Cyber-physics intrinsic modelling for smart systems

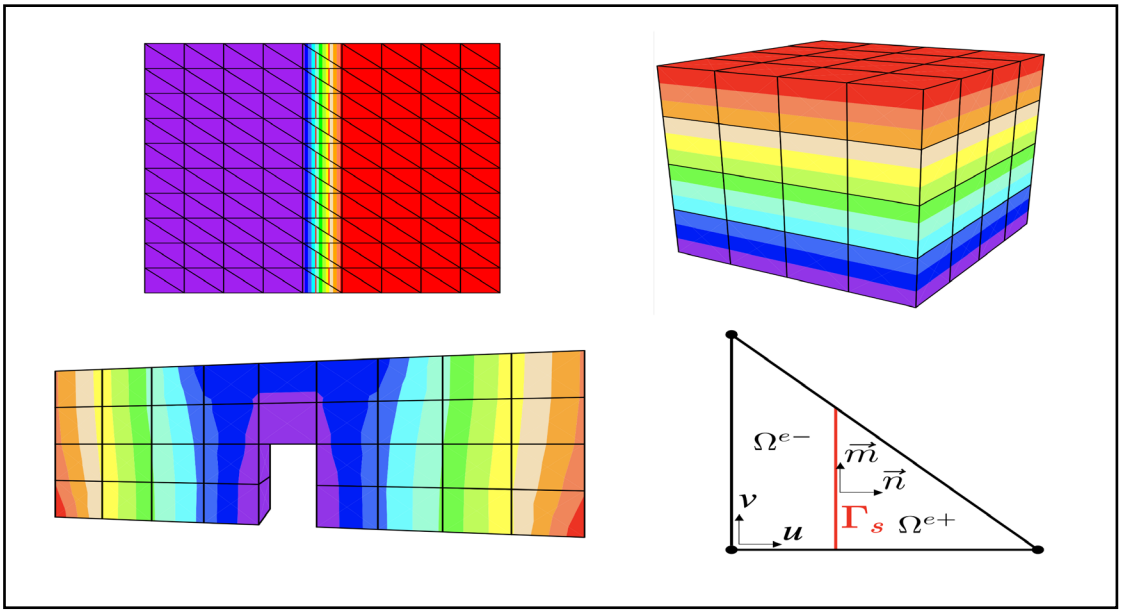
Ivan Rukavina

Université de Technologie Compiègne

Par **Ivan RUKAVINA**

Cyber-physics intrinsic modeling for smart systems

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 29 janvier 2021
Spécialité : Mécanique Numérique : Unité de recherche en Mécanique - Laboratoire Roberval (FRE UTC - CNRS 2012)
D2581

UNIVERSITÉ DE TECHNOLOGIE DE COMPIÈGNE

Ivan Rukavina

CYBER-PHYSICS INTRINSIC MODELLING FOR SMART SYSTEMS

DOCTORAL THESIS

Supervisor:

Prof. Adnan Ibrahimbegovic

29 / 01 / 2021

Compiègne, 2021

DOCTORAL THESIS

Cyber-physics intrinsic modeling for smart systems

Ivan Rukavina

Université de Technologie de Compiègne
Alliance Sorbonne Université

This doctoral thesis was defended on 29/01/2021 at Université the Technologie de Compiègne, in front of the following committee members:

Reviewers:

- Prof. Jose Luis Pérez-Aparicio, Universitat Politècnica de València, Spain
- Prof. Hermann Matthies, Technische Universität Braunschweig, Germany

Members of the jury:

- Prof. Pierre Villon, Université de Technologie de Compiègne, France
- Prof. Anna Kučerová, Czech Technical University in Prague, Czech Republic
- Prof. Alejandro Ospina, Université de Technologie de Compiègne, France
- Prof. Adnan Ibrahimbegovic, Université de Technologie de Compiègne, France

Acknowledgements

The work on this thesis has been made possible jointly by Haut-de-France Region (CR Picardie) (120-2015-RDISTRUCT-000010 and RDISTRUCT-000010) and EU funding (FEDER) for Chaire-de-Mécanique (120-2015-RDISTRUCTF-000010 and RDISTRUCTI-000004). This support is gratefully acknowledged.

There are so many people who were a part of my life during these several years of doctoral thesis, who helped me in many different ways and with whom I shared a lot of great memories for which I'm very grateful.

First of all, I would like to thank my supervisor, professor Adnan Ibrahimbegovic, for his guidance through this thesis. He was always willing to selflessly share his knowledge and expertise.

I would also like to thank the reviewers, professors Hermann Matthies and Jose Luis Pérez-Aparicio, for reading my manuscript and giving me useful feedback. Many thanks to the members of the jury, professors Ana Kučerová, Pierre Villon, and Alejandro Ospina, for all their questions and comments.

Special thanks to professor Delphine Brancherie, for her support and help during my PhD. I also thank all the professors and colleagues from Roberval laboratory and équipe numérique, especially Brigitte for always being there for us.

I am grateful to my friends and colleagues from the Big Office: Pablo for sharing the best coffee every morning (until the Penguin got electrocuted), Adela for all the talks and bike rides, and Uy for not letting the black sheep out of the drawer. Thank you for all the moments we spent together, especially for hunting down the white rabbits to the sounds of *La Traviata*.

I also thank my friends and running teammates Jishuai and Corentin, for all those runs in the rain. To Florian and Maria, who gave me a place to stay when I first came to Compiègne. Also, to the best futsal team at UTC, especially, Alan, Phong, Khalil, and Sitou.

I met some great new friends on this journey, and shared so many beautiful moments with them: Emina, Ismar, Emir, Sara, Milan, Daniel, Abir, Edo, Mijo, Simona, Samir, Nam Do, Luis, Edmundo, Arturo, and many others. These last years would just not be the same without you.

There are some friends who could not be close all the time but who have been always there for me. Thank you Daniela for all the motivation and support, and Mladen for finding together the concept of the funniest thing in the world. Also, to Filip for all the laughs, and Denis and Denis for all the philosophical discussions.

I would like to give a special thanks to my family - my mother and father, and my brother, for all their support, and for teaching me valuable life lessons that made me who I am today.

And last but not least, I would like to thank Tea, for all her love, and for believing in me when I had a hard time believing in myself.

Abstract

In this thesis, a multi-scale and multi-physics coupling computation procedure for a 2D and 3D setting is presented. When modeling the behavior of a structure by a multi-scale method, the macro-scale is used to describe the homogenized response of the structure, and the micro-scale to describe the details of the behavior on the smaller scale of the material where some inelastic mechanisms, like damage or plasticity, can be taken into account. The micro-scale mesh is defined for each macro-scale element in a way to fit entirely inside it. The two scales are coupled by imposing a constraint on the displacement field over their interface. The computation is performed using the operator split solution procedure on both scales, using the standard finite element method.

In a 2D setting, an embedded discontinuity is implemented in the Q4 macro-scale element to capture the softening behavior happening on the micro-scale. For the micro-scale element, a constant strain triangle (CST) is used. In a 3D setting, a macro-scale tetrahedral and hexahedral elements are developed, while on the micro-scale Timoshenko beam finite elements are used.

This multi-scale methodology is extended with a multi-physics functionality, to simulate the behavior of a piezoelectric material. An additional degree of freedom (voltage) is added on the nodes of the 3D macro-scale tetrahedral and hexahedral elements. For the micro-scale element, a Timoshenko beam element with added polarization switching model is used. Also, a multi-scale Hellinger-Reissner formulation for electrostatics has been developed and implemented for a simple electrostatic patch test.

For implementing the proposed procedure, Finite Element Analysis Program (FEAP) is used. To simulate the behavior on both macro and micro-scale, FEAP is modified and two different version of FEAP code are implemented - macroFEAP and microFEAP. For coupling, the two codes are exchanging information between them, and Component Template Library (CTL) is used.

The capabilities of the proposed multi-scale approach in a 2D and 3D pure mechanics settings, but also multi-physics environment have been shown. The theoretical formulation and algorithmic implementation are described, and the advantages of the multi-scale approach for modeling heterogeneous materials are shown on several numerical examples.

Résumé

Dans le cadre de cette thèse, une approche de calcul de couplage multi-échelle et multi-physique en 2D et en 3D est présentée. La modélisation multi-échelle d'une structure consiste de l'échelle macro qui représente la réponse homogénéisée de la structure entière, tandis que l'échelle micro peut capturer les détails du comportement à la petite échelle du matériau, où des mécanismes inélastiques, tels que la plasticité ou l'endommagement, peuvent être pris en compte. L'intérieur de chaque macro-élément est rempli par le maillage à l'échelle micro qui s'y adapte entièrement. Les deux échelles sont couplées à travers le champ de déplacements imposé à l'interface. Le calcul par éléments finis est effectué, en utilisant une procédure de solution *operator-split* sur les deux échelles.

En 2D, une discontinuité dans le champ de déplacements est introduite à l'échelle macro dans un élément fini Q4, pour pouvoir capturer l'adoucissement qui se produit à l'échelle micro, où des éléments finis triangulaires (*CST - Constant Strain Triangle*) sont utilisés. En 3D, des éléments finis de tétraèdre et d'hexaèdre sont développés à l'échelle macro, pendant que des poutres de Timoshenko sont utilisées à l'échelle-micro.

Cette méthodologie multi-échelle est étendue avec des fonctionnalités multi-physiques pour simuler le comportement d'un matériau piézoélectrique. Un degré de liberté supplémentaire qui représente le voltage est ajouté aux nœuds des macro-éléments de tétraèdre et d'hexaèdre en 3D. La poutre de Timoshenko comportant un modèle de commutation de polarisation est utilisée à l'échelle micro. Également, une formulation multi-échelle de Hellinger-Reissner a été développée et implémentée pour un simple patch test en électrostatique.

La procédure proposée est mise en œuvre dans le logiciel de calcul par éléments finis *FEAP - Finite Element Analysis Program*. Pour simuler le comportement aux deux échelles, *FEAP* est modifié, et deux versions différentes du code sont obtenues - *macroFEAP* et *microFEAP*. Le couplage de ces codes est réalisé avec *Component Template Library - CTL* qui rend possible l'échange d'informations entre les deux échelles.

Les capacités de cette approche multi-échelle en 2D et en 3D sont démontrées dans un environnement purement mécanique, mais aussi multi-physique. La

formulation théorique et l'application algorithmique sont présentées, et les avantages de la méthode multi-échelle pour la modélisation des matériaux hétérogènes sont illustrés avec plusieurs exemples numériques.

Contents

1 Introduction

Contents

In this chapter, the motivation and research objectives of this thesis are stated. A general introduction and a literature review are given in the following three sections related to the multi-scale coupling, multi-physics coupling and software code coupling. Finally, the structure of the thesis is outlined in the last section.

1.1 Context and motivation

The study of smart systems and smart materials is an interdisciplinary field that includes different domains, from mechanics, electromagnetics and thermodynamics, to fluid mechanics and chemistry. In smart materials, some multi-physics coupling is present, i.e. in a piezoelectric material, mechanical and electrical domains are coupled. They are integrated in smart systems that take advantage of these effects to deliver the desired behavior, which is different from the traditional non-smart materials. For instance, piezoelectric materials can be used for vibration dampening and precise control of movement in machine tools, lenses, and mirrors (Bengisu and Ferrara, 2018).

Smart material behavior can be modeled from different aspects, and can have a focus on different material properties. Experts from each domain put an accent on the domain they know the best, and try to describe the other domain in terms of notions more familiar to them, which is often not the best or the most natural way. When experts from different domains collaborate and work on the same multi-physics coupling problems, it is often hard for them to find and define the same notation, formulation, methods and tools to be able to obtain optimal solutions.

Smart materials can also be modeled on different scales, from the atomic and crystal structure scale, through the micro and meso-scale, up to the macro-scale, where the behavior of the whole or a part of the structure is modeled. Multi-scale modeling of the material stipulates that the model is defined and that the simulation is carried out for two or more different scales. These scales can communicate during the simulation, or the results from one scale can be used to obtain better results on the other. Each of the scales can be modeled in a different way, using different methods, approximations or material behavior models, which presents a great advantage of this approach.

Numerical multi-scale and multi-physics modeling is carried out by using specialized scientific or commercial software. Some software allows to include different physics in the model, despite the fact that it is not always done in an optimal way. Other software does not have the possibility to include any other physics, and it is designed to tackle only problems from one specific domain. Nevertheless, such software is usually well tested, and gives optimal results for the chosen domain. Developing a new multi-scale or multi-physics formulation brings a question of which software to choose to implement it, and whether it is even possible to do it in a desirable way following the proposed theoretical formulation.

The general motivation and idea behind this thesis is to asses the problem of multi-scale and multi-physics numerical modeling for smart materials in an

interdisciplinary environment. In that way, the behavior of the material is modeled in each particular domain, and the formulation is posed according to the literature from that domain. A coupling solution procedure is then defined that allows for different domains or scales to communicate. Finally, software codes specialized for each domain are used and modified in such a way as to allow them to execute in parallel and provide the final solution of the problem.

There is a special focus on the software aspect, since the final goal is to develop a general software framework that is modular and is able to handle different numerical models, while coupling different scales and physical domains. That can be achieved by using and modifying existing software solutions rather then starting from scratch and writing a completely new software code. The development of such a framework would require many steps, and the work undertaken within the scope of this thesis, together with the achieved scientific contributions, will be exposed in the following chapters.

1.2 Research objectives

This thesis deals with code coupling for the multi-scale and multi-physics numerical modeling. The general idea is to reuse existing software codes, and couple them to execute in parallel to obtain the final solution. The starting point is to develop a numerical formulation for the multi-scale and multi-physics solution procedures using a partitioned approach.

Specifically, the research goals and objectives of this thesis are:

- Develop a multi-scale solution procedure for localized failure where an embedded discontinuity is implemented in the macro-scale element, and which is able to capture the softening behavior happening on the micro-scale;
- Extend the multi-scale solution procedure and implement it in a 3D setting, including the support for the localized failure on the macro-scale;
- Explore the possibilities of coupling different software codes for different physics for a multi-physics solution procedure;
- Develop a multi-scale solution procedure for a 3D setting that is able to capture the electro-mechanic coupling behavior on the micro-scale;
- Develop a modular multi-scale software that is easily upgradeable for new behavior models.

1.3 Literature review

1.3.1 Multi-scale coupling

Multi-scale coupling methods are numerical homogenization methods that are usually used to model the behavior of heterogeneous materials. The micro-scale is used to capture the details of the micro-structure and local fluctuations, while the macro-scale represents the homogenized behavior of the material for computing the global structural response (Ibrahimbegovic, 2009; Gloria, 2012). The homogenized macro-scale model is supported by the micro-scale model that can capture fine microscopic details. Macro-scale constitutive laws and equations are computed by averaging over the micro-scale values (Mei and Vernescu, 2010). There can be many scales defined, based on the characteristics of the material or structure. Those scales can be weakly or strongly coupled. In weak coupling, the scales do not communicate all the time during the analysis. The results obtained on one scale are used as averaged values for calculations on the other scale. In strong coupling, there is a constant communication between the scales and the computations on all of them progress simultaneously. Later in this thesis, strong coupling is used, but both weak and strong coupling will be described in this section, together with advantages and disadvantages of each approach.

Weak coupling

Numerous works in multi-scale coupling have been done using the weak coupling between the scales. One of the multi-scale homogenization methods that is often used to describe nonlinear material behavior of heterogeneous materials is the FE^2 method, like in (Feyel and Chaboche, 2000) or (Kouznetsova et al., 2001), where the scales are weakly coupled and the final solution can be constructed in two separate steps by using standard finite elements on both scales. Once the results are obtained on the micro-scale, a macro-scale analysis can be started using these results. This is one of the advantages of weak coupling, as there is no need for the scales to communicate during the whole analysis and exchange information, leading to simpler implementation. The main assumption when modeling the scales is that the micro-scale length scale is much smaller than the characteristic length over which the macro-scale loading varies in space as stated in (Geers et al., 2010).

The first and the most important step in the FE^2 method is to properly define a micro-structural representative volume element (RVE) that represents the constitutive behavior of the material, meaning that the constitutive equations are defined only on the micro-scale (Ibrahimbegovic, 2009). The representative volume element can be defined as a piece of the matrix material containing one

single heterogeneity (Kouznetsova et al., 2001). The macro-scale deformation gradient tensor is then transferred to the micro-scale in order to define a boundary value problem on a representative volume element. The micro-scale boundary value problem is then solved with a standard procedure, and after, the macroscopic stress tensor can be extracted using standard mathematical averaging equations, as shown in detail in (Geers et al., 2010) and (Kouznetsova, 2004). For example, in (Fish and Yu, 2001) three scales are defined: micro, meso and macro, as an attempt to account for the evolution of damage in heterogeneous microphases. The RVE is defined for the macro-scale and meso-scale, where the fields on each of these scales are calculated as the average over the RVE on the scale below. The FE^2 method can be used to model both plastic behavior as in (Feyel and Chaboche, 2000), and damage behavior as in (Ghosh et al., 2001).

The representative volume element is usually defined for a single point, typically chosen as the Gauss point of the macro element, as presented in (Feyel and Chaboche, 2000). For each of the Gauss points, a micro-scale mesh is defined (as shown in Figure 1.1a), where the micro-scale computations are executed. In this way, a heterogeneous material's behavior can be modeled in more detail and can be defined differently for each of the Gauss points, like presented in (Feyel and Chaboche, 2000) and (Geers et al., 2010). To compute the material response in a Gauss point, the macro-scale values of the stress and strain are calculated by averaging the values of the micro-scale stress and strain over the RVE volume, as shown in (Belytschko et al., 2008) and (Ladeveze and Nouy, 2003). The extension of the RVE multi-scale theory for taking into account the inertia and body forces is proposed in (De Souza Neto et al., 2015). It shows that the macro-scale inertia and body forces can be also computed as the volume average of the micro-scale inertia and body force fields over the representative volume element. In works of (De Souza Neto et al., 2015) and (Blanco et al., 2016), the Method of Multiscale Virtual Power is developed, where the energetic consistency between the two scales is established by requiring the stress virtual power to coincide with the volume average of the micro-scale.

Using weak coupling, different methods and finite element elements can be used on different scales, like the Voronoi cell finite element on the micro-scale in (Raghavan and Ghosh, 2004) or the XFEM method on both scales in (Loehnert and Belytschko, 2007). Also, in (Feyel, 2003), the FE^2 method is used to describe the response of highly non-linear structures using generalized continua. It was shown that the FE^2 method is adequate to build material models for generalized continua, as it does not need to solve difficult macro-scale analytical equations, but only the ones at the micro-scale which are modeled using a classical Cauchy continuum.

In (Özdemir et al., 2008), the FE^2 method is used for thermo-mechanical analysis of heterogeneous solids, where a temperature dependent non-linear thermo-mechanical response is accounted for by solving a boundary value problem at the micro-scale. The FE^2 method has been successfully applied to model a thermo-mechanical coupling problem in (Sengupta et al., 2012). In (Ladevèze et al., 2001), a multi-scale computational strategy is described that is based on the decomposition of the structure into an assembly of substructures and interfaces. An interface between the scales can transfer both the displacements and forces, and this model can successfully represent the linear-elastic behavior of the material.

In (Guidault et al., 2007), a multi-scale strategy for the analysis of cracked structures is presented, with the main focus on separating the local effects from the global effects in order to keep a macro-scale mesh unchanged during the crack's propagation, and to enable the use of a proper fine-scale description only where required. The method was able to represent the crack happening on the micro-scale, but the crack propagation aspect was not addressed.

Strong coupling

Another multi-scale homogenization method, that is used later in this thesis, is the strong coupling approach. In this approach, two scales communicate constantly during the analysis, and computations advance simultaneously at each scale to produce the final result (Markovic and Ibrahimbegovic, 2004). The standard finite element method is used on both scales. The micro-scale mesh, finitely smaller than the macro-scale mesh, is placed inside the macro elements, as shown in Figure 1.1b (Ibrahimbegovic and Markovic, 2003; Niekamp et al., 2009). In this way, the macro element does not need any constitutive equation, and its element arrays are obtained from the micro-scale calculations. Strong coupling, in contrast to weak coupling, can be used in cases where the characteristic size of heterogeneities is not small enough compared to the structure size at the macro-scale (Ibrahimbegovic, 2009).

When modeling the behavior of a structure by a multi-scale method, the macro-scale is used to describe the homogenized response of the structure, and the micro-scale to describe the details of the behavior on the smaller scale of the material, where some inelastic mechanisms, like damage or plasticity, can be defined. Scales can be coupled by imposing that the displacements are the same over the interface, as in (Ibrahimbegovic et al., 2014) or (Ibrahimbegovic and Markovic, 2003), or by imposing that the variation of the stresses is the same over the interface, as in (Markovic and Ibrahimbegovic, 2004). In this thesis,

displacement based coupling is used, where the micro-scale displacements comply with the macro-scale displacements on the interface.

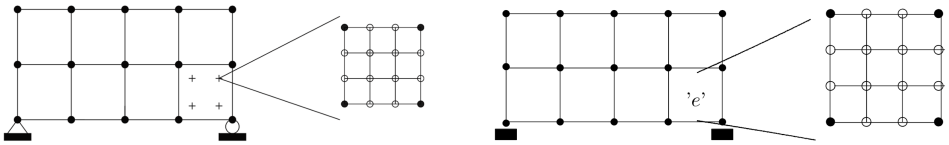


Figure 1.1: (a) FE^2 model with the macro-scale mesh and the micro-scale mesh defined for each macro-scale Gauss point; (b) strong coupling multi-scale model with macro-scale mesh and micro-scale mesh defined for each macro-scale element (Ibrahimbegovic and Markovic, 2003)

The localized Lagrange multiplier method can be used for scale coupling, as in (Markovic et al., 2005). Lagrange multipliers allow to replace the standard computation of the element tangent stiffness matrices and residual vectors by an assembly of micro-scale contributions which are statically condensed at the macro-scale (Hautefeuille et al., 2012). The localized multipliers enforce the condition that the micro-scale interface displacements are calculated as a linear interpolation of the macro-scale displacement.

The scales communicate all the time during the analysis, through the exchange of the values of macro-scale displacements, and the micro-scale condensed stiffness matrix and residual. The finite element models representing the micro-structure communicate between each other exclusively through the degrees of freedom of the macro-scale model, so the advantage is that the micro-scale computations are independent of each other and can be executed in parallel. The negative side is that the scales have to exchange the information after each iteration or time step, increasing the implementation complexity and execution time.

The implementation of the strong coupling multi-scale method applied on modeling of the heterogeneous structures with inelastic constitutive behavior can be found in (Markovic et al., 2005) and (Niekamp et al., 2009). The strong coupling with both displacement based and force based interface is implemented, and it is shown to work for linear-elasticity and hardening, but not for the softening case, which is an issue that will be addressed in this thesis.

In (Hautefeuille et al., 2012) a computational strategy for a strong coupling multi-scale method for heterogeneous material analysis with localized failure is presented, where a cylindrical arc-length procedure at the micro-scale enables the softening phase representation. It does not have a continuous macro-scale displacement representation, and in the case of more complex micro-scale structure

representation, a cylindrical arc-length procedure at the micro-scale is not enough to allow it to enter into the softening phase.

In (Gitman et al., 2008), a multi-scale procedure called coupled-volume approach is developed, where the macro-scale finite element size is the same as the micro-scale cell size. The micro-scale cell size is not linked to an infinitely small macro-scale material point like in the FE^2 method. The scales are coupled in a similar way like the strong coupling method that will be presented in this thesis, through the micro-scale displacements imposed by the macro-scale values. This approach has been developed only for a 1D setting and it has been shown that it works for linear-elasticity, hardening and softening case.

The open question that is addressed later in this thesis is how to allow for the multi-scale strong coupling procedure with displacement based interface to enter into the softening phase, and how to represent the displacement jump on the macro-scale. More details and the complete formulation of the proposed strong coupling multi-scale approach are presented in Chapter 2.

1.3.2 Multi-physics coupling

Multi-physics coupling implies modeling of the behavior of the material that is not described only by mechanical properties, but is also influenced by other physics and fields, like electrical, magnetic, temperature or fluid. It allows integration of complex and smart mechanical sub-systems into interconnected coupled systems (Ibrahimbegovic, 2009). Different domains can be coupled to simulate the behavior of smart materials like piezoelectric, magnetostrictive, shape-memory, electroactive polymers and similar materials (Bengisu and Ferrara, 2018).

Coupled multi-physics systems interact with each other and it is impossible to find a solution for one without solving the other simultaneously. Different physics that are interacting can use different numerical methods to find the final solution, or can have different discretization techniques and meshes used. All this can result in a non-matching interface between them, as it can be seen in Figure 1.2a, and a solution to make these two physics to communicate has to be found. The interfaces are non-matching when the node locations do not coincide, degrees of freedom are not the same, or when the boundary motions are not conforming. Matching meshes on the interface have the same location of the interface nodes, the same number of degrees of freedom per node, and element boundary motions are conforming. In (Park et al., 2002), a non-matching interface treatment is presented, which guarantees preservation of the constant-stress interface patch test when the partitioned sub-domains are connected.

There are two main approaches to solve coupled multi-physics problems: monolithic and partitioned approach. In the monolithic approach, both physics are solved simultaneously, together with the coupling conditions between the domains. Only one set of equations is defined for both physics in the monolithic approach. One example in this thesis is solved using the monolithic approach, although the main focus is on the partitioned approach. The partitioned approach is based on domain decomposition, where the domains can be solved separately, but coupled over the interface. This allows for each domain to be solved using the optimal method, discretization and approximations, as it can be seen in Figure 1.2b.

Some physics, when coupled, share the same spatial domain, while with others there is a clear distinction between the two domains. For example, in fluid-structure interaction, there is a clear interface between the fluid and the structure domain, while in electro-magneto-mechanical coupling the spatial domain is shared by fields from all domains which are present at the same time.

One the most researched multi-physics problems is fluid-structure interaction. The main advantage of using the partitioned approach in this kind of problems is that the standard discretization technique which is most suitable for a particular domain can be used, as shown in (Kassiotis et al., 2011a). The finite element method can be used for the solid, and the finite volume method for the fluid. The solution of the fluid and the structure problems can be decoupled from one another thanks to a suitable splitting of the interface conditions. Information that are passed on the interface are the displacement field (from the structure to the fluid), and the stress field (from the fluid to the structure) (Lombardi et al., 2013). A multi-physics strong coupling solution procedure for fluid–structure interaction is proposed in (Matthies et al., 2006) and simulated on several examples, showing that it achieves the same results as a mono-scale approach. A partitioned solution approach for nonlinear fluid–structure interaction with matching interface, handled by the fixed-point strategy with an adaptive relaxation parameter, is proposed in (Kassiotis et al., 2011b), and demonstrated on a lid driven cavity flow with a flexible bottom example. This code, developed in (Kassiotis et al., 2011b) and provided by the author, was executed and analysed during this thesis, as it was considered for possible improvements by using the mortar element method instead of the radial basis functions, which would guarantee the conservation of the stresses along the interface.

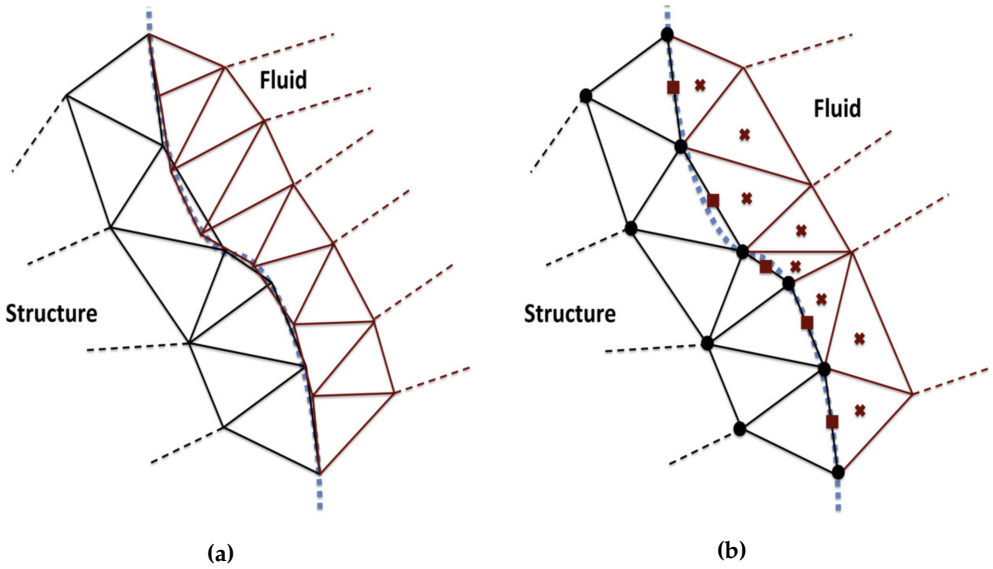


Figure 1.2: (a) Non-matching interface between the fluid and the structure; (b) Different spatial discretizations - finite elements used in the structure part, cell centered finite volumes in the fluid part (Lombardi et al., 2013)

One of the biggest challenges of multi-physics problems is the interface treatment between the domains. The Lagrange multiplier method is on of the most used, like in (Kassiotis et al., 2011a). This method reduces the problem complexity, as the coupled systems become visible to each other only through the global Lagrange multiplier (Ibrahimbegovic, 2009; Park et al., 2002). In (Klöppel et al., 2011), the non-matching interface is treated using the mortar element method. It is a domain decomposition technique which introduces a mortar element between two domains in interaction, as defined in (Belgacem et al., 1998). In this way, a domain is decomposed into two sub-domains, where each of the sub-domains is in interaction only with the mortar element. This approach allows for the use of non-matching meshes and for the connection of different discretizations across the interface in an optimal way, as shown in (Belgacem, 1999).

When different approximations are used in coupled domains, they have to be connected using some of the interface interpolation techniques (Ibrahimbegovic et al., 2014). For example, in fluid-structure interaction, approximations are made on nodal values for the finite element method in the mechanics domain, and on cell centers for the finite volumes method in the fluid domain. The radial basis function method is often used as a technique to interpolate data over non-matching interface grids, as shown in (Lombardi et al., 2013). The main characteristic of radial basis functions is that the value of the function depends only on the Euclidean distance of the argument from the origin (Buhmann, 2003). Radial basis

function implementation guarantees the conservation of the total energy along the interface, but does not guarantee conservation of the stresses, which would lead to better quality results for the stress and displacements for the mechanical domain.

In (Fu et al., 2017), a multi-scale computational method based on the extended multi-scale finite element framework (X-FEM) that simulates the geometrically nonlinear behaviors of heterogeneous piezoelectric materials is developed. The thermal fields are also included in (Lv et al., 2014) to simulate thermo-electro-mechanical coupling behavior of a piezoelectric smart material. In these two works, the thermal, electrical and mechanical fields are described with multi-scale numerical base functions which can capture the micro-scale behavior, and based on that, the response of the heterogeneous piezoelectric smart material is obtained on the macro-scale. A general thermodynamic approach to material constitutive equations for solids and fluids is developed in (Ferrari and Mittica, 2013), and it can be applied to simulate the behavior of piezoelectric and magnetostrictive materials.

Other numerical simulations of piezoelectric material behavior by coupling electrical and mechanical fields and solving the equations simultaneously using a monolithic formulation are shown in (Rochus et al., 2006), (Hansbo and Hansbo, 2004) and (Kamlah and Tsakmakis, 1999).

Multi-physics coupling can be used to perform simulations in many different domains, like a simulation of aortic blood flow in the cardiovascular system in (Crosetto et al., 2011). Here, finite element method is used to model a fluid–structure interaction between blood flow and arterial wall deformation of an aorta.

In (Rukavina and Ibrahimbegovic, 2017), a general fluid-structure interaction coupling problem overview and software-based coupling approach is given. In Chapter 4, electro-mechanic coupling for the simulation of piezoelectric material behavior is presented.

1.3.3 Software code coupling

Scientific software code for a specific problem is usually developed by experts from the same domain. The knowledge about the domain and the internal implementation is often hidden inside the code, and the code is not well documented. Extending existing code to allow it to be applied to some other problems is usually challenging. Also, developing a new software code from scratch is a complex task that is often time consuming. In these cases, for multi-scale and multi-physics coupling, a better solution is to reuse existing code as a software component. This component can then be used as a black box that solves

the desired problem from a specific domain, and couple it with other components to solve complex multi-scale and multi-physics problems (Groen et al., 2013).

The main focus in component based software engineering is the development of software designed for reusability. Existing legacy codes that are developed by domain experts and which are well tested can be reused in some other cases. In the context of numerical modeling for multi-scale and multi-physics problems, this means that the optimal software can be used for each domain or physics. For example, a multi-physics problem of fluid-structure interaction (FSI) can be solved using two software specialised for each domain, as shown in (Kassiotis et al., 2011b). In this work, a FSI problem software solution is implemented in the Finite Element Analysis Program (FEAP) (Taylor, 2014) for structural mechanics and in OpenFOAM (Jasak et al., 2007) for fluid mechanics.

These software can be then executed in parallel and can exchange information during the analysis. An additional software component has to be developed that can coordinate the execution of each software component, help them communicate and exchange data during the analysis. Software codes do not need to be initially developed to work in parallel. For example, in (Kassiotis et al., 2011b), Component Template Library (CTL) (Niekamp, 2005) is used to provide the coupling between two codes.

A component model is a definition of standards for software component implementation, documentation and deployment. An example of component models are Common Object Request Broker Architecture (CORBA), Component Object Model (COM), JavaBeans, or Component Template Library (CTL). Component model implementations provide platform services that allow components written according to the model to communicate. In this way, different software components can be set to communicate and execute in parallel. To use services provided by a model, components are deployed in a container, which provides a set of interfaces to access the service implementations of a component (Sommerville, 2011).

When the existing software is deployed as a software component, and interfaces are defined, they can be coupled and executed in parallel. For this purpose, a specific formulation has to be developed using a partitioned approach. Formulation for each physics (in a multi-physics problem) or for each scale (in a multi-scale problems) has to be implemented to be able to run for each software component. Then, a method for interface treatment between the domains has to be defined, as shown in the previous section. The interface treatment method is then implemented using a component model to allow the two software components to exchange information on the interface during the execution.

There are many works where different software is used and coupled together to solve the multi-scale and multy-physics problems, like in (Sengupta et al., 2012) where the thermo-mechanical coupling problem is modeled and implemented in FEAP and using the Open MPI (Gabriel et al., 2004) version of the classical Message Passing Interface (MPI). In (Peszynska et al., 2000), a multi-physics coupling of codes using a partitioned approach is presented and used to simulate the flow and transport in a porous reservoir. It is built in Integrated Parallel Accurate Reservoir Simulator (IPARS) framework (Lacroix et al., 2001) where the interface algorithm is merged with the framework, allowing each physical model to execute in parallel. MPI implementation is used for parallel execution of the codes.

In (Anciaux et al., 2006), a parallel implementation of a multi-scale coupling method for crack propagation in a material is presented. It uses Lammps (Plimpton et al., 2011) to simulate molecular dynamics on the micro-scale, and a custom developed finite element code based on LibMesh (Kirk and Peterson, 2003) library for continuum mechanics on the macro-scale. MPICH (Gropp and Lusk, 1996) version of MPI specification is used for the communication and parallel execution. In (Guo and Zhao, 2014), two different codes are coupled – Escript (Gross et al., 2007) for finite element method (FEM) analysis, and YADE (Šmilauer et al., 2010) for discrete element method (DEM), to model the behavior of granular media.

A component-based execution environment for multi-scale and multi-physics coupling named Multiscale Coupling Library and Environment (MUSCLE 2) is developed and presented in (Borgdorff et al., 2014). It has programming interfaces for Java, C++, C, Python and Fortran, and it is compatible with MPI, OpenMP and threading codes. It is lightweight and portable, and can be used to execute different codes in parallel. It could be a good solution for code coupling in this thesis, but CTL was chosen due to its simplicity and ease of implementation.

Some other code coupling frameworks worth noting are Common Component Architecture (CCA) (Bernholdt et al., 2006), Model Coupling Toolkit (MCT) (Larson et al., 2005), or OpenPALM coupler (Morel et al., 2019).

In this thesis, Component Template Library (CTL) is used for code coupling because it is lightweight, easy to implement, and it supports FORTRAN programming language. For the mechanical domain, Finite Element Analysis Program (FEAP) is used, and the possibility of using GetDP (Dular and Geuzaine, 2013) for the electrical domain was studied.

1.4 Structure of the thesis

In this thesis, a multi-scale and multi-physics coupling computation procedure for a 2D and 3D setting is developed and presented. It is divided into six chapters,

including the first introductory part. In Chapter 2, the 2D multi-scale coupling procedure is presented. The previously developed multi-scale formulation is shown, and then extended to take into account localized failure. The extension of a 2D multi-scale coupling procedure to a 3D setting is carried out in Chapter 3. In Chapter 4, the 3D multi-scale formulation for electro-mechanical coupling formulation is presented. In every chapter, a description of the micro-scale models used in the multi-scale computations is given. Also, results of several numerical examples are presented in each of the chapters for the proposed formulation. In Chapter 5, software implementation details are presented. Finally, in Chapter 6, some concluding remarks are given, and the perspectives for future work and extensions are outlined.

2

2D multi-scale coupling

Contents

In this chapter, a 2D multi-scale coupling computation procedure is presented. The previously proposed version of the multi-scale formulation is presented in the first section. This model is adequate for representing elastic behavior and hardening, but not for localized failure. To overcome this problem, a multi-scale procedure for localized failure is presented in the second section, where an embedded discontinuity is implemented in the macro-scale element to capture the softening behavior happening on the micro-scale. The proposed procedure represents the main contribution and novelty in this work. In the third section, a damage model used on the micro-scale is described. Finally, the results of several numerical examples are presented in the last section.

2.1 Multi-scale formulation

The previously proposed version of the multi-scale formulation presented in this section was first developed and described in (Ibrahimbegovic and Markovic, 2003), (Ibrahimbegovic, 2009) and (Markovic et al., 2005).

In the proposed multi-scale coupling the two scales are strongly coupled. They are exchanging information during the whole analysis, and the computation advances simultaneously on both scales. At each time step, both macro and micro-scale computations are executed. Only when convergence is obtained at both scales, computations advance to the next time step. The finite element method is used at both scales, which adds to the generality of the method and simplicity of its implementation. The computation is performed using the operator split solution procedure on both scales. The constitutive equations are not defined on the macro element, and its element arrays are obtained from the micro-scale computations. A micro-scale mesh, finitely smaller than the macro-scale mesh, is placed inside each of the macro-scale elements as shown in Figure 2.1. The two scales are strongly coupled using displacement based coupling, where the micro-scale displacements on the interface are imposed by the macro-scale displacements.

For scale coupling, the localized Lagrange multiplier method (Park et al., 2002) is used. It allows to replace the standard computation of the element tangent stiffness matrices and the residual vectors by an assembly of micro-scale contributions which are statically condensed at the macro-scale (Hautefeuille et al., 2012). The finite element models on the micro-scale communicate between each other only through the degrees of freedom defined at the macro-scale. In this way, micro-scale computations are completely independent of each other and can be executed in parallel, significantly reducing execution time and allowing the code to be executed on different processors.

