AN IMPLEMENTATION OF THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR PROBLEMS IN LINEAR ELASTICITY

A DISSERTATION

Submitted in partial fulfilment of the requirements

for the degree of

MASTER OF TECHNOLOGY

IN

CIVIL ENGINEERING

By

DEVASIS LAISHRAM

(11105769)

Supervisor

Mr. Anshul Garg

School of Civil Engineering

LOVELY PROFESSIONAL UNIVERSITY, PHAGWARA

DECLARATION

I, Devasis Laishram (Reg. No. 11105769), hereby declare that this dissertation entitled "**AN IMPLEMENTATION OF THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR PROBLEMS IN LINEAR ELASTICITY**" submitted in the partial fulfilment of the requirements for the award of the degree of Master of Civil Engineering, in the School of Civil Engineering, Lovely Professional University, Phagwara, is my own work. This matter embodied in this report has not been submitted in part or full to any other university or institute for the award of any degree.

Date: Devasis Laishram

Place: Reg. No. 11105769

CERTIFICATE

Certified that this dissertation entitled "**AN IMPLEMENTATION OF THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR PROBLEMS IN LINEAR ELASTICITY**", submitted individually by the student of School of Civil Engineering, Lovely Professional University, Phagwara, carried out the work under my supervision for the Award of Degree. This report has not been submitted to any other university or institution for the award of any degree.

Mr. Anshul Garg

(Assistant Professor)

SUPERVISOR

Mrs. Mandeep Kaur	Dr. V. Rajesh Kumar
HEAD OF DEPARTMENT	**DEAN**
STRUCTURAL ENGINEERING	**SCHOOL OF CIVIL ENGINEERING**

ACKNOWLEDGEMENT

It is the nature of human existence that one walks that path that lies before him essentially alone, that all his thoughts and experiences exist only within himself and is only ever felt and seen by him and by no one else. But what makes this solitary passage through space and time bearable is the attempt at sharing one's solitary being with others, of attempting to understand others, of attempting to be understood, and of attempting to paint the portrait of his own self with the colours of those he meets on his journey. And for that, I am most grateful to all who were there to offer me companionship in this part of my journey. I thank in particular, my supervisor, Mr. Anshul Garg, whose guidance and understanding was invaluable in completing my dissertation. I thank him for the belief he had in me and for all the precious time he dedicated to me. I thank my university for being the ark that sailed me through this voyage, for being the place where I could grow into the man I am now. I thank everyone I have met, everyone who has taught me something, everyone with whom I could share a brief moment during my time here. Lastly, and with the deepest love in my heart, I thank my loved ones for being the ones in this world I can call my very own.

Devasis Laishram

ABSTRACT

The Finite Element Heterogeneous Multiscale Method (FE-HMM) is a numerical method designed to solve physical problems which are multiscale in nature. It comprises of solving the macro scale problem by adding contributions from the data at the microscopic scale. This dissertation implements the FE-HMM to solve multiscale problems in linear elasticity. The implementation is done in MATLAB and the routines for the various components of the FE-HMM algorithm are given. Lastly, results for the numerical experiments of two model problems are given and the theoretical error estimates are verified.

TABLE OF CONTENTS

ACKNOWLEDGEMENT iv

ABSTRACT v

LIST OF FIGURES viii

LIST OF TABLES ix

CHAPTER 1 INTRODUCTION 1

1.1 Introduction 1

1.2 Outline of the Dissertation 2

CHAPTER 2 LITERATURE REVIEW 3

2.1 The Multiscale Problem 3

2.2 Homogenization 3

2.2 Numerical Homogenization Methods 3

2.2.1 Multiscale Finite Element Method (MsFEM) 3

2.2.2 Representative Volume Elements (RVE) 4

2.2.3 Finite Element Heterogeneous Multiscale Method (FE-HMM) 4

CHAPTER 3 PRELIMINARY THEORY 5

3.1 Homogenization in Linear Elasticity 5

3.2 The Finite Element Method 10

CHAPTER 4 THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR LINEAR ELASTICITY 12

4.1 The FE-HMM Algorithm 12

Macro finite element space 12

4.2 A Priori Error Estimates for the FE-HMM 15

CHAPTER 5 MATLAB IMPLEMENTATION OF THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR PROBLEMS IN LINEAR ELASTICITY 16

5.1 Mesh Data Structure 16

5.2 Macrostiffness Matrix 20

5.3 The Micro Problem 22

5.4 Assembling the Macrostiffness Matrix 35

CHAPTER 6 NUMERICAL EXPERIMENTS 40

CONCLUSION 48

REFERENCES 49

LIST OF FIGURES

Figure 4.1. Example of a macro FE space of triangular and quadrilateral elements with sampling domains at the integration nodes..14

Figure.5.1. An example micro mesh and the corresponding periodic node pairs.........................20

Figure 6.1. Plot of $E(y_2)$ for Problem 1. 42

Figure 6.2. Deformed mesh for Problem 1 with 128 elements..43

Figure 6.3. Deformed mesh with tones showing strain energy density across domain for homogenized solution vs. FE-HMM solution for a mesh with 8192 elements. 44

Figure 6.4. Error between $\mathbf{u}^0$ and $\mathbf{u}^H$ for Problem 1. 45

Figure 6.5. Plot of tensor coefficients for Problem 2...47

Figure 6.6. Deformed mesh for the FE-HMM solution of Problem 2. 47

LIST OF TABLES

Table 6.1. The energy and infinity norms for the fine scale solution. 48

Table 6.2. The energy and infinity norms for the FE-HMM solutions. 48

CHAPTER 1
INTRODUCTION

1.1 Introduction

The dissertation deals with the computation of solutions for a certain class of problems known as multiscale problems. The term "multiscale" refers to a certain class of phenomenon that exhibits specific behaviour in different spatial or temporal scales. The multiscale phenomena can be witnessed not only in the physical world (as a physical phenomenon described by mathematical equations) but also in the psychological constructs of the physical phenomena as we experience them in our everyday. Time is organized in terms of days, months and years, the cause of which is the multiscale dynamics of the solar system in which the earth rotates on its own axis in a smaller scale which at the same time revolving around the sun in a larger scale. Even social structures exhibit this multiscale behaviour consisting of families, towns, states, countries and continents.

From the viewpoint of physics, all matter is made up of atoms comprising of the nucleus and electrons that revolve around it and their structure and dynamics determine the entire behaviour of the material. Therefore, a detailed modelling of the atomistic level would allow us to compute all macroscopic behaviour, such as thermal or electrical conductivity, deformation, fracture, wave propagation etc. Physicists often use these type of atomistic level simulations to understand various material properties at the atomic level. However, attempting to determine the macroscopic behaviour using such an approach would result in an impossibly large system of equations, even for the most powerful supercomputers of today. Engineers and material scientists, on the other hand, whose interest lie solely in the macroscopic behaviour of a material, use macroscopic laws and continuum models such as the laws of thermodynamics, the Navier-Stokes equations, the Navier-Cauchy or elastostatic equations and so on. However, with the advancement of science and the development of technologies like nanotechnology, there arises a need to develop more realistic and efficient modelling techniques that can be used to predict the macroscopic behaviour while taking into account the microscopic or atomic properties.

This general philosophy of modelling that exploits the microscopic information while determining the macroscopic behaviour is termed as "multiscale modelling". Such multiscale models are useful in many areas of physical science, geoscience, environmental science and medical science or biology. In fact, almost all scientific problems involves multiscale characteristics, which may tempt one (at least one specializing in multiscale modelling) to claim that multiscale modelling encompasses almost every aspect of modelling.

The dissertation focuses on a multiscale problems modelled as elliptic multiscale partial differential equations belonging to the class of problems known as “homogenization problems”. By homogenization, we mean expressing the multiscale problem with highly oscillating material coefficients into an “averaged” version. Essentially, homogenization describes the microscopically heterogeneous system in terms of a “homogenized equation” in which the material coefficients are replaced by an “averaged” homogeneous version which is valid at the macroscopic level.

Research in homogenization theory has been going on for the past few decades. However, the main difficulty in homogenization is that it is very difficult to find the explicit expressions of the homogenized coefficients. Assumptions such as periodicity are necessary in order to find explicit expressions of the homogenized coefficients. Multiscale methods, for example the finite element heterogeneous method, on the other hand does not require assumptions of periodicity and hence are applicable to a wider range of problems

1.2 Outline of the Dissertation

The chapters are organized as follows.

- Chapter 2 includes a brief literature review of the various methods developed for solving multiscale problems. The dissertation deals with the Finite Element Heterogeneous Multiscale Method which is once such method (FE-HMM).
- Chapter 3 presents a brief review of the preliminary theories of mathematical homogenization for linear elasticity and the (classical) Finite Element Method which are necessary for understanding the FE-HMM.
- Chapter 4 then describes the Finite Element Heterogeneous Multiscale Method. The numerical algorithm, the macro and micro finite element spaces and their coupling conditions, and a priori error estimates are discussed.
- Chapter 5 presents the work done in the dissertation, which is the MATLAB implementation of the FE-HMM for problems in linear elasticity. The various routines involved in implementing FE-HMM are explained in detail.
- Finally, numerical experiments are conducted using the FE-HMM in Chapter 6.

CHAPTER 2
LITERATURE REVIEW

2.1 The Multiscale Problem

Multiscale problems can be modelled using a general class of PDEs written as $L_\varepsilon(u^\varepsilon) = f$, defined on a domain $\Omega \in \mathbb{R}^d, 1 \leq d \leq 3$, and having highly oscillating coefficients that depend on a small parameter $\varepsilon > 0$ which represents the microscopic scale length of the material heterogeneity. The micro solution of these PDEs are given by $u^\varepsilon : \Omega \longrightarrow \mathbb{R}$ which are defined over the entire domain.

The purpose of this chapter is to present a brief glimpse of the various methods are that are used for finding approximate solutions to the solution u^ε of the multiscale problem are reviewed below.

2.2 Homogenization

The general idea behind mathematical homogenization theory is to find a limit denoted by u^0 to which the micro solutions converge as $\varepsilon \to 0$. The macro limit of the solution u^0 is called the homogenized solution if it is also the solution of a homogenized PDE $L_0(u^0) = f$ therefore being independent of the micro scale ε. Homogenization theory deals with finding the analytical homogenized solution u^0 of multiscale PDEs described above. The texts [12], [19] give the necessary introduction into the mathematical theory of homogenization in linear elasticity. A more detailed review of homogenization theory is done in Chapter 3.

2.2 Numerical Homogenization Methods

The homogenized form of a multiscale PDE takes a form $L_0(u^0) = f$ with homogenized coefficients $a^0(x)$. This homogenized PDE can be solved using classical numerical methods. However, in order to apply classical numerical methods, an explicit expression for the homogenized coefficients $a^0(x)$ is necessary. Such an explicit expression for $a^0(x)$ is usually not obtainable. This difficulty led to the development of numerical homogenization methods, some of which are listed below.

2.2.1 Multiscale Finite Element Method (MsFEM)

The MsFEM is based on the ideas proposed in [9] in which the finite element space is modified in order to account for the microstructure of the problem. Special multiscale basis functions are used for this purpose. This method was extend and a detailed description of the method can be found in [16].

The method is aimed at obtaining the full sine scale solution and is therefore very costly computationally. Two other methods are listed below which are much cheaper computationally, but requires certain restrictions such as scale separation and self-similarity of the tensor involved in the multiscale PDE.

2.2.2 Representative Volume Elements (RVE)

The RVE is a well-known method for solving multiscale problems in engineering and structural mechanics. It relies on representative volume elements (RVEs) which simulate the microstructure of the composite. The simulation is then "upscaled" to obtain the effective macroscopic properties.

2.2.3 Finite Element Heterogeneous Multiscale Method (FE-HMM)

The Heterogeneous Multiscale Method (HMM) is a methodology for problems that exhibit scale separation. A detailed description of the HMM can be found in [14], [15].

The HMM consists of the following steps:

- Step 1: A macroscopic model is constructed using data obtained from simulations at the microscopic level.
- Step 2: The macroscopic solution is obtained for the macro model constructed in Step 1.
- Step 3: The fine-scale information is recovered using a post-processing technique.

In Steps 1 and 2, if the Finite Element Method is used to solve the micro and macro models, then the method is termed as the Finite Element Heterogeneous Multiscale Method (FE-HMM)

A detailed description of the FE-HMM can be found in the papers [1]-[6]. The FE-HMM is described in more detail in Chapter 3 for problems in linear elasticity.

CHAPTER 3
PRELIMINARY THEORY

This chapter presents a brief review of the preliminary theory necessary for the Finite Element Heterogeneous Multiscale Method. Section 3.1 gives a brief review of homogenization theory in linear elasticity. Section 3.2 reviews the classical (single-scale) Finite Element Method. The main purpose of this chapter is to set the notations which shall be used throughout the dissertation.

3.1 Homogenization in Linear Elasticity

This section gives a brief review of homogenization and describe the asymptotic behaviour of the linear elasticity system as $\varepsilon \to 0$. A detail description of the theory and proofs of the theorems can be found in [12], [19].

Noting that the Einstein summation convention is used throughout the thesis, we set the following notation.

Notation 3.1.1. *If* $B = \left(b_{ijkh}\right)_{1\le i,j,k,h\le N}$ *is a fourth-order tensor, and* $m = \left(m_{ij}\right)_{1\le i,j\le N}$ $m' = \left(m'_{ij}\right)_{1\le i,j\le N}$ *are second order tensors, represented by square matrices, we set*

$$\begin{cases} B\,m = \left((B\,m)_{i,j}\right)_{1\le i,j,k,h\le N} = \left(\left(b_{ijkh}m_{kh}\right)_{ij}\right)_{1\le i,j,k,h\le N} \\ B\,m\,m' = b_{ijkh}m_{kh}m'_{ij} \\ |m| = \left(\displaystyle\sum_{i,j=1}^{N} m_{ij}^2\right)^{\frac{1}{2}} \end{cases} \tag{3.1.1}$$

Definition 3.1.2. *Let* $\alpha, \beta \in \mathbb{R}$, *such that* $0 < \alpha < \beta$ *and let* $\mathcal{O} \in \mathbb{R}^N$ *be an open set. Then we denote by* $M_e(\alpha, \beta, \mathcal{O})$ *the set of tensors* $B = \left(b_{ijkh}\right)_{1\le i,j,k,h\le N}$ *such that*

$$\begin{cases} i) & b_{ijkh} \in L^\infty(\mathcal{O}), & \text{for any } i,j,k,h = 1,\dots,N \\ ii) & b_{ijkh} = b_{ijhk} = b_{jikh} = b_{khij}, & \text{for any } i,j,k,h = 1,\dots,N \\ iii) & \alpha|m|^2 \le Bmm, & \text{for any symmetric matrix m} \\ iv) & |Bm| \le \beta|m|, & \text{for any matrix matrix m} \end{cases} \tag{3.1.2}$$

a.e. on $\mathcal{O}$.

We define the linearized strain tensor e defined by

$$e(\varphi) = \left(e_{ij}(\varphi)\right)_{1\leq i,j\leq N}, e_{ij}(\varphi) = \frac{1}{2}\left(\frac{\partial \varphi_i}{\partial x_j} + \frac{\partial \varphi_j}{\partial x_i}\right), \qquad \forall\, i,j = 1, \dots, N \tag{3.1.3}$$

for any $\varphi = (\varphi_1, \dots, \varphi_N)$. Then, Hooke's law gives us the strain tensor defined as,

$$\sigma_{ij} = b_{ijkh} e_{kh}(\varphi) = b_{ijkh} \frac{\partial \varphi_k}{\partial x_h}, \tag{3.1.4}$$

where we have used the symmetry property (3.1.2) (ii). We now introduce the reference cell

$$Y =]0, \ell_1[\times \dots \times]0, \ell_N[,$$

where $\ell_1, \dots \ell_N$ are positive numbers.

Definition 3.1.3. *A function f define a.e. on $\mathbb{R}^N$ is said to be Y-$periodic$ iff*

$$f(x + k\ell_i e_i) = f(x) \;\; \text{a.e. on } \mathbb{R}^N, \qquad \forall\, k \in Z, \qquad \forall i = 1, \dots, N,$$

where $\{e_1, \dots, e_N\}$ is the canonical basis of $\mathbb{R}^N$.

Problem 3.1.4. *Let $\Omega \subset \mathbb{R}^N, N = 1,2,3$ denote an open, non-empty, bounded and connected Lipschitz domain with boundary $\partial\Omega$, where the Dirichlet boundary conditions are prescribed on $\Gamma_D \subset \partial\Omega$ and the Neumann boundary conditions on $\Gamma_N = \partial\Omega \setminus \Gamma_D$. If $\overline{\Omega}$ (the closure of Ω) is a region occupied by a linearly elastic material which is in static equilibrium under the action of the body forces $\mathbf{f} \in L^2(\Omega)^N$, surface traction $\mathbf{g} \in H^{-\frac{1}{2}}(\Gamma_N)^N$, find $\mathbf{u}^\varepsilon \in H_0^1(\Omega)^N$ such that*

$$\begin{cases} -\dfrac{\partial}{\partial x_j}\left(a_{ijkh}^\varepsilon \dfrac{\partial u_k^\varepsilon}{\partial x_h}\right) = f_i & \text{in } \Omega, \\ \mathbf{u}^\varepsilon = 0 & \text{on } \Gamma_D, \\ a_{ijkh}^\varepsilon \dfrac{\partial u_k^\varepsilon}{\partial x_h} n_j = g_i & \text{on } \Gamma_N, \end{cases} \tag{3.1.5}$$

for $i = 1, \dots, N$, where ε represents the parameter characterizing the heterogeneity of the material, $\mathbf{n} = (n_1, \dots, n_d)$ is the unit outward normal to $\partial\Omega$ and $A^\varepsilon(x)$ is a fourth order symmetric tensor such that $A^\varepsilon(x) = \left(a_{ijkh}^\varepsilon(x)\right)_{1\leq i,j,k,h\leq N} \in M_e(\alpha, \beta, \Omega)$.

