020 + 0530$

$macro\_microcoupledfile$
commit $dd85f2f36b220d192cd3ef0afafba297504bee9b$
Author: $hari < harikeshava089@gmail.comm >$
Date: $MonFeb1018 : 25 : 192020 + 0530$

macro and micro program updated
commit $be6ba97f61106c75550282ab83e3d1cd65e7baea$
Author: $hari < harikeshava089@gmail.comm >$
Date: $MonFeb1017 : 55 : 282020 + 0530$

macro and micro program changes updated
commit $a2667c4388b65f5db61c106a2f89d30166076 1b9$

Author: $hari < harikeshava089@gmail.comm >$
Date: $WedFeb523 : 20 : 542020 + 0530$

Quadrilateral meshing technique

commit $7491714a09951625f1b95825fc6e66d418cc876d$
Author: $hari < harikeshava089@gmail.comm >$
Date: $WedFeb523 : 17 : 542020 + 0530$

The macro-scale with 2D quadrilateral Element

commit $c1e920613e42f721fe6d8108d028536ba234bb72$
Author: $hari < harikeshava089@gmail.comm >$
Date: $MonFeb322 : 14 : 552020 + 0530$

The macro-scale comments and micro scale elastic part is developed- 50hr

commit $c330e27c6d716fc81cf5b7c1571e49975431a55b$
Author: $hari < harikeshava089@gmail.comm >$
Date: $TueJan2801 : 19 : 372020 + 0530$

Small corrections made

commit $e73ec9f2da8bd61c7ae816996ce3ff34c55390d7$
Author: $hari < harikeshava089@gmail.comm >$
Date: $TueJan2801 : 05 : 522020 + 0530$

20 hrs of work for micro_stiffness pattern

commit $19bef66cd7bb16f1862102e69636cd0661857eea$
Author: $hari < harikeshava089@gmail.comm >$
Date: $FriJan1003 : 18 : 512020 + 0530$

macro programing 70h

commit $e5839cb7bae844225dfa552358c8f317490ef23d$
Author: $hari < harikeshava089@gmail.comm >$
Date: $SatDec2123 : 21 : 192019 + 0530$

10h

commit $ead510c7ba907bfc9a5a371bdc15aeef93e413b6$
Author: $hari < harikeshava089@gmail.comm >$
Date: $SatDec2123 : 18 : 102019 + 0530$

20h

commit $17a1f4173526bbd60c858d062c511270a72cfb37$

Author: $Harikeshava < 52134291 + Harikeshava@users.noreply.github.com >$
Date: $SatDec2117 : 04 : 542019 + 0000$