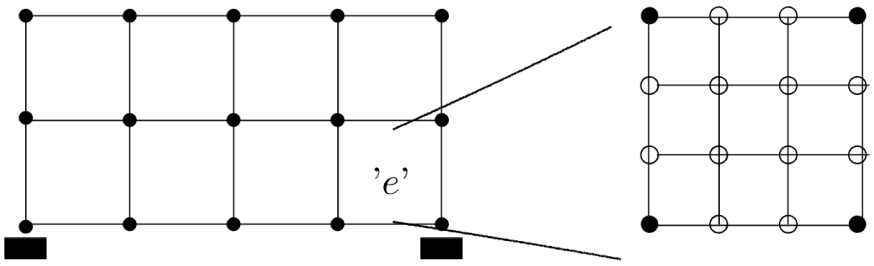


Figure 2.1: Multiscale model with FE mesh at both the macro (on the left) and the micro-scale (on the right) (Ibrahimbegovic and Markovic, 2003)

The formulation for the multi-scale method starts with defining the variational formulation and equation for the total energy of the system that can be written as the sum of energies

$$\Pi(u^M, u^m, \lambda, \bar{\xi}_k) = \Pi^M + \Pi^m + \Pi^{\Gamma^{Mm}} \tag{2.1}$$

where Π^M is the strain energy at the macro-scale, Π^m is the strain energy at the micro-scale, and $\Pi^{\Gamma^{Mm}}$ is the energy at the interface between the two scales.

The external forces are defined and assumed to apply only at the macro-scale, while the constitutive laws and the internal variables are defined only at the micro-scale. Energies for both scales and energy on the interface can be then defined as

$$\begin{aligned} \Pi^M &= -\int_{\Omega^M} \mathbf{u}^M \cdot \mathbf{b}^M dV - \int_{\Gamma_\sigma^M} \mathbf{u}^M \cdot \bar{\mathbf{t}}\, dA \\ \Pi^m &= \int_{\Omega^m} \psi^m(\mathbf{u}^m, \xi_k) dV \\ \Pi^{\Gamma^{Mm}} &= \int_{\Gamma^{Mm}} \boldsymbol{\lambda} \cdot (\mathbf{u}^M - \mathbf{u}^m) dA \end{aligned} \tag{2.2}$$

where $\mathbf{u}^M$ and $\mathbf{u}^m$ are macro and micro-scale displacement vectors respectively, $\mathbf{b}^M$ represents volumetric forces, $\bar{\mathbf{t}}$ is the traction force vector, $\boldsymbol{\lambda}$ is the Lagrange multiplier that is used to couple the scales, ψ^m is the micro-scale strain energy density and ξ_k is the micro-scale internal variable. The macro-scale domain is denoted as Ω^M, the micro-scale domain as Ω^m, Γ_σ^M is part of the macro-scale boundary where tractions are applied and Γ^{Mm} is the interface between the two scales.

Since nonlinear inelastic behavior is considered, an incremental-iterative analysis (e.g. (Ibrahimbegovic, 2009)) is used to obtain the final solution. In any typical increment, the central problem of multi-scale analysis can be posed as

For given $\mathbf{u}_n^M = \mathbf{u}^M(\mathbf{x}^M, t_n)$, $\mathbf{u}_n^m = \mathbf{u}^m(\mathbf{x}^m, t_n)$, $\boldsymbol{\lambda}_n = \lambda(\mathbf{x}^M, t_n)$, $\xi_{k,n} = \xi_k(\mathbf{x}^m, t_n)$, $h = t_{n+1} - t_n$

find $\mathbf{u}_{n+1}^M$, $\mathbf{u}_{n+1}^m$, $\boldsymbol{\lambda}_{n+1}$, $\xi_{k,n+1}$, such that

$$\begin{aligned} 0 &= \frac{d}{d\varepsilon}\bigg|_{\varepsilon=0} \Pi^m(\mathbf{u}_{n+1}^m + \varepsilon \mathbf{w}^m, \cdot) \equiv G^m(\mathbf{u}_{n+1}^m, \boldsymbol{\lambda}_{n+1}, \xi_{k,n+1}; \mathbf{w}^m) = \\ &= \int_{\Omega^m} \nabla^s \mathbf{w}^m \cdot \hat{\boldsymbol{\sigma}}^m(\mathbf{u}_{n+1}^m, \xi_{k,n+1}) dV - \int_{\Gamma^{Mm}} \mathbf{w}^m \cdot \boldsymbol{\lambda} dA \end{aligned} \tag{2.3}$$

$$0 = \frac{d}{d\varepsilon}\Big|_{\varepsilon=0} \Pi^M(\mathbf{u}^M_{n+1} + \varepsilon \mathbf{w}^M, \cdot) \equiv G^M(\mathbf{u}^M_{n+1}, \boldsymbol{\lambda}_{n+1}; \mathbf{w}^M) = \\ = \int_{\Gamma^{Mm}} \mathbf{w}^M \cdot \boldsymbol{\lambda} dA - \int_{\Omega^M} \mathbf{w}^M \cdot \mathbf{b} dV - \int_{\Gamma^M_\sigma} \mathbf{w}^M \cdot \bar{\mathbf{t}} dA \tag{2.4}$$

$$0 = \frac{d}{d\varepsilon}\Big|_{\varepsilon=0} \Pi^{Mm}(\boldsymbol{\lambda}_{n+1} + \varepsilon \boldsymbol{\nu}, \cdot) \equiv G^{Mm}(\mathbf{u}^M_{n+1}, \mathbf{u}^m_{n+1}, \boldsymbol{\nu}) = \\ = \int_{\Gamma^{Mm}} \boldsymbol{\nu} \cdot (\mathbf{u}^M_{n+1} - \mathbf{u}^m_{n+1}) dA \tag{2.5}$$

with $\xi_{k,n+1} = \xi_{k,n} + h\hat{f}(\mathbf{u}^M_{n+1}, \mathbf{u}^m_{n+1}, \boldsymbol{\lambda}_{n+1}, \xi_{k,n+1})$

where $\mathbf{w}^m$, $\mathbf{w}^M$ and $\boldsymbol{\nu}$ are the variations of the micro-scale displacements, macro-scale displacements and Lagrange multipliers, respectively.

At the macro-scale, a isoparametric Q4 element is used as shown in Figure 2.2. The standard finite element shape functions for Q4 element are used

$$N^M_a(\xi, \eta) = \frac{1}{4}(1 + \xi_a \xi)(1 + \eta_a \eta), \quad a = 1, 2, 3, 4 \tag{2.6}$$

where ξ_a and η_a are the natural coordinates of node a.

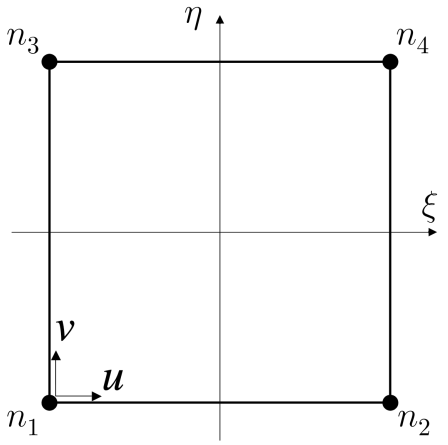


Figure 2.2: Macro-scale Q4 isoparametric element with two degrees of freedom in each node

At the micro-scale, a constant strain triangle (CST) element with one integration point is used, but any other element with the same number of degrees of freedom can be used for the formulation to be still valid. The CST element with implemented damage model is used and described in detail in Section 2.3.

The displacement field approximations at the micro and macro-scale, and the localized Lagrange multipliers are calculated using standard finite element approximations, which can be written as

$$\begin{aligned}
\mathbf{u}^m_{n+1}\Big|_{\Omega^{m,e}}(\mathbf{x}^m) &= \sum_{a=1}^{n^m_{el}} \mathbf{N}^{m,e}_a(\mathbf{x}^m)\mathbf{d}^m_{a,n+1} \\
\mathbf{u}^M_{n+1}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm,E}} \mathbf{N}^{M,E}_a(\mathbf{x}^m)\mathbf{d}^{M,E}_{a,n+1} \\
\boldsymbol{\lambda}_{n+1}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm,E}} \mathbf{P}^{M,E}_a(\mathbf{x}^m)\boldsymbol{\beta}_{a,n+1}
\end{aligned} \tag{2.7}$$

where $\mathbf{d}^m_{a,n+1}$ and $\mathbf{d}^{M,E}_{a,n+1}$ are nodal value of micro and macro-scale displacements in time step $n+1$, $\mathbf{P}^{M,E}_a$ is the Lagrange multipliers' interpolation function, and $\boldsymbol{\beta}_{a,n+1}$ is the nodal value of Lagrange multipliers in time step $n+1$.

Keeping in mind the approximations stated above, the central problem can now be rewritten as follows

For given $\mathbf{d}^{M,E}_n$, $\mathbf{d}^m_n$, $\boldsymbol{\beta}_n$, $\xi_{k,n}$; $\forall\Omega^{M,E}$

find $\mathbf{d}^{M,E}_{n+1}$, $\mathbf{d}^m_{n+1}$, $\boldsymbol{\beta}_{n+1}$, $\xi_{k,n+1}$,

such that $(\forall e \in [1, n^m_{el}], \forall E \in [1, n^M_{el}])$

$$\begin{aligned}
\mathbf{0} &= \mathbf{r}^m(\mathbf{d}^m_{n+1}, \boldsymbol{\beta}_{n+1}, \xi_{k,n+1}) = \\
&= \mathbb{A}_{e=1}^{n^m_{elem}} \Big[\int_{\Omega^{m,e}} \mathbf{B}^{m^T} \cdot \hat{\boldsymbol{\sigma}}(\mathbf{d}^m_{n+1}, \xi_{k,n+1}) dV \Big] - \int_{\Gamma^{Mm,E}} \mathbf{N}^{m,e^T}\mathbf{P}^{M,E}\boldsymbol{\beta}_{n+1} dA
\end{aligned} \tag{2.8}$$

$$\begin{aligned}
\mathbf{0} &= \mathbf{r}^M(\mathbf{d}^M_{n+1}, \boldsymbol{\beta}_{n+1}) = \\
&= \mathbb{A}_{E=1}^{n^M_{elem}} \Big[\int_{\Gamma^{M,E}} \mathbf{N}^{M,E^T}\mathbf{P}^{M,E}\boldsymbol{\beta}_{n+1} dA - \int_{\Omega^{M,E}} \mathbf{N}^{M,E^T}\mathbf{b} dV - \int_{\Gamma^{M,E}} \mathbf{N}^{M,E^T}\bar{\mathbf{t}} dA \Big]
\end{aligned} \tag{2.9}$$

$$\mathbf{0} = \mathbf{p}^{M,E}(\mathbf{d}^{M,E}_{n+1}, \mathbf{d}^m_{n+1}, \boldsymbol{\beta}_{n+1}) = \int_{\Gamma^{Mm,E}} \mathbf{P}^{E^T}(\mathbf{N}^{M,E}\mathbf{d}^{M,E} - \mathbf{N}^m\mathbf{d}^m) dA \tag{2.10}$$

with $\xi_{k,n+1} = \xi_{k,n} + h\hat{f}(\mathbf{d}^{M,E}_{n+1}, \mathbf{d}^m_{n+1}, \boldsymbol{\lambda}_{n+1}, \xi_{k,n+1})$

The localized Lagrange multipliers enforce that the displacements of the interface nodes at the micro-scale are calculated as a linear interpolation of

the nodal values of displacements at the macro-scale, as shown in Figure 2.3. This can be achieved by choosing the Dirac delta functions $\delta(\mathbf{x} - \mathbf{x}_a)$ centered upon the micro-scale interface nodes $x_a \in \Gamma^{M,E}$. By introducing the Dirac delta function into (2.10), it can be obtained

$$\mathbf{p}_a^M = \bar{\mathbf{d}}_{a,n+1}^m - \sum_b \mathbf{N}_b^{M,E}(\mathbf{x}_a)\mathbf{d}_{b,n+1}^{M,E} = \mathbf{0} \tag{2.11}$$

Finally, the micro-scale nodal displacement vector on the interface can be written as

$$\bar{\mathbf{d}}_{n+1}^m\Big|_{\Gamma^{M,E}} = \mathbf{T}^E \mathbf{d}_{n+1}^{M,E} \tag{2.12}$$

where $\mathbf{T}^E$ is the connectivity matrix, and $\mathbf{d}_{n+1}^{M,E}$ are macro-scale nodal displacements.

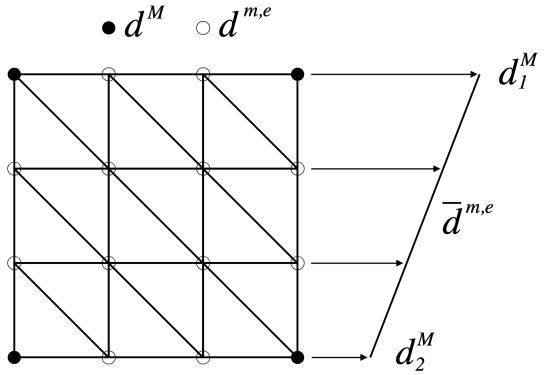


Figure 2.3: Micro-scale interface nodal displacements calculated as a linear interpolation of the macro-scale nodal displacements

The connectivity matrix $\mathbf{T}^E$ is based on the particular values of macro-scale shape functions which correspond to the interface nodes. Namely, that matrix is constructed by simply introducing the isoparametric coordinates of each micro-scale node on the interface into the macro-scale shape functions N_a^M, as it will be explained on an example in the next section.

The standard finite element system of equations for computing the increment of the displacement field on the micro-scale can be written as

$$\begin{bmatrix} \bar{\bar{\mathbf{K}}} & \bar{\mathbf{K}}^T \\ \bar{\mathbf{K}} & \mathbf{K} \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{d}} \\ \Delta\mathbf{d} \end{bmatrix} = - \begin{bmatrix} \bar{\mathbf{r}} \\ \mathbf{r} \end{bmatrix} \tag{2.13}$$

where $\bar{\bar{\mathbf{K}}}$ is the part of the stiffness matrix related only to interface nodes, $\bar{\mathbf{K}}$ is related to interface nodes in relation to free nodes, and $\mathbf{K}$ is related only to free nodes. In the same way, $\Delta\bar{\mathbf{d}}$ and $\bar{\mathbf{r}}$ are the displacement increments and residuals of the interface nodes, and $\Delta\mathbf{d}$ and $\mathbf{r}$ are displacement increments and residuals of free nodes.

Static condensation (e.g. (Ibrahimbegovic, 2009)) can be performed on the previous system of equations. First, the displacement field increment of free nodes can be expressed as

$$\Delta \mathbf{d} = \mathbf{K}^{-1}(-\mathbf{r} - \bar{\mathbf{K}}\Delta\bar{\mathbf{d}}) \tag{2.14}$$

Introducing (4.29) to the first equation in (4.27) it is obtained

$$\left(\bar{\bar{\mathbf{K}}} - \bar{\mathbf{K}}^T\mathbf{K}^{-1}\bar{\mathbf{K}}\right)\Delta\bar{\mathbf{d}} = -\bar{\mathbf{r}} + \bar{\mathbf{K}}^T\mathbf{K}^{-1}\mathbf{r} \tag{2.15}$$

Then, the statically condensed stiffness matrix and residual obtained at the micro-scale can be written as

$$\begin{aligned} \widetilde{\mathbf{K}}^m &= \left(\bar{\bar{\mathbf{K}}} - \bar{\mathbf{K}}^T\mathbf{K}^{-1}\bar{\mathbf{K}}\right) \\ \widetilde{\mathbf{r}}^m &= -\bar{\mathbf{r}} + \bar{\mathbf{K}}^T\mathbf{K}^{-1}\mathbf{r} \end{aligned} \tag{2.16}$$

After the computations on the micro-scale have converged, the final values of the condensed stiffness matrix and residual are used to compute the values of the stiffness matrix and residual that is going to be used at the macro-scale

$$\begin{aligned} \mathbf{K}^{M,E}_{n+1} &= \mathbf{T}^{E,T}\widetilde{\mathbf{K}}^m_{n+1}\mathbf{T}^E \\ \mathbf{r}^{M,E}_{n+1} &= \mathbf{T}^{E,T}\widetilde{\mathbf{r}}^m_{n+1} \end{aligned} \tag{2.17}$$

When the values of the macro-scale stiffness matrix and residual are computed, they are used to update the values of the macro-scale displacement field. The standard finite element system of equations that needs to be solved is

$$\mathbf{K}^M_{n+1}\Delta\mathbf{d}^M_{n+1} = -\mathbf{r}^M_{n+1} \tag{2.18}$$

2.2 Multi-scale formulation for localized failure

The multi-scale formulation described in the previous section cannot take into account the crack propagation. When a localized failure happens on the micro-scale, it cannot be properly transfered to the macro-scale. The multi-scale formulation with localized failure proposed in this section, together with the numerical examples from Section 2.4, are first presented in (Rukavina et al., 2019).

In order to allow for the multi-scale model to represent localized failure, the corresponding incompatible mode function that describes the embedded discontinuity is introduced inside the macro-scale element. In this way, the localized failure can be properly transfered from micro to macro-scale. The discontinuity is positioned at the center of the Q4 isoparametric element. Vectors n and m are the normal and the tangential vector at the discontinuity. In this

work, for simplicity, a model for a tension test is illustrated, where only the crack opening in mode I is taken into account. As a result, only n, the normal vector at the discontinuity, will have a non-zero value. The element domain is divided into two sub-domains, Ω^{e-} and Ω^{e+}, as shown in Figure 2.4a. Hence, the incompatible mode function can be written as

$$M(\xi,\eta) = H_{\Gamma_s}(\xi,\eta) - \sum_{b\in\Omega^{e+}} N_b^M(\xi,\eta) \tag{2.19}$$

where H_{Γ_s} is the Heaviside step function defined as

$$H_{\Gamma_s} = \begin{cases} 0, (\xi,\eta) \in \Omega^{e-} \\ 1, (\xi,\eta) \in \Omega^{e+} \end{cases} \tag{2.20}$$

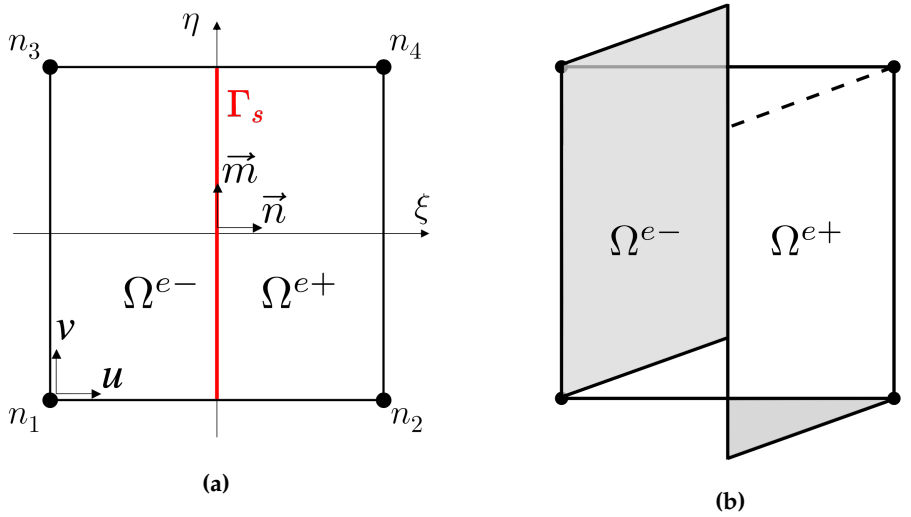


Figure 2.4: (a) Q4 isoparametric element with two sub-domains related to the displacement jump; (b) Incompatible mode shape function **M** for the discontinuity in the middle of the element

Now, the macro-scale displacement field from (2.7) can be rewritten as

$$\mathbf{u}_{n+1}^M\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) = \sum_{a\in\Gamma^{Mm,E}} \mathbf{N}_a^{M,E}(\mathbf{x}^m)\mathbf{d}_{a,n+1}^{M,E} + \mathbf{M}^{M,E}(\mathbf{x}^m)\boldsymbol{\alpha}_{n+1}^{M,E} \tag{2.21}$$

where $\boldsymbol{\alpha}_{n+1}^{M,E}$ is the displacement discontinuity vector.

Now the macro-scale displacement field approximation can be introduced into (2.10)

$$\begin{aligned} \mathbf{0} &= \mathbf{p}^{M,E}(\mathbf{d}_{n+1}^{M,E}, \mathbf{d}_{n+1}^m, \boldsymbol{\beta}_{n+1}) = \\ &= \int_{\Omega^{M,E}} \mathbf{P}^{E^T}(\mathbf{N}^{M,E}\mathbf{d}^{M,E} + \mathbf{M}^{M,E}\boldsymbol{\alpha}^{M,E} - \mathbf{N}^{m,e}\mathbf{d}^m)dA \end{aligned} \tag{2.22}$$

The displacement field on the micro-scale interface nodes is now a function of both the macro-scale displacement field and the displacement discontinuity

$$\mathbf{p}_a^M = \bar{\mathbf{d}}_{a,n+1}^m - \sum_b \mathbf{N}_b^{M,E}(\mathbf{x}_a)\mathbf{d}_{b,n+1}^{M,E} - \mathbf{M}^{M,E}(\mathbf{x}_a)\boldsymbol{\alpha}_{n+1}^{M,E} = \mathbf{0} \tag{2.23}$$

The final expression used to compute the micro-scale nodal displacement field on the interface can be written as

$$\bar{\mathbf{d}}_{n+1}^m\Big|_{\Gamma^{M,E}} = \mathbf{T}^E\mathbf{d}_{n+1}^{M,E} + \mathbf{S}^E\boldsymbol{\alpha}_{n+1}^{M,E} \tag{2.24}$$

where $\mathbf{S}^E$ is the transformation matrix for the incompatible mode, and $\boldsymbol{\alpha}_{n+1}^{M,E}$ is the displacement discontinuity vector. The transformation matrix $\mathbf{S}^E$ is constructed based on the macro-scale incompatible mode function, in a similar way as the transformation matrix $\mathbf{T}^E$. The isoparametric coordinates of each micro-scale interface node are introduced into the macro-scale incompatible mode function M.

On the micro-scale, damage model computations are executed, and after convergence the condensed stiffness matrix $\widetilde{\mathbf{K}}^m$ and residual $\widetilde{\mathbf{r}}^m$ are obtained as explained in (4.31). After the micro-scale computations are finished, the system of equations that needs to be solved on the macro-scale can be written as

$$\begin{bmatrix} \mathbf{K}^M & \mathbf{F}^M \\ \mathbf{F}^{M,T} & \mathbf{H}^M \end{bmatrix} \begin{bmatrix} \Delta\mathbf{d}^M \\ \Delta\boldsymbol{\alpha}^M \end{bmatrix} = - \begin{bmatrix} \mathbf{r}^M \\ \mathbf{h}^M \end{bmatrix} \tag{2.25}$$

where the vector $\mathbf{h}^M$ represents the residual at the discontinuity.

To construct the macro-scale stiffness matrix needed for solving the macro-scale system of equations, submatrices $\mathbf{K}^M$, $\mathbf{F}^M$ and $\mathbf{H}^M$ are computed as

$$\begin{aligned} \mathbf{K}_{n+1}^{M,E} &= \mathbf{T}^{E,T}\tilde{\mathbf{K}}_{n+1}^m\mathbf{T}^E \\ \mathbf{F}_{n+1}^{M,E} &= \mathbf{T}^{E,T}\tilde{\mathbf{K}}_{n+1}^m\mathbf{S}^E \\ \mathbf{H}_{n+1}^{M,E} &= \mathbf{S}^{E,T}\tilde{\mathbf{K}}_{n+1}^m\mathbf{S}^E \end{aligned} \tag{2.26}$$

This corresponds to the way submatrices are computed in the standard finite element procedure on the global level for the incompatible mode method

$$\begin{aligned} \mathbf{K}_{n+1}^e &= \int_{\Omega^e} \mathbf{B}^{e,T}\mathbf{C}_{n+1}^{ed,(i)}\mathbf{B}^e d\Omega \\ \mathbf{F}_{n+1}^e &= \int_{\Omega^e} \mathbf{B}^{e,T}\mathbf{C}_{n+1}^{ed,(i)}\tilde{\mathbf{G}}^e d\Omega \\ \mathbf{H}_{n+1}^e &= \int_{\Omega^e} \tilde{\mathbf{G}}^{e,T}\mathbf{C}_{n+1}^{ed,(i)}\tilde{\mathbf{G}}^e d\Omega \end{aligned} \tag{2.27}$$

where $\mathbf{B}^e$ is the matrix containing the shape functions derivatives, $\tilde{\mathbf{G}}^e$ is the matrix containing incompatible mode function derivatives, and $\mathbf{C}^{ed}$ is the tangent elasto-damage tensor.

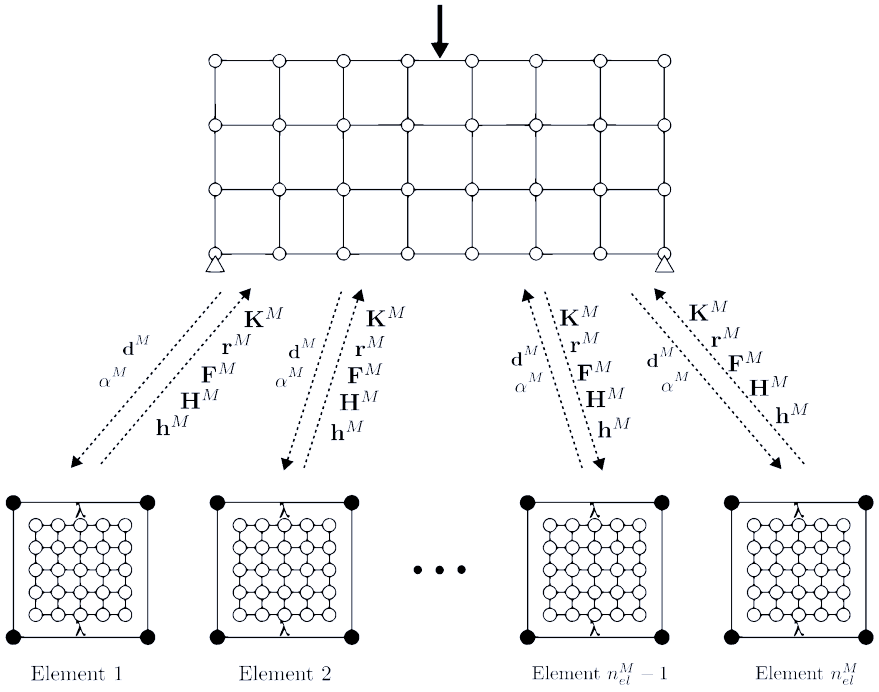


Figure 2.5: The macro-scale and micro-scale data transfered between the scales in every time step (according to (Ibrahimbegovic and Markovic, 2003) with added localization failure)

Also, the macro-scale residuals are computed using the values of micro-scale residuals and transformation matrices $\mathbf{T}$ and $\mathbf{S}$

$$\begin{aligned} \mathbf{r}_{n+1}^{M,E} &= \mathbf{T}^{E,T}\tilde{\mathbf{r}}_{n+1}^{m} \\ \mathbf{h}_{n+1}^{M,E} &= \mathbf{S}^{E,T}\tilde{\mathbf{r}}_{n+1}^{m} \end{aligned} \tag{2.28}$$

which correspond to the standard finite element procedure, where the values of residuals are computed as

$$\begin{aligned} \mathbf{r}_{n+1}^{e} &= \int_{\Omega^e} \mathbf{B}^{e,T}\sigma(d,\alpha)d\Omega \\ \mathbf{h}_{n+1}^{e} &= \int_{\Omega^e} \tilde{\mathbf{G}}^{e}\sigma(d,\alpha)d\Omega \end{aligned} \tag{2.29}$$

The system of equations (2.25) is solved using the operator split procedure (e.g. (Ibrahimbegovic, 2009)). First, the values of the macro-scale displacements are fixed, while the correct value of $\boldsymbol{\alpha}^M$ has to be calculated iteratively. Each iteration consists of calling the micro-scale computations with the imposed micro-scale interface displacement field as a function of the macro-scale displacement field

and displacement discontinuity field. After each micro-scale iteration, the value of the macro-scale displacement jump increment is computed from the equation

$$\mathbf{H}_{n+1}^{M,E(i,j)} \Delta \boldsymbol{\alpha}_{n+1}^{M,E(i,j)} = \mathbf{h}_{n+1}^{M,E(i,j)} \tag{2.30}$$

and then the value of the macro-scale displacement jump is updated as

$$\boldsymbol{\alpha}_{n+1}^{M,E(i,j+1)} = \boldsymbol{\alpha}_{n+1}^{M,E(i,j)} + \Delta \boldsymbol{\alpha}_{n+1}^{M,E(i,j)} \tag{2.31}$$

If the value of the residual $\mathbf{h}_{n+1}^{M,E(i,j)}$ obtained by the micro-scale computations is smaller than the chosen tolerance, the iterations are stopped, since the correct value of $\boldsymbol{\alpha}_{n+1}^{M,E(i)}$ is reached. If the residual $\mathbf{h}_{n+1}^{M,E(i,j)}$ is greater than the tolerance, another micro-scale iteration (j) is started to update the value of $\boldsymbol{\alpha}_{n+1}^{M,E(i)}$. The micro-scale computation is then executed with the updated value of $\boldsymbol{\alpha}_{n+1}^{M,E(i,j+1)}$ from the current iteration, so the micro-scale interface displacement field can be updated again using (2.24). The detailed flow-chart of the proposed multi-scale operator split solution procedure for localized failure is shown in Algorithm 1.

After the value of $\boldsymbol{\alpha}^M$ has converged, the second step of the operator split procedure is activated. In the second step, the value of $\boldsymbol{\alpha}^M$ is fixed and the values of the macro-scale displacement field iterative contributions are computed using

$$\left[\mathbf{K}_{n+1}^{M(i)} - \mathbf{F}_{n+1}^{M(i)} (\mathbf{H}_{n+1}^{M(i)})^{-1} \mathbf{F}_{n+1}^{M(i)^T}\right] \Delta \mathbf{d}_{n+1}^{M(i)} = -\mathbf{r}_{n+1}^{M(i)} \tag{2.32}$$

followed by the corresponding update of the macro displacements

$$\mathbf{d}_{n+1}^{M(i+1)} = \mathbf{d}_{n+1}^{M(i)} + \Delta \mathbf{d}_{n+1}^{M(i)} \tag{2.33}$$

for *n=0,1,2,...* **do**
  for *(i)=1,2,...* **do**
    for *(j)=1,2,...* **do**
      for E=1,2,...,n_{el}^{M} **do**
        $\bar{\mathbf{d}}_{n+1}^{m(i,j,1)}\Big|_{\Gamma^{M,E}} = \mathbf{T}^{E}\mathbf{d}_{n+1}^{M,E(i)} + \mathbf{S}^{E}\boldsymbol{\alpha}_{n+1}^{M,E(i,j)}$
        for *(k)=1,2,...* **do**
          $\mathbb{A}_{e=1}^{n_{el}^{m}}\left[\mathbf{K}_{n+1}^{m,e(i,j,k)}\left(\mathbf{d}_{n+1}^{m,e(i,j,k+1)} - \mathbf{d}_{n+1}^{m,e(i,j,k)}\right) = -\mathbf{r}_{n+1}^{m,e(i,j,k)}\right]$
          if $||\mathbf{r}_{n+1}^{m,e(i,j,k+1)}|| > tol$ **then**
            (k) = (k) + 1
          else
            $\mathbf{r}_{n+1}^{M,E(i,j)} = \mathbf{T}^{E,T}\tilde{\mathbf{r}}_{n+1}^{m}$
            $\mathbf{h}_{n+1}^{M,E(i,j)} = \mathbf{S}^{E,T}\tilde{\mathbf{r}}_{n+1}^{m}$
            $\mathbf{K}_{n+1}^{M,E(i,j)} = \mathbf{T}^{E,T}\tilde{\mathbf{K}}_{n+1}^{m}\mathbf{T}^{E}$
            $\mathbf{F}_{n+1}^{M,E(i,j)} = \mathbf{T}^{E,T}\tilde{\mathbf{K}}_{n+1}^{m}\mathbf{S}^{E}$
            $\mathbf{H}_{n+1}^{M,E(i,j)} = \mathbf{S}^{E,T}\tilde{\mathbf{K}}_{n+1}^{m}\mathbf{S}^{E}$
            exit loop
          end
        end
        E = E + 1
      end
      if $\left\|\mathbf{h}_{n+1}^{M,E(i,j)}\right\| > tol$ **then**
        $\Delta\boldsymbol{\alpha}_{n+1}^{M,E(i,j)} = \mathbf{H}_{n+1}^{M,E(i,j)^{-1}}\mathbf{h}_{n+1}^{M,E(i,j)}$
        $\boldsymbol{\alpha}_{n+1}^{M,E(i,j+1)} = \boldsymbol{\alpha}_{n+1}^{M,E(i,j)} + \Delta\boldsymbol{\alpha}_{n+1}^{M,E(i,j)}$
        (j) = (j) + 1
      else
        exit loop
      end
    end
    $\mathbb{A}_{E=1}^{n_{elem}^{M}}\left[\left(\mathbf{K}_{n+1}^{M,E(i)} - \mathbf{F}_{n+1}^{M,E(i)}\mathbf{H}_{n+1}^{M,E(i)^{-1}}\mathbf{F}_{n+1}^{M,E(i)^{T}}\right)\left(\mathbf{d}_{n+1}^{M,E(i+1)} - \mathbf{d}_{n+1}^{M,E(i)}\right) = -\mathbf{r}_{n+1}^{M,E(i)}\right]$
    if $\left\|\mathbf{r}_{n+1}^{M,E(i)}\right\| > tol$ **then**
      (i) = (i) + 1
    else
      exit loop
    end
  end
  n = n + 1
end

Algorithm 1: Operator split multi-scale iterative solution procedure for localized failure

2.2.1 Transformation matrix T

To illustrate the construction of transformation matrices **T** and **S** that are used to transfer the information between the scales, a simple 2D multi-scale example is given. On the macro-scale, there is a Q4 element with two degrees of freedom in each node. Inside the macro-element, a micro-mesh that consists of 18 CST (constant strain triangle) elements is created, as shown in Figure 2.6.

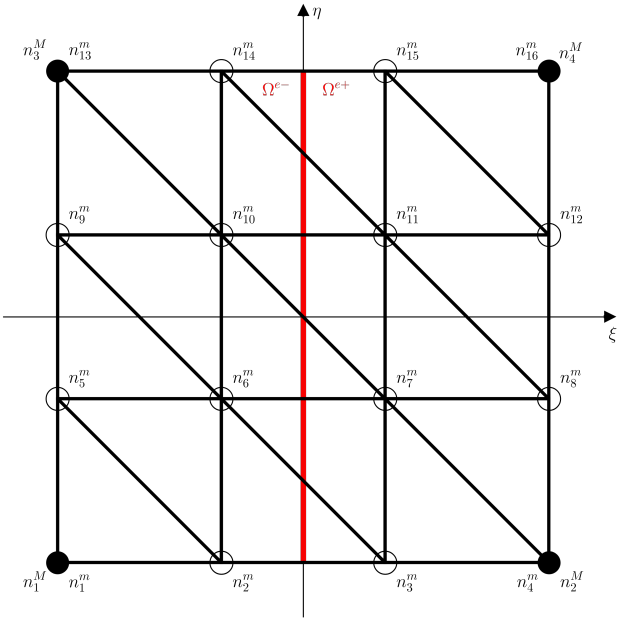


Figure 2.6: Macro-scale Q4 element with 18 CST micro-scale elements that fit inside

To compute the transformation matrix **T**, the isoparametric macro-scale coordinates of each micro-scale node (the position of a micro-scale node inside the macro-scale element) are introduced into the macro-scale shape functions.

The relation that can be used to calculate the displacements of a micro-scale node n_a^m based on the displacements of the macro-scale nodes is shown in equation (2.12). Then, the sum of the shape functions with introduced coordinates of the micro-scale nodes can be written as

$$\mathbf{T}_a = \sum_b N_b^M(\mathbf{x_a}) \tag{2.34}$$

or in matrix notation

$$\mathbf{T}_a = \begin{bmatrix} N_1^M(\mathbf{x_a}) & N_2^M(\mathbf{x_a}) & N_3^M(\mathbf{x_a}) & N_4^M(\mathbf{x_a}) \end{bmatrix} \tag{2.35}$$

In this way, micro-scale node n_1^m with coordinates $\mathbf{x_a} = (-1, -1)$ will lead to matrix $\mathbf{T}_1$ equal to

$$\mathbf{T}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.36}$$

and micro-scale node n_2^m with coordinates $\mathbf{x_a} = (-\frac{1}{3}, -1)$ to matrix $\mathbf{T}_2$ equal to

$$\mathbf{T}_2 = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} & 0 & 0 \end{bmatrix} \tag{2.37}$$

The transformation matrix $\mathbf{T}_a$ is constructed for each of the micro-scale elements in the same manner. The final values of the transformation matrix $\mathbf{T}$ is then equal to

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{2}{3} & \frac{1}{3} & 0 & 0 \\ \frac{1}{3} & \frac{2}{3} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & \frac{2}{3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.38}$$

To be able to write the equation for calculating the values of micro-scale displacements in matrix notation, the transformation matrix $\mathbf{T}$ needs to be expanded to take into account both degrees of freedom

$$\underset{(24\times1)}{\bar{\mathbf{d}}^m}\Big|_{\Gamma^M} = \underset{(24\times8)}{\mathbf{T}} \quad \underset{(8\times1)}{\mathbf{d}^M} \tag{2.39}$$

Since the same shape functions are used to calculate transformation matrix values for both degrees of freedom, the matrix $\mathbf{T}$ that is used to calculate the displacements for both degrees of freedom equals to

$$\mathbf{T} = \underset{(24\times8)}{\begin{bmatrix} 1 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 1 & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 1 & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & 1 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & 1 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 1 \end{bmatrix}^T} \tag{2.40}$$

2.2.2 Transformation matrix S

To compute the transformation matrix $\mathbf{S}$, needed for the multi-scale computation procedure with localized failure, the isoparametric macro-scale coordinates of

each micro-scale node (the position of a micro-scale node inside the macro-scale element) are introduced into the macro-scale incompatible mode function M.

The key point here is to calculate the value of the incompatible mode function (2.19). If the simplest case is taken into account, where the crack opening is only in mode I, then the macro-scale nodes n_1^M and n_3^M are part of the domain Ω^{e-} and the macro-scale nodes n_2^M and n_4^M are part of the domain Ω^{e+}. Then, the incompatible mode function can be written as

$$M(\xi, \eta) = H_\Gamma(\xi, \eta) - \frac{1}{2}(1 + \xi) \tag{2.41}$$

or, taking into account the Heaviside function in (2.20), it can be written as

$$M(\xi) = \begin{cases} -\frac{1}{2}(1 + \xi), & \xi \in [-1, 0\rangle \\ 1 - \frac{1}{2}(1 + \xi), & \xi \in [0, 1] \end{cases} \tag{2.42}$$

In a more general case, when the crack opening is happening under a certain angle, we need to determine if the micro-scale node is inside the domain Ω^{e-} or Ω^{e+} of the macro-scale element, before calculating the value of the Heaviside function. The incompatible mode function will then be a function of both ξ and η (the coordinate of a micro-scale node will be taken into account to calculate the value of the Heaviside function).