Existence and uniqueness.

We define the space $\boldsymbol{\mathcal{V}}$ of admissible displacements as

$$\boldsymbol{\mathcal{V}} = H_0^1(\Omega)^N := \{\mathbf{v} \in H^1(\Omega)^N \,;\; \mathbf{v} = 0 \text{ on } \Gamma_D \}. \tag{3.1.6}$$

Then, due to the *Poincaré Inequality*, $\boldsymbol{\mathcal{V}}$ can be equipped with the norm

$$\|\mathbf{v}\|_{\mathcal{V}} = \left(\sum_{i,j=1}^{N} \|\nabla v_i\|_{L^2(\Omega)^N}^2 \right)^{\frac{1}{2}}, \tag{3.1.7}$$

for $\mathbf{v} = (v_1, \dots, v_N) \in \boldsymbol{\mathcal{V}}$, and is a Hilbert space for the scalar product

$$(\mathbf{u}, \mathbf{v})_{\mathcal{V}} = \sum_{i=1}^{N} (\nabla u_i, \nabla v_i)_{L^2(\Omega)}.$$

The variational form of (3.1.5) can then be written as follows.

$$\begin{cases} \text{Find } \mathbf{u}^\varepsilon \in \boldsymbol{\mathcal{V}} \text{ such that} \\ B_\varepsilon(\mathbf{u}^\varepsilon, \mathbf{v}) := \displaystyle\int_\Omega A^\varepsilon(x)\, e(\mathbf{u}^\varepsilon)\, e(\mathbf{v})\, dx \\ \qquad\qquad = \displaystyle\int_\Omega \mathbf{f} \cdot \mathbf{v}\, dx + \int_{\Gamma_N} \mathbf{g} \cdot \mathbf{v}\, ds = : F(\mathbf{v}), \qquad \forall\, \mathbf{v} \in \boldsymbol{\mathcal{V}} \end{cases} \tag{3.1.8}$$

The bilinear form in (3.1.8) can be shown to be $\boldsymbol{\mathcal{V}}$-coercive by defining a norm on $\boldsymbol{\mathcal{V}}$

$$|||\mathbf{v}||| = \int_\Omega |e(\mathbf{v})|^2 dx \tag{3.1.9}$$

which is equivalent to (3.1.7). This can be done using the *first Korn inequality*

$$\|\mathbf{v}\|_{\mathcal{V}} \leq c_K \left(\int_\Omega |e(\mathbf{v})|^2 dx \right)^{\frac{1}{2}} \tag{3.1.10}$$

which holds for all $\mathbf{v} \in \boldsymbol{\mathcal{V}}$. And since the bilinear form in (3.1.8) is bounded on $\boldsymbol{\mathcal{V}}$ as $A^\varepsilon(x) \in M_e(\alpha, \beta, \Omega)$, we can apply the Lax-Milgram theorem to guarantee the existence and uniqueness of the solution of (3.1.5) and get the estimate

$$\|\mathbf{u}^\varepsilon\|_{\mathcal{V}} \leq C \left(\|\mathbf{f}\|_{L^2(\Omega)^N} + \|\mathbf{g}\|_{H^{-\frac{1}{2}}(\Gamma_N)^N} \right). \tag{3.1.11}$$

In order to derive the homogenization results for the problem (3.1.5), we set

$$a^{\varepsilon}_{ijkh}(x) = a_{ijkh}\left(\frac{x}{\varepsilon}\right) \text{ a.e. on } \mathbb{R}^N, \qquad \forall\, i,j,k,h = 1,\dots,N \tag{3.1.12}$$

and

$$A^{\varepsilon}(x) = A\left(\frac{x}{\varepsilon}\right) = \left(a^{\varepsilon}_{ijkh}(x)\right)_{1\le i,j,k,h\le N} \text{ a.e. on } \mathbb{R}^N \tag{3.1.13}$$

where $A = A(y)$ is a fourth-order tensor such that

$$\begin{cases} a_{ijkh} \text{ is } Y\text{-}periodic, \qquad \forall\, i,j,k,h = 1,\dots,N & (3.1.14)\\ A = \left(a^{\varepsilon}_{ijkh}\right)_{1\le i,j,k,h\le N} \in M_e(\alpha,\beta,Y). & (3.1.15)\end{cases}$$

Auxiliary periodic problems.

In order to derive the homogenized equation for the linear elasticity system, we introduce corrector functions which are solutions to a family periodic boundary value problems posed on the reference cell Y.

First we define, for any $\ell, m \in \{1,\dots,N\}$, the vector valued function $\boldsymbol{P}^{\ell m}(y) = \left(P^{\ell m}_k(y)\right)_{1\le k\le N}$ by

$$P^{\ell m}_k(y) = y_m \delta_{k\ell} \qquad k = 1,\dots,N, \tag{3.1.16}$$

where $\delta_{k\ell}$ is the Kronecker symbol. Then, we introduce the vector valued function $\boldsymbol{\chi}^{\ell m} = \left(\chi^{\ell m}_k\right)_{1\le k\le N}$, a solution of the system

$$\begin{cases} -\dfrac{\partial}{\partial x_j}\left(a_{ijkh}\dfrac{\partial\left(\chi^{\ell m}_k - P^{\ell m}_k\right)}{\partial x_h}\right) = 0 \qquad \text{in Y}, i = 1,\dots,N \\ \chi^{\ell m}_k \quad Y\text{-}periodic \\ \mathcal{M}_Y\left(\chi^{\ell m}_k\right) = 0 \end{cases} \tag{3.1.17}$$

where $\mathcal{M}_Y(\gamma)$ denotes the mean value of an integrable function γ, defined by $\mathcal{M}_Y = \frac{1}{|Y|}\int_Y \gamma(y)dy$.

The variational form of (3.1.16) is as follows.

$$\begin{cases} \text{Find } \boldsymbol{\chi}^{\ell m} \in \boldsymbol{W}_{per}(Y) \text{ such that} \\ \displaystyle\int_Y A(y)\, e\left(\boldsymbol{\chi}^{\ell m}\right) e(\mathbf{v})dy = \int_Y A(y)\, e\left(\boldsymbol{P}^{\ell m}\right) e(\mathbf{v})dy\ , \qquad \forall\, \mathbf{v} \in \boldsymbol{W}_{per}(Y) \end{cases} \tag{3.1.18}$$

where the space $\boldsymbol{W}_{per}(Y)$ is defined as

$$\boldsymbol{W}_{per}(Y) = \left\{\mathbf{v} \in H^1_{per}(Y)^N;\ \mathcal{M}_Y(v_i) = 0, \qquad i = 1, \dots, N\right\} \tag{3.1.19}$$

and $H^1_{per}(Y)$ is the closure of $C^\infty_{per}(Y) \subset C^\infty(\mathbb{R}^N)$, the subset of Y-*periodic* functions, for the H^1-norm. Using the *Poincaré-Wirtinger Inequality* and the *Korn inequality* for the periodic case, the existence and uniqueness for the problem (3.1.18) is given by the Lax-Milgram theorem.

The following theorem gives the homogenized problem for (3.1.5).

Theorem 3.1.5. *Let* $\mathbf{f} \in \boldsymbol{\mathcal{V}}'$, $\boldsymbol{g} \in H^{-\frac{1}{2}}(\Gamma_N)^N$ *and* A^ε *be given by (3.1.12) – (3.1.15). If* $\mathbf{u}^\varepsilon \in \boldsymbol{\mathcal{V}}$ *is the solution of (3.1.5), then*

$$\begin{cases} i) & \mathbf{u}^\varepsilon \rightharpoonup \mathbf{u}^0 \ weakly\ in\ \boldsymbol{\mathcal{V}}. \\ ii) & A^\varepsilon\, e(\mathbf{u}^\varepsilon) \rightharpoonup A^0\, e(\mathbf{u}^0)\ weakly\ in\ L^2(\Omega)^{N\times N} \end{cases}$$

where $\mathbf{u}^0 = (u_1^0, \dots, u_N^0)$ *is the unique solution in* $\boldsymbol{\mathcal{V}}$ *of the homogenized system*

$$\begin{cases} -\dfrac{\partial}{\partial x_j}\left(a^0_{ijkh}\dfrac{\partial u_k^0}{\partial x_h}\right) = f_i & \text{in } \Omega\,, \\ \mathbf{u}^0 = 0 & \text{on } \Gamma_D, \\ a^0_{ijkh}\dfrac{\partial u_k^0}{\partial x_h}\, n_j = g_i & \text{on } \Gamma_N, \end{cases} \tag{3.1.20}$$

for $i = 1, \dots, N$. *The tensor* $A^0 = \left(a^0_{ijkh}\right)_{1\le i,j,k,h\le N}$ *is called the homogenized tensor and is constant, verifies the symmetry property (3.1.2) (ii) and satisfies the coerciveness condition for some constant* α^0. *The elements of* A^0 *are given*

$$a^0_{ijkh} = \frac{1}{|Y|}\int_Y a_{ijkh}(y)\, dy - \frac{1}{|Y|}\int_Y a_{ij\ell m}(y)\frac{\partial \chi_\ell^{kh}}{\partial y_m}(y)\, dy. \tag{3.1.21}$$

The variational formulation for the homogenized problem (3.1.20) is given by

$$\begin{cases} \text{Find } \mathbf{u}^\varepsilon \in \boldsymbol{\mathcal{V}} \text{ such that} \\ B_0(\mathbf{u}^\varepsilon, \mathbf{v}) := \displaystyle\int_\Omega A^\varepsilon(x)\, e(\mathbf{u}^\varepsilon)\, e(\mathbf{v})\, dx = F(\mathbf{v}), \qquad \forall\, \mathbf{v} \in \boldsymbol{\mathcal{V}} \end{cases} \tag{3.1.22}$$

3.2 The Finite Element Method

As mentioned in the introduction, the FE-HMM uses the finite element method to solve the macro and micro problems simulated in the HMM methodology. We therefore present a short review of the finite element method and mention the finite element error estimates which shall also be used in a priori error estimates for the FE-HMM.

For a non-oscillatory (single-scale) version of problem (3.1.1) i.e. with a tensor $A(x) = a_{ijkh}(x)$ in place of a_{ijkh}^{ε}, we write the variational formulation as: Find $\mathbf{u} \in \boldsymbol{\mathcal{V}}$ such that

$$\begin{cases} \text{Find } \mathbf{u} \in \boldsymbol{\mathcal{V}} \text{ such that} \\ B_{fem}(\mathbf{u}, \mathbf{v}) := \int_{\Omega} A(x)\, e(\mathbf{u})\, e(\mathbf{v})\, dx = F(\mathbf{v}), \qquad \forall\, \mathbf{v} \in \boldsymbol{\mathcal{V}} \end{cases} \tag{3.2.1}$$

The existence and uniqueness of the solution for (3.2.1) is guaranteed in the same way as earlier by the Lax-Milgram theorem.

The discretization of the variational form is done in the standard Galerkin framework. Let the domain be partitioned by a triangulation $\mathcal{T}_H$ consisting of either simplicial or quadrilateral elements with diameter H_K and with $H = \max_{K \in \mathcal{T}_H} H_K$ as the maximum diameter of the triangulation. Further, the triangular is assumed to be admissible and shape regular, i.e.,

- $\bigcup_{K \in \mathcal{T}_h} K = \overline{\Omega}$ and the intersection of two elements is either empty, a single shared vertex or single common edge/face (admissible),
- $\exists\, \rho > 0$ such that $H_K / d_K \leq \kappa$, where $d_K = \sup\{d : \text{there is a circle of diameter } d \text{ in } K\}$ (shape regular).

The finite dimensional subspace of $H_0^1(\Omega)^N$ for the partition is defined by

$$\boldsymbol{\mathcal{V}}^p(\Omega, \mathcal{T}_H) = \{\mathbf{v}^H \in H_0^1(\Omega)^N \,;\, \mathbf{v}^H|_K \in \mathcal{R}^p(K)^N, \forall K \in \mathcal{T}_H\} \tag{3.2.2}$$

Where $\mathcal{R}^p(K)$ is either of the spaces $\mathcal{P}^p(K)$ or $\mathcal{Q}^p(K)$ of polynomials for simplicial elements and quadrilateral elements respectively, with degree p in each variable.

The discretized problem is then, as follows.

$$\begin{cases} \text{Find } \mathbf{u}^H \in \boldsymbol{\mathcal{V}}^p(\Omega, \mathcal{T}_H) \text{ such that} \\ B_{fem}(\mathbf{u}^H, \mathbf{v}^H) = F(\mathbf{v}^H), \qquad \forall\, \mathbf{v}^H \in \boldsymbol{\mathcal{V}}^p(\Omega, \mathcal{T}_H). \end{cases} \tag{3.2.3}$$

A priori error estimates for the solution $\mathbf{u}^H$ can be derived from Céa's Lemma which states that

$$\|\mathbf{u}-\mathbf{u}^H\|_{H^1(\Omega)^N} \leq \frac{C}{\lambda}\, \inf_{\mathbf{v}^H \in \boldsymbol{\mathcal{V}}^p(\Omega,\mathcal{T}_H)} \|\mathbf{u}-\mathbf{v}^H\|_{H^1(\Omega)^N}, \tag{3.2.4}$$

which bounds the error $\mathbf{u}-\mathbf{u}^H$ by the interpolation error, implying that it depends on the degree p of the space $\boldsymbol{\mathcal{V}}^p(\Omega,\mathcal{T}_H)$.

If it is provided that the solution $\mathbf{u}$ has regularity $\mathbf{u} \in H^{r+1}(\Omega)^N$ then there exists a constant C such that the errors in the H^1-norm and L^2-norm are given by

$$\|\mathbf{u}-\mathbf{u}^H\|_{H^1(\Omega)^N} \leq CH^p\, |\mathbf{u}|_{H^{r+1}(\Omega)^N} \tag{3.2.5}$$

and

$$\|\mathbf{u}-\mathbf{u}^H\|_{L^2(\Omega)^N} \leq CH^{p+1}\, |\mathbf{u}|_{H^{r+1}(\Omega)^N} \tag{3.2.6}$$

where $|\mathbf{u}|_{H^{r+1}(\Omega)^N}$ is the semi-norm given by

$$|\mathbf{u}|_{H^{r+1}(\Omega)^N} = \left(\sum_{i=1}^{N} \|\nabla u_i\|^2_{L^{r+1}(\Omega)^N}\right)^{\frac{1}{2}}.$$

Quadrature formula. The quadrature formula for a finite element is defined using a C^1-diffeomorphism, F_K from each element $K \in \mathcal{T}_H$ to the reference element $\widehat{K}$ such that $K = F_K(\widehat{K})$. If the quadrature formula on the reference element is given by $\{\hat{x}_\ell, \widehat{\omega}_\ell\}_{\ell=1}^{\mathcal{L}}$, then the corresponding quadrature formula on an element $K \in \mathcal{T}_H$ is defined as $\{x_\ell, \omega_\ell\}_{\ell=1}^{\mathcal{L}}$, where x_ℓ are the quadrature points given by $x_\ell = F_K(\hat{x}_\ell), \ell = 1, \dots, \mathcal{L}$ and ω_ℓ are the quadrature weights given by $\omega_\ell = \widehat{\omega}_\ell |\det(\partial F_K)|, \ell = 1, \dots, \mathcal{L}$. We make the following assumptions in the quadrature formula:

(Q1) $\sum_{\ell=1}^{\mathcal{L}} \widehat{\omega}_\ell |\nabla \hat{q}(\hat{x}_\ell)|^2 \geq \hat{\lambda} \|\nabla \hat{q}\|^2_{L^2(\widehat{\omega}_\ell)}$ for $\widehat{\omega}_\ell > 0, \ell = 1, \dots, \mathcal{L}$, $\hat{\lambda} > 0$ and $\forall\, \hat{q}(\hat{x}) \in \mathcal{R}^p(\widehat{K})$

(Q2) $\int_{\widehat{K}} \hat{q}(\hat{x})d\hat{x} = \sum_{\ell=1}^{\mathcal{L}} \widehat{\omega}_\ell \hat{q}(\hat{x}_\ell)$, $\forall \hat{q}(\hat{x}) \in \mathcal{R}^\sigma(\widehat{K})$ where $\sigma = \max(2p-2, p)$ for simplicial elements and $\sigma = \max(2p-1, p+1)$ for rectangular elements.

For the numerical experiments conducted in this dissertation, we adopt the symmetric Gaussian quadrature rule for simplicial 2D (triangular) elements derived by Dunavant [13] (see also, [17]), as the problems considered are 2-dimensional and triangular Lagrange elements are used for the finite element discretization.

CHAPTER 4
THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR LINEAR ELASTICITY

This chapter presents in detail, the Finite Element Heterogeneous Multiscale Method (FE-HMM) for linear elasticity. As described in Chapter 2 the Heterogeneous Multiscale Method (HMM) solves the multiscale linear elasticity system using a macro solver coupled with a micro solver that estimates the localized microscopic data. Choosing the Finite Element Method as both macro and micro solvers gives the numerical algorithm for the FE-HMM.