Now, the relation used to calculate the displacements of a micro-scale node n_a^m based on the displacements of the macro-scale nodes and the displacement jump $\boldsymbol{\alpha}$ is shown in equation (2.23). Then, the incompatible mode function with introduced coordinates of the micro-scale nodes can be written as

$$S_a = M^M(\mathbf{x_a}) \tag{2.43}$$

In this way, micro-scale node n_1^m with coordinates $\mathbf{x_1} = (-1, -1)$ will lead to S_1 that equals to

$$S_1 = 0 \tag{2.44}$$

and micro-scale node n_2^m with coordinates $\mathbf{x_2} = (-\frac{1}{3}, -1)$ will lead to S_2 that equals to

$$S_1 = -\frac{1}{3} \tag{2.45}$$

The part S_a of the transformation matrix $\mathbf{S}$ is constructed for each of the micro-scale element in the same manner. The final values of the transformation

matrix $\mathbf{S}$ is then equal to

$$\mathbf{S} = \begin{bmatrix} 0 \\ -\frac{1}{3} \\ \frac{1}{3} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -\frac{1}{3} \\ \frac{1}{3} \\ 0 \end{bmatrix} \tag{2.46}$$

To be able to write the equation for calculating the values of micro-scale displacements in matrix notation, the transformation matrix $\mathbf{S}$ needs to be expanded to take into account both normal and tangential component of the crack opening $\boldsymbol{\alpha}$

$$\underset{(24\times1)}{\bar{\mathbf{d}}^m}\Big|_{\Gamma^M} = \underset{(24\times8)}{\mathbf{T}}\ \underset{(8\times1)}{\mathbf{d}^M} + \underset{(24\times2)}{\mathbf{S}}\ \underset{(2\times1)}{\boldsymbol{\alpha}^M} \tag{2.47}$$

Then, the transformation matrix $\mathbf{S}$ that is used to calculate the displacements for both degrees of freedom and both components of crack opening equals to

$$\underset{(24\times2)}{\mathbf{S}} = \begin{bmatrix} 0 & 0 & -\frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \end{bmatrix}^T \tag{2.48}$$

2.3 Micro-scale damage model

The micro-scale model used in the multi-scale computation procedure is described in this section for better understanding, although the theoretical formulation is not developed or improved in any way in the scope of this thesis. The existing formulation and software code developed in (Brancherie, 2003) has been implemented and rewritten as a new software code to improve the efficiency, make it more upgradeable and to better understand the model. The developed code has been used for numerical examples in this thesis, and later further upgraded and used in (Rukavina, 2018).

On the micro-scale, a 2D damage model with hardening and softening is used, as described in (Brancherie, 2003) and (Rukavina, 2018). The chosen finite element

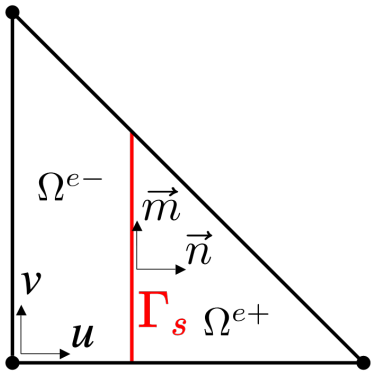


Figure 2.7: CST element with two sub-domains related to the displacement discontinuity

is the constant strain triangle (CST). The hardening part is modeled with an isotropic damage model, while the softening part has an embedded discontinuity implemented in the displacement field that can represent localized failure.

The elasto-damage model with hardening and softening can give a realistic description of the behavior of the material that leads to failure. First, the material is described with a linear elastic behavior. In the hardening phase, the micro-cracks are developing along the fracture process zone. Finally, when the micro-cracks coalesce, a macro-crack appears and starts to propagate, and the material enters into the softening phase (Brancherie and Ibrahimbegovic, 2009; Kucerova et al., 2009). The softening is described with an exponential function. The CST element has one Gauss integration point defined at the barycenter of the triangle, and the discontinuity is positioned in the middle of the element, as it can be seen in Figure 2.7.

The localized failure is formulated within an anisotropic multi-surface model that can take into account the crack opening in mode I (traction), and in mode II (shear) (Rukavina, 2018; Ibrahimbegovic and Wilson, 1991; Kozar et al., 2018).

The standard finite element shape functions for the CST element (as shown in (Zienkiewicz et al., 2005)) are used

$$N_a(x,y) = \frac{1}{2\Delta}(a_a + b_a x + c_a y), \quad a = 1,2,3 \tag{2.49}$$

where coefficients a_a, b_a and c_a are calculated as

$$\begin{aligned} a_1 &= x_2y_3 - x_3y_2, \quad b_1 = y_2 - y_3, \quad c_1 = x_3 - x_2 \\ a_2 &= x_3y_1 - x_1y_3, \quad b_2 = y_3 - y_1, \quad c_2 = x_1 - x_3 \\ a_3 &= x_1y_2 - x_2y_1, \quad b_3 = y_1 - y_2, \quad c_3 = x_2 - x_1 \end{aligned} \tag{2.50}$$

and $\Delta = (x_1 b_1 + x_2 b_2 + x_3 b_3)/2$ is the area od the triangle. Coordinates x_a and y_a are the coordinates of the triangle nodes.

To be able to represent the displacement jump, the incompatible mode function is introduced as

$$M(\mathbf{x}) = H_{\Gamma_s}(\mathbf{x}) - \sum_{a \in \Omega^{e+}} N_a(\mathbf{x}) \tag{2.51}$$

where H_{Γ_s} is the Heaviside step function defined as

$$H_{\Gamma_s}(\mathbf{x}) = \begin{cases} 0, \mathbf{x} \in \Omega^{e-} \\ 1, \mathbf{x} \in \Omega^{e+} \end{cases} \tag{2.52}$$

where sub-domains Ω^{e+} and Ω^{e-} are part of the CST element domain Ω divided by the discontinuity surface Γ_s as shown in Figure 2.7.

Now, with the incompatible mode function introduced, the total displacement field can be written as the sum of the standard and the incompatible part

$$\mathbf{u}(\mathbf{x}) = \sum_{a=1}^{3} N_a(\mathbf{x}) \mathbf{d}_a + M(\mathbf{x}) \boldsymbol{\alpha} \tag{2.53}$$

where N_a is the standard linear shape function of the constant strain triangle element node a, $\mathbf{d}_a$ is the standard displacement field of element node a, and $\boldsymbol{\alpha}$ is the displacement jump field at the discontinuity.

The strain field approximation is computed as a derivative of (2.53) and is also written as a sum of the standard part and the incompatible part

$$\boldsymbol{\varepsilon}(\mathbf{x}) = \sum_{a=1}^{3} \mathbf{B}_a(\mathbf{x}) \mathbf{d}_a + \mathbf{G}_\mathbf{r}(\mathbf{x}) \boldsymbol{\alpha} \tag{2.54}$$

where $\mathbf{B}_a$ is a matrix of the standard linear shape function derivatives of the element node a and $\mathbf{G}_\mathbf{r}$ is a matrix of the derivatives of the incompatible mode function.

The matrix $\mathbf{B}_a$ is defined as

$$\mathbf{B}_a = \begin{bmatrix} \frac{\partial N_a}{\partial x} & 0 \\ 0 & \frac{\partial N_a}{\partial y} \\ \frac{\partial N_a}{\partial y} & \frac{\partial N_a}{\partial x} \end{bmatrix} \tag{2.55}$$

The matrix $\mathbf{G}_\mathbf{r}$ can be split into a regular part and a singular part

$$\mathbf{G}_\mathbf{r}(\mathbf{x}) = \bar{\mathbf{G}}_\mathbf{r}(\mathbf{x}) + \bar{\bar{\mathbf{G}}}_\mathbf{r} \tag{2.56}$$

where the regular part is defined as

$$\bar{\mathbf{G}}_{\mathbf{r}} = \begin{bmatrix} \frac{\partial M}{\partial x} & 0 \\ 0 & \frac{\partial M}{\partial y} \\ \frac{\partial M}{\partial y} & \frac{\partial M}{\partial x} \end{bmatrix} \tag{2.57}$$

and the singular part as

$$\bar{\bar{\mathbf{G}}}_{\mathbf{r}} = \mathbf{n}\delta_{\Gamma} \tag{2.58}$$

where $\mathbf{n}$ is the normal vector at the discontinuity and δ_{Γ} is the derivative of the Heaviside function at the discontinuity.

It has been shown in (Ibrahimbegovic and Wilson, 1991) that the function $\mathbf{G}_{\mathbf{r}}$ has to be modified in order to satisfy the patch test

$$\mathbf{G}_{\mathbf{v}}(\mathbf{x}) = \mathbf{G}_{\mathbf{r}}(\mathbf{x}) - \frac{1}{A^e}\int_{\Omega^e} \mathbf{G}_{\mathbf{r}}(\mathbf{x}) d\Omega^e \tag{2.59}$$

where A^e is the area of the CST finite element.

After (2.56) is introduced in (2.59), as shown in (Brancherie, 2003), the function $\mathbf{G}_{\mathbf{v}}$ can be written as

$$\mathbf{G}_{\mathbf{v}}(\mathbf{x}) = -\frac{l_{\Gamma}^e}{A^e}\mathbf{n} + \mathbf{n}\delta_{\Gamma} \tag{2.60}$$

where l_{Γ}^e is the length of the discontinuity.

The function $\mathbf{G}_{\mathbf{v}}(\mathbf{x})$ then satisfies the patch test condition

$$\int_{\Omega^e} \mathbf{G}_{\mathbf{v}}(\mathbf{x}) d\Omega^e = 0 \tag{2.61}$$

With the previous formulation defined, the system of equations to be finally solved is

$$\begin{aligned} \mathbb{A}_{e=1}^{n}[\mathbf{f_{int}}^e - \mathbf{f_{ext}}^e] &= 0 \\ \mathbf{h}^e &= 0 \end{aligned} \tag{2.62}$$

where $\mathbf{f_{int}}^e$ and $\mathbf{f_{ext}}^e$ are internal and external force vectors, respectively, and $\mathbf{h}^e$ is the residual at the discontinuity. They can be defined as

$$\begin{aligned} \mathbf{f_{int}}^e &= \int_{\Omega^e} \mathbf{B}^T \sigma d\Omega^e \\ \mathbf{f_{ext}}^e &= \int_{\Omega^e} \mathbf{N}^T \mathbf{b} d\Omega^e + \int_{\Gamma_{\Omega}^e} \mathbf{N}^T \mathbf{t}_{\Gamma_{\Omega}^e} d\Gamma \\ \mathbf{h}^e &= \int_{\Omega^e} \bar{\mathbf{G}}_{\mathbf{v}}^T \sigma d\Omega^e + \int_{\Gamma_s} \bar{\bar{\mathbf{G}}}_{\mathbf{v}}^T \mathbf{t}_{\Gamma_s} d\Omega^e \end{aligned} \tag{2.63}$$

where $\boldsymbol{\sigma}$ is the stress in the element, $\mathbf{b}$ is the volume forces vector, $\mathbf{t}_{\Gamma^e_\Omega}$ is the traction at the boundary of the element domain Γ^e_Ω, and $\mathbf{t}_{\Gamma_s}$ is the traction at the discontinuity Γ_s.

When this system is linearized, as shown in (Rukavina, 2018), the following set of equations is obtained

$$\begin{bmatrix} \mathbf{K} & \mathbf{F}_r \\ \mathbf{F}_v & \mathbf{H} + \mathbf{K}_\alpha \end{bmatrix} \begin{bmatrix} \Delta \mathbf{d}_{n+1} \\ \Delta \boldsymbol{\alpha}_{n+1} \end{bmatrix} = - \begin{bmatrix} \mathbf{f}_{\mathbf{int,n+1}} - \mathbf{f}_{\mathbf{ext,n+1}} \\ 0 \end{bmatrix} \tag{2.64}$$

where the tangent stiffness matrices are defined as

$$\begin{aligned} \mathbf{K}^{e,(i)}_{n+1} &= \int_{\Omega^e} \mathbf{B}^{e,T} \mathbf{C}^{ed,(i)}_{n+1} \mathbf{B}^e d\Omega \\ \mathbf{F}^{e,(i)}_{r,n+1} &= \int_{\Omega^e} \mathbf{B}^{e,T} \mathbf{C}^{ed,(i)}_{n+1} \bar{\mathbf{G}}^e_r d\Omega \\ \mathbf{F}^{e,(i)}_{v,n+1} &= \int_{\Omega^e} \bar{\mathbf{G}}^{e,T} \mathbf{C}^{ed,(i)}_{n+1} \mathbf{B}^e_v d\Omega \\ \mathbf{H}^{e,(i)}_{n+1} &= \int_{\Omega^e} \bar{\mathbf{G}}^{e,T}_v \mathbf{C}^{ed,(i)}_{n+1} \bar{\mathbf{G}}^e_r d\Omega \\ \mathbf{K}^{(i)}_{\alpha,n+1} &= l^e_{\Gamma_s} \bar{\bar{\mathbf{C}}}^{ed,(i)}_{n+1} \end{aligned} \tag{2.65}$$

where $\mathbf{C}^{ed,(i)}_{n+1}$ and $\bar{\bar{\mathbf{C}}}^{ed,(i)}_{n+1}$ are the elasto-damage modulus and tangent modulus for the discontinuity, respectively, as explained in detail in (Simo and Taylor, 1985).

After performing static condensation on the set of equations (2.64), the displacement jump can be computed as

$$\Delta \boldsymbol{\alpha}_{n+1} = -(\mathbf{H} + \mathbf{K}_\alpha)^{-1} \mathbf{F}_v \Delta \mathbf{d}_{n+1} \tag{2.66}$$

To compute the increment of the displacement vector, a value of displacement jump is introduced back into (2.64) so the final equation to calculate the displacement vector increments can be written as

$$\left[\mathbf{K} - \mathbf{F}_r (\mathbf{H} + \mathbf{K}_\alpha)^{-1} \mathbf{F}_v\right] \Delta \mathbf{d}_{n+1} = -(\mathbf{f}_{int,n+1} - \mathbf{f}_{ext,n+1}) \tag{2.67}$$

The finite element procedure is described in general, while the details can be found in (Brancherie, 2003). The computations are divided into a bulk computation and a computation at the discontinuity.

The bulk computations are governed by an isotropic damage model. The damage evolution is described with a Lagrange multiplier $\bar{\gamma}$ and a damage function $\bar{\phi}$ that is used to test the admissibility of the stress in the bulk. The conditions for loading and unloading can be presented in Kuhn-Tucker form as stated in (Ibrahimbegovic, 2009)

$$\bar{\phi}(\cdot) \leq 0; \quad \dot{\bar{\gamma}} \geq 0; \quad \dot{\bar{\gamma}} \bar{\phi}(\cdot) = 0 \tag{2.68}$$

where $\dot{\bar{\gamma}}$ represents a derivative of the Lagrange multiplier with respect to time. The internal variables used to describe the damage model are the damage compliance tensor for the bulk $\bar{\boldsymbol{D}}$ and the hardening variable $\bar{\xi}$. A linear hardening function is used to describe the behavior of the material in the hardening phase where the stress-like hardening variable is defined as

$$\bar{q} = -\bar{K}\bar{\xi} \tag{2.69}$$

where $\bar{K}$ is the hardening modulus. The chosen value for the damage function, as described in (Brancherie, 2003), is

$$\bar{\phi}(\boldsymbol{\sigma}, \bar{q}) = \sqrt{\boldsymbol{\sigma} \cdot \boldsymbol{D}^e \cdot \boldsymbol{\sigma}} - \frac{1}{\sqrt{E}}(\bar{\sigma}_f - \bar{q}) \leq 0 \tag{2.70}$$

where $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{D}^e$ is the unchanged elastic compliance tensor for the bulk material and is computed as the inverse of the elastic constitutive matrix, E is the Young's modulus, and $\bar{\sigma}_f$ is the limit stress. The bulk computation starts with an elastic trial step, where there is no evolution of the internal variables. The trial value of the damage function $\bar{\phi}$ is tested, to check if the elastic step is really valid or the damage has already occurred. If the value of the damage function is less than or equal to zero, that means the step is elastic and the trial values can be taken as final. If the value of the damage function is greater than zero, the computations for the damage step have to be executed. In the damage step computation, a new value for the Lagrange multiplier $\bar{\gamma}$ is computed, after which the value for the elasto-damage modulus for the bulk can be computed. The evolution equations of the internal variables in the damage step are calculated as in (Brancherie, 2003)

$$\begin{aligned} \dot{\bar{\mathbf{D}}} &= \dot{\bar{\gamma}} \frac{\bar{\mathbf{D}}}{\sqrt{\boldsymbol{\sigma} \cdot \boldsymbol{D}^e \cdot \boldsymbol{\sigma}}} \\ \dot{\bar{\xi}} &= \dot{\bar{\gamma}} \frac{1}{\sqrt{E}} \end{aligned} \tag{2.71}$$

When the value of stress inside the finite element becomes greater than the value of the ultimate stress, the softening has started inside the element. In the softening phase, the computation at the discontinuity has to be executed as the macro-crack opening starts to form. Once the computation at the discontinuity starts, the bulk computations are not executed any more. In the softening phase, the stress is localized at the discontinuity, and the bulk of the material is unloading elastically. The softening part is defined by an anisotropic multi-surface damage model, and the crack opening can happen in mode I and mode II. For the softening part, two Lagrange multipliers, $\bar{\bar{\gamma}}_1$ and $\bar{\bar{\gamma}}_2$, are defined to describe the damage evolution at the discontinuity. Also, two damage functions, $\bar{\bar{\phi}}_1$ and $\bar{\bar{\phi}}_2$, are defined

to test the admissibility of the traction at the discontinuity in normal and tangential direction. In the same way as for the hardening phase, the conditions for the discontinuity in the softening phase are presented in Kuhn-Tucker form as

$$\begin{aligned} \bar{\bar{\phi}}_1(\cdot) \leq 0; \quad \dot{\bar{\bar{\gamma}}}_1 \geq 0; \quad \dot{\bar{\bar{\gamma}}}_1 \bar{\bar{\phi}}_1(\cdot) = 0 \\ \bar{\bar{\phi}}_2(\cdot) \leq 0; \quad \dot{\bar{\bar{\gamma}}}_2 \geq 0; \quad \dot{\bar{\bar{\gamma}}}_2 \bar{\bar{\phi}}_2(\cdot) = 0 \end{aligned} \tag{2.72}$$

The internal variables used in the softening phase are the damage compliance tensor for the discontinuity $\bar{\bar{\mathbf{Q}}}$ and the softening variable $\bar{\bar{\xi}}$. An exponential softening function is used to describe the behavior of the material in the softening phase where the traction-like softening variable is defined as in (Brancherie, 2003)

$$\bar{\bar{q}} = \bar{\bar{\sigma}}_f \left[1 - \exp\left(- \frac{\bar{\bar{\beta}}}{\bar{\bar{\sigma}}_f} \bar{\bar{\xi}} \right) \right] \tag{2.73}$$

where $\bar{\bar{\sigma}}_f$ is the ultimate stress in normal direction, and $\bar{\bar{\beta}}$ is a material parameter that controls softening and is inversely proportional to the fracture energy. The equations for the damage functions, as described in (Brancherie, 2003), are

$$\begin{aligned} \bar{\bar{\phi}}_1(\boldsymbol{t}_{\Gamma_s}, \bar{\bar{q}}) &= \boldsymbol{t}_{\Gamma_s} \cdot \mathbf{n} - (\bar{\bar{\sigma}}_f - \bar{\bar{q}}) \leq 0 \\ \bar{\bar{\phi}}_2(\boldsymbol{t}_{\Gamma_s}, \bar{\bar{q}}) &= |\boldsymbol{t}_{\Gamma_s} \cdot \mathbf{m}| - (\bar{\bar{\sigma}}_s - \frac{\bar{\bar{\sigma}}_s}{\bar{\bar{\sigma}}_f} \bar{\bar{q}}) \leq 0 \end{aligned} \tag{2.74}$$

where $\boldsymbol{t}_{\Gamma_s}$ is the traction at the discontinuity, $\mathbf{n}$ is the normal direction of the discontinuity, $\mathbf{m}$ is the tangential direction of the discontinuity, and $\bar{\bar{\sigma}}_s$ is the ultimate stress in the tangential direction. The evolution equations of the internal variables for the softening part are calculated as in (Brancherie, 2003)

$$\begin{aligned} \dot{\bar{\bar{\mathbf{Q}}}} &= \dot{\bar{\bar{\gamma}}}_1 \frac{\mathbf{n} \otimes \mathbf{n}}{\mathbf{t} \cdot \mathbf{n}} + \dot{\bar{\bar{\gamma}}}_2 \frac{\mathbf{m} \otimes \mathbf{m}}{|\mathbf{t} \cdot \mathbf{m}|} \\ \dot{\bar{\bar{\xi}}} &= \dot{\bar{\bar{\gamma}}}_1 + \dot{\bar{\bar{\gamma}}}_2 \frac{\bar{\bar{\sigma}}_s}{\bar{\bar{\sigma}}_f} \end{aligned} \tag{2.75}$$

The final solution is obtained using the operator split solution procedure where the computations are divided into a local and a global phase. In the local phase, evolution equations of internal variables defined locally at each Gauss point are computed, using implicit backward Euler time integration scheme. In the global phase, after the local phase has finished, the equilibrium equations are solved and the values of the nodal displacements are computed using the incremental and iterative Newton-Raphson procedure (Kozar et al., 2018).

2.4 Numerical examples

2.4.1 Validation examples for the proposed multi-scale approach

To validate the theoretical formulation with numerical examples, a simple tension test is chosen. The results for the proposed multi-scale method with localized failure are compared against the monolithic solution and the previously developed multi-scale method. The goal is to prove that the multi-scale method with added embedded discontinuity on the macro-scale can represent localized failure and produce the same quality results as the monolithic solution. The simple tension test with boundary conditions and imposed displacement is shown in Figure 2.8.

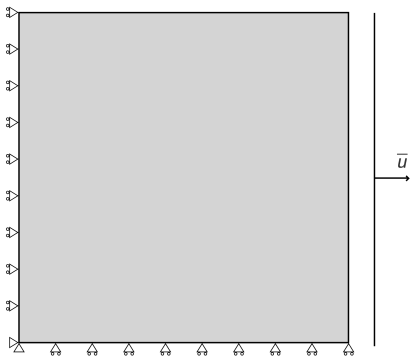
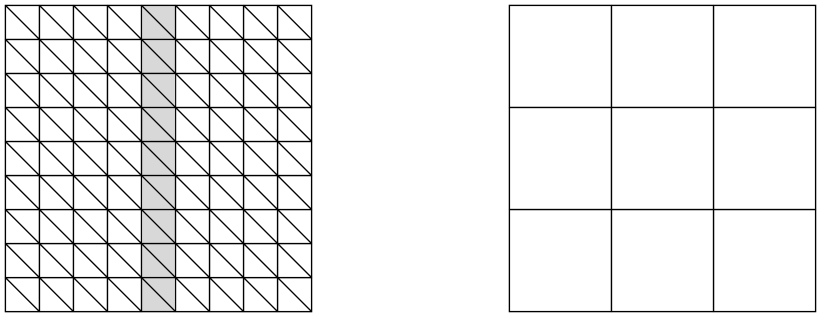


Figure 2.8: Simple tension test: boundary conditions

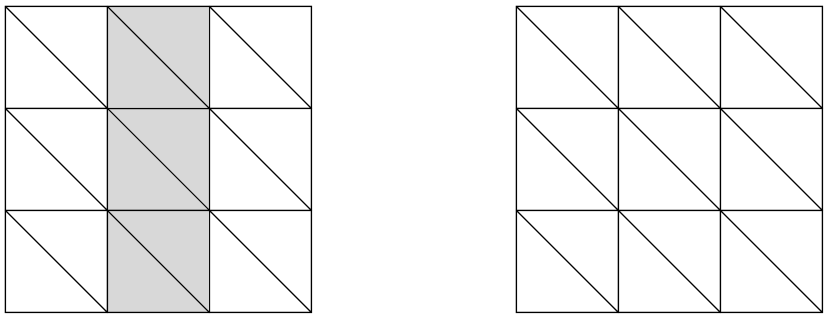
The mesh for the mono-scale example (shown in Figure 2.9a) consist of 18 x 18 CST elements. The weakened elements that are going to crack first are shown in gray. Elements are weakened by simple change of material properties that will cause them to enter softening phase much sooner than other elements. The macro-scale mesh (shown in Figure 2.9b) of the multi-scale example has the same dimensions as the mono-scale mesh, and consists of 3 x 3 Q4 elements.



(a) **(b)**

Figure 2.9: (a) Mono-scale mesh; (b) Macro-scale mesh

Inside each of the macro-scale elements there is a micro-scale mesh that consists of 6 x 6 CST elements. In general, a different micro-scale mesh can be defined for each macro-scale element. The three central macro-scale elements (where the crack should appear) contain the micro-mesh shown in Figure 2.10a. The other six macro-scale elements on the left and the right side contain the micro-mesh without the weakened elements (shown in Figure 2.10b). For the multi-scale example, the boundary conditions are imposed only on the macro-scale.



(a) **(b)**

Figure 2.10: (a) Micro-scale mesh with weakened elements; (b) Micro-scale mesh without weakened elements

In this way, the mono-scale and the multi-scale examples have the same total number of CST elements, the same position of the weakened elements and the same dimensions, and therefore should behave in the same way. The central column of the weakened CST elements inside the micro-scale mesh coincides with the displacement discontinuity of the Q4 macro-scale element.

The following material parameters for the CST damage element are chosen: Young's modulus $E = 38\,000$ MPa, hardening modulus $\bar{K} = 1000$ MPa, Poisson's ratio $\nu = 0$, limit stress for hardening $\bar{\sigma} = 2$ MPa, ultimate stress $\bar{\bar{\sigma}} = 2.55$ MPa, the ratio between the softening parameter and the ultimate stress is 20, and the ratio between the tangential and the normal direction ultimate stress is 0.3. For the weakened elements, the ultimate stress $\bar{\bar{\sigma}}$ is set to 2.5 MPa. The dimensions of the mesh are 300 x 300 mm and the imposed displacement $\bar{u} = 1$ mm.

The example is executed in the Finite Element Analysis Program (FEAP) (Taylor, 2014). On the macro-scale, the Q4 element with localized failure developed in Section 2.2 is used, and the procedure described in Algorithm 1. is implemented. On the micro-scale, the CST element with damage model presented in Section 2.3 is used, which is implemented in FEAP based on the work and previous code from (Brancherie, 2003). The multi-scale framework is developed based on the work from (Markovic, 2004), and upgraded for taking into account localized failure. The software implementation is described in detail in Chapter 5.

Elastic response

When the stress is small enough and only elastic response is obtained, the results for mono-scale and both multi-scale methods, standard and with localized failure, produce the same force-displacement diagram as shown in Figures 2.11 and 2.12. Both multi-scale methods are giving the same results since no crack appears on the micro-scale, so there is no need to represent it on the macro-scale. Only the stiffness matrix $\mathbf{K}^M$ and the residual $\mathbf{r}^M$ are transfered from the micro to the macro-scale.

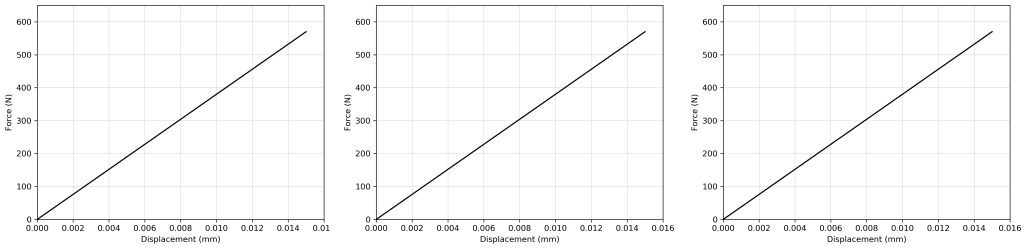


Figure 2.11: Force-displacement diagrams for elastic response: (a) Mono-scale; (b) Multi-scale; (c) Multi-scale with localized failure

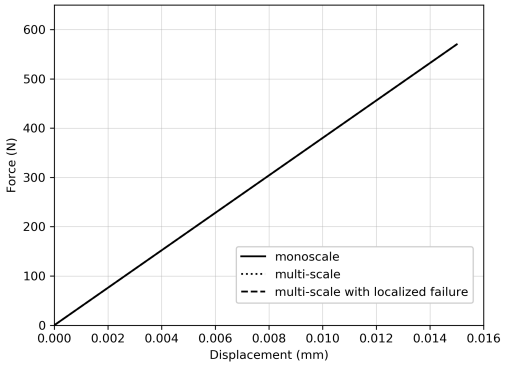


Figure 2.12: Superposed force-displacement diagrams for elastic response computed by three different methods

The multi-scale procedure has also been tested with a simple Q4 linear elastic element implemented in FEAP on the micro-scale, which has produced the same results for the elastic phase. This shows that the proposed multi-scale method can work with different micro-scale elements with little or no modifications at all.

Elasto-damage with hardening

When the structure enters the hardening phase, the force-displacement diagrams for all three methods are still the same, as shown in Figure 2.13. The only change

compared to the elastic phase are the values of the stiffness matrix $\mathbf{K}^M$ due to the introduction of the hardening modulus $\bar{K}$. These changes are successfully captured and transfered from the micro to the macro-scale for both multi-scale methods, as shown in Figures 2.13 and 2.14.

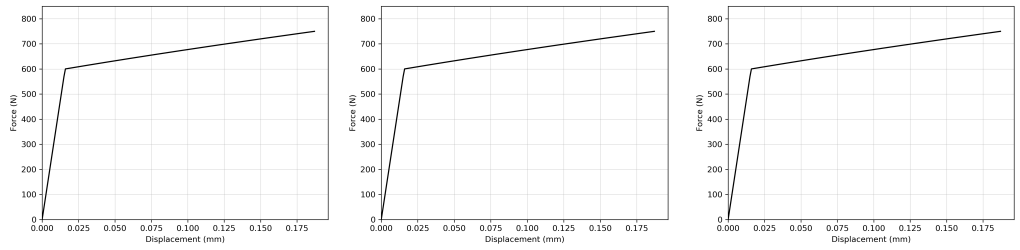


Figure 2.13: Force-displacement diagrams for elasto-damage with hardening: (a) Mono-scale; (b) Multi-scale; (c) Multi-scale with localized failure

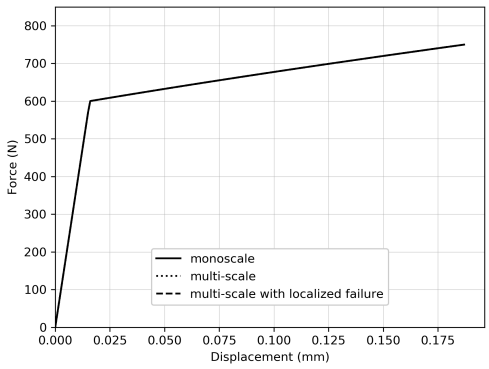


Figure 2.14: Superposed force-displacement diagrams for elasto-damage with hardening computed by three different methods

Elasto-damage with softening

When the value of the ultimate stress in the micro-scale CST element is reached, a crack appears and starts to propagate. The regular multi-scale method cannot transfer the displacement jump on the macro-scale and therefore cannot produce the correct results. The micro-scale nodal displacement field on the interface is not computed correctly, which leads to a concentration of the stresses around the crack, and finally a collapse of the whole structure. This results in a force-displacement diagram shown in Figure 2.15b.

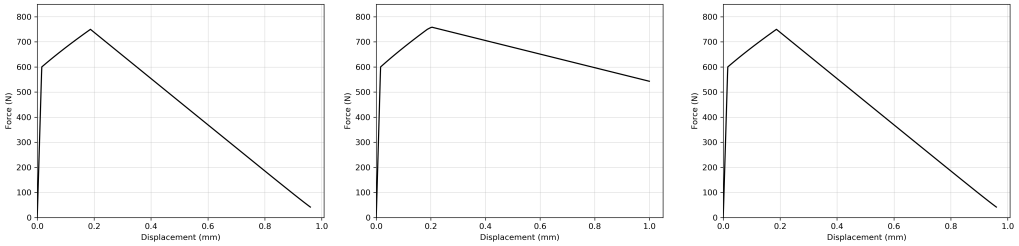


Figure 2.15: Force-displacement diagrams for elasto-damage with softening: (a) Mono-scale; (b) Multi-scale; (c) Multi-scale with localized failure

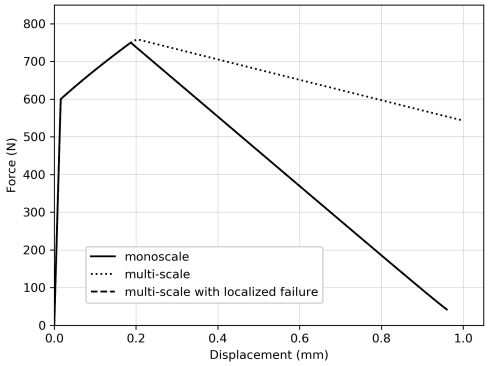
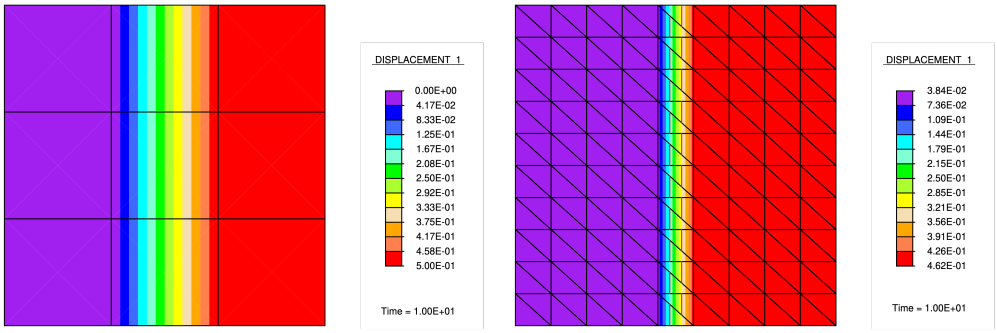


Figure 2.16: Superposed force-displacement diagrams for elasto-damage with softening computed by three different methods

The proposed multi-scale method with localized failure can transfer the displacement jump on the macro-scale and therefore allow the crack opening to increase until the stresses reach zero, as shown in Figure 2.15c. The comparison of all three methods can be seen in Figure 2.16. The results for the mono-scale and multi-scale with localized failure are the same for each time step of the analysis, and the values differ only after the fifth significant figure.

In Figure 2.17a and 2.17b, the final results of FEAP multi-scale code execution are shown for the displacement distribution in direction x for both macro and micro-scale. The macro-scale results are shown for the micro-scale mesh that fits in the macro-scale element positioned in the middle column of the macro-scale mesh, one of those in which the crack appears.

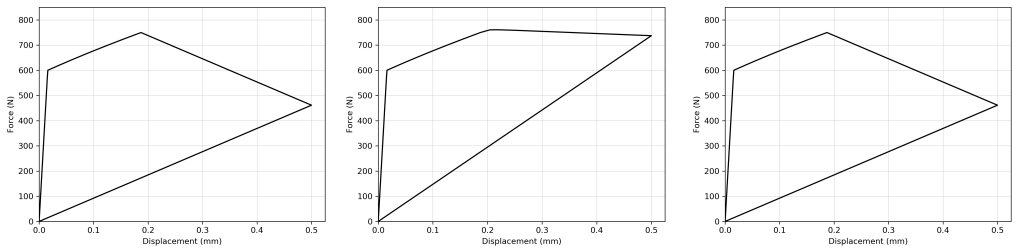


(a) **(b)**

Figure 2.17: (a) Macro-scale displacement distribution in direction x; (b) Micro-scale displacement distribution in direction x with shown CST elements with crack in the middle

Elasto-damage with softening - unloading

In this example the specimen is subjected to unloading after the initial 50% of the load has been applied and the softening phase has started. In the unloading phase the force goes back to zero as there are not any irreversible deformations present. This linear response is different from the linear elastic response in the loading phase as it can be seen in Figure 2.18a. The regular multi-scale method does not produce the correct results starting from the softening phase, although the force goes back to zero in the unloading phase.



(a) **(b)** **(c)**

Figure 2.18: Force-displacement diagrams for elasto-damage with softening - unloading: (a) Mono-scale; (b) Multi-scale; (c) Multi-scale with localized failure

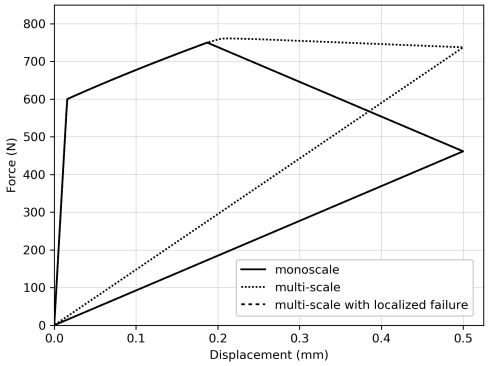


Figure 2.19: Superposed force-displacement diagrams for elasto-damage with softening (unloading) computed by three different methods

The proposed multi-scale method with localized failure gives the same results for the unloading phase as the mono-scale method, as shown in Figure 2.18c. The comparison of all three methods can be seen in Figure 2.19.

2.4.2 Validation examples of mesh objectivity

To prove that the proposed multi-scale method with localized failure is not mesh dependent, it is tested for two more examples with a different number of micro-scale mesh elements. In Figure 2.20a, the micro-scale mesh consists of 6 x 6 CST elements, in Figure 2.20b of 18 x 18 elements and in Figure 2.20c of 54 x 54 CST elements. The weakened elements that are present only in the central macro-scale elements are again shown in gray.

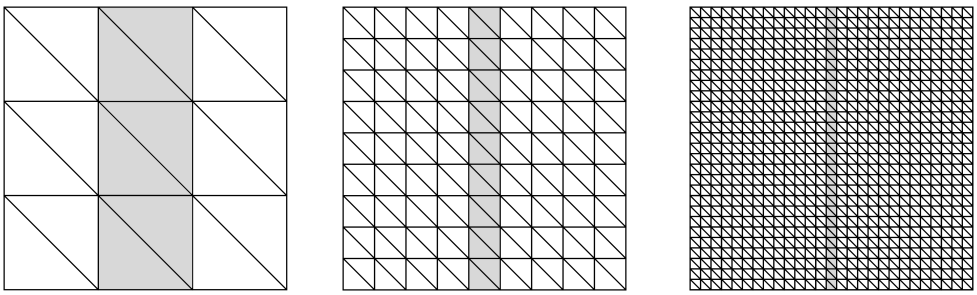


Figure 2.20: Micro-scale mesh: (a) 6 x 6 elements; (b) 18 x 18 elements; (c) 54 x 54 elements

After running the multi-scale examples with different micro-scale meshes, identical results are obtained for all three cases, as shown in Figures 2.21 and 2.22.

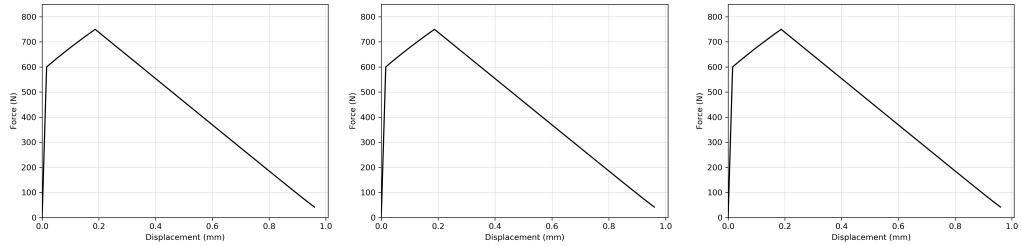


Figure 2.21: Force-displacement diagrams for elasto-damage with softening for multi-scale with localized failure computed with a different number of micro-scale elements: (a) 6 x 6 elements; (b) 18 x 18 elements; (c) 54 x 54 elements

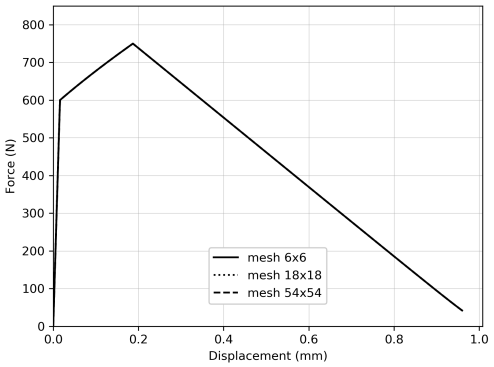


Figure 2.22: Superposed force-displacement diagrams for elasto-damage with softening for multi-scale with localized failure computed with a different number of micro-scale elements

Conclusion

It has been shown that the proposed multi-scale procedure for localized failure can efficiently deal with softening behavior happening on the micro-scale in a 2D setting, all the while being mesh independent. The multi-scale procedure has been implemented in FEAP, and it has been shown on the numerical examples that it produces the same results as the mono-scale procedure for the elastic, hardening and softening phase, as well as in unloading. The numerical examples are performed with a CST damage element on the micro-scale, but the procedure is easily extendable, as it allows for the choice of different finite elements on the micro-scale that can simulate specific behaviors of heterogeneous materials. It can be noted that the macro-scale and micro-scale elements have the same dimension (2D) and the same degrees of freedom per node, which does not have to be the case, as it will be shown in the following chapter.