4.1 The FE-HMM Algorithm

We consider again the problem multiscale problem (3.1.5). The FE-HMM gives an approximate solution without requiring to compute the homogenized tensor A^0. The algorithm comprises of the following components.

Macro finite element space. The macro space is defined as

$$\boldsymbol{\mathcal{V}}_D^p(\Omega, \mathcal{T}_H) = \{\mathbf{v}^H \in H_0^1(\Omega)^N \; ; \; \mathbf{v}^H|_K \in \mathcal{R}^p(K)^N, \;\; \forall K \in \mathcal{T}_H\} \tag{4.1.1}$$

with macro elements $K \in \mathcal{T}_H$ and admissibility and shape regularity are assumed on $\mathcal{T}_H$. The macro mesh size H is allowed to be much larger than the scale length ε.

The following components within each macro element $K \in \mathcal{T}_H$ are used in the FE-HMM algorithm:

- integration nodes $x_{\ell,K} \in K$,
- sampling domains $K_d(x_{\ell,K}) = x_{\ell,K} + \delta I$ around each $x_{\ell,K}$, where $I = \left(-\frac{1}{2}, \frac{1}{2}\right)^N$ and $\delta \geq \varepsilon$,
- quadrature weights $\omega_{\ell,K}$.

The quadrature formula for the macro finite element is given by **(Q1)** and **(Q2)**, presented in Section (3.2). Also, as mentioned earlier, we shall use the symmetric Gaussian quadrature rule from [13].

Macro bilinear form. The macro bilinear form for the FE-HMM is obtained by modifying the bilinear form $B_{fem}(\cdot,\cdot)$ given in (3.2.1) using micro functions obtained from the sampling domains. It is defined by

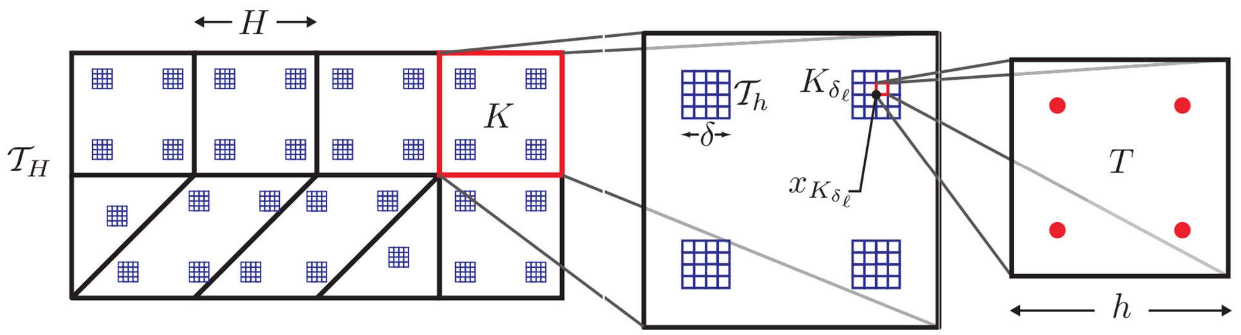
$$B_H(\mathbf{v}^H, \mathbf{w}^H) := \sum_{K \in \mathcal{T}_H} \sum_{\ell=1}^{\mathcal{L}} \frac{\omega_{\ell,K}}{|K_\delta(x_{\ell,K})|} \int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e(\mathbf{v}^h_{\ell,K_\delta})\, e(\mathbf{w}^h_{\ell,K_\delta}) dx, \tag{4.1.2}$$

where $\mathbf{v}^h_{\ell,K_\delta}$ and $\mathbf{w}^h_{\ell,K_\delta}$ are micro functions defined on sampling domains $K_\delta(x_{\ell,K})$ and are obtained by solving the micro problem given in (4.1.3). Since the integrals are defined on $K_\delta(x_{\ell,K})$ instead of K, the quadrature weights are divided by a factor of $|K_\delta(x_{\ell,K})|$ which is the measure of the sampling domain $K_\delta(x_{\ell,K})$.

The variational form for the macro-problem is written as:

$$\begin{cases} \text{Find } \mathbf{u}^H \in \boldsymbol{\mathcal{V}}^p(\Omega, \mathcal{T}_H) \text{ such that} \\ B_H(\mathbf{u}^H, \mathbf{v}^H) = F(\mathbf{v}^H), \qquad \forall\, \mathbf{v}^H \in \boldsymbol{\mathcal{V}}^p(\Omega, \mathcal{T}_H). \end{cases} \tag{4.1.3}$$

The coupling between macro and micro problems is shown below in Figure 4.1.



(a) macro-problem (b) micro-problem (c) quadrature for the micro-problem

Figure 4.1. Example of a macro FE space of triangular and quadrilateral elements with sampling domains at the integration nodes (a). Each sampling domain is discretized with a micro mesh as shown in (b). Quadrature points for the micro problem are depicted in (c). Picture taken from [3]

Micro problem. We let $\mathcal{T}_h$ be the finite element discretization for the micro problem. Then, the microstructure information is added to the macro stiffness matrix for each macro element K by computing the micro functions $\mathbf{v}^h_{\ell,K_\delta}$ (and $\mathbf{w}^h_{\ell,K_\delta}$) on the sampling domain $K_\delta(x_{\ell,K})$, $\ell = 1, \dots, \mathcal{L}$ such that $\left(\mathbf{v}^h_{\ell,K_\delta} - \mathbf{v}^H_{lin,(x_{\ell,K})}\right) \in \boldsymbol{\mathcal{S}}^q(K_\delta(x_{\ell,K}), \mathcal{T}_h)$ and

$$\int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e(\mathbf{v}^h_{\ell,K_\delta})\, e(\mathbf{z}^h_{\ell,K_\delta}) dx = 0, \qquad \forall \mathbf{z}^h \in \boldsymbol{\mathcal{S}}^q(K_\delta(x_{\ell,K}), \mathcal{T}_h) \tag{4.1.4}$$

where

$$\mathbf{v}^H_{lin,(x_{\ell,K})}(x) = \mathbf{v}^H(x_{\ell,K}) + e\left(\mathbf{v}^H(x_{\ell,K})\right)(x - x_{\ell,K}) \tag{4.1.5}$$

is the linearization of the macro function $\mathbf{v}^H$ at the quadrature nodes $x_{\ell,K}$ and the micro finite element space is given by

$$\boldsymbol{\mathcal{S}}^q(K_\delta(x_{\ell,K}), \mathcal{T}_h) = \left\{\mathbf{z}^h \in \boldsymbol{W}\left(K_\delta(x_{\ell,K})\right);\ \boldsymbol{z}^h|_T, \in \mathcal{R}^q(Q)^N, Q \in \mathcal{T}_h\right\} \tag{4.1.6}$$

where the choice of the Sobolev space $\boldsymbol{W}\left(K_\delta(x_{\ell,K})\right)$ determines the micro boundary conditions and the micro-macro coupling conditions.

Coupling Conditions. The coupling conditions considered in the dissertation are

- *Periodic* coupling, with

$$\boldsymbol{W}\left(K_\delta(x_{\ell,K})\right) = \boldsymbol{W}_{per}\left(K_\delta(x_{\ell,K})\right) \tag{4.1.7}$$

  and the micro space $\boldsymbol{\mathcal{S}}^q(K_\delta, \mathcal{T}_h)$ will be denoted by $\boldsymbol{\mathcal{S}}^q_{per}(K_\delta, \mathcal{T}_h)$.

- *Dirichlet* coupling, with

$$\boldsymbol{W}\left(K_\delta(x_{\ell,K})\right) = \boldsymbol{\mathcal{V}}\left(K_\delta(x_{\ell,K})\right) \tag{4.1.8}$$

  and the micro space $\boldsymbol{\mathcal{S}}^q(K_\delta, \mathcal{T}_h)$ will be denoted by $\boldsymbol{\mathcal{S}}^q_{dir}(K_\delta, \mathcal{T}_h)$.

The Sobolev spaces $\boldsymbol{\mathcal{V}}\left(K_\delta(x_{\ell,K})\right)$ and $\boldsymbol{W}_{per}\left(K_\delta(x_{\ell,K})\right)$ are defined in (3.1.6) and (3.1.19) respectively.

4.2 A Priori Error Estimates for the FE-HMM

The error estimate for the FE-HMM can be decomposed into three parts, viz. macro error, micro error and modelling error. We have,

$$\|\mathbf{u}^0 - \mathbf{u}^H\| \leq e_{MAC} + e_{MIC} + e_{MOD}.$$

We assume that the solutions $\boldsymbol{\chi}^{\ell m}$ for the equation (3.1.17) satisfy the following condition

$$\begin{cases} \varepsilon\boldsymbol{\chi}^{\ell m} \in H^{q+1}\left(K_\delta(x_{\ell,K})\right)^N \text{ and} \\ \left\|D^\alpha(\varepsilon\boldsymbol{\chi}^{\ell m})\right\|_{L^\infty\left(K_\delta(x_{\ell,K})\right)} \leq C\varepsilon^{-|\alpha|+1}, \text{for } \alpha \leq q+1, \ell, m = 1, \dots, N. \end{cases} \tag{4.2.1}$$

Theorem 4.2.1. *Let* $\mathbf{u}^0$ *and* $\mathbf{u}^H$ *are solutions of (3.1.22) and (4.1.3) respectively and assume that condition (4.2.1) holds. If* $u^0 \in H^{r+1}(\Omega)^N$, *for some* $r > 0$, *then*

$$\begin{aligned} \|\mathbf{u}^0 - \mathbf{u}^H\|_{H^1(\Omega)^N} &\leq C\left(H^s + \left(\frac{h}{\varepsilon}\right)^{2q} + e_{MOD}\right), \\ \|\mathbf{u}^0 - \mathbf{u}^H\|_{L^2(\Omega)^N} &\leq C\left(H^{s+1} + \left(\frac{h}{\varepsilon}\right)^{2q} + e_{MOD}\right), \qquad s = min(r,p). \end{aligned} \tag{4.2.2}$$

Further, if A^ε *satisfies (3.1.12) – (3.1.15), then the modelling error is given by*

$$\begin{aligned} e_{MOD} &= 0, for\ periodic\ coupling\ with\ {}^{\delta}/_{\varepsilon} \in N^*, \\ e_{MOD} &= \frac{\varepsilon}{\delta}, for\ Dirichlet\ coupling\ with\ d > \varepsilon \end{aligned} \tag{4.2.3}$$

The a priori estimates are verified with numerical experiments in Chapter 6.

CHAPTER 5
MATLAB IMPLEMENTATION OF THE FINITE ELEMENT HETEROGENEOUS MULTISCALE METHOD FOR PROBLEMS IN LINEAR ELASTICITY

This chapter comprises of the main work of the dissertation. The FE-HMM described in Chapter 4 is implemented in MATLAB and the various components and routines are explained within this chapter. We note that the FEM codes that form the basis for the FE-HMM codes presented here are based mainly on [17] while references were made to [7] and [8]. The FE-HMM routines itself (for linear elasticity problems) are based partly on the scalar FE-HMM codes from [5].

5.1 Mesh Data Structure

The finite element mesh for the macro and micro problems are generated using the FE-codes from [17]. We present here only the mesh data structure and refer the reader to [17] for more details.

The routine `MakeMeshElasticity` creates a MATAB structure array `struct` that contains the following fields: (We note that X and Y in each field refer to x and y directions respectively)

- `Degree`: The field storing the degree d of the basis functions.
- `Nodes`: A $\mathrm{Nv} \times 2$ array (where Nv is the total number of nodes) which stores the x and y co-ordinates of the nodes.
- `Edges`: An $\mathrm{Nv} \times (\mathrm{d}+1)$ array which stores the nodes on each edge.
- `Elements`: An $\mathrm{Nt} \times 3$ array which stores the three edges of each triangle.
- `FNodePtrsX(Y)`, `CNodePtrsX(Y)` and `NodePtrsX(Y)`: which are pointers from the *free nodes* to `Nodes`, *constrained nodes* to `Nodes`, and the inverse mapping from `Nodes` to both the *free* and *constrained nodes.*
- `EdgeEls`: A field that specifies whether an edge is *free*, *constrained* or *interior.*
- `FBndyEdgesX(Y)`: A pointer from the free boundary edges to `Edges`.
- `EdgeCFlags`: A flag that specifies the curved boundary edges.
- `CBndyEdgesX(Y)`: This is an additional field that stores pointers from the constrained boundary edges to `Edges`.

This additional field `CBndyEdgesX(Y)` is redundant for a standard FEM code but is used in the FE-HMM code to pair the periodic nodes while enforcing periodic micro boundary conditions.

The routine `RefineMeshElasticity` is used to perform a *uniform refinement* of the mesh. Elements with higher order Lagrange polynomial basis functions are generated using the routine `GeneralLagrangeMeshElasticity`. Both routines are also part of the finite element code from [17].

The micro mesh. The micro mesh is generated for the micro/sampling domain which is a square with having a side length δ.

The routine `MakeMicroMeshElasticity` is given below.

```
function TMicro=MakeMicroMeshElasticity(qpt,delta,bctype,r,dmicro)
if nargin<5
    dmicro=1;
end
TMicro.Degree=1;
TMicro.Nodes=[qpt(1)-delta/2 qpt(2)-delta/2
    qpt(1)+delta/2 qpt(2)-delta/2
    qpt(1)-delta/2 qpt(2)+delta/2
    qpt(1)+delta/2 qpt(2)+delta/2];
TMicro.Edges=[1 2
    2 4
    1 4
    3 4
    1 3];
TMicro.Elements=[1 2 -3;3 -4 -5];
TMicro.EdgeCFlags=zeros(5,1);
switch(lower(bctype))
    case{'periodic'}
        % For enforcing periodic bc, we consider two of the edges as
        % TMirco.FBndyEdges and the other two edges as T.CBndyEdges,
        % which allows us to extract the periodic node pairs.
        TMicro.EdgeEls=[1 -1 -1;1 -2 -1;1 2 2;2 0 0;2 0 0];
        TMicro.NodePtrsX=[-1;1;-2;-3];
        TMicro.NodePtrsY=[-1;1;-2;-3];
        TMicro.FNodePtrsX=2;
        TMicro.FNodePtrsY=2;
        TMicro.CNodePtrsX=[1;3;4];
        TMicro.CNodePtrsY=[1;3;4];
        TMicro.FBndyEdgesX=[1 2];
        TMicro.CBndyEdgesX=[4 5];
        TMicro.FBndyEdgesY=[1 2];
        TMicro.CBndyEdgesY=[4 5];
    case{'dirichlet'}
```

```
          TMicro.EdgeEls=[1 0 0;1 0 0;1 2 2;2 0 0;2 0 0];
          TMicro.NodePtrsX=[-1;-2;-4;-3];
          TMicro.NodePtrsY=[-1;-2;-4;-3];
          TMicro.FNodePtrsX=zeros(0,1);
          TMicro.FNodePtrsY=zeros(0,1);
          TMicro.CNodePtrsX=[1;2;4;3];
          TMicro.CNodePtrsY=[1;2;4;3];
          TMicro.FBndyEdgesX=zeros(0,1);
          TMicro.FBndyEdgesY=zeros(0,1);
          TMicro.CBndyEdgesX=[1 2 4 5];
          TMicro.CBndyEdgesY=[1 2 4 5];
end
% Refine micro mesh r times
for i=1:r
     TMicro=RefineMeshElasticity(TMicro);
end
TMicro=GeneralLagrangeMeshElasticity(TMicro,dmicro);

% Create constraint nodes/node pairs for either bctype
switch (lower(bctype))
     case{'periodic'}
[TMicro.Npairs,TMicro.Dofpairs]=PeriodicNodePairsElasticity(TMicro);
end
end
```

The function takes the following inputs:

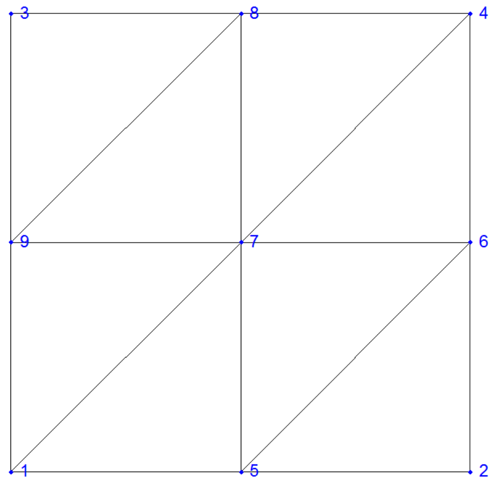
- `bctype`: The micro boundary condition; a string which is either `'periodic'` or `'dirichlet'`.
- `epsilon`: The size of the micro length scale ε.
- `delta`: The size of the sampling domain.
- `r`: The number of refinement for the micro discretization.
- `dmicro`: The degree for the micro discretization.

For the Dirichlet boundary condition, the micro mesh is the same as that for a typical finite element mesh structure with pure Dirichlet boundary condition. For the periodic case, however, a new field becomes necessary that would store the pairs of "periodic" nodes, which are mirror images of each other. This is done by the routine `PeriodicNodePairs` which is given below.

```

function [Npairs]=PeriodicNodePairsElasticity(T)
d=T.Degree;
Nv=size(T.Nodes,1);
ne=length(T.FBndyEdgesX);
```

```
N=d*ne/2+1; % N gives the number of nodes along an edge.
n=N+(N-1);  % N nodes along one edge and N-1 along the other
            % ensuring no redundancy of last (3,4) node pair.
Npairs=zeros(n,2);
% The periodic node pairs are extracted by looping through
% F.BndyEdgesX which comprises of the first pairs of periodic nodes.
for i=1:ne
    if i==ne/2  % The right-most bottom and top edge pairs.
        for j=1:d
            % Note that T.FBndyEdges and T.CBndyEdges have been
            % created in MakeMicroMesh such that T.FBndyEdges(i)
            % pairs with T.CBndyEdges(i).
            Npairs(i+(j-1)*ne,:)=[T.Edges(T.FBndyEdgesX(i),j)...
                               T.Edges(T.CBndyEdgesX(i),j)];
            % The right bottom corner and the top corner nodes are
            % stored as the nth node pair.
            Npairs(n,:)=[T.Edges(T.FBndyEdgesX(i),d+1)...
                              T.Edges(T.CBndyEdgesX(i),d+1)];
        end
    else
        for j=1:d
            Npairs(i+(j-1)*ne,:)=[T.Edges(T.FBndyEdgesX(i),j)...
                               T.Edges(T.CBndyEdgesX(i),j)];
        end
    end
end
end
```



$$\texttt{Npairs}=\begin{bmatrix}1 & 3\\ 5 & 8\\ 2 & 1\\ 6 & 9\\ 2 & 4\end{bmatrix}$$

Figure 5.1. An example micro mesh and the corresponding periodic node pairs.

Figure 5.1 shows an example micro mesh created with a single refinement. The periodic node pairs are stored in the array `Npairs`.

The enforcement of boundary conditions are described in detail in a later section. For now, we only note that the pairs are mirror reflections of each other and one periodic node pair $(4,3)$ is redundant as it is implicitly enforced by $(1,3), (2,1)$ and $(2,4)$.

5.2 Macrostiffness Matrix

This section describes the algorithm for the generation of the macrostiffness matrix. Since the macrostiffness matrix requires the micro functions which are obtained by solving the micro problem, it is preferable to refer to the routines for the micro problem while reading this section.

Macro and micro basis functions. The first objective is to specify notations for the macro and micro basis functions. We note that the domain Ω is assumed to be 2D from here on. The notations are described as follows:

- $\{\boldsymbol{\varphi}_i^H\}_{i=1}^{N_{fx}+N_{fy}}$ denotes the basis of the macro FE space $\boldsymbol{\mathcal{V}}^p(\Omega,\mathcal{T}_H)$ defined in (3.1.2). $N = N_{fx} + N_{fy}$ gives the total number of (*free*) nodes in the discretized macro domain. The basis functions are defined as

$$\boldsymbol{\varphi}_i^H = (\varphi_i^H, 0)\,, \qquad \boldsymbol{\varphi}_{N_{fx+j}}^H = \left(0, \varphi_j^H\right), \quad i = 1, \dots, N_{fx}\,, j = 1, \dots, N_{fy} \tag{5.2.1}$$

  where φ_i^H denote the usual basis functions for the scalar finite element space $V^p(\Omega,\mathcal{T}_H)$ and N_{fx} denote the number of *free* nodes in the 1st or x dimension.

- $\{\boldsymbol{\psi}_{k,K_\delta}^h\}_{k=1}^{2n}$ denotes the basis of the micro FE space $\boldsymbol{\mathcal{S}}^q\left(K_\delta\left(x_{\ell,K}\right),\mathcal{T}_h\right)$ defined in (4.1.6). n gives the total number of nodes in the discretized micro domain. Similar to the macro space, the basis functions $\boldsymbol{\psi}_k^h$ are defined as

$$\boldsymbol{\psi}_k^h = \left(\psi_k^h, 0\right), \qquad \boldsymbol{\psi}_{n+k}^h = \left(0, \psi_k^h\right) \quad k = 1, \dots, n \tag{5.2.3}$$

  where ψ_k^H denote the usual basis functions for scalar micro space $S^q\left(K_\delta\left(x_{\ell,K}\right),\mathcal{T}_h\right)$. It is also noted that for the micro space, the boundaries are considered to be entirely *free*. The required boundary conditions are enforced using Lagrange multipliers instead as will be described shortly.

The superscripts H and h are used throughout the dissertation to differentiate between the macro and micro finite element spaces.

Local macrostiffness matrix. This section describes the local macrostiffness matrix $\boldsymbol{A}_K$ for each $K \in \mathcal{T}_H$. For macro basis functions $\boldsymbol{\varphi}_i^H$ $i = 1, \dots, i_D$, (where i_D is the number of nodes per element) with non-zero support in K, the macrostiffness matrix is given by

$$\begin{aligned}\boldsymbol{A}_K &:= B_H\left(\boldsymbol{\varphi}_i^H, \boldsymbol{\varphi}_j^H\right)_{i,j=1}^{2i_D} \\ &= \left(\sum_{\ell=1}^{\mathcal{L}} \frac{\omega_{\ell,K}}{\left|K_\delta\left(x_{\ell,K}\right)\right|} \int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e\left(\boldsymbol{\varphi}_{\ell,K_\delta}^h\right) e\left(\boldsymbol{\varphi}_{\ell,K_\delta}^h\right) dx\right)_{i,j=1}^{2i_D}\end{aligned} \tag{5.2.4}$$

where $\boldsymbol{\varphi}_{i_{\ell,K_\delta}}^h$ and $\boldsymbol{\varphi}_{j_{\ell,K_\delta}}^h$ are micro functions obtained from solving the micro problem (4.1.4). Writing the micro functions $\boldsymbol{\varphi}_{i_{\ell,K_\delta}}^h$ and $\boldsymbol{\varphi}_{j_{\ell,K_\delta}}^h$ in terms of their basis functions as

$$\boldsymbol{\varphi}_{i_{\ell,K_\delta}}^h = \sum_{k=1}^{2n} \alpha_{k,\ell}^i \boldsymbol{\psi}_k^h \quad \text{and} \quad \boldsymbol{\varphi}_{j_{\ell,K_\delta}}^h = \sum_{k=1}^{2n} \alpha_{k,\ell}^j \boldsymbol{\psi}_k^h \tag{5.2.5}$$

we can write (8.1.2) as

$$\boldsymbol{A}_K = \sum_{\ell=1}^{\mathcal{L}} \frac{\omega_{\ell,K}}{\left|K_\delta\left(x_{\ell,K}\right)\right|} \left(\left(\boldsymbol{\alpha}_\ell^i\right)^T \boldsymbol{B}_{\ell,K_\delta} \boldsymbol{\alpha}_\ell^j\right) \tag{5.2.6}$$

where $\boldsymbol{\alpha}_\ell^i = \left(\alpha_{1,\ell}^i, \dots, \alpha_{2n,\ell}^i\right)^T$ and $\boldsymbol{\alpha}_\ell^j = \left(\alpha_{1,\ell}^j, \dots, \alpha_{2n,\ell}^j\right)^T$ are the coefficients for the micro functions which shall be computed from the micro problem (4.1.4).

$\boldsymbol{B}_{\ell,K_\delta}{}'s$ are microstiffness matrices for the sampling domains $K_\delta\left(\omega_{\ell,K}\right)$ around each quadrature point $\omega_{\ell,K}$. Each B_{ℓ,K_δ} is computed in the similar manner as done for a classical finite element stiffness matrix.

The MATLAB routine for computing the macrostiffness matrix `MacroStiffnessElasticity` is described at end. First, the computation of the micro functions $\boldsymbol{\varphi}_{i_{\ell,K_\delta}}^h$ and $\boldsymbol{\varphi}_{j_{\ell,K_\delta}}^h$ (or the vectors $\boldsymbol{\alpha}_\ell^i$ and $\boldsymbol{\alpha}_\ell^i$) is discussed.

5.3 The Micro Problem

The micro boundary conditions are enforced weakly using Lagrange multipliers. The problem (4.1.6) is restated as a minimization problem: Find

$$\boldsymbol{\varphi}^h_{\ell,K_\delta} = \min \frac{1}{2} \int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e(\mathbf{w}^h)\, e(\mathbf{w}^h) dx \tag{5.3.1}$$

over all functions $\mathbf{w}^h \in \boldsymbol{\mathcal{S}}^p(K_\delta, \mathcal{T}_H)$ satisfying the condition $\left(\mathbf{w}^h - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta}\right) \in \boldsymbol{\mathcal{S}}^q(K_\delta, \mathcal{T}_h)$ which can be

- $\left(\mathbf{w}^h - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta}\right) \in \boldsymbol{\mathcal{S}}^q_{per}(K_\delta, \mathcal{T}_h)$ (periodic coupling) or
- $\left(\mathbf{w}^h - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta}\right) \in \boldsymbol{\mathcal{S}}^q_{dir}(K_\delta, \mathcal{T}_h)$ (Dirichlet coupling).

Periodic coupling. Micro boundary conditions are weakly enforced through Lagrange multipliers. We follow the procedure adopted in [6] to define the constraint functional. To define the constraint functional we first introduce the following notation.

Let $\boldsymbol{\mathcal{S}}^q\left(\partial K_\delta, \mathcal{T}_{\partial,h}\right)$ be a finite element space of degree q defined on the micro boundary ∂K_δ with $(\boldsymbol{\vartheta}^h_s)_{s=1}^{4m}$ as the basis given by

$$\boldsymbol{\vartheta}^h_s = (\vartheta^h_s, 0)\,, \qquad \boldsymbol{\vartheta}^h_{n+s} = (0, \vartheta^h_s) \quad s = 1, \dots, 2m \tag{5.3.2}$$

where ϑ^h_r denote the usual basis functions for the scalar space $S^q\left(\partial K_\delta\left(x_{\ell,K}\right), \mathcal{T}_{\partial,h}\right)$. The first 2m basis functions belong to the x dimension. These basis functions are ordered such that the first m nodes p_r are on two edges of ∂K_δ that intersect. Then, for a node p_r the node on the opposite edge by mirror symmetry is denoted by $p_{\sigma(r)}$. The remaining $2m$ basis are for the y dimension and are ordered similarly.

The constraint functional is then defined by $G\left(\boldsymbol{\varphi}^h_{\ell,K_\delta} - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta}, \boldsymbol{\vartheta}, \boldsymbol{\kappa}\right) = 0$, where G is given by

$$G(\boldsymbol{\psi}, \boldsymbol{\vartheta}, \boldsymbol{\kappa}) = \sum_{r=1}^{m} \left(\boldsymbol{\psi}(p_r) - \boldsymbol{\psi}\left(p_{\sigma(r)}\right)\right) \cdot \boldsymbol{\vartheta}(p_r) + \boldsymbol{\kappa} \cdot \int_{K_\delta} \boldsymbol{\psi} dx \tag{5.3.3}$$

and $\boldsymbol{\vartheta} \in \boldsymbol{\mathcal{S}}^q\left(\partial K_\delta, \mathcal{T}_{\partial,h}\right)$, $\boldsymbol{\psi} \in \boldsymbol{\mathcal{S}}^q(K_\delta, \mathcal{T}_h)$ and $\boldsymbol{\kappa} \in \mathbb{R}^d$.

The constraint allows the micro problem to be defined as a saddle point problem (see [6]):

$$\begin{cases} \text{Find } \boldsymbol{\varphi}^h_{\ell,K_\delta} \in \boldsymbol{\mathcal{S}}^q(K_\delta,\mathcal{T}_h) \text{ and } \boldsymbol{\vartheta}_0 \in \boldsymbol{\mathcal{S}}^q(\partial K_\delta,\mathcal{T}_{\partial,h}), \boldsymbol{\kappa}_0 \in \mathbb{R}^d \text{ such that} \\ \displaystyle\int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e\big(\boldsymbol{\varphi}^h_{\ell,K_\delta}\big)\, e(\boldsymbol{\psi})dx + G(\boldsymbol{\psi},\boldsymbol{\vartheta}_0,\boldsymbol{\kappa}_0) = 0, \quad \forall \boldsymbol{\psi} \in \boldsymbol{\mathcal{S}}^q(K_\delta,\mathcal{T}_h), \\ G\big(\boldsymbol{\varphi}^h_{\ell,K_\delta} - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta},\boldsymbol{\vartheta},\boldsymbol{\kappa}\big) = 0, \quad \forall \boldsymbol{\vartheta} \in \boldsymbol{\mathcal{S}}^q(\partial K_\delta,\mathcal{T}_{\partial,h}), \quad \forall \boldsymbol{\kappa} \in \mathbb{R}^d. \end{cases} \tag{5.3.4}$$

Now, writing in terms of the basis functions, as

$$\boldsymbol{\varphi}^h_{i,\ell,K_\delta} = \sum_{k=1}^{2n} \alpha^i_{k,\ell}\boldsymbol{\psi}^h_k, \qquad \text{and } \boldsymbol{\vartheta}_0 = \sum_{s=1}^{4m} \lambda_s \boldsymbol{\vartheta}^h_s$$

and noting that $\boldsymbol{\vartheta}^h_s(p_r) = (1,0)$ and $\boldsymbol{\vartheta}^h_{n+s}(p_r) = (0,1)$ for $s = r$ or $2m + r$, and 0 everwhere else, for $k = 1, \dots, 2n$, we have,

$$\sum_{k'=1}^{2n} \int_{K_\delta(x_{\ell,K})} A^\varepsilon(x)\, e\big(\alpha^i_{k',\ell}\boldsymbol{\psi}^h_{k'}\big)\, e\big(\boldsymbol{\psi}^h_k\big)dx + G\big(\boldsymbol{\psi}^h_k,\boldsymbol{\vartheta}_0,\boldsymbol{\kappa}_0\big) = 0$$

$$\Rightarrow \sum_{k'=1}^{2n} \big(\boldsymbol{B}_{\ell,K_\delta}\big)_{k'k}\alpha^i_{k',\ell} + \sum_{r=1}^{m}\sum_{s=1}^{4m}\Big(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\big(p_{\sigma(r)}\big)\Big)\cdot \lambda_s\boldsymbol{\vartheta}^h_s(p_r) + \int_{K_\delta} \psi^h_k dx = 0$$

$$\Rightarrow \sum_{k'=1}^{2n} \big(\boldsymbol{B}_{\ell,K_\delta}\big)_{k'k}\alpha^i_{k',\ell} + \sum_{r=1}^{m}\Big(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\big(p_{\sigma(r)}\big)\Big)\cdot\Big(\lambda_r\boldsymbol{\vartheta}^h_r(p_r) + \lambda_{2m+r}\boldsymbol{\vartheta}^h_{2m+r}(p_r)\Big)$$
$$+\int_{K_\delta} \psi^h_k dx = 0 \tag{5.3.5}$$

Now, for the second term can be written as

$$\sum_{r=1}^{m}\bigg(\Big(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\big(p_{\sigma(r)}\big)\Big)\cdot\lambda_r\boldsymbol{\vartheta}^h_r(p_r) + \Big(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\big(p_{\sigma(r)}\big)\Big)\cdot\lambda_{2m+r}\boldsymbol{\vartheta}^h_{2m+r}(p_r)\bigg)$$

and, for all $k = 1, \dots, 2n$ can be written in the form

$$\left[\begin{array}{ccc:ccc} \big(\boldsymbol{\psi}^h_1(p_1) - \boldsymbol{\psi}^h_1(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_1(p_1) & \cdots & \big(\boldsymbol{\psi}^h_1(p_m) - \boldsymbol{\psi}^h_1(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_m(p_m) & \big(\boldsymbol{\psi}^h_1(p_1) - \boldsymbol{\psi}^h_1(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \big(\boldsymbol{\psi}^h_1(p_m) - \boldsymbol{\psi}^h_1(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_{3m}(p_m) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \big(\boldsymbol{\psi}^h_n(p_1) - \boldsymbol{\psi}^h_n(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_1(p_1) & \cdots & \big(\boldsymbol{\psi}^h_n(p_m) - \boldsymbol{\psi}^h_n(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_m(p_m) & \big(\boldsymbol{\psi}^h_n(p_1) - \boldsymbol{\psi}^h_n(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \big(\boldsymbol{\psi}^h_n(p_m) - \boldsymbol{\psi}^h_n(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_{3m}(p_m) \\ \hdashline \big(\boldsymbol{\psi}^h_{n+1}(p_1) - \boldsymbol{\psi}^h_{n+1}(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_1(p_1) & \cdots & \big(\boldsymbol{\psi}^h_{n+1}(p_m) - \boldsymbol{\psi}^h_{n+1}(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_m(p_m) & \big(\boldsymbol{\psi}^h_{n+1}(p_1) - \boldsymbol{\psi}^h_{n+1}(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \big(\boldsymbol{\psi}^h_{n+1}(p_m) - \boldsymbol{\psi}^h_{n+1}(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_{3m}(p_m) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \big(\boldsymbol{\psi}^h_{2n}(p_1) - \boldsymbol{\psi}^h_{2n}(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_1(p_1) & \cdots & \big(\boldsymbol{\psi}^h_{2n}(p_m) - \boldsymbol{\psi}^h_{2n}(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_m(p_m) & \big(\boldsymbol{\psi}^h_{2n}(p_1) - \boldsymbol{\psi}^h_{2n}(p_{\sigma(1)})\big)\cdot\boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \big(\boldsymbol{\psi}^h_{2n}(p_m) - \boldsymbol{\psi}^h_{2n}(p_{\sigma(m)})\big)\cdot\boldsymbol{\vartheta}^h_{3m}(p_m) \end{array}\right]\begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \\ \hdashline \lambda_{m+1} \\ \vdots \\ \lambda_{3m} \end{bmatrix}$$

We denote this $2n \times 2m$ matrix by $\widetilde{\boldsymbol{D}}^T$ and partition it into four sub-matrices. Recalling the definitions of the basis functions $\boldsymbol{\psi}_k^h$ (5.2.5), and since $\boldsymbol{\vartheta}_s^h(p_r) = (1,0)$ and $\boldsymbol{\vartheta}_{n+s}^h(p_r) = (0,1), s = 1, \dots, m$ we can write $\widetilde{\boldsymbol{D}}^T$ as

$$\widetilde{\boldsymbol{D}}^T = \begin{bmatrix} [\widetilde{D}]_{m\times n} & [\text{zeros}]_{m\times n} \\ [\text{zeros}]_{m\times n} & [\widetilde{D}]_{m\times n} \end{bmatrix}_{2m\times 2n} \tag{5.3.6}$$

where,

$$\widetilde{D}_{kr} = \psi_k^h(p_r) - \psi_k^h\left(p_{\sigma(r)}\right) \; r = 1, \dots, m. \tag{5.3.7}$$

Now, since the values of the micro basis functions evaluated at the periodic nodes p_r can only be either 0 or 1, it follows that $\widetilde{D}$ will have values of either $\widetilde{D}_{kr} = 1$ if $k = p_r$ or $\widetilde{D}_{kr} = -1$ if $k = p_{\sigma(r)}$ and 0 otherwise.