3

3D multi-scale coupling

Contents

In this chapter, the multi-scale coupling computation procedure is extended to a 3D setting. In the first section, the multi-scale formulation is presented for a tetrahedral macro-scale element, and then extended to take into account the localized failure for both the tetrahedral and hexahedral macro-scale element. In the second section, a plasticity model of Timoshenko beam with localized failure, which is used on the micro-scale, is presented. Finally, numerical computations are performed and presented in the last section, showing that the multi-scale computation procedure can produce the same quality results as the mono-scale computations.

3.1 Multi-scale formulation

For the multi-scale computation procedure in a 3D setting, a macro-scale isoparametric tetrahedral and hexahedral elements are developed. Most of the equations from the formulation described in Chapter 2 are still valid, and here will be listed only the ones that are different, or the ones that are crucial for the understanding of the procedure. In some cases, the dimensions of tensors are listed under the symbols for more clarity. Adding the additional degree of freedom also increases the complexity of the implementation into a software code, including the manipulation of the data that has to be exchanged between the scales in each iteration and time step.

The standard finite element shape functions for the 3D isoparametric tetrahedral element (e.g. (Ibrahimbegovic, 2009)) are used at the macro-scale

$$\begin{aligned} N_1^M(\xi,\eta,\zeta) &= 1-\xi-\eta-\zeta \\ N_2^M(\xi,\eta,\zeta) &= \xi \\ N_3^M(\xi,\eta,\zeta) &= \eta \\ N_4^M(\xi,\eta,\zeta) &= \zeta \end{aligned} \tag{3.1}$$

and the standard finite element shape functions for the 3D isoparametric hexahedral element

$$N_a^M(\xi,\eta,\zeta) = \frac{1}{8}(1+\xi_a\xi)(1+\eta_a\eta)(1+\zeta_a\zeta), \quad a = 1..8 \tag{3.2}$$

where ξ_a, η_a and ζ_a are the coordinates of node a, and can take values -1 or 1.

At the micro-scale, a Timoshenko beam element is used, but any other element that has at least three translational degrees of freedom can be used for the formulation to be valid. The Timoshenko beam element with implemented plasticity model is used and described in detail in Section 3.3.

The displacement field approximations at the micro and macro-scale, and localized Lagrange multipliers are computed with standard finite element approximations (2.7), which remain the same and are defined as

$$\begin{aligned} \mathbf{u}^m_{n+1}\Big|_{\Omega^{m,e}}(\mathbf{x}^m) &= \sum_{a=1}^{n^m_{el}} \mathbf{N}^{m,e}_a(\mathbf{x}^m)\mathbf{d}^m_{a,n+1} \\ \mathbf{u}^M_{n+1}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm,E}} \mathbf{N}^{M,E}_a(\mathbf{x}^m)\mathbf{d}^{M,E}_{a,n+1} \\ \boldsymbol{\lambda}_{n+1}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm,E}} \mathbf{P}^{M,E}_a(\mathbf{x}^m)\boldsymbol{\beta}_{a,n+1} \end{aligned} \tag{3.3}$$

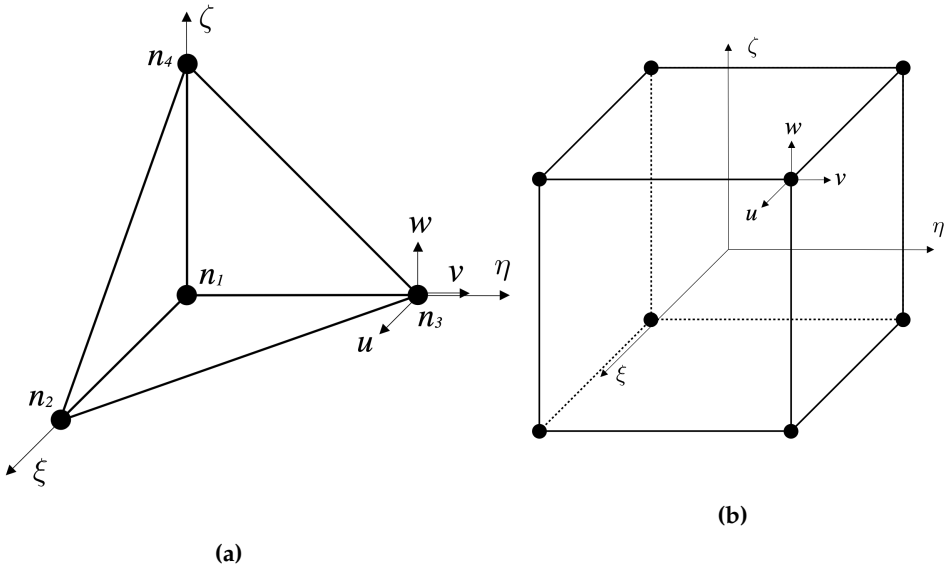


Figure 3.1: (a) 3D isoparametric tetrahedral element; (b) 3D isoparametric hexahedral element

As explained in Section 2.1, the local Lagrange multipliers are introduced, which enforce the condition that the micro-scale interface nodal displacements are calculated as a linear interpolation of the macro-scale displacement nodal values. This gives us the micro-scale nodal displacement field on the interface

$$\bar{\mathbf{d}}^{m}_{n+1}\Big|_{\Gamma^{M,E}} = \mathbf{T}^{E}\mathbf{d}^{M,E}_{n+1} \tag{3.4}$$

By following the same procedure as in 2D, we arrive at the standard finite element system of equations for computing the increment of the displacement field on the micro-scale, as in (4.27)

$$\begin{bmatrix} \underset{(n_\Gamma \times n_\Gamma)}{\bar{\bar{\mathbf{K}}}} & \underset{(n_\Gamma \times n_f)}{\bar{\mathbf{K}}^T} \\ \underset{(n_f \times n_\Gamma)}{\bar{\mathbf{K}}} & \underset{(n_f \times n_f)}{\mathbf{K}} \end{bmatrix} \begin{bmatrix} \underset{(n_\Gamma \times 1)}{\Delta\bar{\mathbf{d}}} \\ \underset{(n_f \times 1)}{\Delta\mathbf{d}} \end{bmatrix} = - \begin{bmatrix} \underset{(n_\Gamma \times 1)}{\bar{\mathbf{r}}} \\ \underset{(n_f \times 1)}{\mathbf{r}} \end{bmatrix} \tag{3.5}$$

Here, the dimensions of the submatrices are written below each element. The value n_Γ is the number of interface micro-scale nodes multiplied by the number of the degrees of freedom in every node, and the value n_f is the number of free micro-scale nodes multiplied by the number of the degrees of freedom in every node.

The final values of the condensed stiffness matrix and residual at the macro-scale are given in (2.17) and listed here with the value of N standing for the number of macro-scale nodes for one element multiplied by the number of

degrees of freedom in every node

$$\begin{aligned} \underset{(N\times N)}{\mathbf{K}^{M,E}_{n+1}} &= \underset{(N\times n_\Gamma)}{\mathbf{T}^{E,T}}\ \underset{(n_\Gamma\times n_\Gamma)}{\widetilde{\mathbf{K}}^{m}_{n+1}}\ \underset{(n_\Gamma\times N)}{\mathbf{T}^{E}} \\ \underset{(N\times 1)}{\mathbf{r}^{M,E}_{n+1}} &= \underset{(N\times n_\Gamma)}{\mathbf{T}^{E,T}}\ \underset{(n_\Gamma\times 1)}{\widetilde{\mathbf{r}}^{m}_{n+1}} \end{aligned} \tag{3.6}$$

In the case of the tetrahedral element with three degrees of freedom in every node the value of N, which defines the dimension of the matrix, equals 12. For the hexahedral element, N is equal to 24.

As in (2.18), the standard finite element system of equations is solved

$$\mathbf{K}^{M}_{n+1}\Delta\mathbf{d}^{M}_{n+1} = -\mathbf{r}^{M}_{n+1} \tag{3.7}$$

after which the corresponding values of the macro-scale displacement are updated with

$$\mathbf{d}^{M(i+1)}_{n+1} = \mathbf{d}^{M(i)}_{n+1} + \Delta\mathbf{d}^{M(i)}_{n+1} \tag{3.8}$$

Using this formulation, a multi-scale element is not able to take into account the localized failure happening on the micro-scale. The improvement of the multi-scale formulation with added embedded discontinuity is described in the following section.

3.2 Multi-scale formulation for localized failure

The embedded discontinuity finite element method is implemented on the macro-scale tetrahedral and hexahedral element. In order to allow the localized failure representation on the macro-scale, the incompatible mode shape function is introduced, as it has been already shown for a 2D setting in Section 2.2. The discontinuity is positioned inside the macro-scale element with n and m representing the normal and the tangential vector at the discontinuity. In the scope of this work, only the crack opening in mode I is taken into account, so only the normal direction vector at the discontinuity will have a non-zero value. The element domain is divided into two sub-domains, Ω^{e-} and Ω^{e+}, as shown in Figure 3.2a and 3.2b. The incompatible mode function for a macro-scale element in 3D can be written as

$$M(\xi,\eta,\zeta) = H_{\Gamma_s}(\xi,\eta,\zeta) - \sum_{b\in\Omega^{e+}} N^{M}_{b}(\xi,\eta,\zeta) \tag{3.9}$$

where H_{Γ_s} is the Heaviside step function defined as

$$H_{\Gamma_s} = \begin{cases} 0, & (\xi,\eta,\zeta)\in\Omega^{e-} \\ 1, & (\xi,\eta,\zeta)\in\Omega^{e+} \end{cases} \tag{3.10}$$

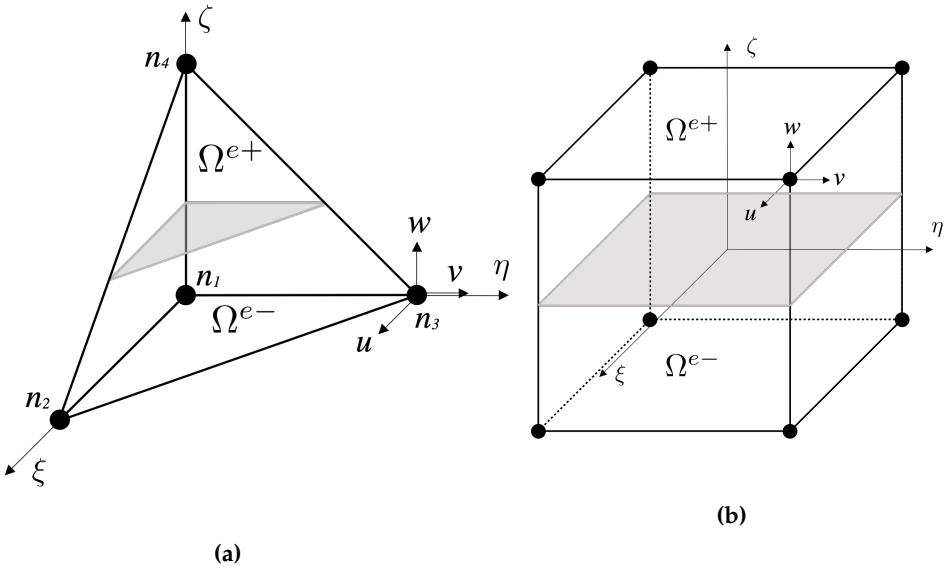


Figure 3.2: (a) 3D isoparametric tetrahedral element with discontinuity; (b) 3D isoparametric hexahedral element with discontinuity

When the incompatible mode function M and displacement jump $\boldsymbol{\alpha}$ are introduced, the macro-scale displacement field from (3.3) can be then rewritten as in (2.21)

$$\mathbf{u}_{n+1}^{M}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) = \sum_{a\in\Gamma^{Mm,E}} \mathbf{N}_a^{M,E}(\mathbf{x}^m)\mathbf{d}_{a,n+1}^{M,E} + \mathbf{M}^{M,E}\boldsymbol{\alpha}_{n+1}^{M,E} \tag{3.11}$$

The micro-scale nodal displacement field on the interface is the same as for 2D setting

$$\bar{\mathbf{d}}_{n+1}^{m}\Big|_{\Gamma^{M,E}} = \mathbf{T}^E\mathbf{d}_{n+1}^{M,E} + \mathbf{S}^E\boldsymbol{\alpha}_{n+1}^{M,E} \tag{3.12}$$

The final system of equations on the macro-scale can be defined as

$$\begin{bmatrix} \underset{(N\times N)}{\mathbf{K}^M} & \underset{(N\times 3)}{\mathbf{F}^M} \\ \underset{(3\times N)}{\mathbf{F}^{M,T}} & \underset{(3\times 3)}{\mathbf{H}^M} \end{bmatrix} \begin{bmatrix} \underset{(N\times 1)}{\Delta\mathbf{d}^M} \\ \underset{(3\times 1)}{\Delta\boldsymbol{\alpha}^M} \end{bmatrix} = - \begin{bmatrix} \underset{(N\times 1)}{\mathbf{r}^M} \\ \underset{(3\times 1)}{\mathbf{h}^M} \end{bmatrix} \tag{3.13}$$

where the value N is equal to the number of macro-scale nodes for one element multiplied by the number of degrees of freedom in every node on the macro-scale.

To construct the macro-scale stiffness matrix needed for solving the macro-scale

system of equations, submatrices $\mathbf{K}^M$, $\mathbf{F}^M$ and $\mathbf{H}^M$ are computed as

$$\begin{aligned}
\underset{(N\times N)}{\mathbf{K}^{M,E}_{n+1}} &= \underset{(N\times n_\Gamma)}{\mathbf{T}^{E,T}}\ \underset{(n_\Gamma\times n_\Gamma)}{\widetilde{\mathbf{K}}^{m}_{n+1}}\ \underset{(n_\Gamma\times N)}{\mathbf{T}^{E}} \\
\underset{(N\times 3)}{\mathbf{F}^{M,E}_{n+1}} &= \underset{(N\times n_\Gamma)}{\mathbf{T}^{E,T}}\ \underset{(n_\Gamma\times n_\Gamma)}{\widetilde{\mathbf{K}}^{m}_{n+1}}\ \underset{(n_\Gamma\times 3)}{\mathbf{S}^{E}} \\
\underset{(3\times 3)}{\mathbf{H}^{M,E}_{n+1}} &= \underset{(3\times n_\Gamma)}{\mathbf{S}^{E,T}}\ \underset{(n_\Gamma\times n_\Gamma)}{\widetilde{\mathbf{K}}^{m}_{n+1}}\ \underset{(n_\Gamma\times 3)}{\mathbf{S}^{E}}
\end{aligned} \tag{3.14}$$

where the value n_Γ is the number of interface micro-scale nodes multiplied by the number of the degrees of freedom in every node, and the value n_f is the number of free micro-scale nodes multiplied by the number of the degrees of freedom in every node.

In the same way, the macro-scale residuals are computed using the values of micro-scale residuals and transformation matrices $\mathbf{T}$ and $\mathbf{S}$

$$\begin{aligned}
\underset{(N\times 1)}{\mathbf{r}^{M,E}_{n+1}} &= \underset{(N\times n_\Gamma)}{\mathbf{T}^{E,T}}\ \underset{(n_\Gamma\times 1)}{\widetilde{\mathbf{r}}^{m}_{n+1}} \\
\underset{(3\times 1)}{\mathbf{h}^{M,E}_{n+1}} &= \underset{(3\times n_\Gamma)}{\mathbf{S}^{E,T}}\ \underset{(n_\Gamma\times 1)}{\widetilde{\mathbf{r}}^{m}_{n+1}}
\end{aligned} \tag{3.15}$$

The final equations to be solved are the same as the ones described in (2.30)-(2.33) and will be given here for completeness of the presented 3 formulation. The local equation from which the macro-scale displacement jump is computed is the following

$$\mathbf{H}^{M,E(i,j)}_{n+1}\Delta\boldsymbol{\alpha}^{M,E(i,j)}_{n+1} = \mathbf{h}^{M,E(i,j)}_{n+1} \tag{3.16}$$

with the corresponding update

$$\boldsymbol{\alpha}^{M,E(i,j+1)}_{n+1} = \boldsymbol{\alpha}^{M,E(i,j)}_{n+1} + \Delta\boldsymbol{\alpha}^{M,E(i,j)}_{n+1} \tag{3.17}$$

The detailed flow-chart of the proposed multi-scale operator split solution procedure is the same as for 2D setting and is shown in Algorithm 1 in Section 2.2.

The global equation from which the values of the macro-scale displacements are computed is

$$[\mathbf{K}^{M(i)}_{n+1} - \mathbf{F}^{M(i)}_{n+1}(\mathbf{H}^{M(i)}_{n+1})^{-1}\mathbf{F}^{M(i)^T}_{n+1}]\Delta\mathbf{d}^{M(i)}_{n+1} = -\mathbf{r}^{M(i)}_{n+1} \tag{3.18}$$

with the corresponding update

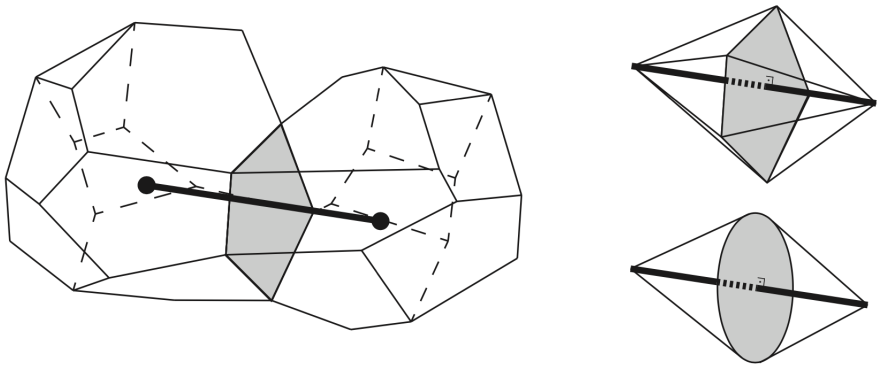
$$\mathbf{d}^{M(i+1)}_{n+1} = \mathbf{d}^{M(i)}_{n+1} + \Delta\mathbf{d}^{M(i)}_{n+1} \tag{3.19}$$

The reason why both macro-scale tetrahedral and hexahedral elements are developed is that they both have their advantages and disadvantages. The discontinuity position and orientation is calculated in a different way. Calculations for tetrahedral elements are more simple as they have only one Gauss point compared to four Gauss points for hexahedral element. On the other side, for the hexahedral element, it is easier to detect interface nodes between the scales, so it is a better option for obtaining results in the elasticity and hardening phase. This specificities together with the discussion of results for both types of macro-scale elements will be shown in Section 3.4.

3.3 Micro-scale Timoshenko beam plasticity model

On the micro-scale, the Timoshenko beam finite element is used, developed in the works of (Hadzalic, 2019), (Karavelic et al., 2017) and (Nikolic, 2015). The model is described for a better understanding and interpretation of the numerical results for the multi-scale solution procedure, although the theoretical formulation is not improved in any way in the scope of this thesis. Also, for the numerical examples, the FEAP code from (Hadzalic, 2019) is used for the micro-scale. The plasticity model is able to represent localized failure using the embedded strong discontinuity (Karavelic et al., 2017). A discrete beam lattice model is used, which is based on the Voronoi cell representation of the domain with inelastic Timoshenko beam finite elements acting as cohesive links.

To define a finite element mesh for a micro-scale problem, the following procedure is used, as described in (Hadzalic, 2019). A 3D mesh is constructed using the duality property between the Voronoi cell representation and Delaunay tetrahedralization of the domain. First, the domain of the problem is divided into polyhedral regions. Then, performing the Delaunay tetrahedralization, each of the centers of the adjacent Voronoi cells are connected. In this way, a dual mesh of tetrahedra is constructed. Along each edge of the tetrahedral element, a cohesive link is placed that is modelled as a one-dimensional Timoshenko beam finite element. The beam is perpendicular to the polygon shared between the two Voronoi cells, and the polygon divides the beam in two equal parts. Due to this, it is more practical to simulate the displacement jump in the middle of the beam element as the crack is happening only at the intersection of the Voronoi cells and the cell does not need to be divided. To calculate the cross-section of a beam, a polygon area is used. To simplify the computation of the area of the polygon, it is approximated with an equivalent circular cross-section (Moreno-Navarro, 2019; Hadzalic, 2019).



(a) (b)

Figure 3.3: (a) Two adjacent Voronoi cells connected by the cohesive link; (b) Timoshenko beam element and the circular cross-section approximation (Moreno-Navarro, 2019)

It can be noted that there is a different number of degrees of freedom on the micro and the macro-scale. The Timoshenko beam element has six degrees of freedom (three displacements and three rotations), and the macro-scale elements have just three degrees of freedom in each node. For the multi-scale formulation, only the translational micro-scale degrees of freedom are taken into account, while the rotations are ignored. The numerical examples and the mesh will have to be constructed later with having this in mind, to limit the significance that rotations have on the computations and the final result.

The behavior of cohesive links is modelled with inelastic Timoshenko beam finite elements with embedded strong discontinuity, as shown in (Nikolic and Ibrahimbegovic, 2015) and (Nikolic, 2015). They are capable of representing the crack in mode I, mode II and mode III. Mode I relates to crack opening, mode II relates to in-plane crack sliding, and mode III relates to the out-of-plane shear sliding (Hadzalic, 2019). The Timoshenko beam finite element has one Gauss integration point which is placed in the middle of the element.

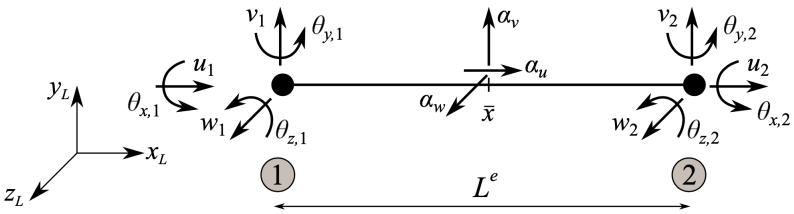


Figure 3.4: Timoshenko beam finite element with 6 degrees of freedom in every node (Hadzalic, 2019)

The finite element interpolation of the total displacement field can be written as the sum of the regular and the incompatible part

$$\mathbf{u}(x) = \sum_{a=1}^{2} N_a(x)\mathbf{d}_a + M(x)\boldsymbol{\alpha} \tag{3.20}$$

where N_a is the standard linear shape function, $\mathbf{d}_a$ is the displacement vector containing displacements in all three directions and three rotations of cross section around local axes, M is the incompatible mode function for the Timoshenko beam, and $\boldsymbol{\alpha}$ is the vector of displacement jumps at the discontinuity in all three directions.

The shape functions for the Timoshenko beam are defined as

$$\begin{aligned} N_1(x) &= 1 - \tfrac{x}{L^e} \\ N_2(x) &= \tfrac{x}{L^e} \end{aligned} \tag{3.21}$$

where L^e is the length of the beam. The vector of displacements is defined as

$$\mathbf{d}_a = [u_a \; v_a \; w_a \; \varphi_a \; \psi_a \; \theta_a]^T \tag{3.22}$$

where u_a, v_a and w_a are the displacements of the node a along the local axes x, y and z, and φ_a, ψ_a and θ_a are the rotations of the node a around the local axes x, y and z, respectively.

The incompatible mode function is defined as

$$M(x) = H_{\bar{x}}(x) - N_2(x) \tag{3.23}$$

The value of the Heaviside function depends on the position relative to the discontinuity

$$H_{\bar{x}}(x) = \begin{cases} 0, & x \leq \bar{x} \\ 1, & x > \bar{x} \end{cases} \tag{3.24}$$

where $\bar{x}$ is the position of the discontinuity, which is in this case placed in the middle of the element. The final value of the incompatible mode function can then be written as

$$M(x) = \begin{cases} -\frac{x}{L^e}, & x \leq \bar{x} \\ 1 - \frac{x}{L^e}, & x > \bar{x} \end{cases} \tag{3.25}$$

The vector of the displacement jumps is defined as

$$\boldsymbol{\alpha} = [\alpha_u \; \alpha_v \; \alpha_w \; 0 \; 0 \; 0]^T \tag{3.26}$$

where α_u, α_v and α_w are the displacement jumps in axial, in-plane transverse and out-of-plane transverse direction (Hadzalic, 2019). The rotations are not concerned with the displacement jumps.

For a geometrically linear Timoshenko beam, the strain field is defined as

$$\boldsymbol{\varepsilon} = [\varepsilon_{xx}\ \gamma_{xy}\ \gamma_{xz}\ \kappa_x\ \kappa_y\ \kappa_z]^T \tag{3.27}$$

where ε_{xx} is the axial strain, γ_{xy} and γ_{xz} are the shear strains, and κ_x, κ_y and κ_z are the curvatures. They are defined as in (Karavelic et al., 2017)

$$\begin{aligned} \varepsilon_{xx} &= \frac{du}{dx} & \gamma_{xy} &= \frac{dv}{dx} - \theta_z & \gamma_{xz} &= \frac{dw}{dx} + \theta_y \\ \kappa_x &= \frac{d\varphi}{dx} & \kappa_y &= \frac{d\psi}{dx} & \kappa_z &= \frac{d\theta}{dx} \end{aligned} \tag{3.28}$$

Then, the interpolated enhanced strain field can be written as

$$\boldsymbol{\varepsilon} = \sum_{a=1}^{2} \mathbf{B}_a \mathbf{d}_a + \mathbf{G}\boldsymbol{\alpha} \tag{3.29}$$

where the matrix $\mathbf{B}_a$ is defined as

$$\mathbf{B}_a = \begin{bmatrix} B_a & 0 & 0 & 0 & 0 & 0 \\ 0 & B_a & 0 & 0 & 0 & -N_a \\ 0 & 0 & B_a & 0 & N_a & 0 \\ 0 & 0 & 0 & B_a & 0 & 0 \\ 0 & 0 & 0 & 0 & B_a & 0 \\ 0 & 0 & 0 & 0 & 0 & B_a \end{bmatrix} \tag{3.30}$$

The values B_a are the derivatives of the linear shape functions of element node a and can be written as

$$\begin{aligned} B_1(x) &= \tfrac{dN_1}{dx} = -\tfrac{1}{L^e} \\ B_2(x) &= \tfrac{dN_2}{dx} = \tfrac{1}{L^e} \end{aligned} \tag{3.31}$$

The matrix $\mathbf{G}$ is defined as

$$\mathbf{G} = \begin{bmatrix} G & 0 & 0 & 0 & 0 & 0 \\ 0 & G & 0 & 0 & 0 & 0 \\ 0 & 0 & G & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.32}$$

where G is the derivative of the incompatible mode function that can be split into a regular and a singular part

$$G(x) = \bar{G}(x) + \bar{\bar{G}}(x), \quad \bar{G}(x) = -\frac{1}{L^e}, \quad \bar{\bar{G}}(x) = \delta_{\bar{x}}(x) \tag{3.33}$$

where $\delta_{\bar{x}}$ is the derivative of the Heaviside function at the discontinuity

$$\delta_{\bar{x}}(x) = \begin{cases} 0, & x \in [\,0, \bar{x}\rangle \cup \langle \bar{x}, L^e\,] \\ \infty, & x = \bar{x} \end{cases} \tag{3.34}$$

The patch test condition needs to be fulfilled by enforcing the orthogonality between the enhanced strain and constant stress (Karavelic et al., 2017), so the matrix $\mathbf{G}_{\mathbf{v}}(x)$ is introduced and defined as

$$\mathbf{G}_{\mathbf{v}}(x) = \mathbf{G}(x) - \frac{1}{L^e}\int_0^{L^e} \mathbf{G}(x)dx \tag{3.35}$$

Since the Timoshenko beam has only one Gauss integration point it can be proved that $\mathbf{G}_{\mathbf{v}} = \mathbf{G}$ (Karavelic et al., 2017).

The system of equations to be solved can be written as

$$\mathbb{A}[\mathbf{f}^e_{int} - \mathbf{f}^e_{ext}] = 0 \tag{3.36}$$

$$\mathbf{h}^e = 0 \tag{3.37}$$

where $\mathbf{f}^e_{int}$ is the internal force vector, $\mathbf{f}^e_{ext}$ is the external force vector, and $\mathbf{h}^e$ is the residual at the discontinuity. They can be defined as

$$\mathbf{f}^e_{int} = \int_0^{L^e} \mathbf{B}^T \boldsymbol{\sigma} dx \tag{3.38}$$

$$\mathbf{h}^e = \int_0^{L^e} \bar{\mathbf{G}}^T \boldsymbol{\sigma} dx + \mathbf{t} \tag{3.39}$$

where $\boldsymbol{\sigma}$ is the stress in the element, and $\mathbf{t}$ is the traction vector at the discontinuity $\bar{x}$. In order to have a smooth stress field, the condition $\mathbf{h}^e = 0$ needs to be enforced, which leads to the traction vector being defined as (Nikolic, 2015)

$$\mathbf{t} = -\int_0^{L^e} \bar{\mathbf{G}} \boldsymbol{\sigma} dx \tag{3.40}$$

To solve this problem, the system (3.36) is linearized and can be then written as

$$\begin{bmatrix} \mathbf{K} & \mathbf{F} \\ \mathbf{F}_v + \mathbf{K}_d & \mathbf{H} + \mathbf{K}_\alpha \end{bmatrix} \begin{bmatrix} \Delta \mathbf{d}_{n+1} \\ \Delta \boldsymbol{\alpha}_{n+1} \end{bmatrix} = - \begin{bmatrix} \mathbf{f}_{int,n+1} - \mathbf{f}_{ext,n+1} \\ \mathbf{h} \end{bmatrix} \tag{3.41}$$

where the tangent stiffness matrices are defined as

$$\begin{aligned} \mathbf{K}^{e,(i)}_{n+1} &= \int_0^{L^e} \mathbf{B}^{e,T} \mathbf{C}^{(i)}_{n+1} \mathbf{B}^e dx \\ \mathbf{F}^{e,(i)}_{n+1} &= \int_0^{L^e} \mathbf{B}^{e,T} \mathbf{C}^{(i)}_{n+1} \bar{\mathbf{G}}^e dx \\ \mathbf{F}^{e,(i)}_{v,n+1} &= \int_0^{L^e} \bar{\mathbf{G}}^{e,T} \mathbf{C}^{(i)}_{n+1} \mathbf{B}^e dx \\ \mathbf{H}^{e,(i)}_{n+1} &= \int_0^{L^e} \bar{\mathbf{G}}^{e,T} \mathbf{C}^{(i)}_{n+1} \bar{\mathbf{G}}^e dx \end{aligned} \tag{3.42}$$

where $\mathbf{C}_{n+1}^{(i)}$ is the elasto-plastic modulus for the Timoshenko beam. The values of the matrices $\mathbf{K}_d$ and $\mathbf{K}_\alpha$ depend on the fact whether the current step in softening is elastic or plastic. For an elastic step $\mathbf{K}_\alpha = 0$, and for a plastic step $\mathbf{K_d} = 0$. The procedure and the calculated values for $\mathbf{K_d}$ and $\mathbf{K}_\alpha$ can be seen in (Hadzalic, 2019).

After performing the static condensation on the set of equations (3.41), the displacement jump can be computed as

$$\Delta\boldsymbol{\alpha}_{n+1} = -(\mathbf{H} + \mathbf{K}_\alpha)^{-1}(\mathbf{F}_v + \mathbf{K}_d)\Delta\mathbf{d}_{n+1} \tag{3.43}$$

To compute the increment of the displacement vector, a value of displacement jump is introduced back into (3.41) so the final equation to solve can be written as

$$[\mathbf{K} - \mathbf{F}(\mathbf{H} + \mathbf{K}_\alpha)^{-1}(\mathbf{F}_v + \mathbf{K}_d)]\Delta\mathbf{d}_{n+1} = -(\mathbf{f}_{int,n+1} - \mathbf{f}_{ext,n+1}) \tag{3.44}$$

The general finite element procedure for the plasticity model is described here, while the details can be found in (Hadzalic, 2019). The computations are divided into a bulk computation and a computation at the discontinuity.

The bulk computation starts with an elastic step in both axial and transverse direction that is described with the elasto-viscoplastic constitutive model with implemented linear hardening (Ibrahimbegovic, 2009) and Fredrick-Armstrong nonlinear kinematic hardening law (Armstrong and Frederick, 1966). The behavior of the element in bending and torsion is purely linear elastic (Karavelic et al., 2017).

When the value of the stress becomes greater than the value of the ultimate stress, the computation at the discontinuity starts and the softening is described with an exponential function. The displacement jump is then activated and computed in each step. In the softening phase, the plastic deformation remains localized at the discontinuity and the bulk of the material is unloading elastically (Hadzalic, 2019).

The final solution is obtained using the operator split solution procedure. The computation is divided into a local and a global phase. In the local phase, evolution equations of internal variables defined locally at each Gauss point are computed, using implicit backward Euler time integration scheme. In the global phase, after the local phase is done, the equilibrium equations are solved, and the values of the nodal displacements are computed using the incremental and iterative Newton-Raphson procedure (Hadzalic, 2019; Kozar et al., 2018).

3.4 Numerical examples

Several numerical examples are designed to validate the multi-scale theoretical formulation in a 3D setting. At first, a simple tension test with a cube shaped

specimen is chosen, where the displacement at the top face is imposed as shown in Figure 3.5. The multi-scale results are compared to the mono-scale solution. The goal is to prove that the 3D multi-scale method can produce the same quality results as the mono-scale solution. Timoshenko beam elements described in Section 3.3 are used both for the mono-scale example and for the micro-scale of the multi-scale example. First, a multi-scale numerical example with the macro-scale tetrahedral element is shown, and after that an example with the macro-scale hexahedral element.

The geometry of the specimen is a cube with sides of 5 cm. The top face has an imposed displacement $\bar{u}$ of 0.06 cm. The planes $x = 0$, $y = 0$ and $z = 0$, have fixed displacements in the same direction to simulate symmetry boundary conditions. All rotation degrees of freedom of the beams are set free.

The following material parameters for the Timoshenko beam element are chosen: the Young's modulus $E = 210\,000$ MPa, the Poisson's ratio $\nu = 0.3$, the axial and shear yield stresses $\bar{\sigma}_x = 200$ MPa, $\bar{\sigma}_y = 50$ MPa and $\bar{\sigma}_z = 50$ MPa, the axial and shear hardening moduli $\bar{K}_x = 0$ MPa, $\bar{K}_y = 0$ MPa and $\bar{K}_z = 0$ MPa. For the weakened elements, the axial and shear ultimate stresses are $\bar{\bar{\sigma}}_x = 300$ MPa, $\bar{\bar{\sigma}}_y = 60$ MPa and $\bar{\bar{\sigma}}_z = 60$ MPa, and the axial and shear fracture energies are $G_{f,x} = 5$ N/mm, $G_{f,y} = 5$ N/mm and $G_{f,z} = 5$ N/mm. Ultimate stresses and fracture energies for non-weakened elements are set to high values that are never reached, so they never enter into the softening phase.

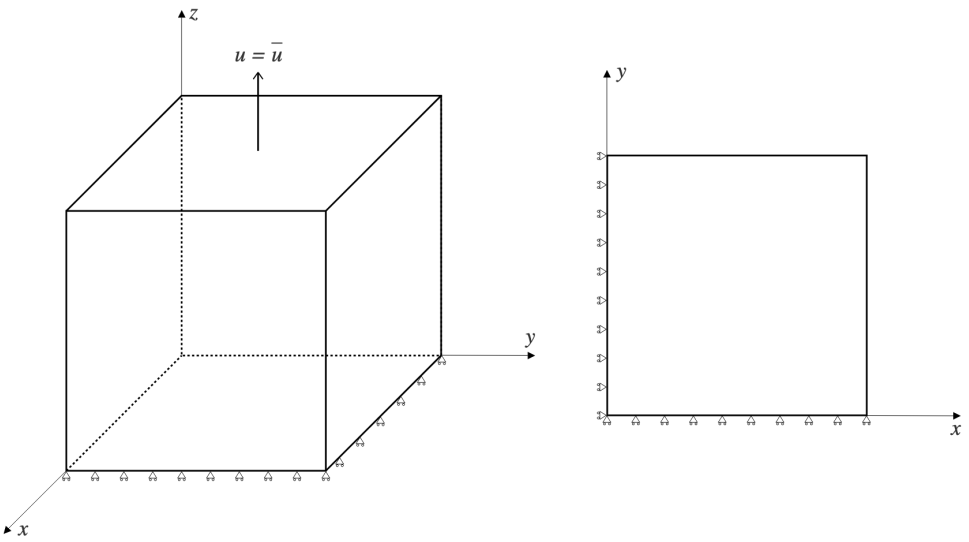


Figure 3.5: Simple tension test - boundary conditions

3.4.1 Numerical example for macro-scale tetrahedron

For the proposed geometry, six tetrahedral elements are used to define the macro-scale mesh as shown in Figure 3.6a. A cube shaped specimen can be divided into a minimum of six congruent tetrahedra (each tetrahedron can be obtained from the other by translation, rotation or reflection). For each of the tetrahedral elements, a micro-scale mesh is defined as shown in Figure 3.6b. It is composed of Timoshenko beam elements formed so they can fit inside a tetrahedral element. Since the tetrahedra are congruent, theoretically one micro-scale mesh can be defined and used for each of the macro-scale tetrahedral elements. In this case, microFEAP code should be changed in a way so that the micro-scale mesh conform to the interface (rotation and translation of the mesh). In this example, for the sake of simplicity, six different micro-scale meshes are defined.

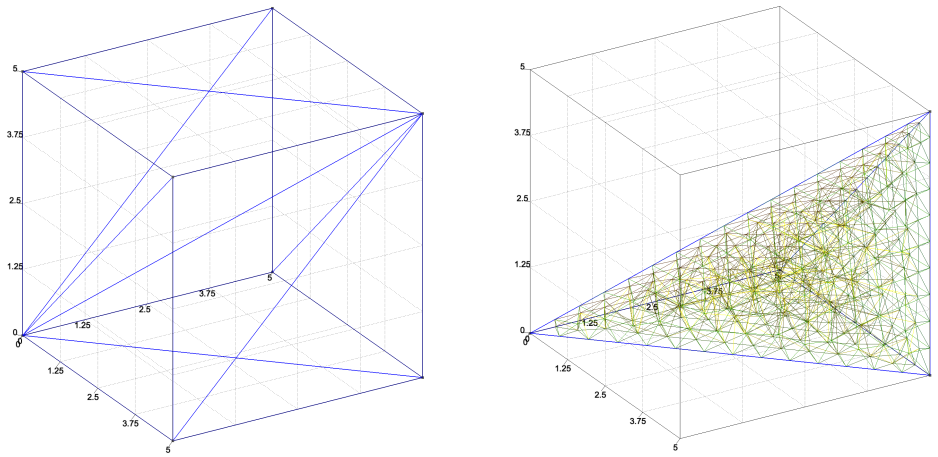


Figure 3.6: (a) Macro-scale mesh consisting of six tetrahedral elements; (b) micro-scale mesh for one tetrahedral element consisting of Timoshenko beam elements

A micro-scale mesh composed of Timoshenko beam elements is created using the procedure explained in Section 3.3. Weakened elements are positioned around the middle of each macro-scale tetrahedral element parallel to the cube's facet with imposed displacement. Domain division into polyhedral regions and Delauney tetrahedralization are done in Matlab using the existing code developed as part of the work (Karavelic et al., 2017). An additional Matlab code is developed to calculate the cross-section of beam elements and arrange the mesh data into the FEAP input file format.