An example of $\widetilde{D}$ for the micro mesh shown in Figure (5.1) is

$$\widetilde{D} = \begin{pmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Combining with the last term in (5.3.5) expressed as $\left(\int_T \psi_1^h, \dots, \int_T \psi_{2n}^h\right) = b = (b_1, \dots, b_{2n})$ we can write (5.3.5) as

$$\boldsymbol{B}_{\ell,K_\delta}\boldsymbol{\alpha}_\ell^i + \boldsymbol{D}^T\lambda = 0,$$

where

$$\boldsymbol{D} = \begin{pmatrix} b_1 & \cdots & b_{2n} \\ & \widetilde{\boldsymbol{D}} & \end{pmatrix}, \boldsymbol{\alpha}_\ell^i = \left(\alpha_{1,\ell}^i, \dots, \alpha_{2n,\ell}^i\right)^T \text{ and } \lambda = (\kappa_0, \lambda_1, \dots, \lambda_{2m})^T. \tag{5.3.8}$$

Now, for the second equation in (5.3.4), noting that $\int_{K_\delta}(\boldsymbol{\varphi}_{i,\ell,K_\delta}^h - \boldsymbol{\varphi}_{i,lin,\ell,K_\delta}^H)dx = 0$ for periodic boundary conditions, we have, for $i = 1, \dots, i_D$ (recall that i_D is the number of nodes per element)

$$G\left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta} - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}, \boldsymbol{\vartheta}, \boldsymbol{\kappa}\right) = 0$$

$$\Rightarrow \sum_{r=1}^{m} \left(\left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta} - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\right)(p_r) - \left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta} - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\right)\left(p_{\sigma(r)}\right)\right) \cdot \left(\boldsymbol{\vartheta}^h_r(p_r) + \boldsymbol{\vartheta}^h_{2m+r}(p_r)\right)$$

$$+ \int_{K_\delta} \left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta} - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\right) dx = 0$$

$$\Rightarrow \sum_{r=1}^{m} \left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta}(p_r) - \left(\boldsymbol{\varphi}^h_{i,\ell,K_\delta} -\right)\left(p_{\sigma(r)}\right)\right) \cdot \left(\boldsymbol{\vartheta}^h_r(p_r) + \boldsymbol{\vartheta}^h_{2m+r}(p_r)\right)$$

$$= \sum_{r=1}^{m} \left(\boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(p_r) - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\left(p_{\sigma(r)}\right)\right) \cdot \left(\boldsymbol{\vartheta}^h_r(p_r) + \boldsymbol{\vartheta}^h_{2m+r}(p_r)\right)$$

Expanding in terms of the basis functions we can write the above equation as,

$$\sum_{r=1}^{m} \sum_{k=1}^{2n} \alpha^i_{k,\ell} \left(\left(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\left(p_{\sigma(r)}\right)\right) \cdot \boldsymbol{\vartheta}^h_r(p_r) + \left(\boldsymbol{\psi}^h_k(p_r) - \boldsymbol{\psi}^h_k\left(p_{\sigma(r)}\right)\right) \cdot \boldsymbol{\vartheta}^h_{2m+r}(p_r)\right)$$

$$= \sum_{r=1}^{m} \left(\left(\boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(p_r) - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\left(p_{\sigma(r)}\right)\right) \cdot \boldsymbol{\vartheta}^h_r(p_r) + \left(\boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(p_r) - \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}\left(p_{\sigma(r)}\right)\right) \cdot \boldsymbol{\vartheta}^h_{2m+r}(p_r)\right)$$

or, $\left(\boldsymbol{D}\boldsymbol{\alpha}^i_\ell\right)_j = \boldsymbol{D}^\beta_{ji}$.

where, the matrix $\boldsymbol{D}^\beta$ can be partitioned and written as

$$\boldsymbol{D}^\beta = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \left[\boldsymbol{D}^{\beta 11}\right]_{m\times i_D} & & \left[\boldsymbol{D}^{\beta 12}\right]_{m\times i_D} & \\ \left[\boldsymbol{D}^{\beta 21}\right]_{m\times i_D} & & \left[\boldsymbol{D}^{\beta 22}\right]_{m\times i_D} & \end{bmatrix}_{(2m+1)\times 2i_D} \quad (5.3.9)$$

where the sub-matrices have the expressions

$$\boldsymbol{D}^{\beta 11}$$

$$= \begin{bmatrix} \left(\boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_1(p_1) & \cdots & \left(\boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_1(p_1) \\ \vdots & \ddots & \vdots \\ \left(\boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_m(p_m) & \cdots & \left(\boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_m(p_m) \end{bmatrix}$$

$$\boldsymbol{D}^{\beta 21}$$

$$= \begin{bmatrix} \left(\boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \left(\boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_{2m+1}(p_1) \\ \vdots & \ddots & \vdots \\ \left(\boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{1,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_{3m}(p_m) & \cdots & \left(\boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{i_D,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_{3m}(p_m) \end{bmatrix}$$

$$\boldsymbol{D}^{\beta 12}$$

$$= \begin{bmatrix} \left(\boldsymbol{\varphi}^H_{i_D+1,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{i_D+1,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_1(p_1) & \cdots & \left(\boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_1(p_1) \\ \vdots & \ddots & \vdots \\ \left(\boldsymbol{\varphi}^H_{i_D+1,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{i_D+1,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_m(p_1) & \cdots & \left(\boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_m(p_m) \end{bmatrix}$$

$$\boldsymbol{D}^{\beta 22}$$

$$= \begin{bmatrix} \left(\boldsymbol{\varphi}^H_{i_{Dx}+1,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{i_{Dx}+1,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_{2m+1}(p_1) & \cdots & \left(\boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_1) - \boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_{\sigma(1)})\right) \cdot \boldsymbol{\vartheta}^h_{2m+1}(p_1) \\ \vdots & \ddots & \vdots \\ \left(\boldsymbol{\varphi}^H_{i_{Dx}+1,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{i_{Dx}+1,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_{3m}(p_m) & \cdots & \left(\boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_m) - \boldsymbol{\varphi}^H_{2i_D,lin,\ell,K_\delta}(p_{\sigma(m)})\right) \cdot \boldsymbol{\vartheta}^h_{3m}(p_m) \end{bmatrix}$$

Now, from (4.1.5), we get,

for $1 \le i \le i_D$

$$\boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(x) = \boldsymbol{\varphi}^H_i(x_{\ell,K}) + e\left(\boldsymbol{\varphi}^H_i(x_{\ell,K})\right)(x - x_{\ell,K})$$

$$\Rightarrow \begin{bmatrix} \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(x)_1 \\ \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(x)_2 \end{bmatrix} = \begin{bmatrix} \varphi^H_i(x_{\ell,K}) \\ 0 \end{bmatrix} + \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) & {}^1\!/_2\, \varphi^H_{i,y}(x_{\ell,K}) \\ {}^1\!/_2\, \varphi^H_{i,y}(x_{\ell,K}) & 0 \end{bmatrix} \begin{bmatrix} x - x_{\ell,K} \\ y - y_{\ell,K} \end{bmatrix}$$

$$= \begin{bmatrix} \varphi^H_i(x_{\ell,K}) \\ 0 \end{bmatrix} + \begin{bmatrix} \left[(x - x_{\ell,K}) \quad {}^1\!/_2\,(y - y_{\ell,K})\right] \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \\ \left[(y - y_{\ell,K}) \quad {}^1\!/_2\,(x - x_{\ell,K})\right] \begin{bmatrix} 0 \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \end{bmatrix}, \qquad (5.3.10)$$

and for $i_D < i \le 2i_D$,

$$\begin{bmatrix} \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(x)_1 \\ \boldsymbol{\varphi}^H_{i,lin,\ell,K_\delta}(x)_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \varphi^H_i(x_{\ell,K}) \end{bmatrix} + \begin{bmatrix} 0 & {}^1\!/_2\, \varphi^H_{i,x}(x_{\ell,K}) \\ {}^1\!/_2\, \varphi^H_{i,x}(x_{\ell,K}) & \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \begin{bmatrix} x - x_{\ell,K} \\ y - y_{\ell,K} \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ \varphi^H_i(x_{\ell,K}) \end{bmatrix} + \begin{bmatrix} \begin{bmatrix} {}^1\!/_2\,(y - y_{\ell,K}) & (x - x_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ 0 \end{bmatrix} \\ \begin{bmatrix} {}^1\!/_2\,(x - x_{\ell,K}) & (y - y_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \end{bmatrix}. \tag{5.3.11}$$

Hence, we can write the expressions of the sub-matrices of $\boldsymbol{D}^\beta$ as,

$$\boldsymbol{D}^{\beta 11}_{ri} = \left(\varphi^H_i(x_{\ell,K}) + \begin{bmatrix} (x - x_{\ell,K}) & {}^1\!/_2\,(y - y_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \right)(p_r)$$
$$- \left(\varphi^H_i(x_{\ell,K}) + \begin{bmatrix} (x - x_{\ell,K}) & {}^1\!/_2\,(y - y_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \right)(p_{\sigma(r)}),$$
$$1 \le i \le i_D, 1 \le r \le m \tag{5.3.12}$$

$$\boldsymbol{D}^{\beta 21}_{ri} = \left({}^1\!/_2\,(x - x_{\ell,K}) \varphi^H_{i,y}(x_{\ell,K}) \right)(p_r) - \left({}^1\!/_2\,(x - x_{\ell,K}) \varphi^H_{i,y}(x_{\ell,K}) \right)(p_{\sigma(r)}),$$
$$1 \le i \le i_D, 1 \le r \le m \tag{5.3.13}$$

$$\boldsymbol{D}^{\beta 12}_{ri} = \left({}^1\!/_2\,(y - y_{\ell,K}) \varphi^H_{i,x}(x_{\ell,K}) \right)(p_r) - \left({}^1\!/_2\,(y - y_{\ell,K}) \varphi^H_{i,x}(x_{\ell,K}) \right)(p_{\sigma(r)}),$$
$$i_D < i \le 2i_D, 1 \le r \le m \tag{5.3.14}$$

$$\boldsymbol{D}^{\beta 11}_{ri} = \left(\varphi^H_i(x_{\ell,K}) + \begin{bmatrix} {}^1\!/_2\,(x - x_{\ell,K}) & (y - y_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \right)(p_r)$$
$$- \left(\varphi^H_i(x_{\ell,K}) + \begin{bmatrix} {}^1\!/_2\,(x - x_{\ell,K}) & (y - y_{\ell,K}) \end{bmatrix} \begin{bmatrix} \varphi^H_{i,x}(x_{\ell,K}) \\ \varphi^H_{i,y}(x_{\ell,K}) \end{bmatrix} \right)(p_{\sigma(r)}),$$
$$i_D < i \le 2i_D, 1 \le r \le m \tag{5.3.15}$$

Now, we can finally express the saddle problem (5.3.4) in matrix form as

$$\begin{cases} \text{Find } \boldsymbol{\alpha}_\ell^i \text{ such that} \\ \boldsymbol{B}_{\ell,K_\delta}\boldsymbol{\alpha}_\ell^i + \boldsymbol{D}^T\lambda = 0 \\ \left(\boldsymbol{D}\boldsymbol{\alpha}_\ell^i\right)_j = \boldsymbol{D}_{ji}^\beta \end{cases} \tag{5.3.16}$$

where $\boldsymbol{B}_{\ell,K_\delta}, \boldsymbol{\alpha}_\ell^i, \boldsymbol{D}, \boldsymbol{D}^\beta$ and λ are defined by (5.3.21) (5.3.8) and (5.3.9).

The MATLAB routine for computing $\widetilde{\boldsymbol{D}}$ and $\boldsymbol{D}^\beta$ is given below.

```
function [D_tilda BetaD]=MicroConstraintsPeriodicElasticity...
                    (TMicro,Qpt,VQpts,grad,j)

% The function MicroConstraintsPeriodic returns the matrices D_tilda
% and BetaD. BetaD gives the right hand side of the constraint
% equation.

%Create the matrix D_tilda
np=size(TMicro.Npairs,1);
Nv=size(TMicro.Nodes,1);

D_tilda=sparse(repmat(1:np,1,2),[TMicro.Npairs(:,1)...
                TMicro.Npairs(:,2)],[ones(np,1);-ones(np,1)],np,Nv);

D_tilda=[D_tilda sparse(np,Nv);sparse(np,Nv) D_tilda];

node1=TMicro.Nodes(TMicro.Npairs(:,1),:);
node2=TMicro.Nodes(TMicro.Npairs(:,2),:);

X11=zeros(size(node1));
X11_sigma=zeros(size(node1));

X21=zeros(size(node1));
X21_sigma=zeros(size(node1));

X12=zeros(size(node1));
X12_sigma=zeros(size(node1));

X22=zeros(size(node1));
X22_sigma=zeros(size(node1));

X1=node1-repmat(Qpt,size(node1,1),1);
X2=node2-repmat(Qpt,size(node2,1),1);

X11(:,1)=X1(:,1);
X11(:,2)=X1(:,2)/2;
X11_sigma(:,1)=X2(:,1);
X11_sigma(:,2)=X2(:,2)/2;
```

```
X21(:,2)=X1(:,1)/2;
X21_sigma(:,2)=X2(:,1)/2;

X12(:,1)=X1(:,2)/2;
X12_sigma(:,1)=X2(:,2)/2;

X22(:,1)=X1(:,1)/2;
X22(:,2)=X1(:,2);
X22_sigma(:,1)=X2(:,1)/2;
X22_sigma(:,2)=X2(:,2);

% phi_lin and phi_lin_sigma give the values of the linearized Macro
% basis functions at the periodic nodes pairs.

phi_lin11=repmat(VQpts(j,:),size(X1,1),1) + X11*grad(:,:,j);
phi_lin11_sigma=repmat(VQpts(j,:),size(X2,1),1) +...
            X11_sigma*grad(:,:,j);

phi_lin21= X21*grad(:,:,j);
phi_lin21_sigma= X21_sigma*grad(:,:,j);

phi_lin12= X12*grad(:,:,j);
phi_lin12_sigma= X12_sigma*grad(:,:,j);

phi_lin22=repmat(VQpts(j,:),size(X1,1),1) + X22*grad(:,:,j);
phi_lin22_sigma=repmat(VQpts(j,:),size(X2,1),1) +...
            X22_sigma*grad(:,:,j);

BetaD11= phi_lin11 - phi_lin11_sigma;
BetaD21= phi_lin21 - phi_lin21_sigma;

BetaD12= phi_lin12 - phi_lin12_sigma;
BetaD22= phi_lin22 - phi_lin22_sigma;

BetaD=[BetaD11 BetaD12;BetaD21 BetaD22];
BetaD=[zeros(1,(size(BetaD,2)));BetaD];

end
```

Dirichlet coupling. Dirichlet micro boundary conditions is enforced in a similar manner by changing the constraint functional. For Dirichlet coupling we have,

$$\left(\boldsymbol{\varphi}^h_{\ell,K_\delta} - \boldsymbol{\varphi}^H_{lin,\ell,K_\delta}\right) \in \boldsymbol{\mathcal{S}}^q_{dir}(K_\delta, \mathcal{T}_h) \tag{5.3.17}$$

Thus, we get

$$G(\boldsymbol{\psi},\boldsymbol{\vartheta}) = \sum_{r=1}^{2m} \left(\boldsymbol{\psi}(p_r)\right) \cdot \boldsymbol{\vartheta}(p_r) \tag{5.3.18}$$

where r in this case sums through all the $2m$ boundary nodes. The resulting constraint matrix $\boldsymbol{D}$ is given by

$$\boldsymbol{D}_{kr} = \boldsymbol{\psi}_k^h(p_r) \quad r = 1, \dots 2m. \tag{5.3.19}$$

And the matrix $\boldsymbol{D}^\beta$ for the Dirichlet case becomes

$$\boldsymbol{D}_{ri}^\beta = \boldsymbol{\varphi}_{i,lin,\ell,K_\delta}^H(p_r) \tag{5.3.20}$$