Finally, a mono-scale mesh is defined using Timoshenko beam elements, as shown in Figure 3.7a. Weakened elements are positioned around the middle of the specimen parallel to the cube's facet with imposed displacement. Mono-scale

and multi-scale meshes have approximately the same total number of beam elements, the same position of the weakened elements and the same dimensions, and therefore should behave in a similar way.

To be able to couple the two scales through the degrees of freedom, three micro-scale rotations are ignored when the scales communicate. Due to this, the micro-scale mesh has to be defined in such a way so that rotations do not have a significant effect on computations.

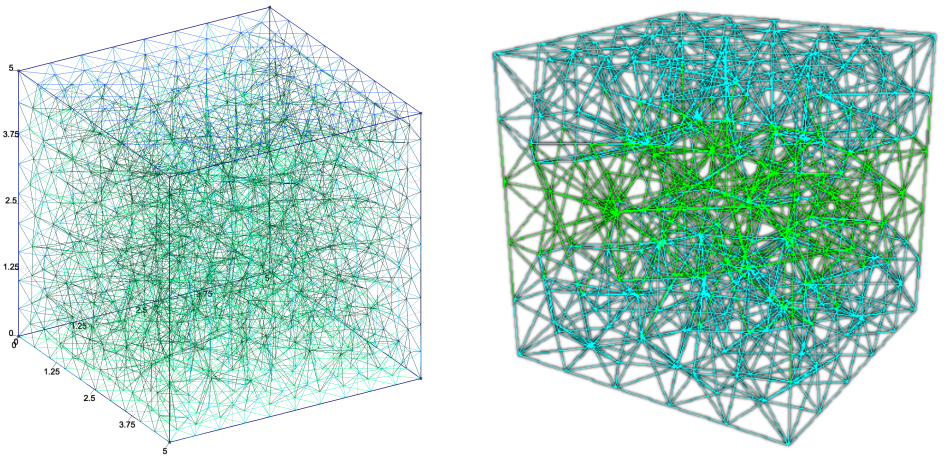


Figure 3.7: (a) Mono-scale mesh consisting of Timoshenko beam elements (b) Weakened elements in the middle of the specimen (shown in green)

Elastic response

When the material is in the elastic phase, the same response is obtained using the multi-scale and the mono-scale approach, as can be seen in the force-displacement diagram shown in Figure 3.8. In the multi-scale code, only the stiffness matrix $\mathbf{K}^M$ and the residual $\mathbf{r}^M$ are transfered from the micro to the macro-scale, and no localized failure is activated.

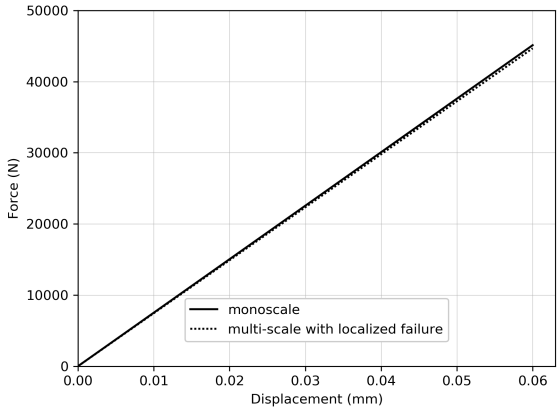


Figure 3.8: Force-displacement diagram for the elasticity phase obtained using the mono-scale and the multi-scale method with macro-scale tetrahedral element

Elasto-plasticity with hardening

When the micro-scale beam elements enter into hardening, a similar response is obtained for the mono and the multi-scale method, as shown on the force-displacement diagram in Figure 3.9. Some of the micro-scale beams can enter into the softening phase and crack, but the specimen has still not entered into softening globally. Localized failure is not activated in the macro-scale tetrahedral elements. In the multi-scale code, only the stiffness matrix $\mathbf{K}^M$ and the residual $\mathbf{r}^M$ are transfered from the micro to the macro-scale, but this time with modified values of stiffness matrix due to the micro-scale beams entering into the hardening and softening phase.

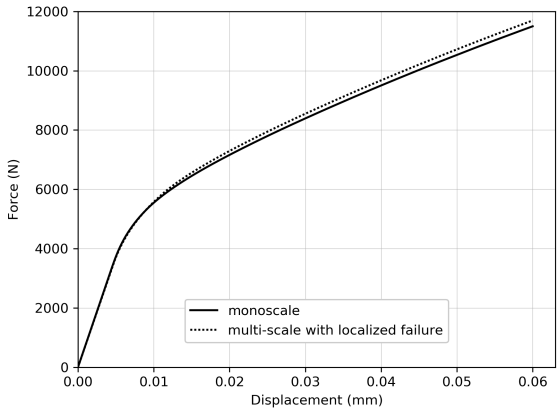


Figure 3.9: Force-displacement diagram for the hardening phase obtained using the mono-scale and the multi-scale method with macro-scale tetrahedral element

Elasto-plasticity with softening

In the softening phase, the correct results cannot be obtained with the multi-scale code, as shown on the force-displacement diagram in Figure 3.10 (dashed line). The current formulation is not sufficient to allow the macro-scale tetrahedral elements to enter into the softening phase when the crack is not perfectly parallel to the cube's facet with imposed displacement. To overcome this, it it necessary to add the functionality of crack opening for both mode I and mode II for the macro-scale element.

The crack opening is not parallel to the cube's facet with imposed displacement due to the geometry of the tetrahedra and the position of the crack in the macro-scale tetrahedral elements. The crack is positioned in the center of each tetrahedron, which is not the same as the position of the weakened elements in the micro-scale mesh. A micro-scale mesh could be additionally modified so that weakened elements fit perfectly with the macro-scale crack just to verify that the formulation is working for this case.

Instead of that, a macro-scale hexahedral element is developed, where that problem does not exist, and the results will be presented in the following section.

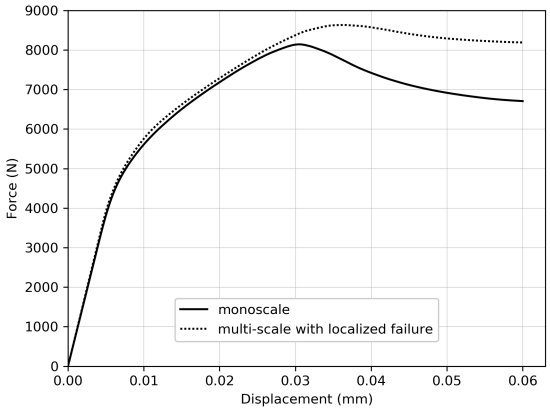


Figure 3.10: Force-displacement diagram for the softening phase obtained using the mono-scale and the multi-scale method with macro-scale tetrahedral element

3.4.2 Numerical example for macro-scale hexahedron

For the proposed geometry, a hexahedral element is used to define the macro-scale mesh as shown in Figure 3.11a. For this hexahedral element, a micro-scale mesh is defined as shown in Figure 3.11b. The micro-scale mesh is composed of Timoshenko beam elements formed so they can fit inside a hexahedral element.

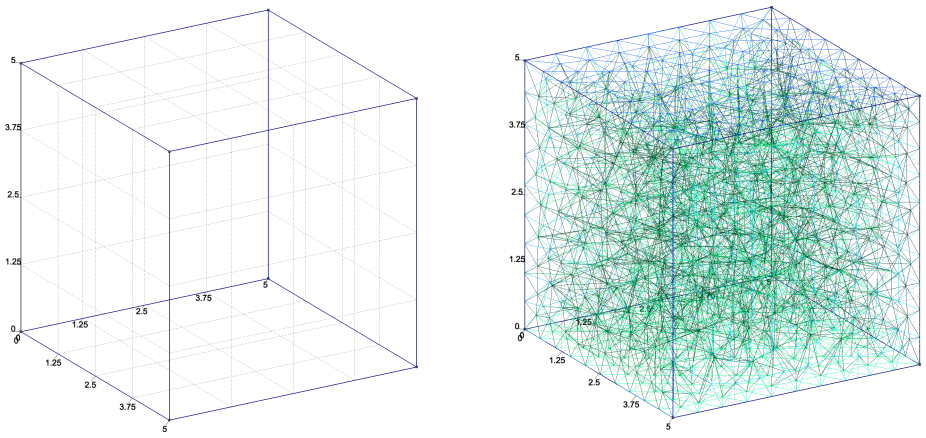


Figure 3.11: (a) Macro-scale mesh consisting of a hexahedral element; (b) micro-scale mesh for one hexahedral element consisting of Timoshenko beam elements

The micro-scale mesh of Timoshenko beams with weakened elements is created using the same procedure as in the previous section. The mono-scale and multi-scale meshes have the same total number of beam elements, the same position of the weakened elements and the same dimensions; therefore, they should behave in a similar way.

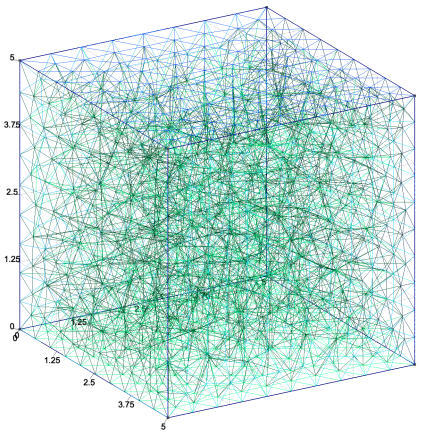


Figure 3.12: Mono-scale mesh consisting of Timoshenko beam elements

Elastic response

For the elastic phase, almost the same response is obtained for both mono-scale and multi-scale approaches, as it can be seen on the force-displacement diagram shown in Figure 3.13. There is a small difference between the obtained force values.

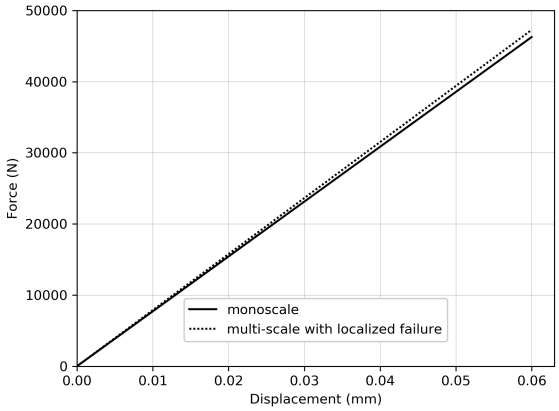


Figure 3.13: Force-displacement diagram for the elasticity phase obtained using the mono-scale and the multi-scale method with macro-scale hexahedral element

To test if the difference will get smaller as the number of elements increases, another micro-scale and mono-scale mesh are created, with an increased number of elements. Now, with a finer mesh, the difference between the obtained force values is smaller than with the coarse mesh, as it can be seen in Figure 3.14.

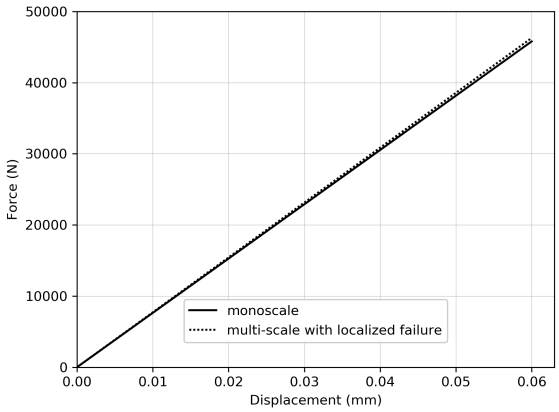


Figure 3.14: Force-displacement diagram for the elasticity phase obtained using the mono-scale and the multi-scale method with macro-scale hexahedral element for a finer mesh

A superposed force-displacement diagram for both coarse and fine mesh can be seen in Figure 3.15. It can be seen that the mono-scale method itself returns slightly different results for different meshes. To conclude, mono-scale and multi-scale methods are in a better agreement when a finer mesh is used.

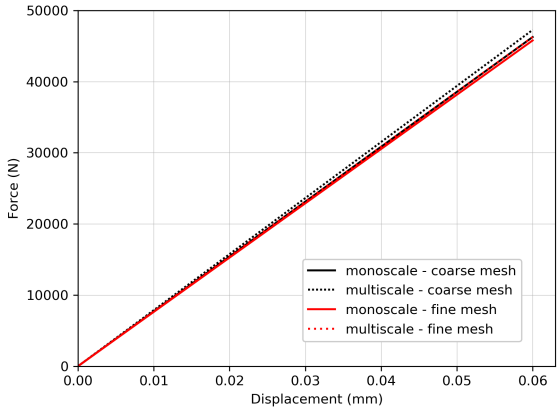


Figure 3.15: Force-displacement diagram for the elasticity phase obtained using the mono-scale and the multi-scale method with macro-scale hexahedral element - comparison of coarse and fine mesh

The multi-scale procedure has also been tested with a simple hexahedral elastic element implemented in FEAP on the micro-scale, and it produces the same results for the elastic phase, showing that the multi-scale procedure can work with different micro-scale elements.

Elasto-plasticity with hardening

For the hardening phase, both multi-scale and mono-scale methods are giving the same results, with the same small differences in obtained force values, as can be seen in Figure 3.16.

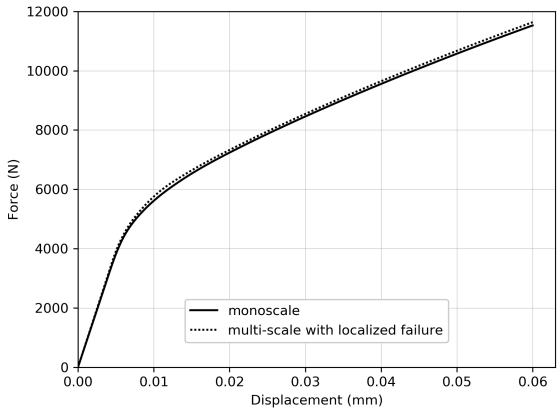


Figure 3.16: Force-displacement diagram for the hardening phase obtained using the mono-scale and the multi-scale method with macro-scale hexahedral element

Elasto-plasticity with softening

In the softening phase, when beams start to crack and the specimen is entering into the softening phase, the same behavior is captured with both mono-scale and multi-scale codes, as it can be seen in Figure 3.17. Due to the weakened beam element being aligned with the discontinuity position inside the macro-scale hexahedral elements, the position of the crack is the same on both micro and macro-scale. The crack opening is also parallel to the cube's face with imposed displacement. The macro-scale hexahedral element can enter into softening and compute the proper value of the displacement jump on the macro-scale.

This shows that with the hexahedral element we are capable of overcoming the limitations of the tetrahedra example, which allows us to obtain realistic results for all the phases of material behavior, including softening.

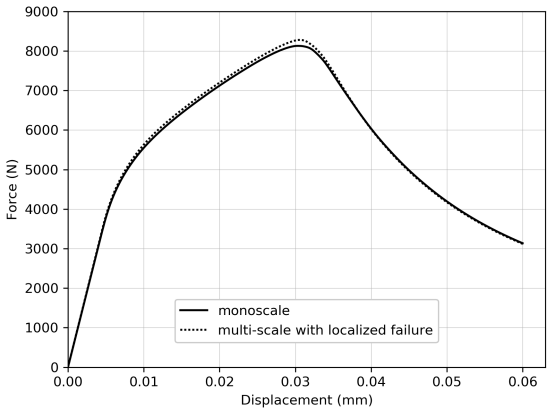


Figure 3.17: Force-displacement diagram for the softening phase obtained using the mono-scale and the multi-scale method with macro-scale hexahedral element

3.4.3 Validation examples of mesh objectivity

To prove that the proposed multi-scale method is mesh independent, it is tested on several more examples – tension and compression test with a different number of macro-scale and micro-scale elements.

3.4.3.1 Tension test

A simple tension test has the same geometry and boundary conditions as presented in Section 3.4.

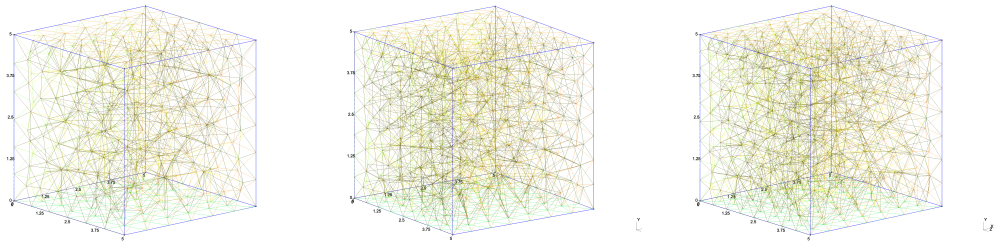
The following material parameters for the Timoshenko beam element are chosen: Young's modulus E = 210 000 MPa, Poisson's ratio ν = 0.3, axial and

shear yield stresses $\bar{\sigma}_x = 200$ MPa, $\bar{\sigma}_y = 50$ MPa and $\bar{\sigma}_z = 50$ MPa, axial and shear hardening moduli $\bar{K}_x = 0$ MPa, $\bar{K}_y = 0$ MPa and $\bar{K}_z = 0$ MPa.

As the example is tested only for the elasticity and hardening case, there are no weakened elements, so there is no need to define their material properties (i.e. ultimate stress and fracture energy).

Mono-scale computation

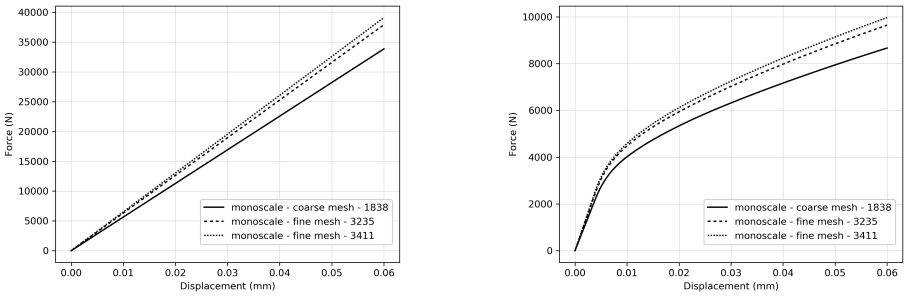
Three meshes with different number of elements are generated and used for the mono-scale computations. The mesh with the smallest (coarse mesh) and the mesh with the biggest number of elements (fine mesh) are used for comparison with the multi-scale computations. The mono-scale mesh is defined using Timoshenko beam elements, as shown in Figure 3.18.



(a) (b) (c)

Figure 3.18: Mono-scale mesh consisting of (a) 1838 Timoshenko beam elements - coarse mesh; (b) 3235 Timoshenko beam elements; (b) 3411 Timoshenko beam elements - fine mesh

It can be seen in Figure 3.19 that for the mono-scale examples, the force displacement diagram results are similar for elasticity and hardening phase. As the number of elements in the mesh increases, the results are approaching to the same value.



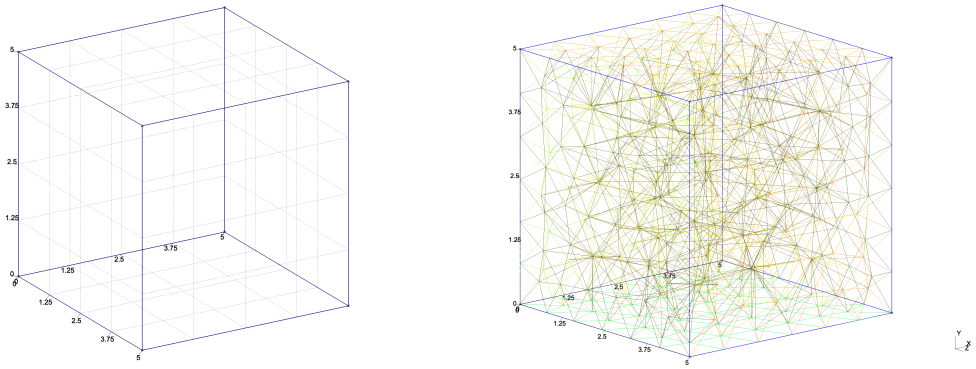
(a) (b)

Figure 3.19: Tension test: Force-displacement diagram obtained using the mono-scale method for three different meshes (a) elasticity; (b) hardening

Multi-scale computation

One macro-scale hexahedron - coarse micro-scale mesh

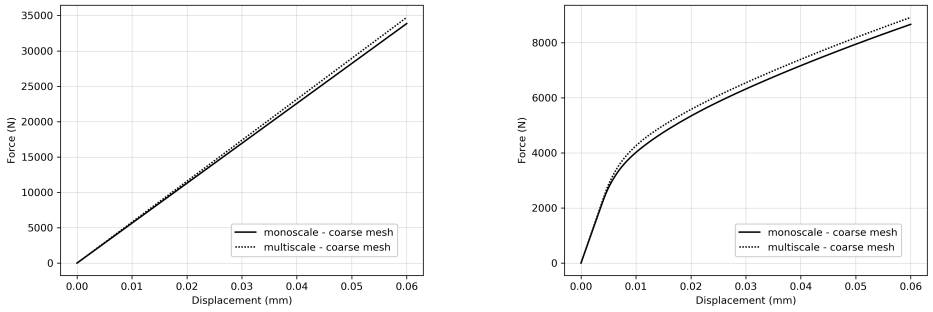
For the proposed geometry, a hexahedral element is used to define the macro-scale mesh as shown in Figure 3.20a. The micro-scale mesh is defined so as to fit into a macro-scale hexahedral element. It has the same number of Timoshenko beams (1838) as the coarse mesh in the mono-scale example.



(a) **(b)**

Figure 3.20: (a) Macro-scale mesh consisting of one hexahedral element; (b) coarse micro-scale mesh for one hexahedral element consisting of 1838 Timoshenko beam elements

Multi-scale and mono-scale examples are executed and their results compared for the meshes with the same number of Timoshenko beam elements. It can be seen in Figure 3.21 that the results show a good fit for the elasticity and hardening phase.

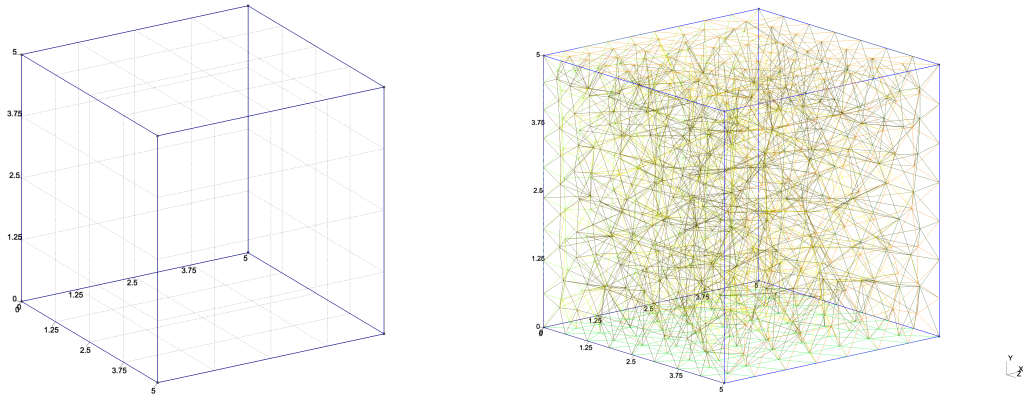


(a) **(b)**

Figure 3.21: Tension test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the coarse micro-scale mesh (a) elasticity; (b) hardening

One macro-scale hexahedron - fine micro-scale mesh

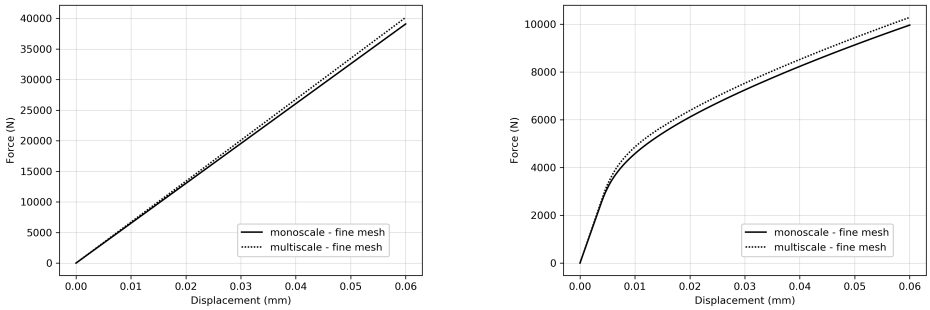
A fine micro-scale mesh with the same number of Timoshenko elements (3411 elements) as the fine mesh in the mono-scale example is defined. Multi-scale and mono-scale examples are executed and results compared.



(a) **(b)**

Figure 3.22: (a) Macro-scale mesh consisting of one hexahedral element; (b) fine micro-scale mesh for one hexahedral element consisting of 3411 Timoshenko beam elements

It can be seen in Figure 3.23 that the obtained forces are approximately the same when the same number of Timoshenko elements are used on the mono-scale and the micro-scale level.

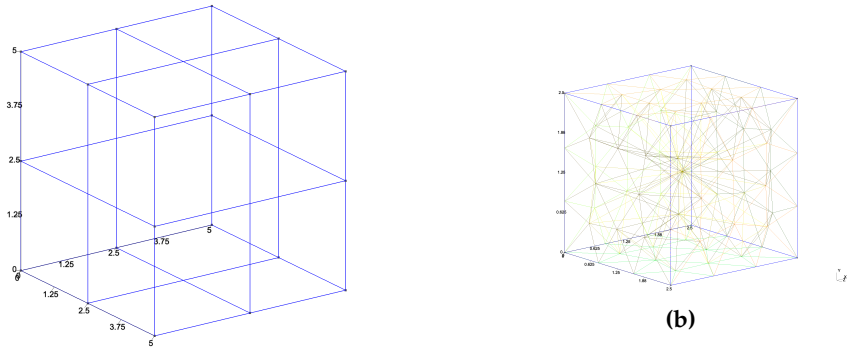


(a) **(b)**

Figure 3.23: Tension test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the fine micro-scale mesh (a) elasticity; (b) hardening

Eight macro-scale hexahedra - coarse micro-scale mesh

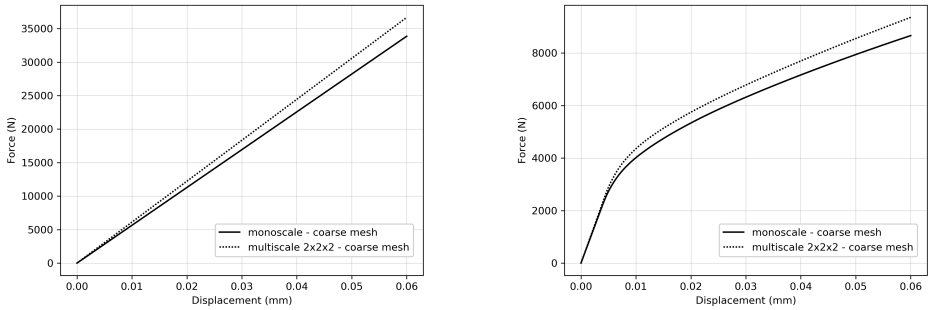
Now, on the macro-scale, eight hexahedral elements are defined. Inside each of the hexahedra, there is a micro-scale mesh defined in a way that the total number of Timoshenko beams for the multi-scale example is approximately the same as the number of beams in the mono-scale example (the elements on the boundaries between the macro-scale elements are excluded).



(a) **(b)**

Figure 3.24: (a) Macro-scale mesh consisting of eight hexahedral elements; (b) coarse micro-scale mesh for one hexahedral element consisting of 376 Timoshenko beam elements

Comparable force-displacement diagrams are obtained for the elasticity and hardening case as shown in Figure 3.25.

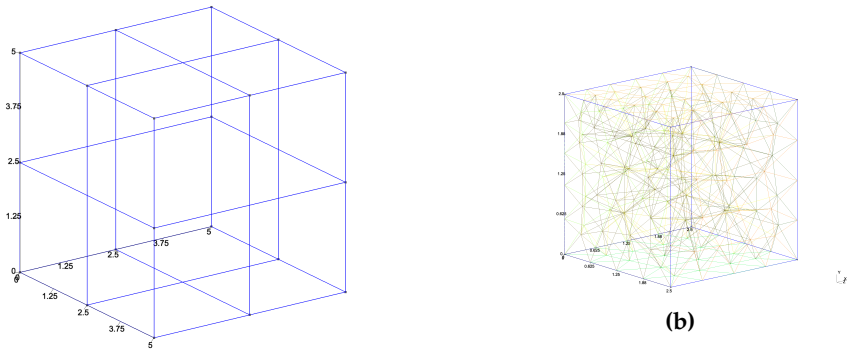


(a) **(b)**

Figure 3.25: Tension test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the coarse micro-scale mesh (a) elasticity; (b) hardening

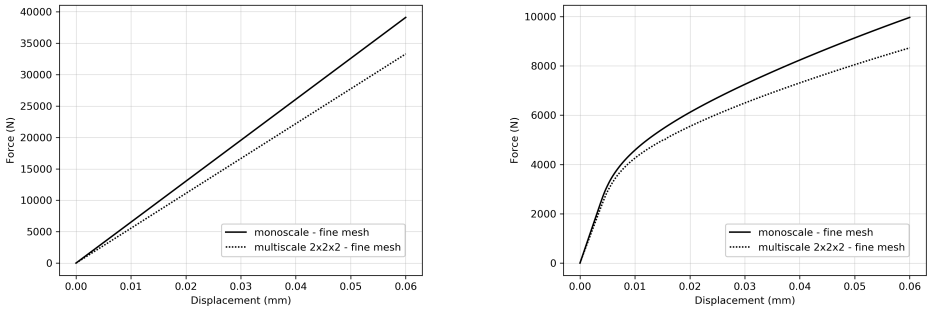
Eight macro-scale hexahedra - fine micro-scale mesh

For eight macro-scale elements, the micro-scale mesh is generated to be finer, with a larger number of elements.



(a) **(b)**

Figure 3.26: (a) Macro-scale mesh consisting of eight hexahedral elements; (b) fine micro-scale mesh for one hexahedral element consisting of 769 Timoshenko beam elements



(a) **(b)**

Figure 3.27: Tension test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the fine micro-scale mesh (a) elasticity; (b) hardening

Superposed diagram for multi-scale method examples

The following diagrams in Figures 3.28 and 3.29 show the results of the execution for all the multi-scale computations, with one and eight macro-scale elements, using both coarse and fine micro-scale meshes, for elasticity and hardening.

Keeping in mind that the mono-scale method itself gives slightly different results for different meshes, the results obtained with the multi-scale method are in agreement with it.

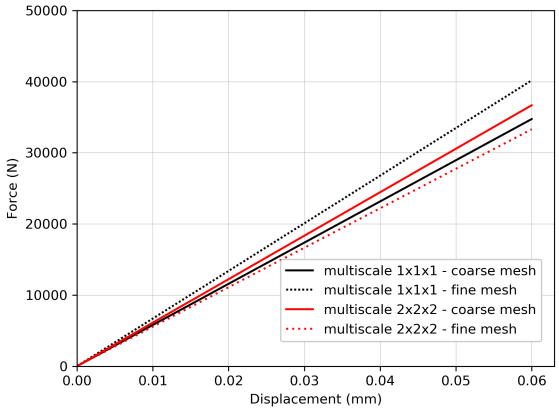


Figure 3.28: Tension test: Force-displacement diagram for the elasticity phase obtained using the mono-scale method for one macro-scale hexahedron and eight macro-scale hexahedra for both coarse and fine micro-scale mesh

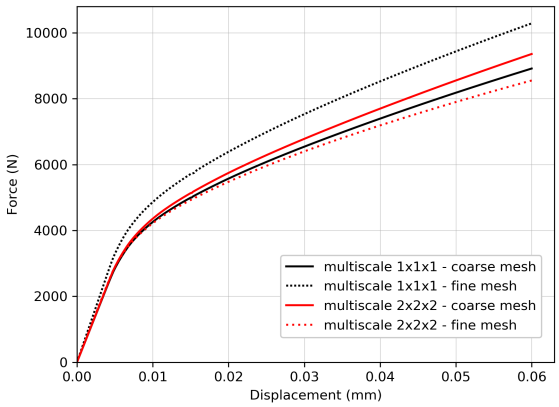


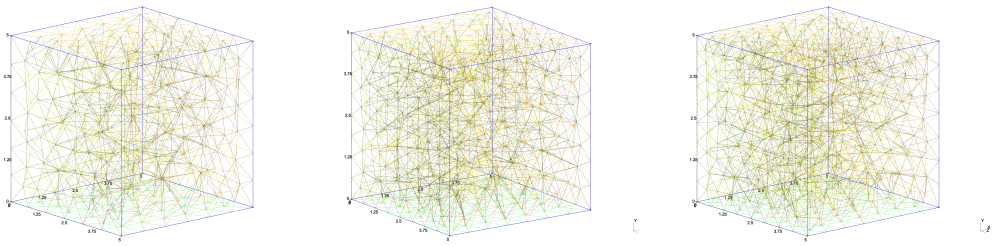
Figure 3.29: Tension test: Force-displacement diagram for the hardening phase obtained using the mono-scale method for one macro-scale hexahedron and eight macro-scale hexahedra for both coarse and fine micro-scale mesh

3.4.3.2 Compression test

The second numerical example is a compression test with the same geometry as in the tension test. The displacement $\bar{u}$ is imposed at the top face and has a negative value of -0.06 cm. The other boundary conditions and the material properties remain the same as in the tension test.

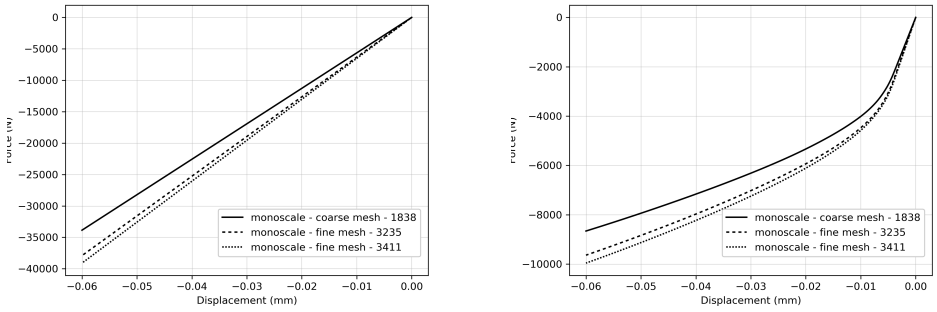
Mono-scale computation

Three meshes with a different number of elements are generated and used for the mono-scale computations. The mesh with the smallest (coarse mesh) and the mesh with the biggest number of elements (fine mesh) are used for comparison with the multi-scale computations.



(a) (b) (c)

Figure 3.30: Mono-scale mesh consisting of (a) 1838 Timoshenko beam elements - coarse mesh; (b) 3235 Timoshenko beam elements; (b) 3411 Timoshenko beam elements - fine mesh



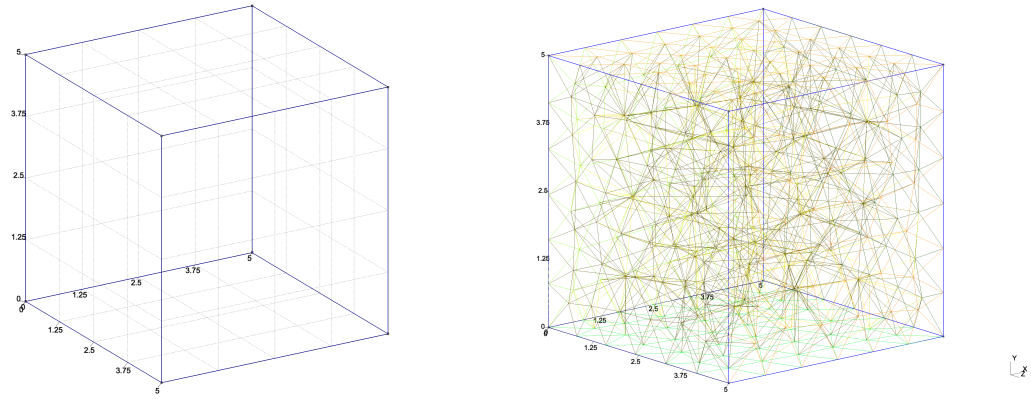
(a) (b)

Figure 3.31: Compression test: Force-displacement diagram obtained using the mono-scale method for three different meshes (a) elasticity; (b) hardening

Multi-scale computation

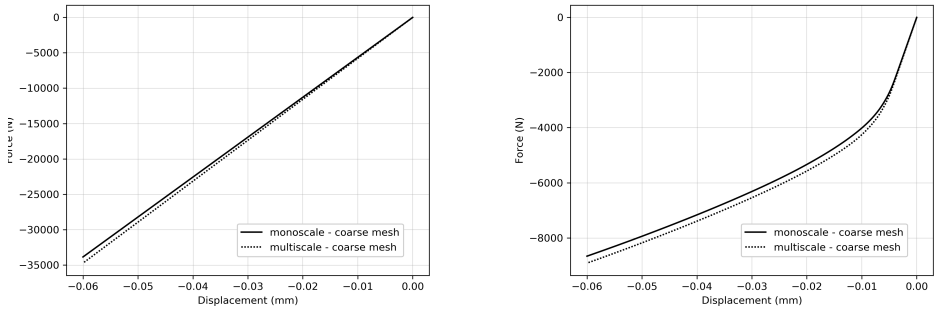
One macro-scale hexahedron - coarse micro-scale mesh

The micro-scale mesh is defined so as to fit into a macro-scale hexahedral element. Multi-scale and mono-scale examples are executed and their results compared for the meshes with the same number (1838) of Timoshenko beam elements.



(a) **(b)**

Figure 3.32: (a) Macro-scale mesh consisting of one hexahedral element; (b) coarse micro-scale mesh for one hexahedral element consisting of 1838 Timoshenko beam elements

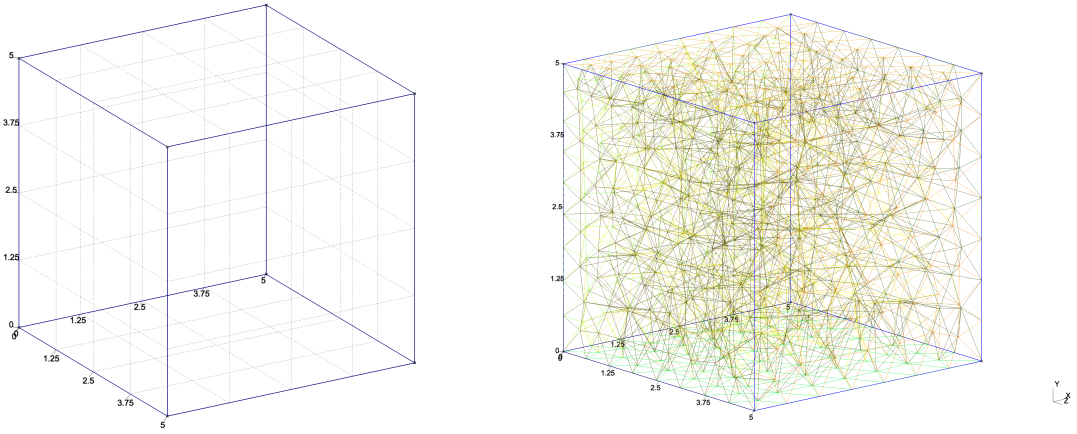


(a) **(b)**

Figure 3.33: Compression test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the coarse micro-scale mesh (a) elasticity; (b) hardening

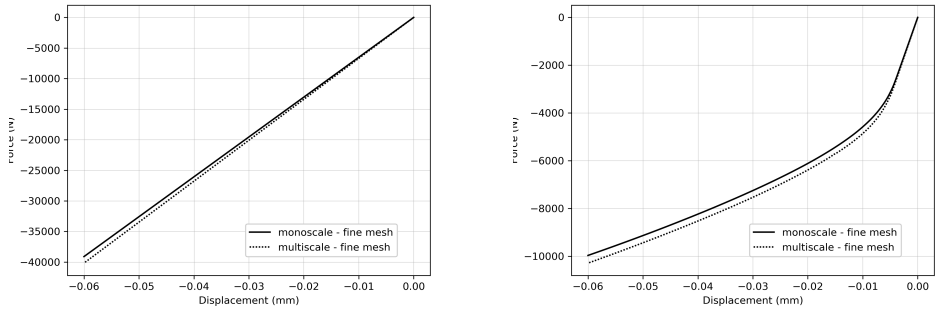
One macro-scale hexahedron - fine micro-scale mesh

The micro-scale mesh is defined so as to fit into a macro-scale hexahedral element. Multi-scale and mono-scale examples are executed and their results compared for the meshes with the same number (3411) of Timoshenko beam elements.



(a) **(b)**

Figure 3.34: (a) Macro-scale mesh consisting of one hexahedral element; (b) fine micro-scale mesh for one hexahedral element consisting of 3411 Timoshenko beam elements

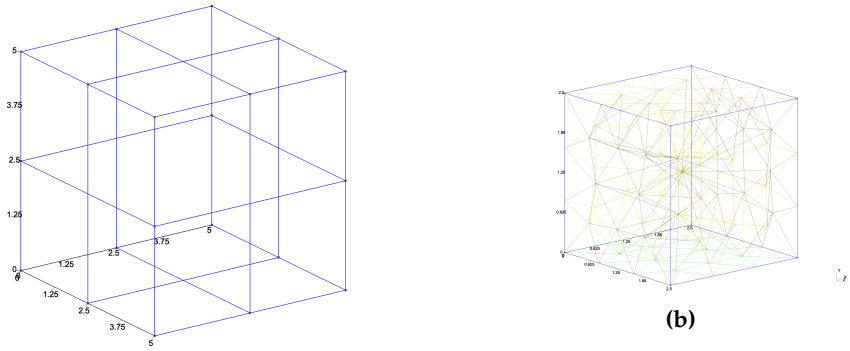


(a) **(b)**

Figure 3.35: Compression test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the fine micro-scale mesh (a) elasticity; (b) hardening

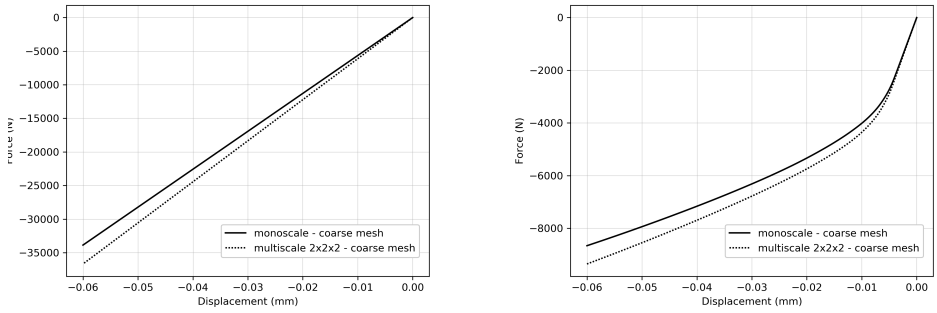
Eight macro-scale hexahedra - coarse micro-scale mesh

The micro-scale mesh is defined so as to fit into each of the macro-scale hexahedral elements. Multi-scale and mono-scale examples are executed and their results compared for the meshes with approximately the same total number of Timoshenko beam elements (the elements on the boundaries between the macro-scale elements are excluded).



(a) **(b)**

Figure 3.36: (a) Macro-scale mesh consisting of eight hexahedral elements; (b) coarse micro-scale mesh for one hexahedral element consisting of 376 Timoshenko beam elements

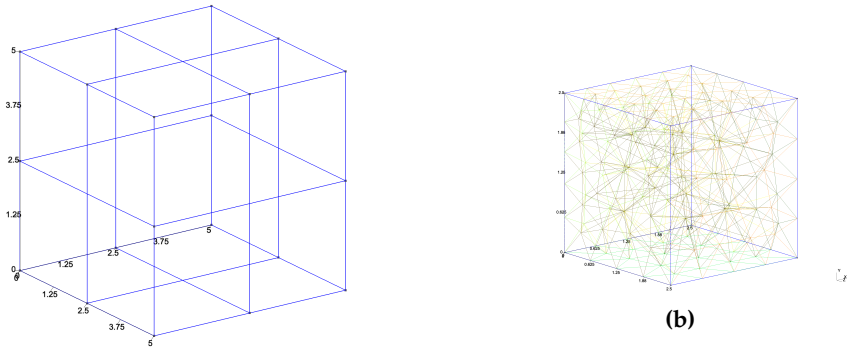


(a) **(b)**

Figure 3.37: Compression test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the coarse micro-scale mesh (a) elasticity; (b) hardening

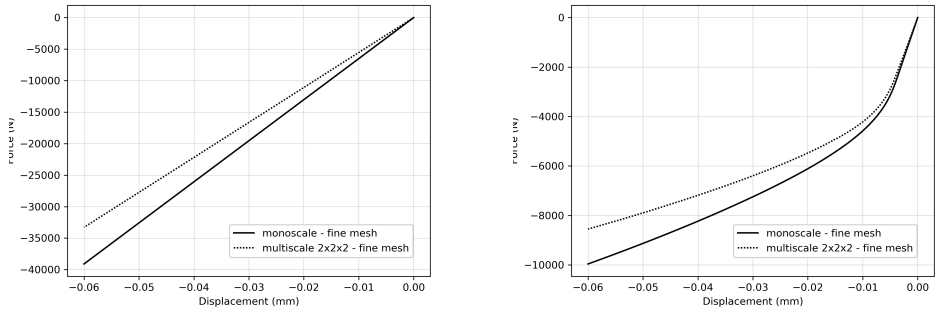
Eight macro-scale hexahedra - fine micro-scale mesh

The micro-scale mesh is defined so as to fit into each of the macro-scale hexahedral elements. Multi-scale and mono-scale examples are executed and their results compared for the meshes with approximately the same total number of Timoshenko beam elements (the elements on the boundaries between the macro-scale elements are excluded).



(a) **(b)**

Figure 3.38: (a) Macro-scale mesh consisting of eight hexahedral elements; (b) fine micro-scale mesh for one hexahedral element consisting of 769 Timoshenko beam elements



(a) **(b)**

Figure 3.39: Compression test: Force-displacement diagram obtained using the mono-scale and the multi-scale method for the fine micro-scale mesh (a) elasticity; (b) hardening

Superposed diagram for multi-scale method examples

Figure 3.40 shows the results of execution for all multi-scale computations for compression for the elasticity phase, with one and eight macro-scale elements, using both coarse and fine micro-scale meshes.

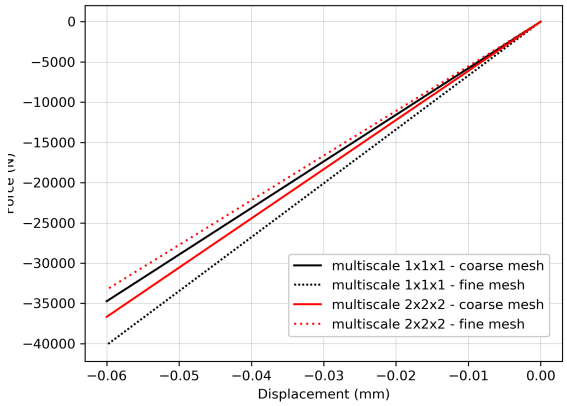


Figure 3.40: Compression test: Force-displacement diagram for the elasticity phase obtained using the mono-scale method for one macro-scale hexahedron and eight macro-scale hexahedra for both coarse and fine micro-scale mesh

The same results for hardening are shown in the Figure 3.41.

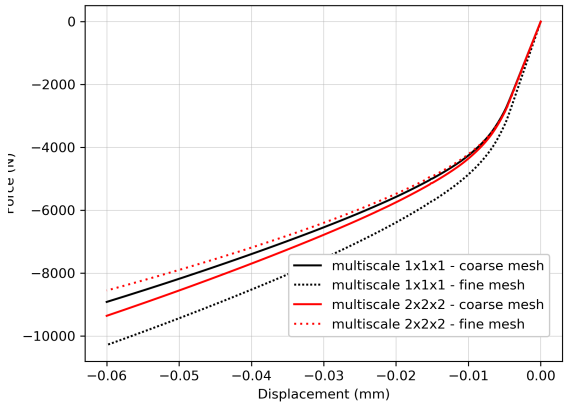


Figure 3.41: Compression test: Force-displacement diagram for the hardening phase obtained using the mono-scale method for one macro-scale hexahedron and eight macro-scale hexahedra for both coarse and fine micro-scale mesh

3.4.4 Three point bending test

To test the ability of the multi-scale code to perform the calculations where a heterogeneous stress field is present, a three point bending test numerical example is executed. Also, here we tested how the multi-scale code is working with a larger number of macro-scale elements. The results are compared to the mono-scale computations.

The geometry of the specimen is a beam with a notch, as shown in Figure 3.42. There is an imposed displacement of 0.1 cm on the top of the specimen above the notch. The specimen is supported on its ends on the bottom side. The reaction forces are measured on the supports in z-direction.

The following material parameters for the Timoshenko beam element are chosen: Young's modulus E = 210 000 MPa, Poisson's ratio ν = 0.3, axial and shear yield stresses $\bar{\sigma}_x$ = 200 MPa, $\bar{\sigma}_y$ = 50 MPa and $\bar{\sigma}_z$ = 50 MPa, axial and shear hardening moduli $\bar{K}_x$ = 0 MPa, $\bar{K}_y$ = 0 MPa and $\bar{K}_z$ = 0 MPa.

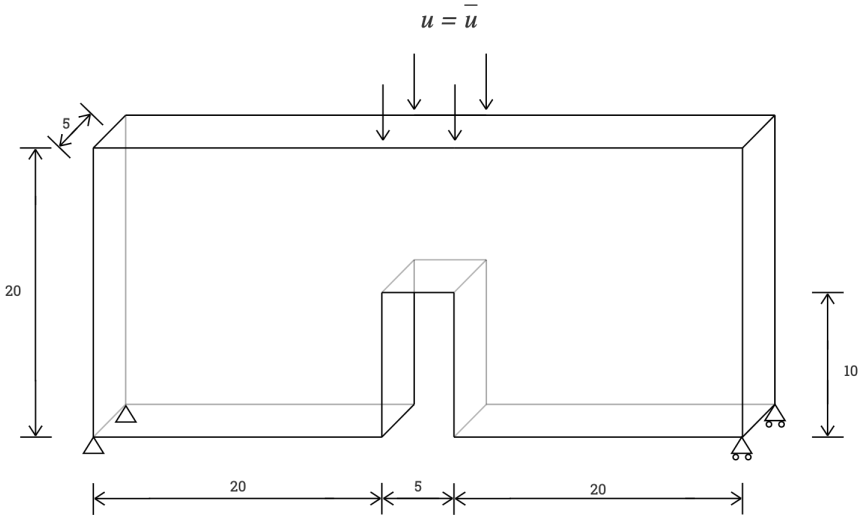


Figure 3.42: Three point bending test - geometry and boundary conditions

Mono-scale computation

The mono-scale mesh is composed of Timoshenko beam elements. In order to simulate the existence of the notch, the elements shown in red in Figure 3.44 are weakened, so they crack at the beginning of the simulation.

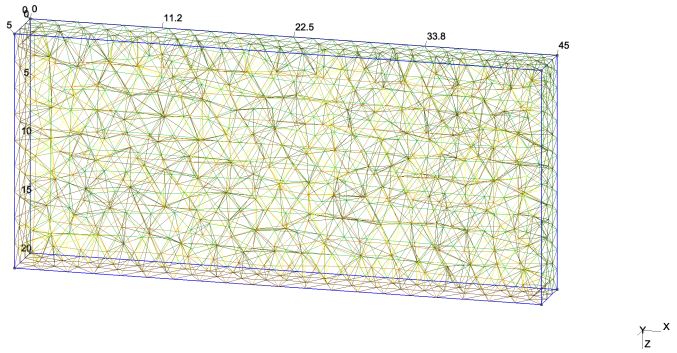


Figure 3.43: Mono-scale mesh consisting of Timoshenko beam elements

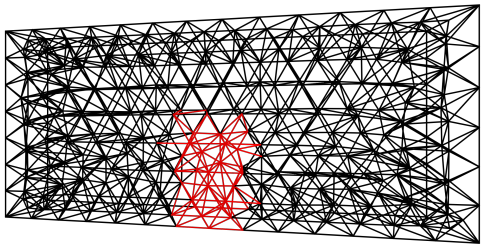


Figure 3.44: Weakened elements inside the mono-scale mesh

The computations are performed for the elasticity and hardening phase and the obtained force-displacement diagrams can be seen in Figure 3.45.

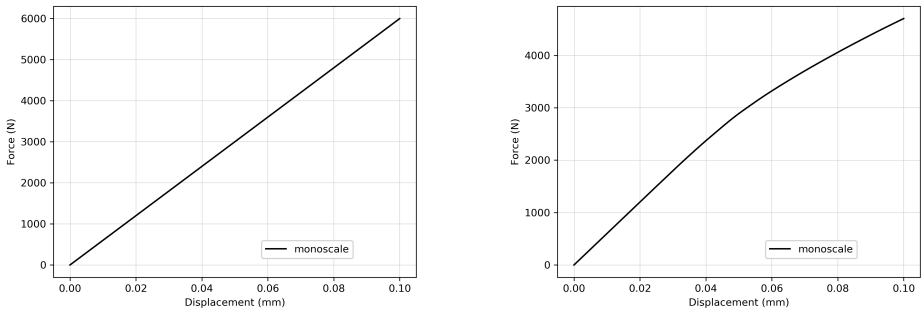
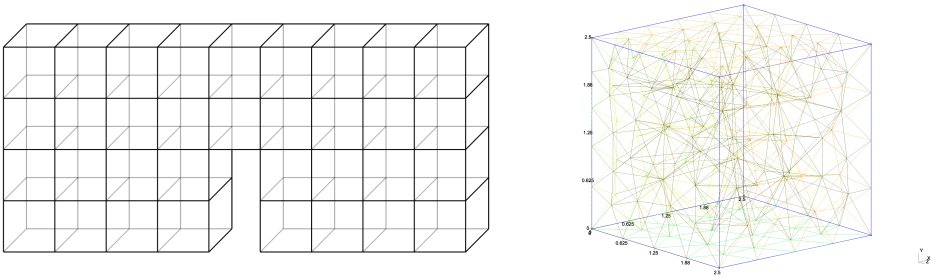


Figure 3.45: Three point bending test: force-displacement diagram obtained using the mono-scale method (a) elasticity; (b) hardening

Multi-scale computation

The macro-scale mesh consists of 34 hexahedral elements that exactly follow the proposed geometry. There are no elements on the notch, and in this way there is

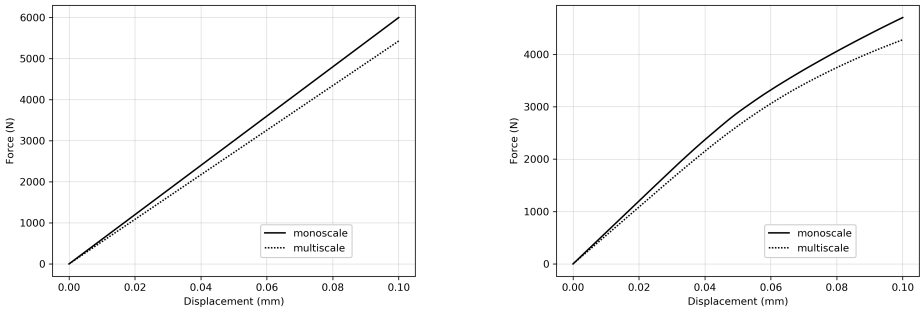
no need to weaken the beams in order to simulate it. Inside each of the macro-scale elements there is a micro-scale mesh of Timoshenko beams as shown in Figure 3.46.



(a) **(b)**

Figure 3.46: (a) Macro-scale mesh consisting of 34 hexahedral elements; (b) micro-scale mesh for one hexahedral element consisting of Timoshenko beams

Keeping in mind that the mesh density of the Timoshenko beams influences the results, and that the notch in the mono-scale example is simulated by weakening the elements, the result comparison between the two computations is only qualitative. It can be seen in Figure 3.47 that the force-displacement diagrams look approximately the same for the elasticity and hardening case.



(a) **(b)**

Figure 3.47: (a) Three point bending test: Force-displacement diagram obtained using the multi-scale method, compared to mono-scale; (b) micro-scale mesh for one hexahedral element consisting of Timoshenko beams

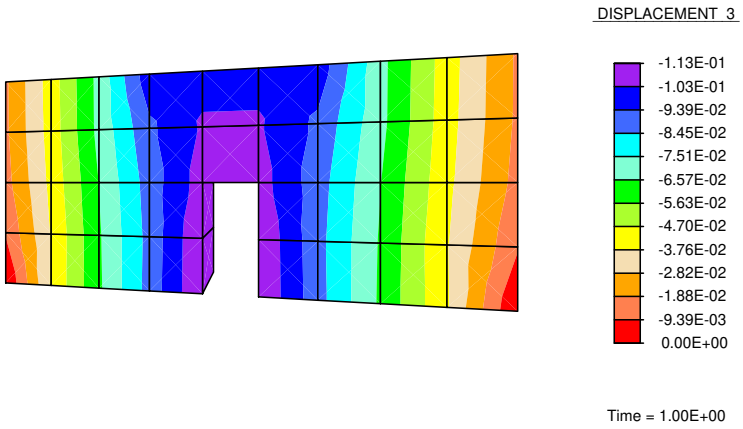


Figure 3.48: Three point bending test multi-scale computation – displacements on the macro-scale in z-direction

Conclusion

It has been shown that the multi-scale procedure with localized failure can capture the micro-scale localized failure in a 3D setting. Two types of macro-scale elements have been developed, a tetrahedral and a hexahedral element, for which the results of numerical examples were presented. The macro-scale hexahedral element was able to capture the softening behavior, with a condition that the crack opening happens in mode I only, and that the micro-scale crack openings are aligned to the macro-scale crack. Validation examples for the tension and compression test are shown, followed by the three point bending test. The numerical examples are performed with the previously developed Timoshenko beam finite element with implemented plasticity model used on the micro-scale. The scales were successfully coupled – the macro-scale element with three degrees of freedom in each node and the micro-scale element with six degrees of freedom in each node. The multi-scale solution procedure can be used with different types of micro-scale elements, as long as they can be coupled through the corresponding degrees of freedom on the interface. Also, different behavior models can be used on the micro-scale, as it was demonstrated in Chapter 2.1 for the micro-scale damage model, and in Chapter 3 for the the micro-scale plasticity model.

4

3D multi-physics coupling

Contents

In this chapter, the 3D multi-scale formulation without localized failure is extended to a multi-physics setting, where different fields from the mechanical and electrical domains can be coupled. In the first section, the multi-scale electro-mechanic formulation is presented for both tetrahedral and hexahedral macro-scale elements. A micro-scale polarization switching model implemented for Timoshenko beams is presented. Numerical examples are performed and presented, showing that the multi-scale electro-mechanic computation procedure can produce the same quality results as the mono-scale computations. In the second section, a multi-scale Hellinger-Reissner procedure for electrostatic is presented, where two independent fields are introduced – scalar electric potential and electric displacement. Also, the mono-scale formulation used on the micro-scale is described. Finally, a patch test for electrostatics is performed, and the results are compared to the mono-scale and analytical computations.

4.1 Multi-physics electro-mechanic coupling

The previously developed multi-scale model is able to capture micro-scale behavior through its three displacement degrees of freedom. In this way, it can simulate the behavior of a material in the mechanical domain only. This model can be further extended to accommodate additional degrees of freedom to implement more complex behavior and capture a multi-physics behavior on the micro-scale, where thermal, electric or magnetic fields can be present. Within the scope of this thesis, to prove that the multi-scale model can be extended to take into account additional physics, a multi-scale electro-mechanic formulation is developed and tested on several examples.

4.1.1 Multi-scale electro-mechanic formulation

To be able to capture a micro-scale behavior where different fields from mechanical and electrical domain are coupled, another degree of freedom is introduced on the macro-scale. Besides the standard three displacements contained in the displacement vector $\mathbf{u}$, the electric potential V is added. It is defined with the kinematic equation

$$\mathbf{E} = -\nabla V \tag{4.1}$$

where $\mathbf{E}$ is the electric field.

Regarding the multi-scale formulation, in addition to the field approximations defined in (3.3) for a 3D setting, a macro-scale voltage approximation is defined and calculated with standard finite element approximations

$$V^M_{n+1}\Big|_{\Gamma^{Mm,E}}(\mathbf{x}^m) = \sum_{a \in \Gamma^{Mm,E}} N^{M,V,E}_a(\boldsymbol{x}^m) v^{M,E}_{a,n+1} \tag{4.2}$$

where $N^{M,V,E}_a$ is the standard finite element shape function for a 3D tetrahedral or hexahedral element. Both types of macro-scale elements are implemented and tested with this formulation. The standard finite element shape functions for the 3D isoparametric tetrahedral element are given in (3.1) and (3.2).

The localized Lagrange multipliers enforce that the voltage on the interface nodes at the micro-scale is calculated as a linear interpolation of the nodal values of voltage at the macro-scale, the same as for the displacements, as explained in Section 2.1. In addition to the micro-scale nodal displacements defined in (3.4)

$$\underset{(3n\times 1)}{\bar{\mathbf{d}}^m_{n+1}}\Big|_{\Gamma^{M,E}} = \underset{(3n\times 3N)}{\mathbf{T}^E}\,\underset{(3N\times 1)}{\mathbf{d}^{M,E}_{n+1}} \tag{4.3}$$

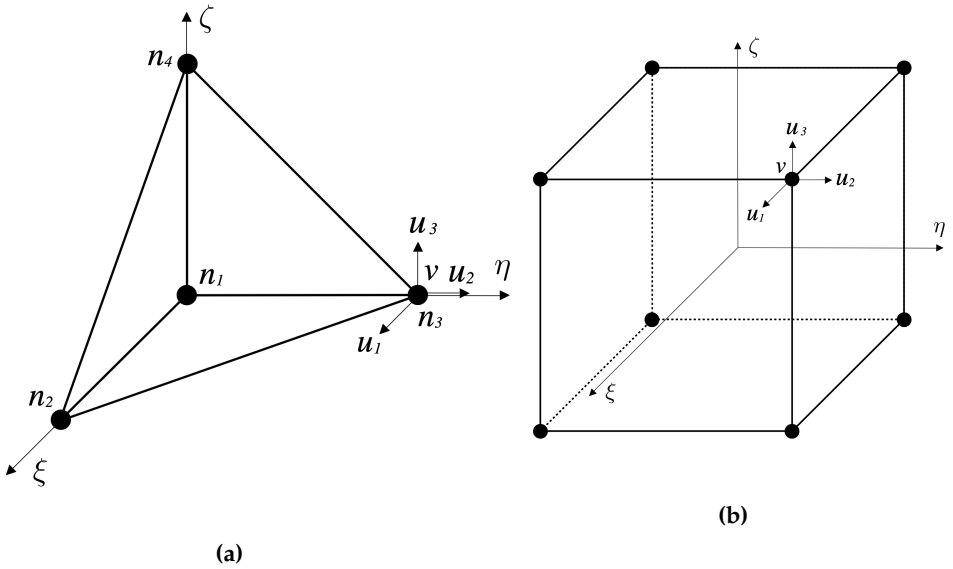


Figure 4.1: (a) 3D isoparametric tetrahedral element with added voltage; (b) 3D isoparametric hexahedral element with added voltage

the micro-scale voltage can be calculated as

$$\underset{(n\times 1)}{\bar{v}^{m}_{n+1}}\Big|_{\Gamma^{M,E}} = \underset{(n\times N)}{\mathbf{U}^{E}}\ \underset{(N\times 1)}{v^{M,E}_{n+1}} \tag{4.4}$$

Here, $\mathbf{T}^E$ is the connectivity matrix for displacements, as defined in Section 2.2.1, $\mathbf{U}^E$ is the connectivity matrix for voltage, the value n is equal to the number of interface micro-scale nodes, and the value N is equal to the number of macro-scale nodes for one element. In the case of the macro-scale tetrahedral element, N equals 4, and in the case of the macro-scale hexahedral element, N equals 8.

The connectivity matrices are based on the particular values of macro-scale shape functions which correspond to the interface nodes, as explained in detail in Section 2.2.1. In this way, the micro-scaled voltage on the interface is imposed by the macro-scale voltage, as a linear interpolation between the macro-scale nodal values.

To simplify the calculations, three components of displacements and one component of voltage can be written as a new vector representing four degrees of freedom in a node n_a

$$\hat{\boldsymbol{d}}_a = [u_{1,a} \quad u_{2,a} \quad u_{3,a} \quad v_a]^T \tag{4.5}$$

Since the transformation matrix values for three nodal displacement and nodal value of voltage are calculated with the same shape functions, a new transformation matrix $\mathbf{R}^E$ can be constructed, which contains values from both matrix $\mathbf{T}^E$ and $\mathbf{U}^E$. It is constructed in the same way as transformation matrix $\mathbf{T}^E$,

as described in detail in Section 2.2.1, but with added voltage as the fourth degree of freedom. Then, the sum of the shape functions with introduced coordinates of the micro-scale interface node n_a^m can be written as

$$\mathbf{R}_a^E = \sum_b N_b^M(\mathbf{x}_a) \tag{4.6}$$

for the three displacements degrees of freedom and for the voltage, where $\mathbf{x}_a$ are the coordinates of interface micro-scale node n_a^m. The part of the transformation matrix $\mathbf{R}^E$ related to the micro-scale interface node n_a^m for all four degrees of freedom can be then written in matrix notation as

$$\underset{(4\times 4N)}{\mathbf{R}_a^E} = \begin{bmatrix} N_1^M(\mathbf{x}_a) & 0 & 0 & 0 \\ 0 & N_1^M(\mathbf{x}_a) & 0 & 0 \\ 0 & 0 & N_1^M(\mathbf{x}_a) & 0 \\ 0 & 0 & 0 & N_1^{M,V}(\mathbf{x}_a) \\ N_2^M(\mathbf{x}_a) & 0 & 0 & 0 \\ 0 & N_2^M(\mathbf{x}_a) & 0 & 0 \\ 0 & 0 & N_2^M(\mathbf{x}_a) & 0 \\ 0 & 0 & 0 & N_2^{M,V}(\mathbf{x}_a) \\ N_3^M(\mathbf{x}_a) & 0 & 0 & 0 \\ 0 & N_3^M(\mathbf{x}_a) & 0 & 0 \\ 0 & 0 & N_3^M(\mathbf{x}_a) & 0 \\ 0 & 0 & 0 & N_3^{M,V}(\mathbf{x}_a) \\ N_4^M(\mathbf{x}_a) & 0 & 0 & 0 \\ 0 & N_4^M(\mathbf{x}_a) & 0 & 0 \\ 0 & 0 & N_4^M(\mathbf{x}_a) & 0 \\ 0 & 0 & 0 & N_4^{M,V}(\mathbf{x}_a) \end{bmatrix}^T \tag{4.7}$$

where $N_b^M(\mathbf{x}_a)$ is the value of the standard shape function for displacements with coordinates of the micro-scale node n_a^m, and $N_b^{M,V}(\mathbf{x}_a)$ is the value of the standard shape function for voltage with coordinates of the micro-scale node n_a^m.

Taking this into account, (4.3) and (4.4) can be written in one expression as

$$\underset{(4n\times 1)}{\bar{\hat{\boldsymbol{d}}}_{n+1}^m}\Big|_{\Gamma^{M,E}} = \underset{(4n\times 4N)}{\mathbf{R}^E}\ \underset{(4N\times 1)}{\hat{\boldsymbol{d}}_{n+1}^{M,E}} \tag{4.8}$$

Then, the standard finite element system of equations for computing the increment of the displacement field and voltage on the micro-scale can be written as

$$\begin{bmatrix} \underset{(4n_\Gamma\times 4n_\Gamma)}{\bar{\bar{\mathbf{K}}}} & \underset{(4n_\Gamma\times 4n_f)}{\bar{\mathbf{K}}^T} \\ \underset{(4n_f\times 4n_\Gamma)}{\bar{\mathbf{K}}} & \underset{(4n_f\times 4n_f)}{\mathbf{K}} \end{bmatrix} \begin{bmatrix} \underset{(4n_\Gamma\times 1)}{\Delta\bar{\hat{\boldsymbol{d}}}} \\ \underset{(4n_f\times 1)}{\Delta\hat{\boldsymbol{d}}} \end{bmatrix} = - \begin{bmatrix} \underset{(4n_\Gamma\times 1)}{\bar{\mathbf{r}}} \\ \underset{(4n_f\times 1)}{\mathbf{r}} \end{bmatrix} \tag{4.9}$$

where $\bar{\bar{\mathbf{K}}}$ is the part of the micro-scale stiffness matrix related only to interface nodes, $\bar{\mathbf{K}}$ is related to interface nodes in relation to free nodes and $\mathbf{K}$ is related only to free nodes. In the same way, $\Delta\bar{\hat{\boldsymbol{d}}}$ is displacement and voltage increments on the interface nodes, and $\bar{\mathbf{r}}$ is the residual on the interface nodes. Also, $\Delta\hat{\boldsymbol{d}}$ is displacement and voltage increments on free nodes, and $\mathbf{r}$ is the residual on free nodes. The value n_Γ is the number of interface micro-scale nodes, and the value n_f is the number of free micro-scale nodes.

After performing the static condensation in the same way as in Section 3.1, and after the computations on the micro-scale have converged, the final values of the micro-scale condensed stiffness matrix and residual are used to compute the values of the stiffness matrix and residual to be used at the macro-scale

$$\begin{aligned} \underset{(4N\times 4N)}{\mathbf{K}^{M,E}_{n+1}} &= \underset{(4N\times 4n_\Gamma)}{\mathbf{R}^{E,T}} \; \underset{(4n_\Gamma\times 4n_\Gamma)}{\widetilde{\mathbf{K}}^{m}_{n+1}} \; \underset{(4n_\Gamma\times 4N)}{\mathbf{R}^{E}} \\ \underset{(4N\times 1)}{\mathbf{r}^{M,E}_{n+1}} &= \underset{(4N\times 4n_\Gamma)}{\mathbf{R}^{E,T}} \; \underset{(4n_\Gamma\times 1)}{\widetilde{\mathbf{r}}^{m}_{n+1}} \end{aligned} \tag{4.10}$$

where $\widetilde{\mathbf{K}}^{m}_{n+1}$ and $\widetilde{\mathbf{r}}^{m}_{n+1}$ are the statically condensed stiffness matrix and residual obtained at the micro-scale, as explained in Section 2.1.

When the values of the macro-scale stiffness matrix and residual are computed, they are used to update the values of the macro-scale displacement field and voltage. The standard finite element system of equations needs to be solved

$$\mathbf{K}^{M}_{n+1}\Delta\hat{\boldsymbol{d}}^{M}_{n+1} = -\mathbf{r}^{M}_{n+1} \tag{4.11}$$

after which the corresponding values of the macro-scale displacement and voltage are updated with

$$\hat{\boldsymbol{d}}^{M(i+1)}_{n+1} = \hat{\boldsymbol{d}}^{M(i)}_{n+1} + \Delta\hat{\boldsymbol{d}}^{M(i)}_{n+1} \tag{4.12}$$

where $\hat{\boldsymbol{d}}^{M(i+1)}_{n+1}$ is a vector that contains three values of displacement and one value of voltage in time step $n+1$ and iteration $(i+1)$.

With the additional value of voltage stored on the macro-scale, the model is able to capture any micro-scale behavior resulting in an altered value of voltage. Its application will be shown in several examples with piezoelectric effect and polarization switching model at the micro-scale.

It has to be noted that the multi-physics coupling is implemented in a multi-scale setting in such a way that the macro-scale element itself does not simulate the multi-physics behavior, and does not couple the mechanical and electrical domain. It only serves as an empty macro-scale element, without any constitutive equations defined, that is able to capture the micro-scale behavior that couples the electrical and the mechanic fields. On the macro-scale, that effects have to be detectable through four degrees of freedom - three values of displacement and one value of voltage.

4.1.2 Micro-scale polarization switching model

On the micro-scale, a Timoshenko beam finite element that can capture the piezoelectric coupling along with a non-linear constitutive behavior for electric and mechanic fields is used. It was developed as part of the work from (Moreno-Navarro, 2019), where additional formulation and implementation details can be found.

Kinematics equations are defined as

$$\begin{aligned} \boldsymbol{\varepsilon} &= \frac{1}{2}\left[\nabla \otimes \mathbf{u} + (\nabla \otimes \mathbf{u})^T\right] = \nabla^{sy}\mathbf{u} \\ \mathbf{E} &= -\nabla V \end{aligned} \tag{4.13}$$

where $\boldsymbol{\varepsilon}$ is the strain tensor, $\mathbf{u}$ is the displacement field, $\mathbf{E}$ is the electric field and V is electric potential.

The model of the beam is reduced in a way that it takes into account only the axial direction. Then, the only strain tensor components that are left are the axial strain ε_{xx}, shear strains γ_{xy} and γ_{xz}, the curvatures κ_x, κ_y and κ_z, and the axial electric field E_x. The strain tensor calculation from the kinematics equation (4.13) for the Timoshenko beam finite element, as shown in (Moreno-Navarro et al., 2018), can be then written

$$\begin{bmatrix} \varepsilon_{xx} \\ \gamma_{xy} \\ \gamma_{xz} \\ \kappa_x \\ \kappa_y \\ \kappa_z \\ E_x \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial}{\partial x} & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & \frac{\partial}{\partial x} & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial}{\partial x} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ \varphi \\ \psi \\ \theta \\ V \end{bmatrix} \tag{4.14}$$

where u, v and w are the displacements along the local axes x, y and z, and φ, ψ and θ are the rotations around the local axes x, y and z, respectively.

If the displacement and electric potential vector from (4.14) is compared to the macro-scale displacement field defined in (4.5) it can be seen that the micro-scale displacement field has additional three rotational degrees of freedom that are ignored during the scale coupling, as explained in the examples in the next section.

Conservation equations for the beam where the body forces are neglected and a quasistatic approach is taken can be written as

$$\begin{aligned} \boldsymbol{\sigma}\nabla &= 0 \\ \nabla \cdot \mathbf{D} &= 0 \end{aligned} \tag{4.15}$$

where σ is the stress tensor and $\mathbf{D}$ is the electric displacement.

In the constitutive equations for the piezoelectric material, it can be seen that the electrical and mechanical variables are coupled as they influence each other. They can be derived from the free-energy potential as explained in (Moreno-Navarro et al., 2018)

$$\begin{aligned} \boldsymbol{\sigma} &= \boldsymbol{C}\boldsymbol{\varepsilon} - \mathbf{e}^{e}\mathbf{E} \\ \boldsymbol{D} &= \boldsymbol{\epsilon}\boldsymbol{E} + \mathbf{e}^{e}\boldsymbol{\varepsilon} \end{aligned} \tag{4.16}$$

where C is the stiffness tensor, $\mathbf{e}^{e}$ is the piezoelectric tensor, and ϵ is the permittivity tensor. To simplify the constitutive equations applied to beams, some of the components can be left our from the final equations. For the relation between the strain and the electric variables, only the axial component is taken into account. Then the constitutive equations for the beam can be written as

$$\begin{bmatrix} N_x \\ N_y \\ N_z \\ M_x \\ M_y \\ M_z \\ Q_x \end{bmatrix} = \begin{bmatrix} \breve{E}A & 0 & 0 & 0 & 0 & 0 & -e_{11}A \\ 0 & k_c GA & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & k_c GA & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & GJ & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \breve{E}I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \breve{E}I & 0 \\ e_{11}A & 0 & 0 & 0 & 0 & 0 & \epsilon_1 A \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \gamma_{xy} \\ \gamma_{xz} \\ \kappa_x \\ \kappa_y \\ \kappa_z \\ E_x \end{bmatrix} \tag{4.17}$$

where N_i is the force in direction i, M_i is the moment in direction i, Q_x is the electric charge in direction x, $k_c$is the shear correction factor, G is the shear modulus, J is the polar moment of inertia, I is the moment of inertia, ϵ_1 is the permittivity in axial direction, e_{11} is the piezoelectric coefficient and $\breve{E}$ is the first term of the stiffness tensor that is defined as

$$\breve{E} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \tag{4.18}$$

where E is the Young's modulus and ν is the Poisson's coefficient.

Beams developed as a part of the work from (Moreno-Navarro, 2019) have more complex constitutive equations where plasticity and ferroelectricity behavior are modeled. Ferroelectricity is a property of certain dielectrics that exhibit spontaneous electric polarization that can be reversed in direction by applying an electric field (Schwartz, 2002). In the ferroelectric model, the polarization $\mathbf{P}$ is introduced, a macroscopic magnitude that accumulates the microscopic electric dipole moments in a material. The polarization P is a measure of the degree of

piezoelectricity in the material. There are two types of polarization sources present in the material. The first one is generated in the presence of the electric field and it is proportional to the applied value. The second is a consequence of a specific material microstructure and it is a permanent value called remanent polarization $\mathbf{P}^r$. Remanent polarization has only the possibility of domain switching or changing the orientation (Moreno-Navarro et al., 2018; Schwartz, 2002).

There are three types of behavior regarding the polarization: dielectric, paraelectric and ferroelectric. For both dielectric and paraelectric behaviors the polarization depends only on the electric field. The first one is linear and the second one is a non-linear dependency. In the ferroelectrics behavior, the polarization is non-linearly dependent on the electric field, but also on the superposed remanent polarization that causes hysteresis phenomena. This non-linearity in polarization is defined as hysteresis. The existence of spontaneous polarization together with polarization reversal is generally accepted as proof of ferroelectricity (Schwartz, 2002). Ferroelectricity always implies a coupling with the mechanical field, since only piezoelectric materials can be ferroelectrics (Moreno-Navarro, 2019).

Ferroelectricity is a subset of pyroelectricity; it exhibits spontaneous polarization below a transition temperature known as the Curie temperature. In ferroelectric materials, this spontaneous polarization can be reoriented by application of an electric field. Full reversal of the polarization is called domain switching. The direction of spontaneous polarization is defined by the crystal symmetry of the material and is most significant well below the Curie temperature (Said et al., 2017).

There are two switching types possible, 180° and 90° switching, and it depends on the angle between the old and the new $\mathbf{P}^r$ vector. An electric field can induce both switches, while stress can only induce a 90° switch. Domain switch criteria are extracted from (Keip and Schröder, 2011), where the combination of both electric field and stress is taken into account.

The model used to describe the polarization switching is able to represent the 180° switch, since beams can only take into account axial variations in voltage. In this model, every beam is in non-polarized state at the beginning. When the electric field has reached the coercive value E_c the beam gets positively polarized. In the same way, if the electric field reaches the value $-E_c$ the beam gets negatively polarized. Once the beam is polarized, it cannot go back into a non-polarized state any more. If the coercive electrical field reaches the opposite value, it can switch to the opposite polarized state. The diagram of the polarization state switching for a beam is shown in Figure 4.2.