As before, we can write $\boldsymbol{D}^\beta$ into four sub-matrices defined, in this case, by

$$\boldsymbol{D}_{ri}^{\beta 11} = \left(\varphi_i^H(x_{\ell,K}) + \left[(x - x_{\ell,K}) \quad {}^1\!/_2\,(y - y_{\ell,K})\right] \begin{bmatrix} \varphi_{i,x}^H(x_{\ell,K}) \\ \varphi_{i,y}^H(x_{\ell,K}) \end{bmatrix}\right)(p_r), \quad 1 \le i \le i_D, 1 \le r \le m \tag{5.3.21}$$

$$\boldsymbol{D}_{ri}^{\beta 21} = \left({}^1\!/_2\,(x - x_{\ell,K})\varphi_{i,y}^H(x_{\ell,K})\right)(p_r), \qquad 1 \le i \le i_D, 1 \le r \le m \tag{5.3.22}$$

$$\boldsymbol{D}_{ri}^{\beta 12} = \left({}^1\!/_2\,(y - y_{\ell,K})\varphi_{i,x}^H(x_{\ell,K})\right)(p_r), \qquad i_D < i \le 2i_D, 1 \le r \le m, \tag{5.3.23}$$

$$\boldsymbol{D}_{ri}^{\beta 11} = \left(\varphi_i^H(x_{\ell,K}) + \left[{}^1\!/_2\,(x - x_{\ell,K}) \quad (y - y_{\ell,K})\right] \begin{bmatrix} \varphi_{i,x}^H(x_{\ell,K}) \\ \varphi_{i,y}^H(x_{\ell,K}) \end{bmatrix}\right)(p_r), \quad i_D < i \le 2i_D, 1 \le r \le m \tag{5.3.24}$$

The MATLAB routine for computing $\boldsymbol{D}$ and $\boldsymbol{D}^\beta$ for the Dirichilet boundary condition is given below.

```
function [D BetaD]=MicroConstraintsDirichletElasticity(TMicro,Qpt,...
               VQpts,grad,j)

% The function MicroConstraintsDirichlet returns the matrices D and
% BetaD. BetaD gives the right hand side of the constraint equation.

%Create the matrix D
ndirn=size(TMicro.CNodePtrsX,1); % No. of Dirichlet Nodes.
Nv=size(TMicro.Nodes,1);
```

```
D=sparse(1:ndirn,TMicro.CNodePtrsX,ones(ndirn,1),ndirn,Nv);
D=[D zeros(size(D));zeros(size(D)) D];

% bnodes imply boundary nodes

bnodes=TMicro.Nodes(TMicro.CNodePtrsX,:);

X11=zeros(size(bnodes));
X21=zeros(size(bnodes));
X12=zeros(size(bnodes));
X22=zeros(size(bnodes));

X=bnodes-repmat(Qpt,size(bnodes,1),1);

X11(:,1)=X(:,1);
X11(:,2)=X(:,2)/2;

X21(:,2)=X(:,1)/2;

X12(:,1)=X(:,2)/2;

X22(:,1)=X(:,1)/2;
X22(:,2)=X(:,2);

BetaD11=repmat(VQpts(j,:),size(X,1),1) + X11*grad(:,:,j);
BetaD21= X21*grad(:,:,j);
BetaD12= X12*grad(:,:,j);
BetaD22=repmat(VQpts(j,:),size(X,1),1) + X22*grad(:,:,j);

BetaD=[BetaD11 BetaD12;BetaD21 BetaD22];

end
```

Microstiffness matrix. The microstiffness $\boldsymbol{B}_{\ell,K_\delta}$ is simply a finite element stiffness matrix for linear elasticity and hence, its computation is done in the same manner as for a classical finite element algorithm.

The routine for computing the microstiffness based on [17]. However, minor improvements have been made which include:

- Allows computation of stiffness for non-isotropic materials.
- Vectorization of the codes which improves not only speed but makes the code more intuitive mathematically.
- Adopts faster assembly method for sparse matrices.

The stiffness matrix $\boldsymbol{B}_{\ell,K_\delta}$ is partitioned as follows.

$$\boldsymbol{B}_{\ell,K_\delta} = \begin{bmatrix} B^{11}_{\ell,K_\delta} & B^{12}_{\ell,K_\delta} \\ B^{21}_{\ell,K_\delta} & B^{22}_{\ell,K_\delta} \end{bmatrix} \tag{5.3.25}$$

where, the sub-matrices B^{11}_{ℓ,K_δ}, B^{12}_{ℓ,K_δ}, B^{21}_{ℓ,K_δ} and B^{22}_{ℓ,K_δ} are given by

$$\left(B_{\ell,K_\delta}\right)^{11}_{ij} = \sum_{T\in\mathcal{T}_h} \int_T A^\varepsilon(x)\, e\left(\boldsymbol{\psi}^h_i\right) e\left(\boldsymbol{\psi}^h_j\right) dx\,, \qquad 1 \le i \le n,\ 1 \le j \le n \tag{5.3.26}$$

$$\left(B_{\ell,K_\delta}\right)^{12}_{ij} = \sum_{T\in\mathcal{T}_h} \int_T A^\varepsilon(x)\, e\left(\boldsymbol{\psi}^h_i\right) e\left(\boldsymbol{\psi}^h_j\right) dx\,, \qquad 1 \le i \le n,\ n < j \le 2n \tag{5.3.27}$$

$$\left(B_{\ell,K_\delta}\right)^{21}_{ij} = \sum_{T\in\mathcal{T}_h} \int_T A^\varepsilon(x)\, e\left(\boldsymbol{\psi}^h_i\right) e\left(\boldsymbol{\psi}^h_j\right) dx\,, \qquad n < i \le 2n,\ 1 \le j \le n \tag{5.3.28}$$

$$\left(B_{\ell,K_\delta}\right)^{22}_{ij} = \sum_{T\in\mathcal{T}_h} \int_T A^\varepsilon(x)\, e\left(\boldsymbol{\psi}^h_i\right) e\left(\boldsymbol{\psi}^h_j\right) dx\,, \qquad n < i \le 2n,\ n < j \le 2n \tag{5.3.29}$$

Recalling the definitions of the micro basis functions i.e. $\boldsymbol{\psi}^h_i = \left(\psi^h_i, 0\right)$, $\boldsymbol{\psi}^h_{n+i} = (0, \psi^H_i)$, $i = 1, \dots, n$, and writing in Voigt notation,

$$\left(B_{\ell,K_\delta}\right)^{11}_{ij} = \left(\begin{bmatrix} a_{1111} & a_{1122} & a_{1112} \\ a_{2211} & a_{2222} & a_{2212} \\ a_{1211} & a_{1222} & a_{1212} \end{bmatrix} \begin{bmatrix} \psi^h_{i,x} \\ 0 \\ \psi^h_{i,y} \end{bmatrix} \right) \cdot \begin{bmatrix} \psi^h_{i,x} \\ 0 \\ \psi^h_{i,y} \end{bmatrix}$$

$$= \left(a_{1111}\psi^h_{i,x} + a_{1112}\psi^h_{i,y}\right)\psi^h_{i,x} + \left(a_{1211}\psi^h_{i,x} + a_{1212}\psi^h_{i,y}\right)\psi^h_{i,y} \tag{5.3.30}$$

$$\left(B_{\ell,K_\delta}\right)^{12}_{ij} = \left(\begin{bmatrix} a_{1111} & a_{1122} & a_{1112} \\ a_{2211} & a_{2222} & a_{2212} \\ a_{1211} & a_{1222} & a_{1212} \end{bmatrix} \begin{bmatrix} \psi^h_{i,x} \\ 0 \\ \psi^h_{i,y} \end{bmatrix} \right) \cdot \begin{bmatrix} 0 \\ \psi^h_{i,y} \\ \psi^h_{i,x} \end{bmatrix}$$

$$= \left(a_{2211}\psi^h_{i,x} + a_{2212}\psi^h_{i,y}\right)\psi^h_{i,y} + \left(a_{1211}\psi^h_{i,x} + a_{1212}\psi^h_{i,y}\right)\psi^h_{i,x} \tag{5.3.31}$$

$$\left(B_{\ell,K_\delta}\right)^{21}_{ij} = \left(\begin{bmatrix} a_{1111} & a_{1122} & a_{1112} \\ a_{2211} & a_{2222} & a_{2212} \\ a_{1211} & a_{1222} & a_{1212} \end{bmatrix} \begin{bmatrix} 0 \\ \psi^h_{i,y} \\ \psi^h_{i,x} \end{bmatrix} \right) \cdot \begin{bmatrix} \psi^h_{i,x} \\ 0 \\ \psi^h_{i,y} \end{bmatrix}$$

$$= \left(a_{1122}\psi^h_{i,y} + a_{1112}\psi^h_{i,x}\right)\psi^h_{i,x} + \left(a_{1222}\psi^h_{i,y} + a_{1212}\psi^h_{i,x}\right)\psi^h_{i,y} \tag{5.3.32}$$

$$\left(B_{\ell,K_\delta}\right)_{ij}^{22} = \left(\begin{bmatrix} a_{1111} & a_{1122} & a_{1112} \\ a_{2211} & a_{2222} & a_{2212} \\ a_{1211} & a_{1222} & a_{1212} \end{bmatrix} \begin{bmatrix} 0 \\ \psi_{i,y}^h \\ \psi_{i,x}^h \end{bmatrix}\right) \cdot \begin{bmatrix} 0 \\ \psi_{i,y}^h \\ \psi_{i,x}^h \end{bmatrix}$$

$$= \left(a_{2222}\psi_{i,y}^h + a_{2212}\psi_{i,x}^h\right)\psi_{i,y}^h + \left(a_{1222}\psi_{i,y}^h + a_{1212}\psi_{i,x}^h\right)\psi_{i,x}^h \tag{5.3.33}$$

The component matrices of $\boldsymbol{B}_{\ell,K_\delta}$ are evaluated using the equations (5.3.30) – (5.3.33). The integration is done using the symmetric Gaussian quadrature rule from [13] (see also, [17])

The MATLAB routine `MicroStiffnessElasticity` is given below.

We note that in the routine below, we have used a few basic routines from [17] (but renamed). `RefTriangle` creates a reference triangle with $(0,0), (1,0)$ and $(0,1)$ as vertices. `QuadPts` returns the quadrature points and weights which are based on [13]. `getNodes` returns the coordinates of the nodes in a triangle, in this case, for the reference triangle. `BasisFcnVals(inodes,enodes)` evaluates the values of the basis functions at `enodes` for a triangle having nodal coordinates `inodes`. Similarly, `BasisFcnGradVals(inodes,enodes)` evaluates the values of the gradients of the basis functions. We refer the reader to [17] for more details regarding these routines.

```
function [AMicro,b]=MicroStiffnessElasticity(T,A_tensor,epsilon,Qpt)

% Qpts are the coordinates of the MACRO quadrature nodes and
% qpts are the coordinates of the MICRO quadrature nodes

% Note: The function MicroStiffness returns the vector b along
% with micro stiffness matrix Amicro. b is computed inside
% MicroStiffness as it requires V, qwts and trans.j which
% are already computed within the function.

d=T.Degree;
id=round((d+2)*(d+1)/2);
Nt=size(T.Elements,1);
Nv=size(T.Nodes,1);

indexi=zeros(4*id*id*Nt,1);
indexj=zeros(4*id*id*Nt,1);
vals=zeros(4*id*id*Nt,1);

TR=RefTriangle(d);
[qpts_ref,qwts]=QuadPts(2*d-2);
npts=length(qwts);
```

```
inodes=getNodes(TR,1);
[Vs,Vt]=BasisFcnGradVals(inodes,qpts_ref);

b=zeros(1,2*Nv);

% Evaluate Basis function values at qpts, which are required
% for computing b.

V=BasisFcnVals(inodes,qpts_ref);


for i=1:Nt

   [coords,~,~,ll]=getNodesElasticity(T,i);

   vert=coords(1:d:2*d+1,1:2);

   trans=TransferToRefTri(vert);

   Vx=trans.JinvT(1,1)*Vs + trans.JinvT(1,2)*Vt;
   Vy=trans.JinvT(2,1)*Vs + trans.JinvT(2,2)*Vt;

   qpts=repmat(trans.z1,npts,1)+(trans.J*(qpts_ref'))';

   A11=feval(A_tensor{1,1},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;
   A12=feval(A_tensor{1,2},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;
   A13=feval(A_tensor{1,3},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;
   %A21=A12;
   A22=feval(A_tensor{2,2},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;
   A23=feval(A_tensor{2,3},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;
   %A31=A13;
   %A32=A23;
   A33=feval(A_tensor{3,3},qpts(:,1)/epsilon,qpts(:,2)/epsilon,...
                 Qpt(1),Qpt(2)).*qwts*trans.j;

   A11=sparse(diag(A11));
   A12=sparse(diag(A12));
   A13=sparse(diag(A13));
   %A21=A12;
   A22=sparse(diag(A22));
   A23=sparse(diag(A23));
   %A31=A13;
   %A32=A23;
   A33=sparse(diag(A33));
```

```
    % Compute the partitions of B
    B11= (A11*Vx + A13*Vy)'*Vx + (A13*Vx + A33*Vy)'*Vy;
    B12= (A12*Vx + A23*Vy)'*Vy + (A13*Vx + A33*Vy)'*Vx;
    B21= (A12*Vy + A13*Vx)'*Vx + (A23*Vy + A33*Vx)'*Vy;
    B22= (A22*Vy + A23*Vx)'*Vy + (A23*Vy + A33*Vx)'*Vx;

    M=[B11 B12;B21 B22];

    ll=[ll;ll+Nv]';
    nll=length(ll);


    tempi=repmat(ll,1,nll);
    tempj=repmat(ll,nll,1);

    indexi(1+(i-1)*4*id*id:i*4*id*id)=tempi(:);
    indexj(1+(i-1)*4*id*id:i*4*id*id)=tempj(:);
    vals(1+(i-1)*4*id*id:i*4*id*id)=M(:);

    % Compute b, which shall be used to make the constraint matrix D

    phivals=trans.j*V'*qwts;
    b(ll)=[phivals;phivals];
end

AMicro=sparse(indexi,indexj,vals);
end
```

5.4 Assembling the Macrostiffness Matrix

We have finally computed the necessary components and can now compute and assemble the local macrostiffness matrix. We recall the expression for the macrostiffness matrix (5.2.4).

$$A_K = \sum_{\ell=1}^{\mathcal{L}} \frac{\omega_{\ell,K}}{\left|K_\delta\left(x_{\ell,K}\right)\right|}\left(\left(\boldsymbol{\alpha}_\ell^i\right)^T B_{\ell,K_\delta}\boldsymbol{\alpha}_\ell^j\right)$$

The local macrostiffness matrix comprises of contributions from the sampling domain K_δ around each quadrature point. The MATLAB code given below computes contributions from each sampling domain and adds it to the the local macrostiffness matrix for the element. Then, the local stiffness matrices are assembled as usual for the finite element method to give the final macrostiffness matrix.

We also note the implementation of isoparametric finite elements for curved edges and the computation of their Jacobians (which is not constant). For elements with straight edges, the routine `TransferToRefTri` (also from [17]) is used to compute the Jacobian.

The appropriate weights $\omega_{\ell,K}$ for each macro quadrature point are also computed and stored in the array `W`.

Finally we note that for periodic micro boundary conditions, solving the constrained equation for certain problems (such as Problem 1 of Chapter 6) results in a highly ill-conditioned system (with the condition number ranging from E+16 – E+22). This is a difficulty inherent in the problem and is overcome by forcefully increasing the precision number in MATLAB computations. The function `vpa` is used for this purpose which by default increases the precision number to 34 (or to any other degree using digits). This implies that although $12 - 20$ digits are lost in the computation, the remaining digits are recovered. However, the main drawback is that the computational cost is highly increased. But despite the very large condition number, there doesn't seem to be any problem with the formulation, as solving the ill conditioned system (versus solving against quadruple precision) produce almost exactly the same solution and error. This problem, however, does not occur while enforcing Dirichlet micro boundary conditions.