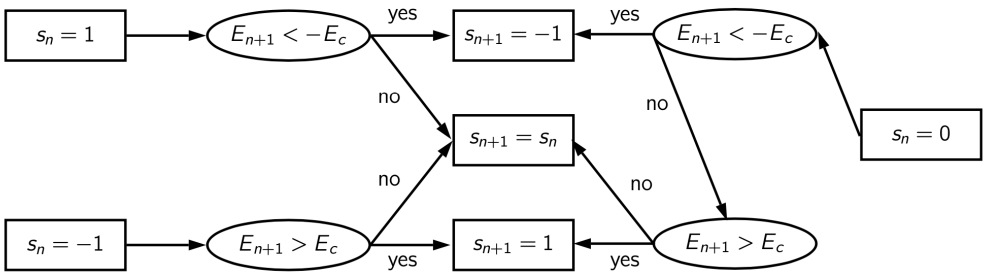


Figure 4.2: Diagram for the polarization switch model to determine the switch-state for the next time step s_{n+1} based on the previous value s_n and the current value of electric field $E_{1,n+1}$ (Moreno-Navarro, 2019)

Further description of the implementation details of finite element procedure for ferroelectric, plasticity and softening behavior of the beam model can be found in (Moreno-Navarro, 2019).

4.1.3 Numerical examples

4.1.3.1 Piezoelectric example

In this section, a simple piezoelectric mono-scale example is modeled, as presented previously in (Rukavina et al., 2017). It is based on kinematics, conservation and constitutive equations from Section 4.1.2, but without the polarization switching model taken into account. There are three cases presented, one with imposed displacement and two where a constant electric potential is imposed, where one has the boundary conditions slightly modified.

The mono-scale formulation is implemented in GetDP software that is primarily used for modeling electromagnetic problems, but can also be used for thermal, mechanical and acoustic problems (Dular and Geuzaine, 2013). The initial idea was to test and analyze GetDP and its source code for possible coupling possibilities with FEAP or other software. In this way, the electro-magnetic part of the computations would be executed in GetDP, and the mechanical part would be executed in FEAP. In the end, these examples are implemented in GetDP and compared to the same formulation implemented in FEAP. The latter is presented in (Moreno-Navarro et al., 2018).

The geometry of the specimen is represented as a rectangular cuboid of dimensions 6 × 6 × 2 mm as shown in Figure 4.3. Specific values of the Young's modulus E, Poisson's coefficient ν, piezoelectric tensor $\mathbf{e}^e$ and dielectric permittivity tensor ϵ for the proposed piezoelectric material BaTiO3 have been taken from (Ramirez et al., 2006).

Example with imposed voltage

In the first example, the electric potential $V = 10\ V$ is imposed at the top face of the cuboid, and $V = 0\ V$ at the bottom face. The vertical component of electric field $\mathbf{E}$ is present due to the electric potential gradient. Only an eighth of this geometry is taken, since symmetry conditions have been taken in planes $x = 0$, $y = 0$ and $z = 0$, as shown in Figure 4.3. As a result of the piezoelectric effect, a linear distribution of displacements is present. The displacements are negative on axis z and positive on axes x and y due to the piezoelectric tensor from the material properties, as it can be seen in Figure 4.4.

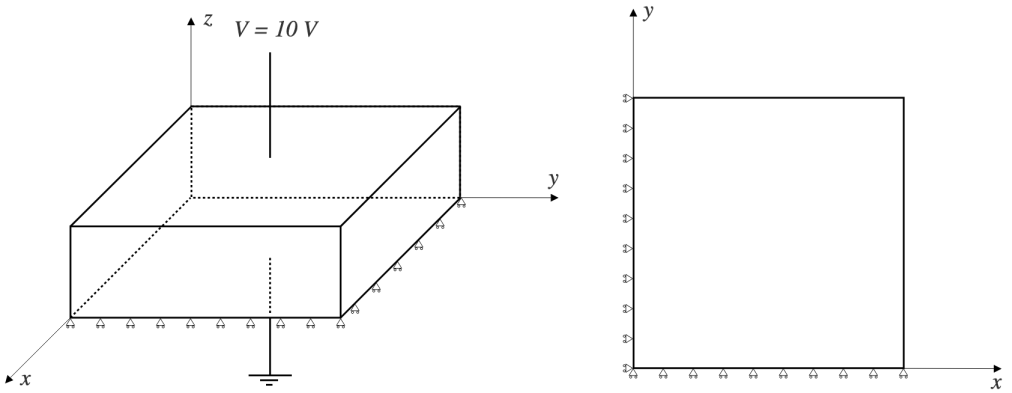


Figure 4.3: A cuboid with imposed boundary conditions - voltage $V = 0$ at the bottom face and $V = 10\ V$ at the top face

Example with imposed displacement

In the second example, a displacement $u = 10^{-4}\ m$ is imposed at the top face of the cuboid in the direction z. Also, only an eighth of the specimen geometry is taken, as shown in Figure 4.3. Due to the imposed displacement, the piezoelectric material generates linear electric potential distribution along the axis z, as it can be seen in Figure 4.5. There are also negative displacements in the directions x and y due to the Poisson effect.

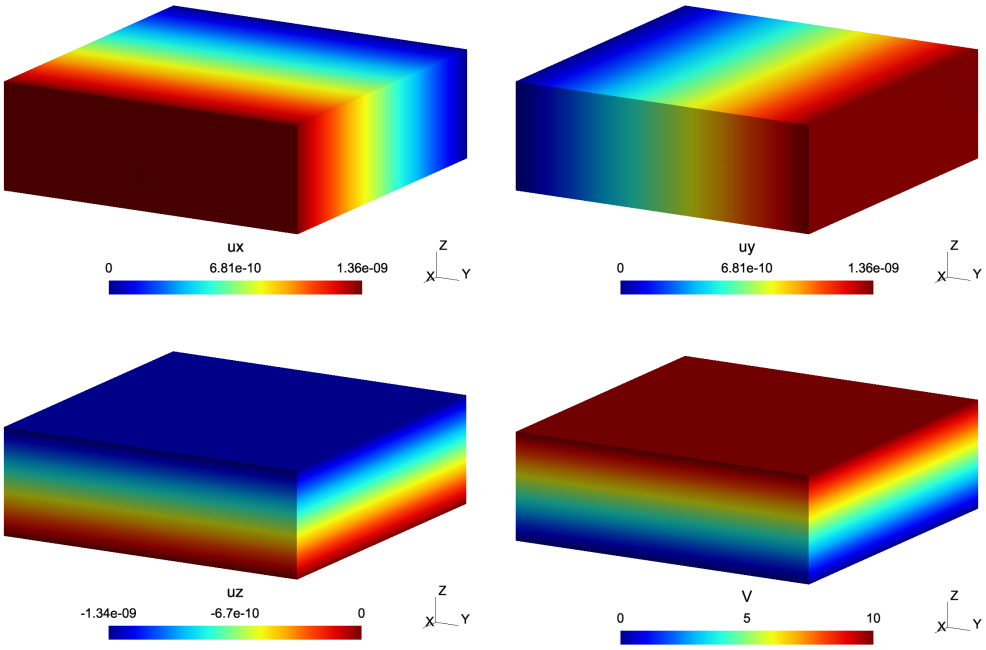


Figure 4.4: Example with imposed voltage: (a) Displacement in direction x; (b) Displacement in direction y; (c) Displacement in direction z; (d) Electric potential distribution

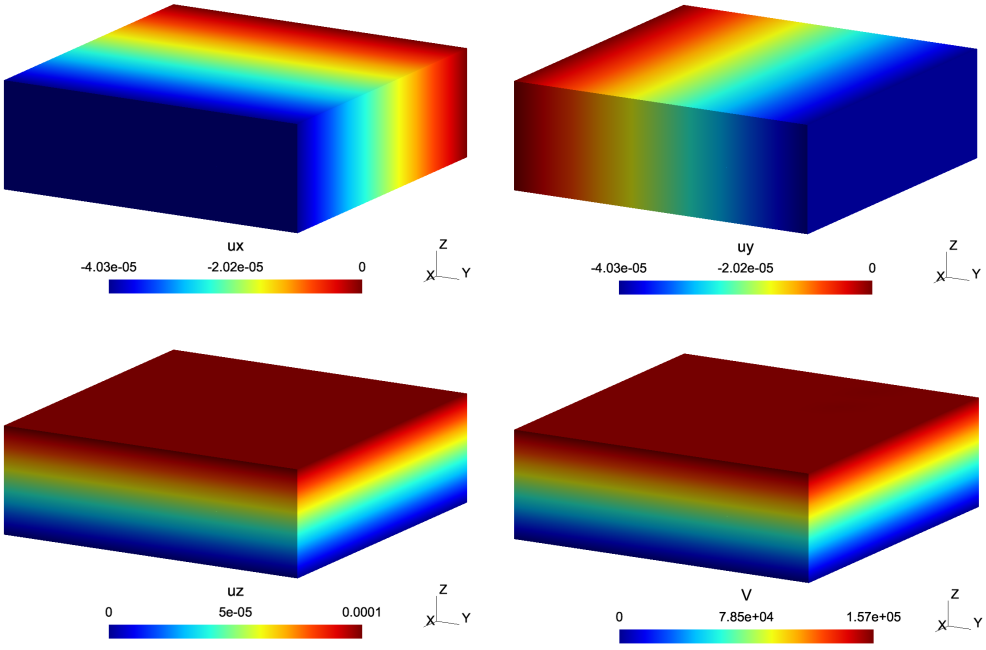


Figure 4.5: Example with imposed displacement: (a) Displacement in direction x; (b) Displacement in direction y; (c) Displacement in direction z; (d) Electric potential distribution

Example with imposed voltage - modified boundary conditions

In the third example, a quarter of the specimen is taken where the left side face and the back face are fixed in directions y and x respectively. Also, the electric potential $V = 10\ V$ is imposed at the top face of the cuboid, and $V = 0\ V$ at the bottom face. The boundary condition for the bottom face of the cuboid is changed, and it is fixed in all direction as shown in Figure 4.6.

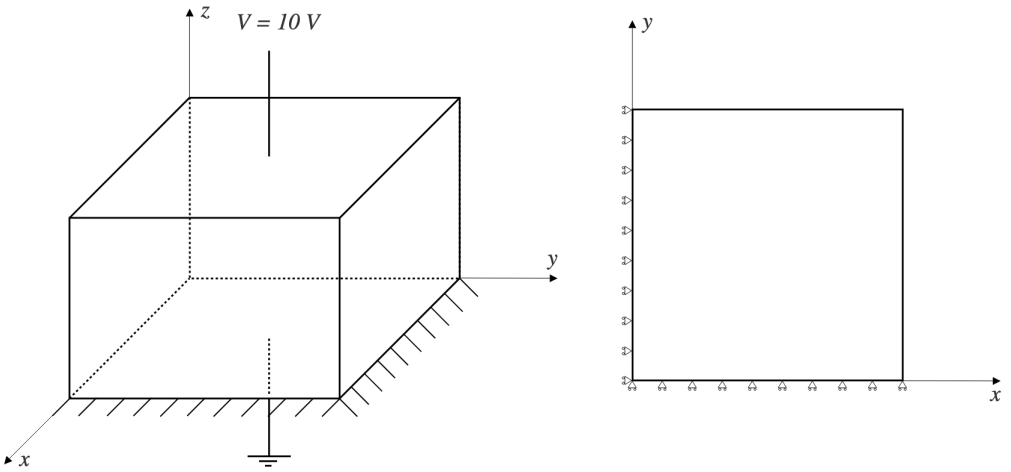


Figure 4.6: A cuboid with imposed boundary conditions - voltage $V = 0$ at the bottom face and $V = 10\ V$ at the top face

In this example, the expansion in transversal directions is restricted at the bottom, resulting in stresses being concentrated, as can be seen in Figure 4.7 The irregular distribution of the displacements in direction z is a consequence of the transversal direction displacement. The stress concentrations also appear at the bottom for this component, as well. The electric potential may seem linear, but it can be observed that near the bottom the isolines are closer than at the top. Also, the isolines are not straight and they are diverging when closer to the free edge as can be seen in Figure 4.8.

The results obtained in these examples are the same as the ones obtained with FEAP software presented in (Moreno-Navarro et al., 2018) within tolerances that depend on the chosen mesh and the finite element discretization technique.

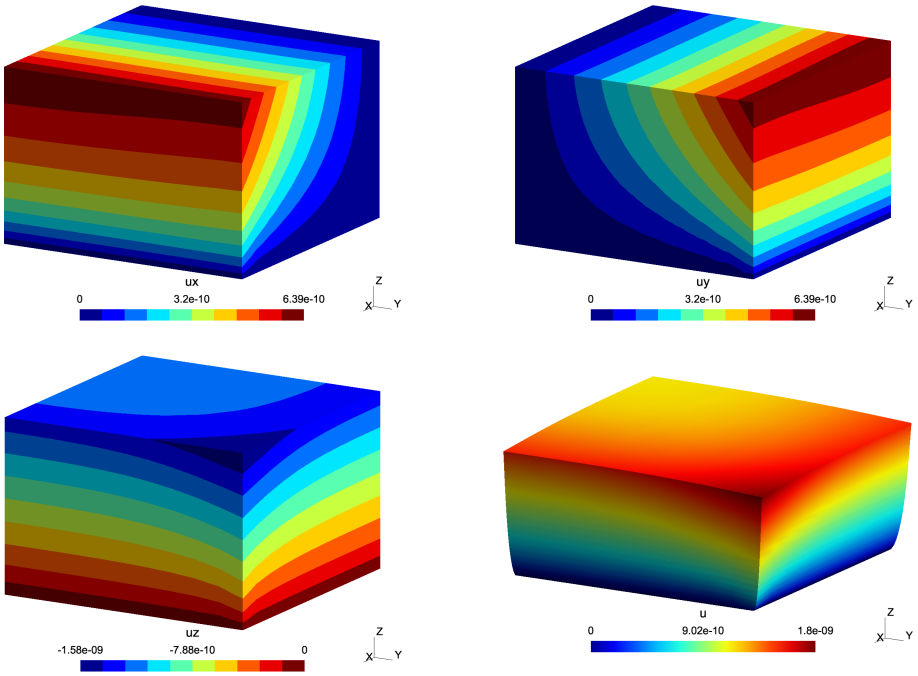


Figure 4.7: Example with imposed voltage - modified boundary conditions: (a) Displacement in direction x; (b) Displacement in direction y; (c) Displacement in direction z; (d) Total displacements

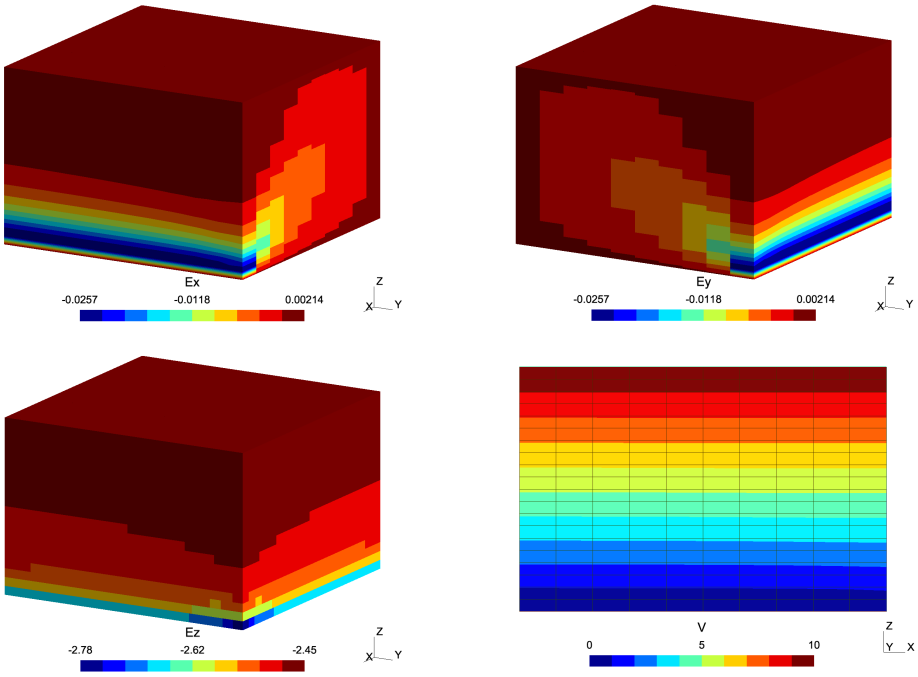


Figure 4.8: Example with imposed voltage - modified boundary conditions: (a) Electric field in direction x; (b) Electric field in direction y; (c) Electric field in direction z; (d) Electric potential distribution

4.1.3.2 Polarization switching model example

For the polarization switching model, a simple numerical example is performed. The results for the proposed multi-scale method with polarization switching are compared against the monolithic solution from (Moreno-Navarro, 2019). The goal is to prove that the multi-scale method with added degrees of freedom for voltage can capture the micro-scale polarization switching behavior and produce the same quality results as the monolithic solution.

On the micro-scale, a one-dimensional Timoshenko beam with added polarization switching model is used, that is developed in (Moreno-Navarro, 2019). The Timoshenko beam element has seven degrees of freedom, three displacements, three rotations and one voltage. To be able to couple the two scales, three micro-scale rotations are ignored when the scales communicate. Due to this, the micro-scale mesh has to be defined in a way so that rotations do not have a significant effect on computations. Also, the advantage of this kind of implementation is that any other micro-scale element developed in FEAP that models some different electro-mechanical coupling behavior can be used instead of the proposed model, with little or no modifications at all.

The geometry of the specimen is a cube with sides of 20 cm as can be seen in Figure 4.9. The bottom face has imposed voltage of $V = 0\ V$, and the top face $V = V(t)$. The voltage at the top face is imposed in a triangular pattern in time, starting from the value of $0\ MV$, increasing to $1\ MV$, then decreasing to $-1\ MV$, going back again to $1\ MV$, and finally decreasing down to $0\ MV$, as shown in Figure 4.10. The planes $x = 0$, $y = 0$ and $z = 0$, have fixed displacements in the same direction to simulate symmetry boundary conditions. There is a constant force imposed at the top face. All rotation degrees of freedom of the beams are set free. The polarization switch for each beam is set to zero. The material properties are taken from (Hwang et al., 1995).

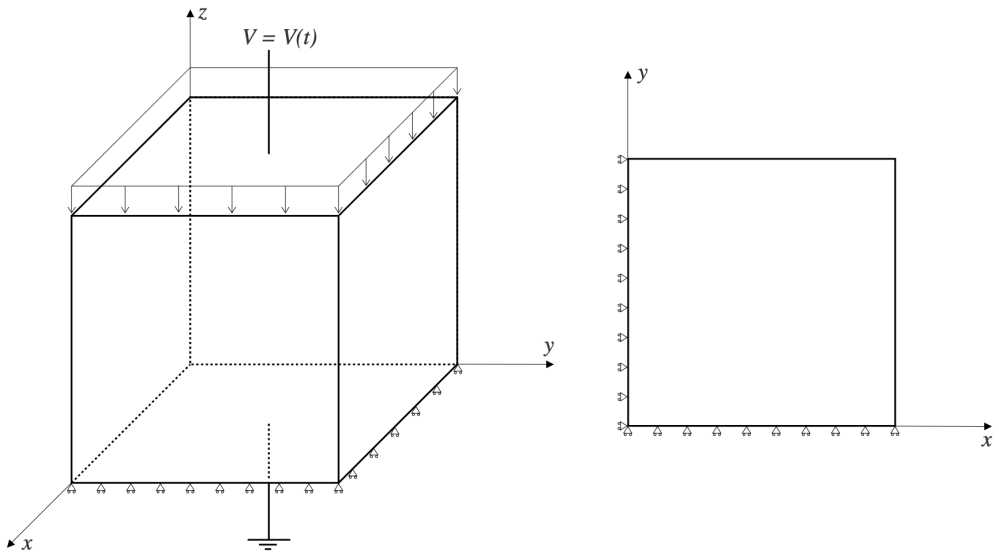


Figure 4.9: A cube with imposed boundary conditions: voltage $V = 0$ at the bottom face and $V = V(t)$ at the top face

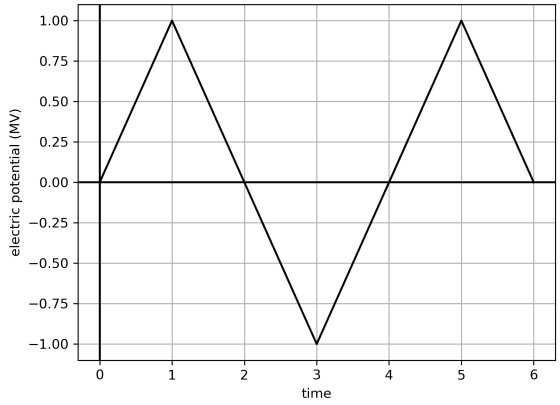
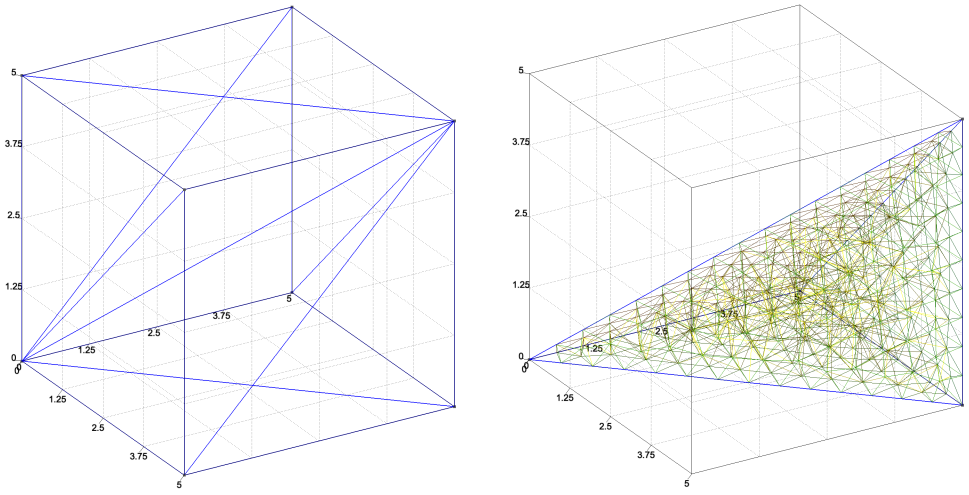


Figure 4.10: Imposed triangular voltage $V = V(t)$ at the top face of the cube

For this example, two multi-scale simulations are executed. One with macro-scale tetrahedral elements, and another with macro-scale hexahedral elements.

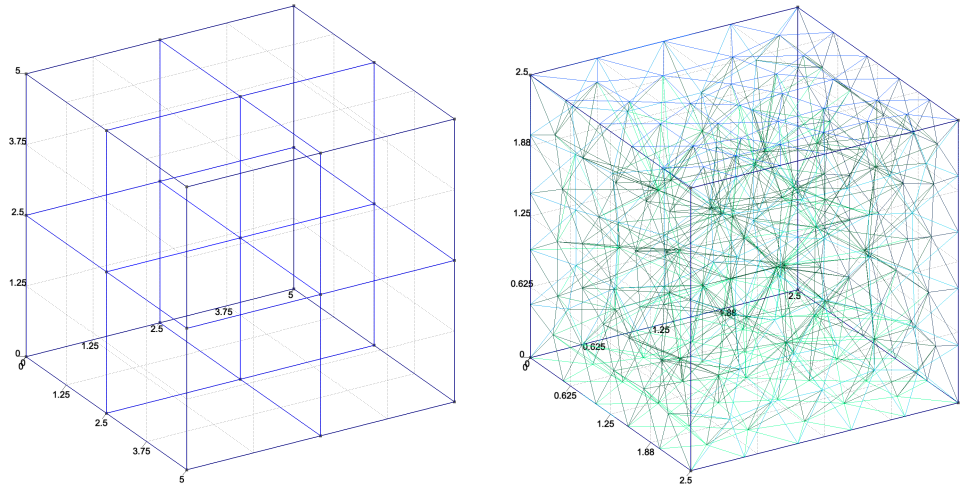
For the first example, six tetrahedral elements are used to define the macro-scale mesh as shown in Figure 4.11a. For each of the tetrahedral elements, a micro-scale mesh composed of Timoshenko beams is defined as shown in Figure 4.11b for one tetrahedral element. Timoshenko beam elements implement the polarization switching model described in Section 4.1.2.



(a) **(b)**

Figure 4.11: (a) Macro-scale mesh consisting of six tetrahedral elements; (b) micro-scale mesh for one tetrahedral element consisting of Timoshenko beam elements

For the second example, eight hexahedral elements are used to define the macro-scale mesh as shown in Figure 4.12a. For each of the hexahedral element, a micro-scale mesh consisting of Timoshenko beam elements is defined as shown in Figure 4.12b.



(a) **(b)**

Figure 4.12: (a) Macro-scale mesh consisting of six tetrahedral elements; (b) micro-scale mesh for one tetrahedral element consisting of Timoshenko beam elements

Finally, a mono-scale mesh is constructed of Timoshenko beams described in Section 4.1.2 (shown in Figure 3.7a). Mono-scale and multi-scale meshes have approximately the same total number of beam elements and the same dimensions,

and therefore should behave in a similar way.

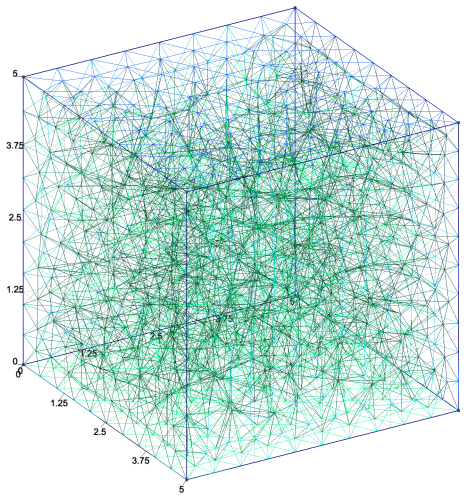
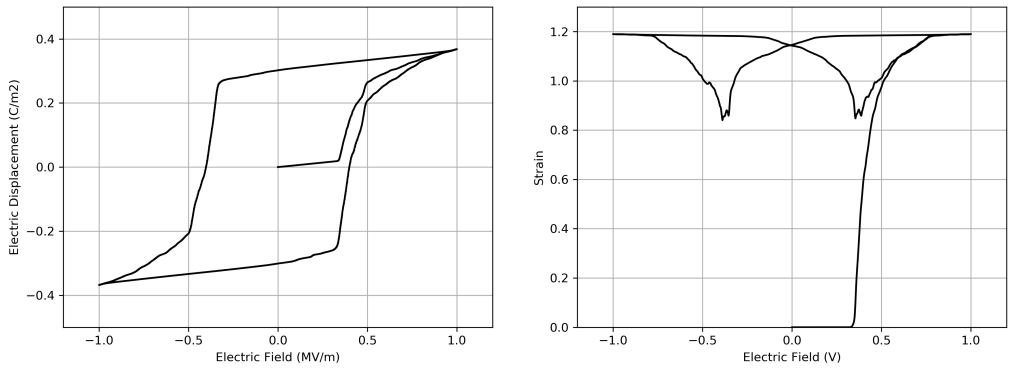


Figure 4.13: Mono-scale mesh consisting of Timoshenko beam elements

For the mono-scale example, the electric and mechanic variables calculated with the beam model have to be expressed in the global frame and averaged with the volume to interpret the macro response of the material using (Moreno-Navarro, 2019)

$$\bar{\xi} = \frac{\int_{\Omega^e} \xi d\Omega}{\int_{\Omega^e} d\Omega} \tag{4.19}$$

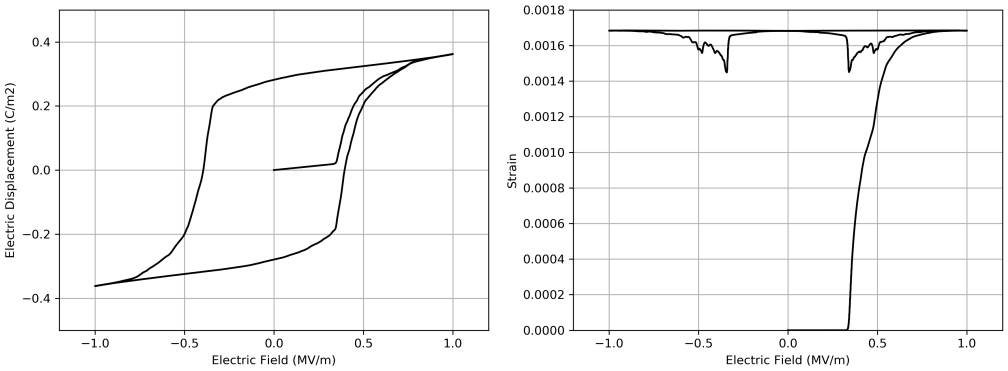
In Figure 4.14, the averaged values of hysteresis loops for vertical electrical displacement and strain obtained for the mono-scale example are shown.



(a) **(b)**

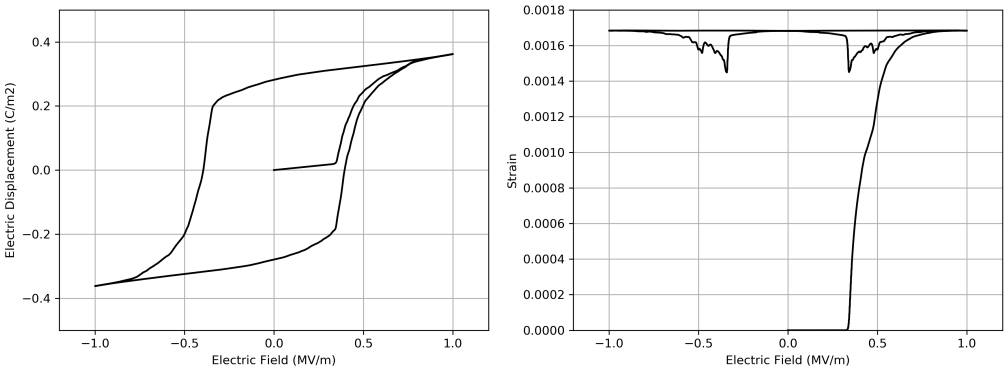
Figure 4.14: Polarization switching mono-scale example hysteresis loops: (a) electric displacement; (b) strain

This results are compared with the results obtained for two multi-scale examples, using hexahedral and tetrahedral macro-scale elements. The hysteresis loops for electrical displacement and strain are shown in Figure 4.15 and Figure 4.16. It can be seen that similar loops are obtained for both macro-scale examples and the mono-scale example. The biggest difference is the values of strain on the hysteresis loop. This is due to the values of strain being approximated for the mono-scale example, while for the multi-scale they are obtained directly on the macro-scale for the whole specimen. The values of strain obtained for the multi-scale examples are closer to the referent values computed in (Hwang et al., 1995).



(a) **(b)**

Figure 4.15: Polarization switching multi-scale example with macro-scale tetrahedral element hysteresis loops: (a) electric displacement; (b) strain



(a) **(b)**

Figure 4.16: Polarization switching multi-scale example with macro-scale hexahedral element hysteresis loops: (a) electric displacement; (b) strain

Superposed hysteresis loops for mono-scale example and both multi-scale examples are shown in Figure 4.17.

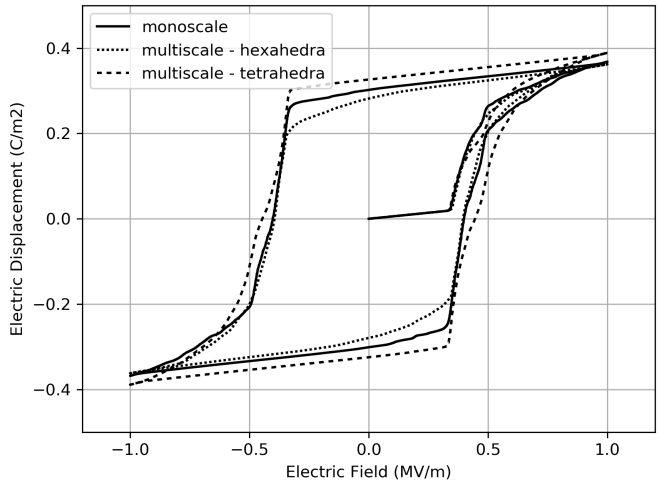


Figure 4.17: Polarization switching example with superposed mono-scale and multi-scale examples - hysteresis loops for the electric displacement

4.2 Multi-scale electrostatics coupling

4.2.1 Multi-scale Hellinger-Reissner formulation for electrostatics

A mixed formulation that is based on the Hellinger-Reissner principle for mechanics is adapted to electrostatics using two independent fields, V (scalar electric potential) and D (electric displacement). It is developed in (Moreno-Navarro et al., 2020) and will be briefly presented in following section.

The formulation is modified and applied in the multi-scale setting. It is used on both macro and micro-scale while the scales are strong coupled, which means that the solution has to be computed simultaneously. As in the previous multi-scale formulations, there are no constitutive equations (in this case, the relation between electric field E and electric displacement D) defined on the macro-scale. This relation has to be obtained on the micro-scale where a complete Hellinger-Reissner formulation for electrostatics with the constitutive equations is defined.

A 27-node isoparametric hexahedral element is defined on the macro-scale, with eight electric potential degrees of freedom v_i^M, $i = 1..8$ defined on the hexahedron vertices, and six degrees of freedom d_i^M, $i = 1..6$ that are associated with the electric displacement D defined on each facet, as shown in Figure 4.18. For this case, a theoretical definition of a 27-node hexahedron is used because it can store the unknowns in the center of the facets. All the other nodes (except the ones defined on the vertices and facets) are ignored and are not used in the finite element computation.

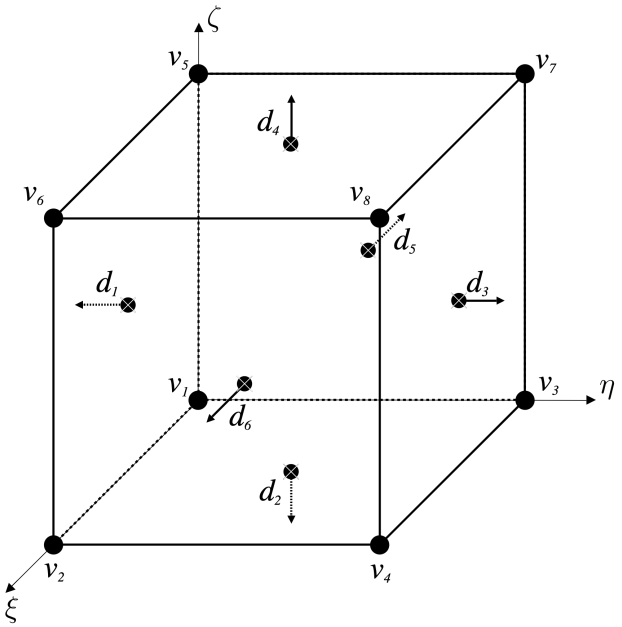


Figure 4.18: Macro-scale hexahedron with eight electric potential degrees of freedom and six degrees of freedom for the electric displacement

On the micro-scale, a 15-node isoparametric tetrahedron is defined with the degrees of freedom defined on the vertices, edges and facets, as shown in Fig 4.21 in the following section. This element is developed in FEAP and presented in detail in (Moreno-Navarro et al., 2020).

The variational formulation and the equations for the total energy which is a sum of macro-scale energy, micro-scale energy and energy on the interface can now be written for electrostatics as

$$\begin{aligned}\Pi(V^M, V^m, d^M, d^m) = \Pi^M(V^M, d^M) + \Pi^m(V^m, d^m) \\ + \Pi^{\Gamma^{Mm}}(V^M, V^m, d^M, d^m)\end{aligned} \tag{4.20}$$

The energy on the interface is defined as

$$\Pi^{\Gamma^{Mm}}(V^M, V^m, d^M, d^M) = \int_{\Gamma^{Mm}} \lambda_V(V^M - V^m) + \lambda_d(d^M - d^m)dA \tag{4.21}$$

where λ_V and λ_d are the Lagrange multipliers for electric potential and electric displacement, respectively, that are used to couple two different scales, and Γ^{Mm} is the interface surface between the scales.

Using the stationarity condition of the potential, $\delta\Pi = 0$, the weak formulation can be obtained for the interface condition and written as

$$\int_{\Gamma^{Mm}} \nu_V(V^M - V^m) + \nu_d(d^M - d^m)dA = 0 \tag{4.22}$$

where v_V and v_d are the variations of the Lagrange multipliers for electric potential and electric displacement, respectively.

Using Whitney elements (as it will be explained more in detail in the following section), electric potential and electric displacement approximation for both macro-scale and micro-scale, and Lagrange multiplier approximations can be written as

$$\begin{aligned}
V^M_{n+1}\Big|_{\Gamma^{Mm}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm}} {}^0\boldsymbol{\mathcal{N}}^M_a(\mathbf{x})v^M_{a,n+1}\\
D^M_{n+1}\Big|_{\Gamma^{Mm}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm}} {}^2\boldsymbol{\mathcal{N}}^M_a(\mathbf{x})d^M_{a,n+1}\\
V^m_{n+1}\Big|_{\Omega^m}(\mathbf{x}^m) &= \sum_{a=1}^{n^m_{el}} {}^0\boldsymbol{\mathcal{N}}^m_a(\mathbf{x})v^m_{a,n+1}\\
D^m_{n+1}\Big|_{\Omega^m}(\mathbf{x}^m) &= \sum_{a=1}^{n^m_{el}} {}^2\boldsymbol{\mathcal{N}}^m_a(\mathbf{x})d^m_{a,n+1}\\
\lambda_{V,n+1}\Big|_{\Gamma^{Mm}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm}} P_{V,a}(\mathbf{x})\beta_{V,a,n+1}\\
\lambda_{d,n+1}\Big|_{\Gamma^{Mm}}(\mathbf{x}^m) &= \sum_{a\in\Gamma^{Mm}} P_{d,a}(\mathbf{x})\beta_{d,a,n+1}
\end{aligned} \tag{4.23}$$

where $v^M_{a,n+1}$ and $v^m_{a,n+1}$ are nodal values of macro-scale and micro-scale electric potential in time step $n+1$, and $d^M_{a,n+1}$ and $d^m_{a,n+1}$ are the macro-scale and micro-scale facet values associated with the electric displacement in time step $n+1$. ${}^0\boldsymbol{\mathcal{N}}$ and ${}^2\boldsymbol{\mathcal{N}}$ are shape functions as defined in the mono-scale formulation in equations (4.36) and (4.37) in the following section. P^M_V and P^M_d are the Lagrange multipliers' interpolation functions, and $\beta_{V,n+1}$ and $\beta_{d,n+1}$ are the nodal values of Lagrange multipliers in time step $n+1$.