The MATLAB routine `MacroStiffnessElasticity` is given below

```
function AMacro=MacroStiffnessElasticity(T,A_tensor,bctype,...
                epsilon,delta,r,dmicro)

d=T.Degree;
id=round((d+2)*(d+1)/2);
Nt=size(T.Elements,1);
NfX=length(T.FNodePtrsX);

indexi=zeros(4*id*id*Nt,1); % Maximum size
indexj=zeros(4*id*id*Nt,1); % Maximum size
vals=zeros(4*id*id*Nt,1);   % Maximum size

TR=RefTriangle(d);
[Qpts_ref,Qwts]=QuadPts(2*d-2);
Npts=length(Qwts);

inodes=getNodes(TR,1);
VQpts=BasisFcnVals(inodes,Qpts_ref);
[Vs,Vt]=BasisFcnGradVals(inodes,Qpts_ref);
```

```
nllp0=0;

for i=1:Nt

    % By convention, it is made sure that all curved edges
    % are the second edge in T.Elements.

    CurvedEdge=T.EdgeCFlags(abs(T.Elements(i,2))) && d>1;

    [coords,llx,lly,~]=getNodesElasticity(T,i);
    vert=coords(1:d:2*d+1,1:2);

    if CurvedEdge

        Qpts=VQpts*coords;

        Vsx=Vs*coords(:,1);
        Vsy=Vs*coords(:,2);
        Vtx=Vt*coords(:,1);
        Vty=Vt*coords(:,2);
        grad=zeros(2,id,Npts);

        detJ=zeros(Npts,1);

        for ii=1:Npts
            J=[Vsx(ii),Vtx(ii);Vsy(ii),Vty(ii)];
            grad(:,:,ii)=J'\[Vs(ii,:);Vt(ii,:)];
            detJ(ii)=abs(det(J));
        end

        %Calculate Macro weights:
        W=detJ.*Qwts;
    else

        trans=TransferToRefTri(vert);
        Qpts=repmat(trans.z1,Npts,1)+(trans.J*(Qpts_ref'))';

        Vx=trans.JinvT(1,1)*Vs + trans.JinvT(1,2)*Vt;
        Vy=trans.JinvT(2,1)*Vs + trans.JinvT(2,2)*Vt;

        grad=zeros(2,id,Npts);
        grad(1,:,:)=Vx';
        grad(2,:,:)=Vy';

        %Calculate Macro Quadrature weights
        W=trans.j*Qwts;

    end

    A=zeros(2*id,2*id);
```

```
%Loop over each quadrature point
for j=1:Npts

    TMicro=MakeMicroMeshElasticity(Qpts(j,:),delta,...
                   bctype,r,dmicro);
    [AMicro,b]=MicroStiffnessElasticity(TMicro,A_tensor,...
                   epsilon,Qpts(j,:));
    switch(lower(bctype))
        case{'periodic'}
            [D_tilda BetaD]=MicroConstraintsPeriodicElasticity...
                            (TMicro,Qpts(j,:),VQpts,grad,j);
            D=[b;D_tilda];
        case{'dirichlet'}
            [D BetaD]=MicroConstraintsDirichletElasticity...
                            (TMicro,Qpts(j,:),VQpts,grad,j);
    end

    AConstr=[AMicro D';D sparse(size(D,1),size(D,1))];
    RhsConstr=[zeros(2*size(TMicro.Nodes,1),2*id);BetaD];

    switch(lower(bctype))
        case{'periodic'}
            % digits(50);
            % soln=vpa(AConstr)\vpa(RhsConstr);
            soln=AConstr)\RhsConstr;
        case{'dirichlet'}
            soln=AConstr\RhsConstr;
    end

    alpha=soln(1:2*size(TMicro.Nodes,1),:);

    %Calculate Weights for each Macro Qpt
    K_macro=W(j);
    K_micro=delta^2;

    %Add contribution of each Qpt to A(2*id,2*id)

    A= A + K_macro/K_micro *alpha'*AMicro*alpha;
end

%Assemble using llp for only Free Nodes

lly(lly>0)= lly(lly>0) + NfX;
ll=[llx;lly]';
llp=ll(ll>0);

nllp=length(llp);
tempi=repmat(llp,1,nllp);
tempj=repmat(llp,nllp,1);
```

```
    indexi(1+nllp0:nllp0+nllp*nllp)=tempi(:);
    indexj(1+nllp0:nllp0+nllp*nllp)=tempj(:);

    A(ll<0,:)=[];
    A(:,ll<0)=[];

    vals(1+nllp0:nllp0+nllp*nllp)=A(:);

    nllp0=nllp0+nllp*nllp;
end

% Delete extra rows from the initial estimated size
indexi=indexi(1:nllp0,:);
indexj=indexj(1:nllp0,:);
vals=vals(1:nllp0,:);

% Assemble the macrostiffness matrix
AMacro=sparse(indexi,indexj,vals);

end
```

This ends Chapter 5 and the MATLAB implementation of the FE-HMM algorithm. The next Chapter presents the numerical experiments of multiscale problems in linear elasticity that are solved using FE-HMM.

CHAPTER 6
NUMERICAL EXPERIMENTS

We shall conduct numerical experiments for two problems within the dissertation. The first problem assumes that the material property varies along only one direction and that the composite is isotropic. Although this is practically not probable, it is done so that we can explicitly compute the homogenized solution. The second problem does not assume any isotropy (i.e. anisotropic) and the material property varies along both directions and is non-uniformly periodic.

Before we give the results we first present an expression for the homogenized tensor for a linear elastic, isotropic composite material whose material property varies along only one direction. For more details we refer to [20].

Property 6.1. *Consider the problem (3.1.4), where the tensor A^{ε} is assumed to represent a linear elastic isotropic material in plane strain condition. Further, assume that A^{ε} satisfy (3.1.12) – (3.1.15) and its coefficients vary along only one direction. Then, the coefficients of the corresponding homogenized tensor A^0 satisfying equation (3.1.20) is given by*

$$\begin{gathered} a^0_{1111} = \frac{1}{\mathcal{M}_Y\left({}^{1}/_{a_{1111}}\right)}, \qquad a_{1112} = a_{1222} = 0, \\ a^0_{1122} = \frac{\mathcal{M}_Y\left({}^{a_{1122}}/_{a_{1111}}\right)}{\mathcal{M}_Y\left({}^{1}/_{a_{1111}}\right)}, \qquad a^0_{1212} = \frac{1}{\mathcal{M}_Y\left({}^{1}/_{a_{1212}}\right)}, \\ a^0_{2222} = \mathcal{M}_Y(a_{2222}) - \mathcal{M}_Y\left(\frac{a^2_{1122}}{a_{1111}}\right) + \frac{\left(\mathcal{M}_Y\left({}^{a_{1122}}/_{a_{1111}}\right)\right)^2}{\mathcal{M}_Y\left({}^{1}/_{a_{1111}}\right)}. \end{gathered} \tag{6.1}$$

where the components of $A(y_2)$ are given by

$$\begin{gathered} a_{1111} = a_{2222} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}, \qquad a_{1122} = \frac{E\nu}{(1+\nu)(1-2\nu)} \\ a_{1212} = \frac{E}{2(1+\nu)}, \qquad a_{1112} = a_{1222} = 0 \end{gathered} \tag{6.2}$$

and $E = E(y_2), \nu = \nu(y_2)$ are the Young's modulus and Poisson's ratio of the material respectively.

Problem 1. Consider the problem (3.1.4), where the domain Ω is defined as a two dimensional region occupied by the points $(0,0)$, $(4800,4400)$, $(0,4400)$, $(4800,6000)$, where the units of distance is mm. Dirichlet boundary conditions are imposed at the end $x = 0$, and is body is subjected to a shearing load $\mathbf{g} = (0,5)\mathrm{N/mm^2}$ at the end $x = 4800$. The volume force $\mathbf{f}$ is assumed to vanish and the material is assumed to be linear elastic and isotropic. Further, A^{ε} is assumed to satisfy (3.1.12) – (3.1.15) and its coefficients vary along only one direction. Finally, the Young's modulus and Poisson's ratio are defined by the functions

$$E(y_2) = 8.35\left(\frac{1 + (0.2 + 0.1\cos(2\pi y_2))}{(0.2 + 0.15\sin(2\pi y_2))}\right)\mathrm{KN/mm^2}, \tag{6.3}$$

$$\nu(y_2) = 0.2 + 0.1\cos(2\pi y_2). \tag{6.4}$$

We note that the $E(y_2)$ and $\nu(y_2)$ are chosen such that their values have the bounds

$$\max\big(E(y_2)\big) = 200.2631\,\mathrm{KN/mm^2} \text{ and } \min\big(E(y_2)\big) = 28.3971\,\mathrm{KN/mm^2}$$

$$\max\big(\nu(y_2)\big) = 0.3 \text{ and } \min\big(\nu(y_2)\big) = 0.1$$

The functions are chosen so that the material might be thought of as practically representing a composite consisting of M28 concrete and Fe415 steel.

A plot of $E(y_2)$ is given below.

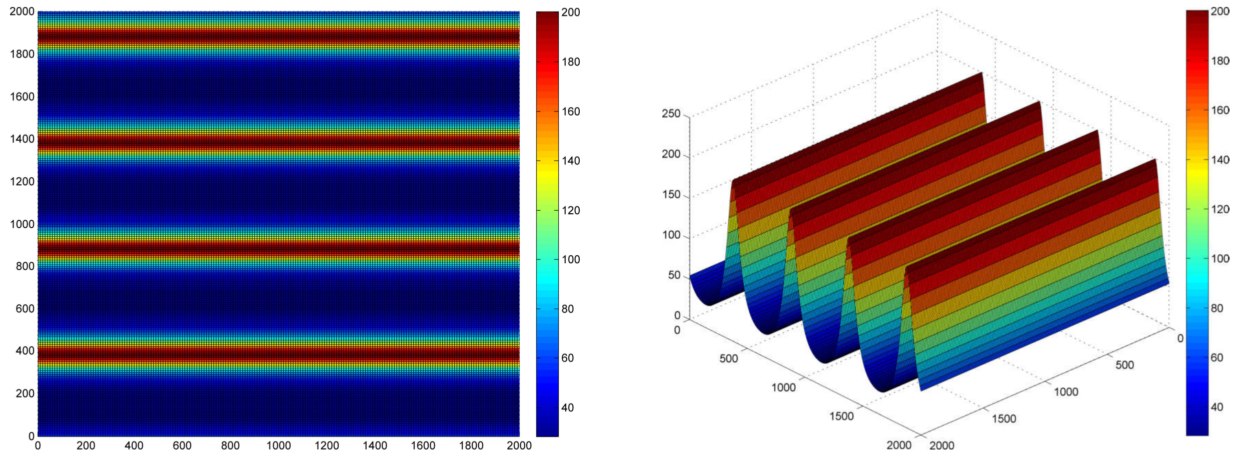


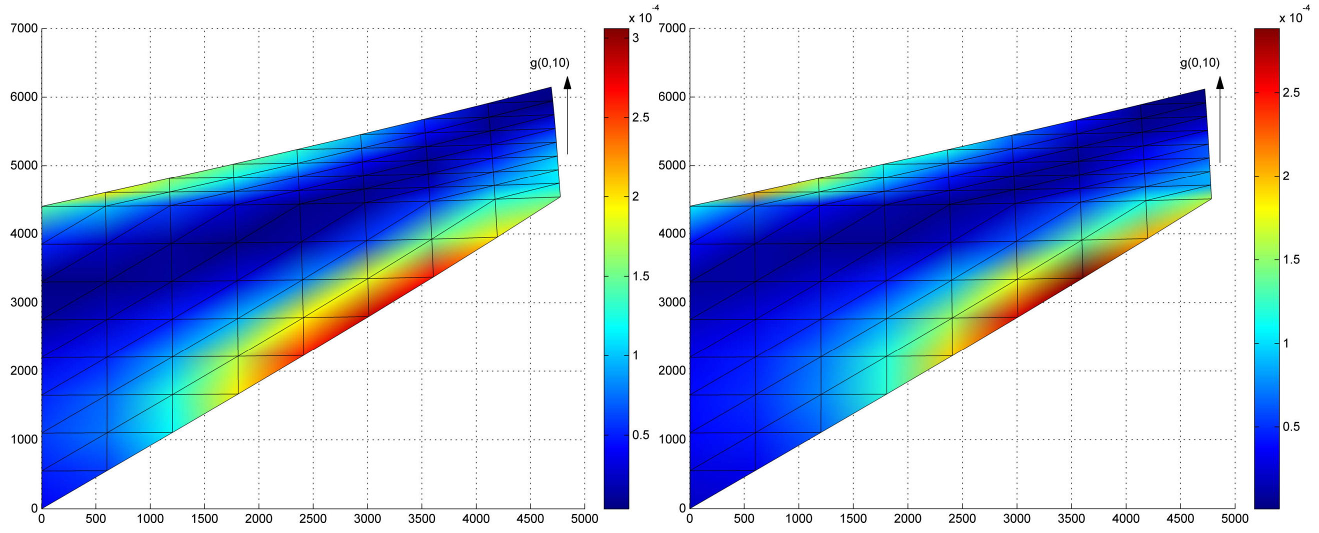
Figure 6.1. Plot of $E(y_2)$ for Problem 1.

From (6.1), we get the homogenized coefficients and the tensor A^0 can be written (in Voigt Notation) as

$$A^0 = \begin{pmatrix} 56.4099 & 14.6599 & 0 \\ 14.6599 & 83.2061 & 0 \\ 0 & 0 & 20.8750 \end{pmatrix}$$

The homogenized solution $\mathbf{u}^0$ is obtained by solving the homogenized system with A^0 as given above. The error of the FE-HMM solution with respect to the homogenized solution is given in Figure 6.4. First we show the deformed mesh for the problem.

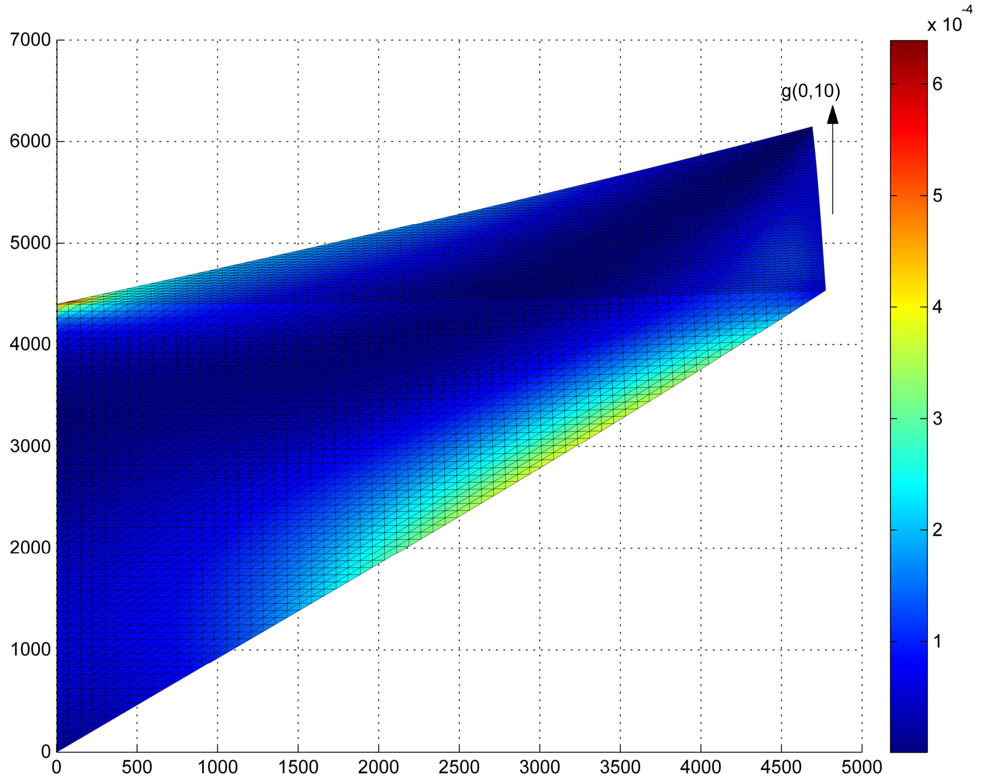
Figures 6.2 and 6.3 show the deformed mesh (multiplied by a factor of 10) for the Problem 1. The colour tones depict the strain energy density across the domain. For a mesh with a larger degree of freedom, the effect of the oscillating material property of the composite (Figure 6.1) can be seen by the corresponding oscillation of the strain energy density.



a) Homogenized solution

b) FE-HMM solution

Figure 6.2. Deformed mesh for Problem 1 with 128 elements.



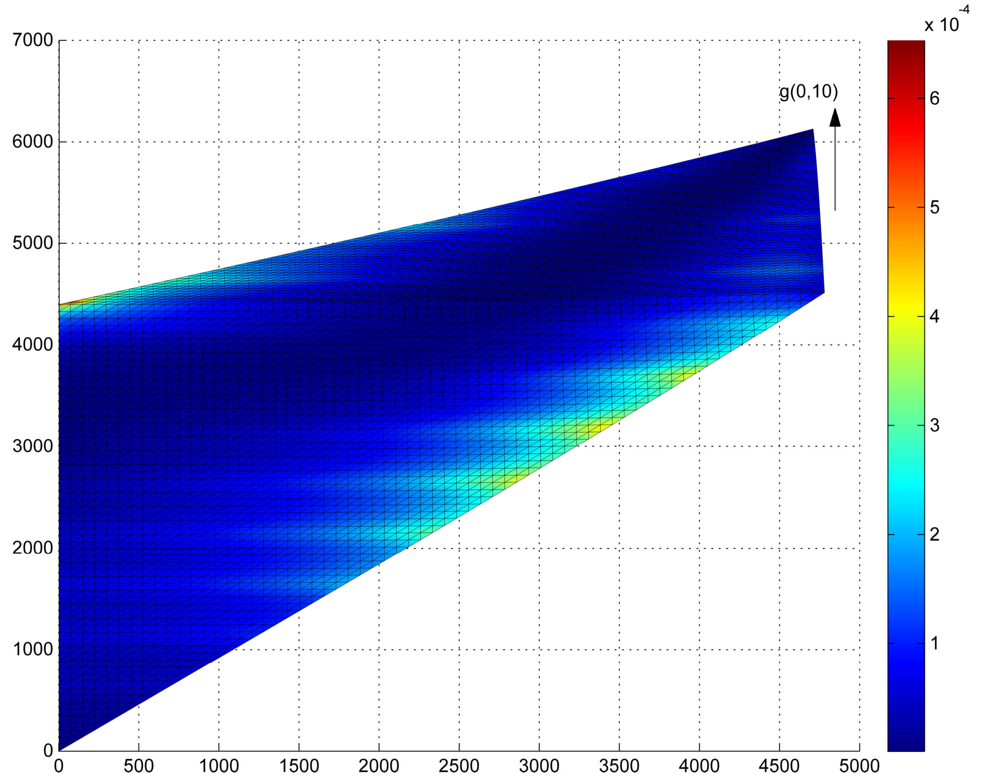


Figure 6.3. Deformed mesh with tones showing strain energy density across domain for homogenized solution vs. FE-HMM solution for a mesh with 8192 elements.

The estimates (4.2.2) imply linear and quadratic convergence rates for $h \propto H^{1/2}$ or $n \propto N^{1/2}$ in the H^1-norm and $h \propto H$ or $n \propto N$ in the L^2 norm respectively, where n and N are the total degree of freedoms of the micro and micro discretization. This can be seen using P^1 macro and micro spaces for Problem 1 as shown in Figure 6.4. However, convergence rates for P^2 macro and micro spaces cannot be seen as the solution $\mathbf{u}^0$ is not smooth enough.

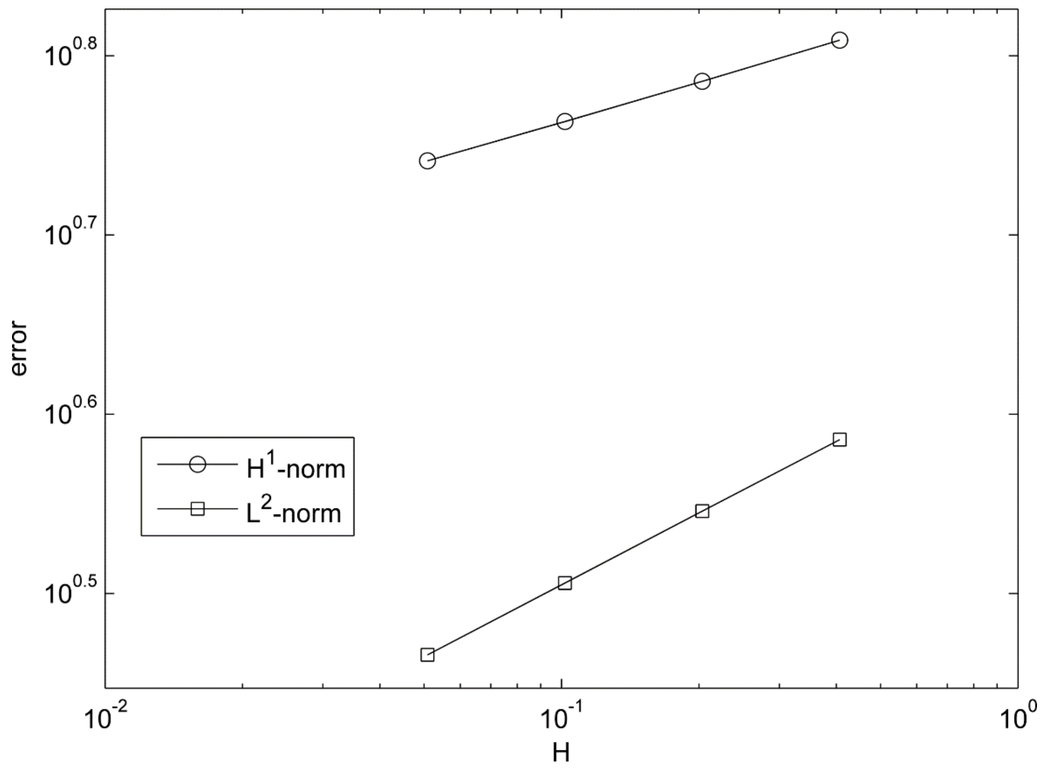


Figure 6.4. Error between $\mathbf{u}^0$ and $\mathbf{u}^H$ for Problem 1. (H before refinement=6.5115m)

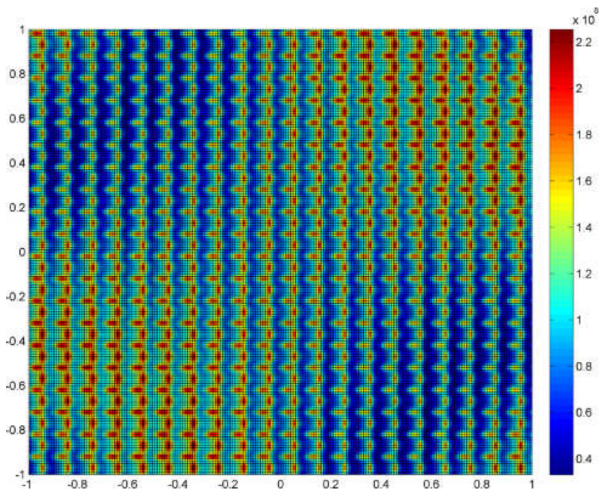
Problem 2. Consider the problem (3.1.4), where the domain is defined by $\Omega = (0,1) \times (1,0) \backslash B(0,0.5)$, where the units of distance is metres. Dirichlet boundary conditions in the x-dimension are imposed at the end $x = 0$, and in the y-dimension at $y = 0$. The body is subjected to s shearing load $\mathbf{g} = (1\mathrm{E}+5,0)\mathrm{KN/m^2}$ at the end $y = 1$ and the volume force $\mathbf{f}$ is assumed to vanish and the material is assumed to be linear elastic and isotropic. Further, A^{ε} is assumed to satisfy (3.1.12) – (3.1.15) and that its coefficients are non-uniformly periodic. The coefficients of A^{ε} are given by

$$a_{1111} = a_{2222} = (3\mathrm{E}+7)\left(\frac{1.5+\sin(2\pi x/\varepsilon)}{1.5+\sin(2\pi y/\varepsilon)} + \frac{1.5+\sin(2\pi y/\varepsilon)}{1.5+\cos(2\pi x/\varepsilon)} + \sin(4xy) + 1\right)$$

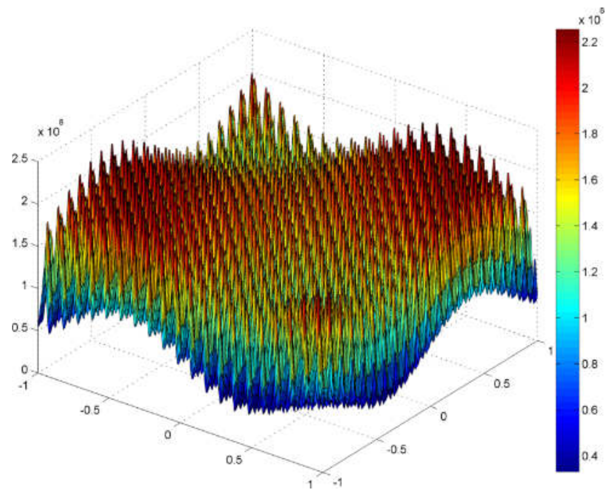
$$a_{1122} = (1\mathrm{E}+7)\left(\frac{1.5+\sin(2\pi y/\varepsilon)}{1.5+\cos(2\pi x/\varepsilon)} + \sin(4xy)\right) \tag{6.5}$$

$$a_{1212} = (1\mathrm{E}+7)\left(2.1 + \frac{2}{1.5+\cos(2\pi x/\varepsilon)} + \sin(4xy)\right)$$

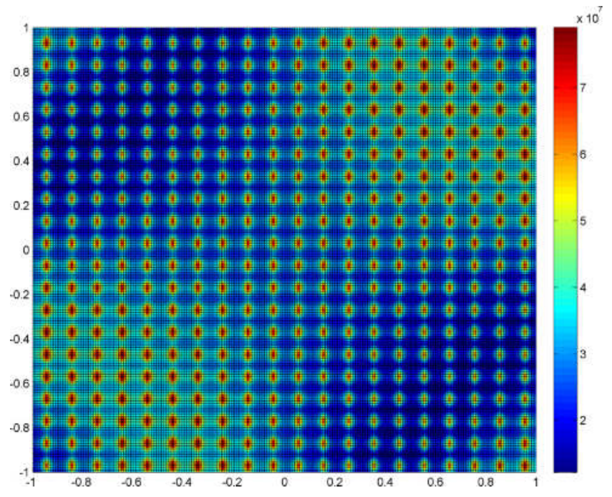
A plot of a_{1111},a_{1122} and a_{1212} are given below.



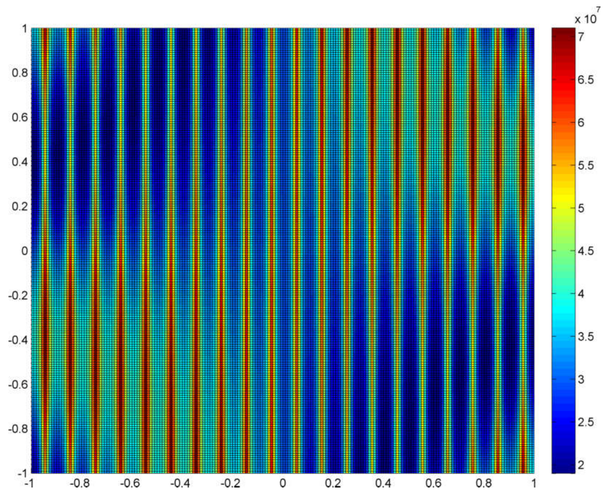
(a) a_{1111}



(b) a_{1111} 3D view



(c) a_{1122}



(d) a_{1212}

Figure 6.5. Plot of tensor coefficients for Problem 2.

The deformed mesh for Problem 2 is given in Figure 6.9 below.

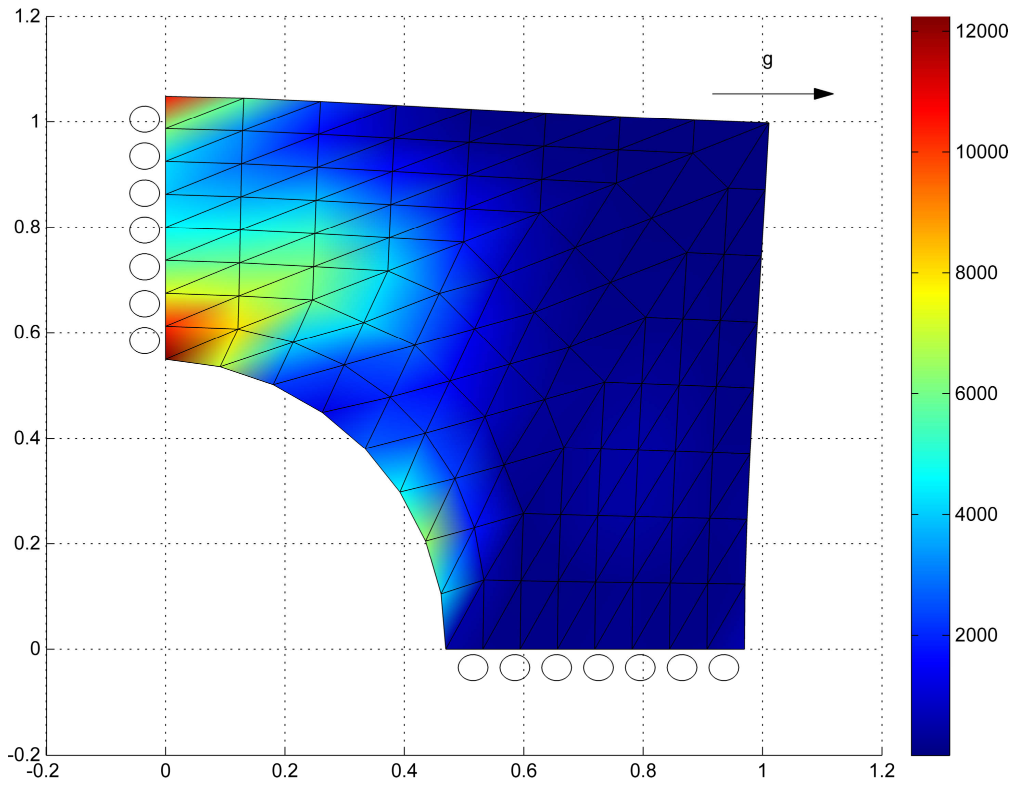


Figure 6.6. Deformed mesh for the FE-HMM solution of Problem 2.

Problem 2 is solved using a classical finite element with a mesh with a large degree of freedom of 7.87968E+5. The energy and infinity norms of the fine scale solution is given below in Table 6.1. The energy and infinity norms of solutions using the FE-HMM by changing the macro and micro degree of freedoms, degree of spaces etc. are given in Table 6.2 and can be compared to the fine scale solution.

fine scale solution	N=7.87968E+5, p=4
$\|\|\mathbf{u}\|\|_E$	42.4995
$\|\|\mathbf{u}\|\|_\infty$	0.0124173

Table 6.1. The energy and infinity norms for the fine scale solution.

n=9	N=9	N=25	N=81	N=289
$\|\|\mathbf{u}\|\|_E$	22.1207	30.5297	36.7621	39.6875
$\|\|\mathbf{u}\|\|_\infty$	0.00405464	0.00733412	0.0100366	0.0112694

(a)

N=25	n=9	n=25	n=81	n=289
$\|\|\mathbf{u}\|\|_E$	30.5297	30.8687	30.9272	31.2386
$\|\|\mathbf{u}\|\|_\infty$	0.00733412	0.00753471	0.00756353	0.00773842

(b)

N=25, n=25	p=1, q1	p=2, q=2	p=3, q=3	p=4, q=4
$\|\|\mathbf{u}\|\|_E$	30.5297	39.7142	40.642	41.475
$\|\|\mathbf{u}\|\|_\infty$	0.00733412	0.0114865	0.0118599	0.0120318

(c)

Table 6.2. The energy and infinity norms for the FE-HMM solutions.

CONCLUSION

We have implemented the Finite Element Heterogeneous Multiscale Method for problems in linear elasticity. Two problems have been solved using the FE-HMM and their results have been compared with the homogenized and fine scale solutions. We have verified the convergence rates for the first problem as predicted by theoretical estimates. Future work in this area may include analysis and implementation of the FE-HMM for plate bending problems and problems in non-linear elasticity. Also, ideas from the FE-HMM may also be adapted to develop a multiscale method for analyzing fracture and crack propagation in composite materials.

REFERENCES

[1] A. Abdulle. *Analysis of a Heterogeneous Multiscale FEM for Problems in Elasticity*. Math. Models Methods Appl. Sci. 16:4 (2006), pp. 615–635.

[2] A. Abdulle. *On A Priori Error Analysis of Fully Discrete Heterogeneous Multiscale FEM*. SIAM Multiscale Mod. Simul., 4(2) (2005), pp. 447–459.

[3] A. Abdulle. *The finite element heterogeneous multiscale method: a computational strategy for multiscale PDEs*. GAKUTO Int. Series, Math. Sci. Appl., 31 (2009), pp. 135–181.

[4] A. Abdulle and B. Engquist. *Finite element heterogeneous multiscale methods with near optimal computational complexity.* SIAM Multiscale Modeling & Simulation, 6(4) (2007), pp. 1059– 1084.

[5] A. Abdulle and A. Nonnenmacher. *A short and versatile finite element multiscale code for homogenization problems*. Computer Methods in Appl. Mech. and Engrg, 198(37-40) (2009), pp. 2839–2859.

[6] A. Abdulle and Ch. Schwab. *Heterogeneous Multiscale FEM for Diffusion Problems on Rough Surfaces*. SIAM Multiscale Mod. Simul., 3(1) (2005), pp. 195–220.

[7] J. Alberty, C. Carstensen, and S.A. Funken. *Remarks around 50 lines of Matlab: short finite element implementation*. Numerical Algorithms, 20(2) (1999), pp. 117–137.

[8] J. Alberty, C. Carstensen, S. A. Funken, and R. Klose. *Matlab implementation of the finite element method in elasticity*. Computing, 69(3) (2002), pp. 239–263.

[9] I. Babuška, G. Caloz, and J.E. Osborn. *Special finite element methods for a class of second order elliptic problems with rough coefficients*. SIAM Journal on Numerical Analysis, 31(4) (1994), pp. 945–981.

[10] S. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, Berlin, 2008.

[11] P. G. Ciarlet, *Mathematical Elasticity, Vol. I: Three-Dimensional Elasticity*. North-Holland, 1988.

[12] D. Cioranescu and P. Donato. *An Introduction to Homogenization*, Oxford Lecture Series in Mathematics and Its Applications, 17, Oxford University Press, USA, 2000.

[13] D. A. Dunavant. *High degree efficient symmetrical Gaussian quadrature rules for the triangle*. Int. J. Numer. Methods Engrg., 21 (1985), pp. 1129-1148.

[14] W. E, P. Ming, and P. Zhang. *Analysis of the heterogeneous multiscale method for elliptic homogenization problems*. AMS, 18(1) (2005), pp. 121–156.

[15] W. E and B. Engquist. *The Heterogeneous Multi-Scale Methods*. Comm. Math. Sci., 1 (2003), pp. 87-132.

[16] Y. Efendiev and T. Y. Hou. *Multiscale Finite Element Methods: Theory and Applications*. Springer, Berlin, 2009.

[17] M. S. Gockenbach. *Understanding and Implementing the Finite Element Method.* SIAM, 2006.

[18] O. Jecker and A. Abdulle. Numerical experiments for multiscale problems in linear elasticity. ENUMATH, 2015.

[19] O. A. Oleinik, A. S. Shamaev and G. A. Yosifian, *Mathematical problems in elasticity and homogenization.* North-Holland, Amsterdam, 1992.

[20] M. Patrício, R. Mattheij and G. de With. *Homogenisation with application to layered materials*. Math. Comp. Simul., 79 (2008), pp. 288-305.