By the choice of both Lagrange multipliers to be the Dirac delta function $\delta(x - x_a)$, it should be ensured that the the values of macro-scale and micro-scale electric potential fields coincide at the nodes, while the electric diplacements coincide at the center of each facet on the interface. Keeping this in mind, and introducing the approximations into equation (4.21), the interface term of the central problem for the Hellinger-Reissner electrostatic multi-scale formulation can be written as

$$\begin{aligned}
\mathbf{0} = \mathbf{p}^{M,E}(\mathbf{V}^M_{n+1}, \mathbf{V}^m_{n+1}, \boldsymbol{d}^M_{f,n+1}, \boldsymbol{d}^m_{f,n+1}) =\\
= \int_{\Gamma^{Mm}} P^T_V({}^0\boldsymbol{\mathcal{N}}^{M,E}\mathbf{V}^{M,E} - {}^0\boldsymbol{\mathcal{N}}^m\mathbf{V}^m)dA\\
+ \int_{\Gamma^{Mm}} P^T_d({}^2\boldsymbol{\mathcal{N}}^{M,E^T}\mathbf{d}^{M,E} - {}^2\boldsymbol{\mathcal{N}}^{m,E^T}\mathbf{d}^m)dA
\end{aligned} \tag{4.24}$$

where P^M_V and P^M_d are the Lagrange multipliers' interpolation functions, and ${}^0\boldsymbol{\mathcal{N}}$ and ${}^2\boldsymbol{\mathcal{N}}$ are 0-form and 2-form interpolation functions for the tetrahedral

Whitney's element. Introducing the values of Lagrange multipliers' interpolation of Dirac delta function, it can be obtained

$$\begin{aligned} \bar{\mathbf{V}}^m_{a,n+1}\Big|_{\Gamma^{Mm}} &= \sum_b {}^0\boldsymbol{\mathcal{N}}^{M,E}_b(\mathbf{x}_a)\mathbf{V}^M_{b,n+1} \\ \bar{\mathbf{d}}^m_{f,a,n+1}\Big|_{\Gamma^{Mm}} &= \sum_b {}^2\boldsymbol{\mathcal{N}}^{M,E}_b(\mathbf{x}_f)\cdot n_f\mathbf{d}^M_{b,n+1} \end{aligned} \tag{4.25}$$

where $\mathbf{x}_a$ is the nodal coordinate of the micro-scale node on the interface, $\mathbf{x}_f$ is the center coordinate of the micro-scale facet (calculated as the average of 3 corresponding nodal values) on the interface, and n_f is the unit exterior normal to the macro-scale element facet. $\bar{\mathbf{V}}^m$ are the values of electric potential on the micro-scale interface nodes, and $\bar{\mathbf{d}}^m$ are the values of electric displacement on the micro-scale interface facets. These values can be written in more compact format by using the connectivity matrices

$$\begin{aligned} \bar{\mathbf{V}}^m_a\Big|_{\Gamma^{Mm}} &= \mathbf{T}_{ab}\mathbf{V}^M_b,\ b = 1..4 \\ \bar{\mathbf{d}}^m_a\Big|_{\Gamma^{Mm}} &= \mathbf{R}_{ab}\mathbf{d}^M_b,\ b = 1 \end{aligned} \tag{4.26}$$

Connectivity matrices $\mathbf{T}_{ab}$ and $\mathbf{R}_{ab}$ are based on the particular values of macro-scale shape functions (${}^0\boldsymbol{\mathcal{N}}^M$ and ${}^2\boldsymbol{\mathcal{N}}^M$) which correspond to the interface nodes and interface facets. Namely, the matrices are constructed by introducing the isoparametric coordinates of each micro-scale node on the interface into the macro-scale shape functions ${}^0\boldsymbol{\mathcal{N}}^M$, and by introducing the isoparametric coordinates of each micro-scale facet's center on the interface into the macro-scale shape functions ${}^2\boldsymbol{\mathcal{N}}^M$.

In practice, this translates to the values of the micro-scale electric potential values being calculated as follows

- If the micro-scale node has the same coordinates as the macro-scale node, the value of electric potential is the same as for the macro-scale node;
- If the micro-scale node is on the macro-scale edge, its value is calculated as a linear combination of the values of the electric potential of the two macro-scale nodes that this edge is connecting;
- If the micro-scale node is on the macro-scale face, its value is calculated as a linear combination of the values of the electric potential of the four macro-scale nodes that this face is connecting;

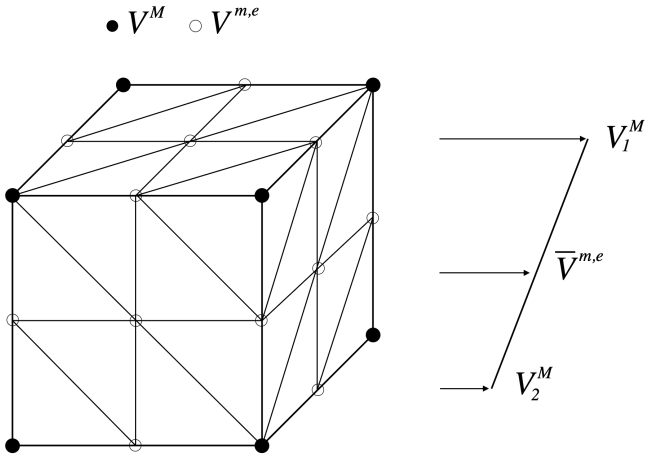


Figure 4.19: Micro-scale electric potential on the interface nodes calculated as a linear interpolation of the macro-scale electric potential nodal values

In a similar way, the implementation of the micro-scale electric displacements' calculations can be described as follows. It is first necessary to compute the areas of the triangles that are the faces of the micro-scale tetrahedral elements on the interface. The value of the micro-scale electric displacement is then a proportional part of the total area of the macro-scale hexahedral element face multiplied by the value of the macro-scale electric displacement on that face. The directions of the micro-scale electric displacements are the same as the macro-scale electric displacement on the corresponding face as shown in Figure 4.20.

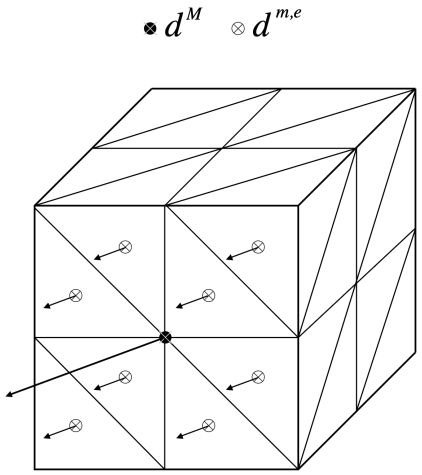


Figure 4.20: Micro-scale electric displacements on the interface facets calculated from the macro-scale electric displacement value, proportional to the micro-scale facets' areas

Finally, after all the values of electric potential and electric displacements are calculated, the system of Hellinger-Reissner equations to be solved on the micro-scale is

$$\begin{bmatrix} \bar{\bar{\mathbf{K}}}^{DD} & \bar{\bar{\mathbf{K}}}^{DV} & \bar{\mathbf{K}}^{DD,T} & \bar{\mathbf{K}}^{DV,T} \\ \bar{\bar{\mathbf{K}}}^{VD} & 0 & \bar{\mathbf{K}}^{VD,T} & 0 \\ \bar{\mathbf{K}}^{DD} & \bar{\mathbf{K}}^{DV} & \mathbf{K}^{DD} & \mathbf{K}^{DV} \\ \bar{\mathbf{K}}^{VD} & 0 & \mathbf{K}^{VD} & 0 \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{d}} \\ \Delta\bar{\mathbf{V}} \\ \Delta\mathbf{d} \\ \Delta\mathbf{V} \end{bmatrix} = - \begin{bmatrix} \bar{\mathbf{r}}_d \\ \bar{\mathbf{r}}_v \\ \mathbf{r}_d \\ \mathbf{r}_v \end{bmatrix} \tag{4.27}$$

The system can be written with the stiffness matrix expressed in a compact form as

$$\begin{bmatrix} \bar{\bar{\mathbf{K}}} & \bar{\mathbf{K}}^T \\ \bar{\mathbf{K}} & \mathbf{K} \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{d}} \\ \Delta\bar{\mathbf{V}} \\ \Delta\mathbf{d} \\ \Delta\mathbf{V} \end{bmatrix} = - \begin{bmatrix} \bar{\mathbf{r}}_d \\ \bar{\mathbf{r}}_v \\ \mathbf{r}_d \\ \mathbf{r}_v \end{bmatrix} \tag{4.28}$$

where each submatrix is defined as in the Hellinger-Reissner formulation for the mono-scale case.

Submatrix $\bar{\bar{\mathbf{K}}}$ is the part of the stiffness matrix related only to interface nodes and facets, $\bar{\mathbf{K}}$ is related to interface nodes and facets in relation to free nodes and facets, and $\mathbf{K}$ is related only to free nodes and facets. In the same way, $\Delta\bar{\mathbf{d}}$ and $\bar{\mathbf{r}}_d$ are the electric displacement increments and residuals of the interface facets, $\Delta\bar{\mathbf{V}}$ and $\bar{\mathbf{r}}_v$ are the electric potential increments and residuals of the interface nodes, $\Delta\mathbf{d}$ and $\mathbf{r}_d$ are the electric displacement increments and residuals of the free facets, $\Delta\mathbf{V}$ and $\mathbf{r}_v$ are the electric potential increments and residuals of the free nodes.

Static condensation can be performed on the previous system of equations defined in (4.28). First, the electric potential and electric displacement field increments of free nodes can be expressed as

$$\begin{bmatrix} \Delta\mathbf{d} \\ \Delta\mathbf{V} \end{bmatrix} = \mathbf{K}^{-1} \left(- \begin{bmatrix} \mathbf{r}_d \\ \mathbf{r}_v \end{bmatrix} - \bar{\mathbf{K}} \begin{bmatrix} \Delta\bar{\mathbf{d}} \\ \Delta\bar{\mathbf{V}} \end{bmatrix} \right) \tag{4.29}$$

Introducing (4.29) to the first equation in (4.28) it is obtained

$$\left(\bar{\bar{\mathbf{K}}} - \bar{\mathbf{K}}^T \mathbf{K}^{-1} \bar{\mathbf{K}} \right) \begin{bmatrix} \Delta\bar{\mathbf{d}} \\ \Delta\bar{\mathbf{V}} \end{bmatrix} = - \begin{bmatrix} \bar{\mathbf{r}}_d \\ \bar{\mathbf{r}}_v \end{bmatrix} + \bar{\mathbf{K}}^T \mathbf{K}^{-1} \begin{bmatrix} \mathbf{r}_d \\ \mathbf{r}_v \end{bmatrix} \tag{4.30}$$

Then, the statically condensed stiffness matrix and residual obtained at the micro-scale can be written as

$$\begin{aligned} \widetilde{\mathbf{K}}^m &= \left(\bar{\bar{\mathbf{K}}} - \bar{\mathbf{K}}^T \mathbf{K}^{-1} \bar{\mathbf{K}} \right) \\ \widetilde{\mathbf{r}}^m &= - \begin{bmatrix} \bar{\mathbf{r}}_d \\ \bar{\mathbf{r}}_v \end{bmatrix} + \bar{\mathbf{K}}^T \mathbf{K}^{-1} \begin{bmatrix} \mathbf{r}_d \\ \mathbf{r}_v \end{bmatrix} \end{aligned} \tag{4.31}$$

After the computations on the micro-scale have converged, the final values of the condensed stiffness matrix and residual are used to compute the values of the stiffness matrix that is going to be used at the macro-scale

$$\begin{aligned}\mathbf{K}_{n+1}^{DD,M,E} &= \mathbf{R}^{E,T}\widetilde{\mathbf{K}}_{n+1}^{m}\mathbf{R}^{E}\\ \mathbf{K}_{n+1}^{DV,M,E} &= \mathbf{R}^{E,T}\widetilde{\mathbf{K}}_{n+1}^{m}\mathbf{T}^{E}\\ \mathbf{K}_{n+1}^{VD,M,E} &= \mathbf{T}^{E,T}\widetilde{\mathbf{K}}_{n+1}^{m}\mathbf{R}^{E}\end{aligned} \tag{4.32}$$

The macro-scale values of residuals are computed in the same manner as

$$\begin{aligned}\mathbf{r}_{d,n+1}^{M,E} &= \mathbf{R}^{E,T}\widetilde{\mathbf{r}}_{n+1}^{m}\\ \mathbf{r}_{v,n+1}^{M,E} &= \mathbf{T}^{E,T}\widetilde{\mathbf{r}}_{n+1}^{m}\end{aligned} \tag{4.33}$$

When the values of the macro-scale stiffness matrix and residual are computed, they are used to update the values of the macro-scale electric potential and electric displacement field. The Hellinger-Reissner finite element system of equations that needs to be solved on the macro-scale is

$$\begin{bmatrix}\mathbf{K}_{n+1}^{DD,M} & \mathbf{K}_{n+1}^{DV,M}\\ \mathbf{K}_{n+1}^{VD,M} & 0\end{bmatrix}\begin{bmatrix}\Delta\mathbf{d}_{n+1}^{M}\\ \Delta\mathbf{V}_{n+1}^{M}\end{bmatrix} = -\begin{bmatrix}\mathbf{r}_{d,n+1}^{M}\\ \mathbf{r}_{v,n+1}^{M}\end{bmatrix} \tag{4.34}$$

The presented formulation is implemented in FEAP multi-scale code, where the main contribution is the development of the macro-scale element. The increased complexity in comparison to the formulations presented in previous chapters arises from the additional nodes on the micro-scale tetrahedal element defined on the faces. During the interface detection procedure and information exchange, it was necessary to detect the type of the node (vertex or facet), to know which degrees of freedom are related to that node on the interface, and finally perform the appropriate interface interpolation (for the electric potential or the electric displacement).

4.2.2 Hellinger-Reissner electrostatics formulation for the micro-scale

On the micro-scale, a Hellinger-Reissner formulation for electrostatics whose discrete approximation is based on Whitney's interpolations for a tetrahedral element is used. It is developed and presented in (Moreno-Navarro et al., 2020), and only the important details of the finite element implementation will be outlined in this section for a better understanding of the multi-scale procedure.

In the variational formulation, the fields $\boldsymbol{V}$ and $\boldsymbol{D}$ are independent. The finite element discrete approximations for Whitney's element are constructed by using

differential forms (i-forms). 0-form are associated with vertices, 1-form with edges, 2-form with faces and 3-form with volume of the tetrahedral Whitney's element, as it can be seen in Figure 4.21. To construct the discrete approximation of a field for electric potential (defined on the vertices) and electric displacement (defined on the faces), the following expressions can be used

$$\begin{aligned} \mathbf{V} &= \sum_{a=1}^{n_v} {}^0\boldsymbol{\mathcal{N}}_a v_a \\ \mathbf{D} &= \sum_{a=1}^{n_f} {}^2\boldsymbol{\mathcal{N}}_a d_a \end{aligned} \tag{4.35}$$

where n_v and n_f are the number of vertices and faces on the element, respectively, and ${}^0\boldsymbol{\mathcal{N}}$ and ${}^2\boldsymbol{\mathcal{N}}$ are interpolation functions for node and facet.

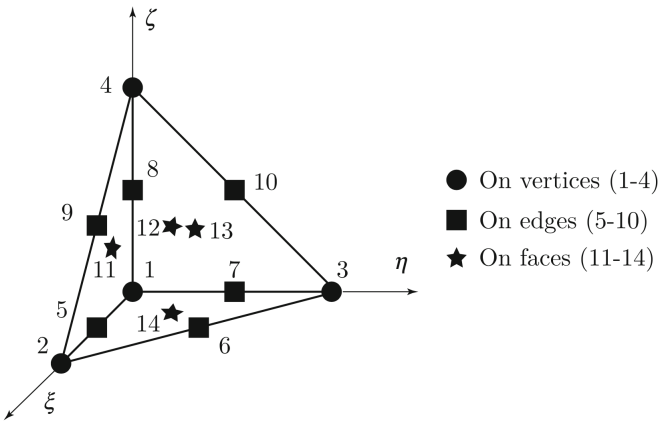


Figure 4.21: Macro-scale hexahedron with eight electric potential degrees of freedom and six degrees of freedom for the electric displacement

The interpolation functions for 0-form are

$$\begin{aligned} {}^0\mathcal{N}_1 &= 1 - \xi - \eta - \zeta \\ {}^0\mathcal{N}_2 &= \xi \\ {}^0\mathcal{N}_3 &= \eta \\ {}^0\mathcal{N}_4 &= \zeta \end{aligned} \tag{4.36}$$

The interpolation functions for 2-form, for facets, are computed as a combination of 0-form and the cross product of the gradients. Facet a is defined with a corresponding set of nodes i, j, k

$${}^2\boldsymbol{\mathcal{N}}_a = {}^2\boldsymbol{\mathcal{N}}_{i\to j\to k} = 2\left({}^0\mathcal{N}_i {}^0\boldsymbol{\mathcal{B}}_j \times {}^0\boldsymbol{\mathcal{B}}_k + {}^0\mathcal{N}_j {}^0\boldsymbol{\mathcal{B}}_k \times {}^0\boldsymbol{\mathcal{B}}_i + {}^0\mathcal{N}_k {}^0\boldsymbol{\mathcal{B}}_i \times {}^0\boldsymbol{\mathcal{B}}_j\right) \tag{4.37}$$

where the gradient ${}^0\boldsymbol{\mathcal{B}}_i$ for the node i is defined as

$$ {}^0\boldsymbol{\mathcal{B}}_i = \mathbf{j}^{-1}\nabla_\xi {}^0\mathcal{N}_i \tag{4.38} $$

where $\mathbf{j}$ is the Jacobian matrix and ∇_ξ are gradients with respect to natural coordinates.

Whitney's element implementation in FEAP is done using a 14-node isoparametric tetrahedron with the approximation for the electric displacement defined as

$$ \mathbf{D} \approx \sum_{i=11}^{14} {}^2\boldsymbol{\mathcal{N}}_i(\mathbf{x}) d_i \tag{4.39} $$

where d_i are the degrees of freedom associated with $\mathbf{D}$ which correspond to the values on the faces of the tetrahedral element. Introducing this approximation into the variational formulation, the residuals related to the electric potential and the electric displacement can be computed. Linearizing this equation, the finite element system to solve can be written as

$$ \boldsymbol{\mathcal{K}} \begin{bmatrix} \boldsymbol{d} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{f} \end{bmatrix} \tag{4.40} $$

where the stiffness matrix $\boldsymbol{\mathcal{K}}$ can be written as

$$ \boldsymbol{\mathcal{K}}_{ab} = \begin{bmatrix} \mathcal{K}^{DD}_{ab} & \mathcal{K}^{DV}_{ab} \\ \mathcal{K}^{VD}_{ab} & 0 \end{bmatrix} \tag{4.41} $$

The sub-matrices can be calculated as

$$ \begin{aligned} \mathcal{K}^{DD}_{ab} &= \int_\Omega {}^2\boldsymbol{\mathcal{N}}_a^T \, \boldsymbol{\epsilon}^{-1} \, {}^2\boldsymbol{\mathcal{N}}_b \, d\Omega \, ; \quad && a, b = 11, 12, 13, 14 \\ \mathcal{K}^{DV}_{ab} &= \int_\Omega {}^2\boldsymbol{\mathcal{N}}_a^T \, {}^0\boldsymbol{\mathcal{B}}_b \, d\Omega \, ; && a = 11, 12, 13, 14; \; b = 1, 2, 3, 4 \, ; \\ \mathcal{K}^{VD}_{ab} &= \int_\Omega {}^0\boldsymbol{\mathcal{B}}_a^T \, {}^2\boldsymbol{\mathcal{N}}_b \, d\Omega \, ; && a = 1, 2, 3, 4; \; b = 11, 12, 13, 14 \end{aligned} \tag{4.42} $$

where $\boldsymbol{\epsilon}$ is the permittivity of the material.

The force vector $\boldsymbol{f}$ for node a is calculated as

$$ f_a = -\int_\Omega {}^0\mathcal{N}_a \rho_q^f d\Omega + \int_{\Gamma_D} {}^0\mathcal{N}_a \bar{D} d\Gamma \, ; \quad a = 1, 2, 3, 4 \tag{4.43} $$

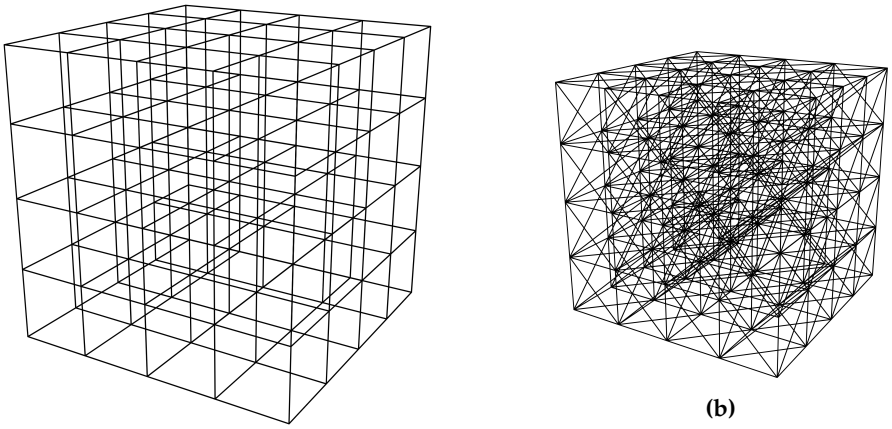
where ρ_q^f is the free electric charge density, and $\bar{D}$ is the electric displacement imposed at the Neumann boundary.

4.2.3 Numerical examples

To test the proposed multi-scale Hellinger-Reissner formulation for electrostatics, a simple validation test example is constructed and compared to the mono-scale computation performed in (Moreno-Navarro et al., 2020). It is assumed that $\rho_q^f = 0$, the electric field is the negative gradient of the scalar electric potential $\boldsymbol{E} = -\Delta V$, and that the constitutive relation for the electric displacement is $\boldsymbol{D} = -\boldsymbol{\epsilon}\Delta V$. The example represents the patch test for electrostatics.

The geometry of the example is a cube with edge length of 0.002 m. At the bottom face, there is an imposed electric potential of 0 V, and at the top face the electric potential is set to 20 V. The electric displacement is not allowed to flow in and out of the lateral faces of the cube. The value of the permittivity $\boldsymbol{\epsilon}$ is 15×10^{-12} F/m.

On the macro-scale, the previously described hexahedral element is used. The mesh is composed of 4x4x4 macro-scale elements, as shown in Figure 4.22a. On the micro-scale, there is a Whitney tetrahedral element with Hellinger-Reissner electrostatics formulation. For each of the macro-scale elements, a micro-scale mesh is defined which consists of 384 tetrahedral elements, as shown in Figure 4.22b.



(a) **(b)**

Figure 4.22: (a) Macro-scale mesh of 64 hexahedral elements; (b) Micro-scale mesh of 384 tetrahedral elements that fits inside a macro-scale element

After performing the computations, it can be seen that the distribution of electric potential is linear between the values of 0 V and 20 V. The value of electric displacement in z-direction is homogeneous, with a value of -1.5×10^{-7} C/m^2. The results are in agreement with both analytical and mono-scale computations obtained in (Moreno-Navarro et al., 2020). The same value of the electric displacement in direction z is obtained by the analytical solution as

$$D_z = -\epsilon \frac{V_z}{l_z} = -15 \cdot 10^{-12} \cdot \frac{20}{0.002} = -1.5 \cdot 10^{-7} \tag{4.44}$$

where l_z is the length of the cube edge in direction z.

In Figure 4.23 the linear distribution of the electric potential obtained on the macro-scale is shown. The same results are obtained for the mono-scale computation shown in Figure 4.24a.

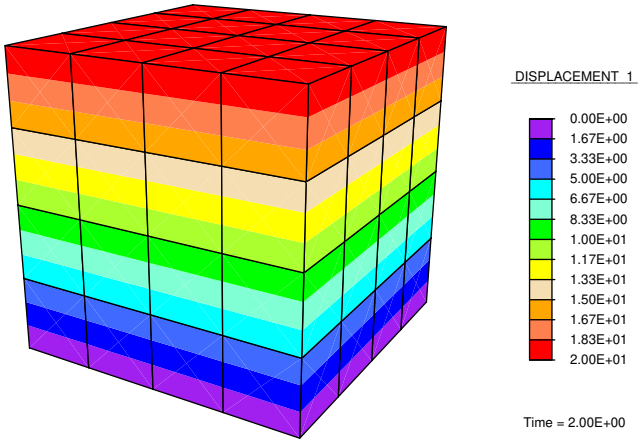


Figure 4.23: Multi-scale results for the macro-scale - linear distribution of the electric potential

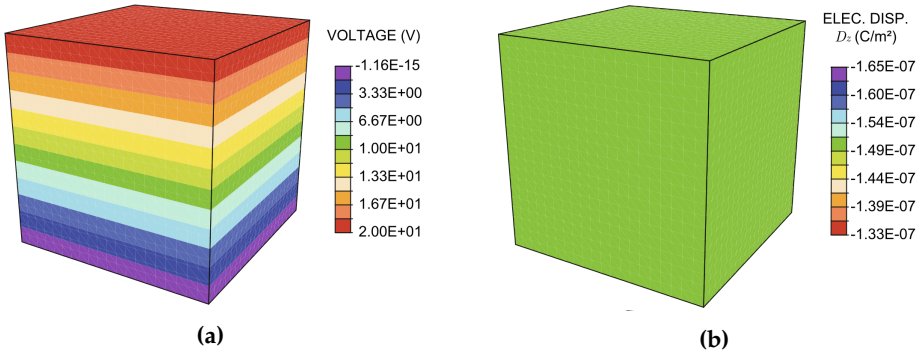
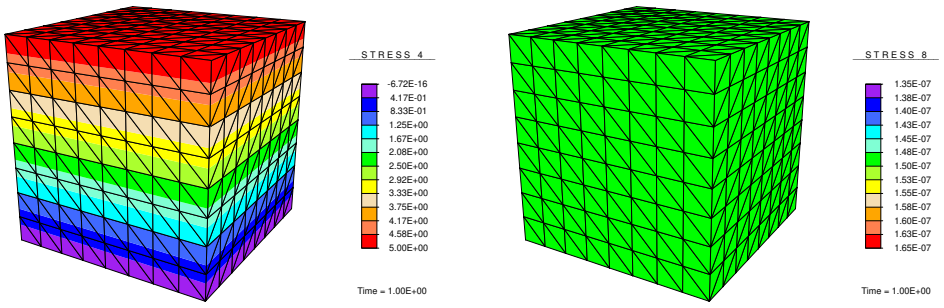


Figure 4.24: Mono-scale results (a) electric potential distribution; (b) electric displacement (Moreno-Navarro et al., 2020)

The value of the electric displacement is constant as shown in Figure 4.25b for micro-scale mesh defined for one macro-scale element. The value of the electric displacement is the same for micro-scale computations for every macro-scale element, so they are not shown here.

It can be seen in Figure 4.25a that the values of the electric potential go from 0 V to 5 V. This is due to the fact that the micro-scale mesh is taken for the macro-scale element positioned in the bottom row of the macro-scale mesh. Others micro-scale meshes also give linear distribution of the electric potential,

but the values go from 5 V to 10 V, 10 V to 15 V, or 15 V to 20 V, depending on their position in the macro-scale mesh.



(a) **(b)**

Figure 4.25: Multi-scale results for the micro-scale computation for one macro-scale element (positioned in the bottom row of the macro-scale mesh) (a) electric potential distribution; (b) electric displacement

Conclusion

By introducing the voltage on the macro-scale as an additional degree of freedom, the multi-scale formulation has been successfully extended to a multi-physics setting, and has been shown to work for the electro-mechanical coupling. Macro-scale tetrahedral and hexahedral elements have been developed. It has been shown that the multi-scale procedure with added voltage on the macro-scale produces the same results for the simple piezoelectric effect and polarization switching model as the mono-scale procedure. Without any macro-scale constitutive relation or direct coupling between the electrical and mechanical field, the macro-scale element was able to capture the micro-scale behavior. Numerical examples are executed with the previously developed Timoshenko beam finite element with implemented polarization switching model used on the micro-scale. Also, Hellinger-Reissner formulation for electrostatics has been applied in the multi-scale settings. Two degrees of freedom, scalar electric potential and electric displacement, are defined both on the macro and micro-scale, and used to couple the scales on the interface. Whitney's elements that are based on the differential forms, which are previously developed, are used on the micro-scale. Numerical example is executed and compared with the monoscale computations. This is the first step in extending the multi-scale procedure for different multi-physics couplings, like electro-magneto-mechanical or thermo-mechanical coupling.

5 Software implementation

Contents

In this chapter, software implementation of the multi-scale and multi-physics codes is presented. In the first section, FEAP software used for multi-scale coupling is described. It includes the description of the standard FEAP code and solution procedure, the differences introduced by the multi-scale FEAP code, and the macro-scale and micro-scale FEAP components. In the second section, a multi-physics fluid-structure code coupling is presented, with the description of OpenFOAM, the software components used for fluid mechanics and structural mechanics, and also the control software component used for the interface coupling and execution order. In the last section, CTL library, which is used for both code couplings, is described

The coupling of different scientific codes is a subject of many recent research papers. The complexity of such couplings lies in the need of understanding the architecture of the existing codes and the knowledge of the physics behind them. In this thesis, a multi-scale coupling is implemented using FEAP software code that is used for both macro and micro-scale. The details of this implementation are presented in section 5.1.

In general, such concept of coupling can be used and implemented using more than one software code. In the scope of this thesis, several such codes are examined, like OpenFoam (Jasak et al., 2007) and GetDP (Dular and Geuzaine, 2013). A fluid-structure interaction coupling implementation with OpenFOAM and FEAP developed in (Kassiotis et al., 2011b) is executed, and possibilities for its improvements are studied. This implementation for fluid-structure interaction, that can also be applied to other multi-physics simulations using different codes, is explained in section 5.2.

Scientific software code for a specific problem is usually developed by domain experts and software developers with deep understanding of the physical phenomena behind it, which means it should give optimal results for this domain. It uses a particular discretization technique, method, programming language, or any other choice that solves this problem in optimal way. To modify and reuse this code for a partitioned approach where the domains can be solved separately is a complex and challenging task.

The main idea behind the multi-physics code coupling is that both the theoretical formulation that describe the system and the component based software development are divided in three parts: (Matthies et al., 2006)

- Software component that solves the first domain;
- Software component that solves the second domain;
- Software compoment for coupling between the domains on the interface and data exchange.

This approach consists of modifying the existing software codes and/or writing additional components that encapsulate the behavior of such a code, and offer an interface that can be used to interact with it. The interface coupling software component has to be developed in order to exchange the data, and make data interpolations or other data manipulations. It can be also used to control the execution order of components for each domain. This makes it the entry point of the execution, the component that is run first and which stores the logic of when other components should be called.

This approach makes software components loosely coupled which means that they are not aware of the implementation details of each other and they communicate only through the interface. To allow these software components to communicate, a communication protocol is needed. The communication protocol defines a set of rules and its software implementation allows the data transmission between the components. The diagram of such an architecture is shown in Figure 5.1. This architecture is used in the fluid-structure interaction problem from (Kassiotis et al., 2011b) and is presented in section 5.2.

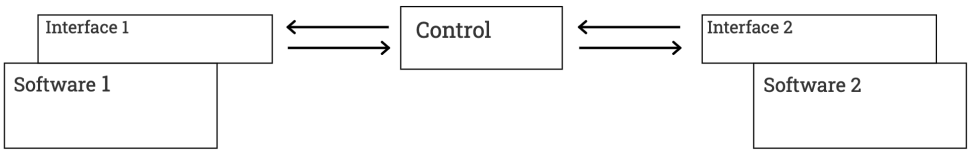


Figure 5.1: Software component architecture for code coupling with control component

The interface coupling software component and the logic of execution order can also be stored in one of the software components. In this case, that software component is a master code, which sends requests and receives data from other software components. This type of architecture is shown in Figure 5.2. It is used in multi-scale software coupling of FEAP code implemented in this thesis and explained more in detail in the following section.

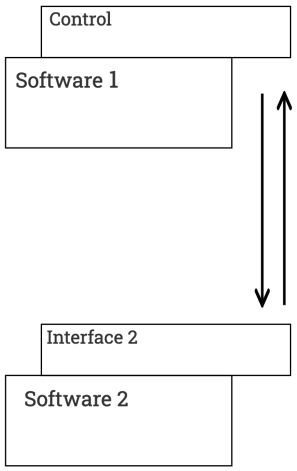


Figure 5.2: Software component architecture for code coupling with master code

5.1 FEAP software code coupling

The details of the software implementation are presented here for better understanding of the multi-scale solution procedure and to give an overview of the implemented software code architecture. Although a technical aspect, it is a key point for obtaining the results and proving that the multi-scale formulation is valid. It can be used as a starting point for possible future developments, like implementing new material models or including different physics. The code snippets are not included, as it would be too long and specific, so it would not contribute to the overall clarity.

5.1.1 FEAP

Before showing how the multi-scale FEAP implementation is working, a standard FEAP code and solution procedure are described. Finite Element Analysis Program (FEAP) (see (Zienkiewicz et al., 2005)) is a general purpose program to solve problems using the finite element method, mostly in solid mechanics but can be extended to other problems that can be solved using partial differential equations. It is written in Fortran programming language and its source code can be easily modified (Taylor, 2014).

A model of a problem to solve in FEAP is defined in a textual input file that contains all the necessary steps to perform a finite element analysis. In the input file, the problem is defined with a mesh specification and a problem solution algorithm. The mesh specification consists of a definition of spatial coordinates, elements, material definition for all the elements, and boundary conditions. The problem solution algorithm defines how the finite element analysis is going to be performed. Namely, it has commands to define the time step increment, to compute the residuals for the governing equations, to loop to repeat the execution of solution commands, and to compute the tangent matrix or solve the set of partial differential equations. Each command defined in the input file translates to a procedure call inside FEAP. To illustrate this, a flowchart diagram for a typical problem executed in FEAP is shown in Figure 5.3, and a typical input file is shown in Figure 5.4.

The standard FEAP code is taken and modified to work for both macro and micro-scale solution procedures. In the micro-scale code, a CTL is used to implement the interface in order to allow to communicate and exchange the information between the two codes. CTL is also used to implement the coupling of the two codes and execute them in parallel.

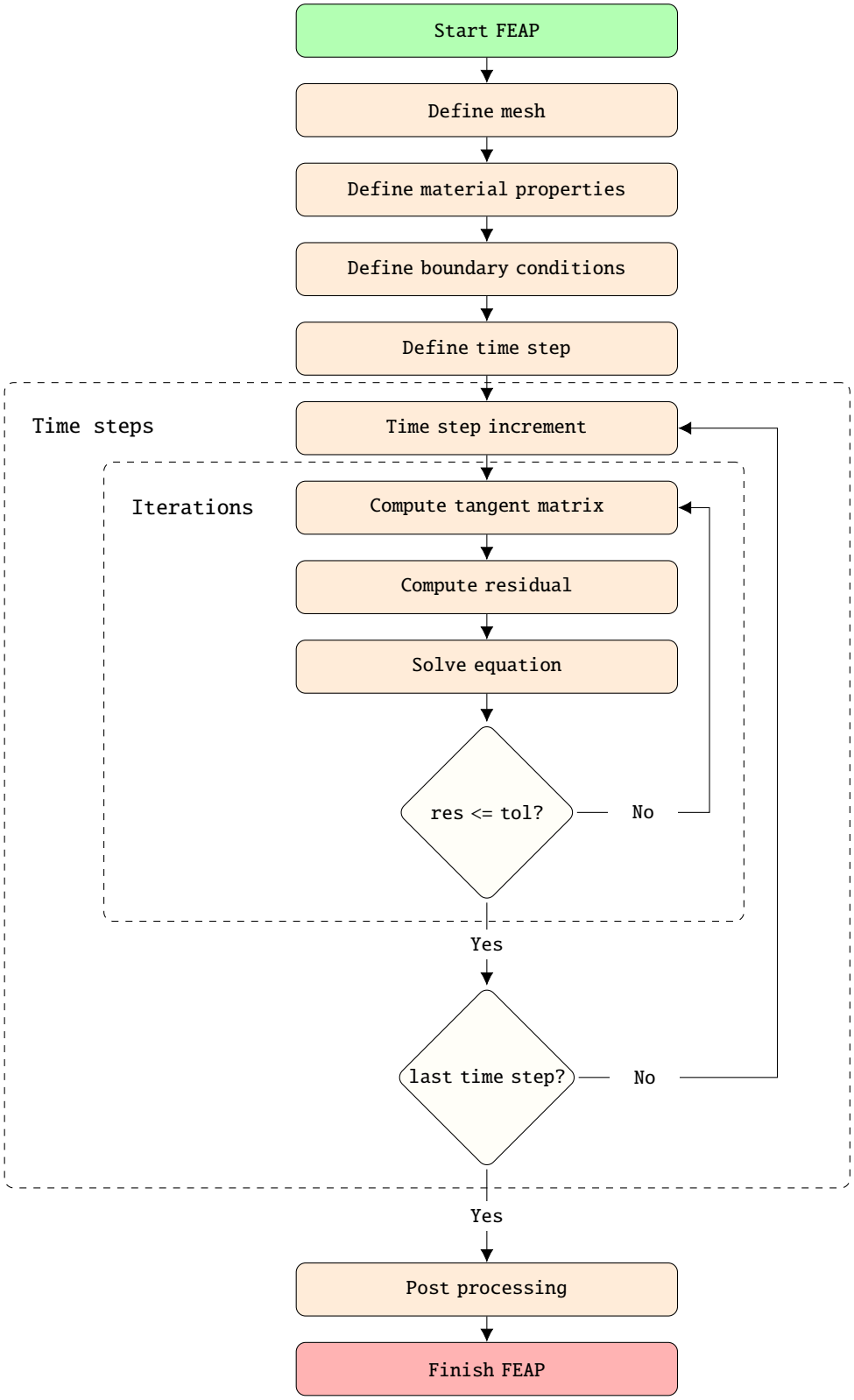


Figure 5.3: Flowchart for standard FEAP problem execution

```
FEAP
0, 0, 0, 2, 2, 4
! node num, elem num, mat num, dim, dof/node, node/elem

COORdinates  ! define nodal coordinates
...

ELEMent  ! define elements
...

BOUNdary  ! define boundary restraint conditions
...

DISPlacements  ! define nodal boundary displacements
...

MATErial 1  ! define material properties
SOLId
ELAStic ISOTropic 38e3 0.18
PLANe STRAin

END

! define problem solution algorithm
BATCh
dt,,0.1 ! define time step
loop,,100 ! time step loop
time  ! time step increment
loop,,20  ! iteration loop
tang  ! compute global tangent matrix
form  ! compute global residual
solv  ! solve equation
next
next
end

STOP  ! finish FEAP

```

Figure 5.4: Standard FEAP input file example

5.1.2 Multi-scale FEAP

For implementing the proposed multi-scale procedure, the Finite Element Analysis Program (FEAP) is used. It is chosen because different material behaviors can be easily implemented by changing the existing code. To simulate the behavior on both the macro and the micro-scale, two different versions of FEAP code are implemented, as explained in (Niekamp et al., 2009). macroFEAP is used on the macro-scale, and its behavior is different from the standard version in the sense that it can initiate the execution of microFEAP instances and use the obtained results. It is used as a master code that initiates all the actions and communication between the codes, as shown in the Figure 5.5.

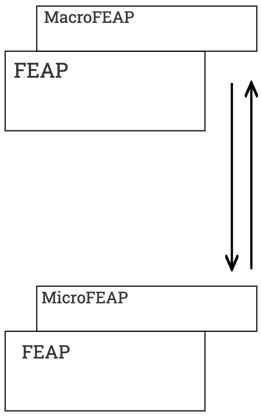


Figure 5.5: Software component architecture for code coupling with macroFEAP acting as a master code

microFEAP was modified in a way that it implements an interface that can take requests and its execution can be controlled from the macroFEAP code. For coupling the two codes, developing the interfaces and allowing them to exchange the information between them, Component Template Library (CTL) is used (see (Niekamp et al., 2014)).

This code is developed based on the work of Damijan Markovic (Markovic, 2004). In this thesis, the existing code was adapted, and mostly rewritten, to work for the latest version of FEAP (version 8.4). This was important because all the micro-scale elements that are developed in the scope of this thesis or used from others' work are developed for this version of FEAP, and therefore could not work with the old multi-scale code. Also, the multi-scale code is modified and partly rewritten in a way that it can work with the new version of CTL, and compiled on a modern Ubuntu operating system with the latest versions of Fortran and C++ compilers.

To solve a multi-scale problem example, one instance of macroFEAP process and n^M_{elem} instances of microFEAP processes are created, as shown in Figure 5.6, where n^M_{elem} is the number of elements in the macro mesh. microFEAP instances are executed in parallel, as they do not need any communication between them, since they are not using the same data. Input data for the macro-scale consists of defining the macro mesh and boundary conditions. The constitutive properties of the material do not have to be defined, as they will be obtained from the micro-scale computations. On the micro-scale, the mesh has to be defined, but no boundary conditions, as those are automatically imposed by the multi-scale solution procedure.

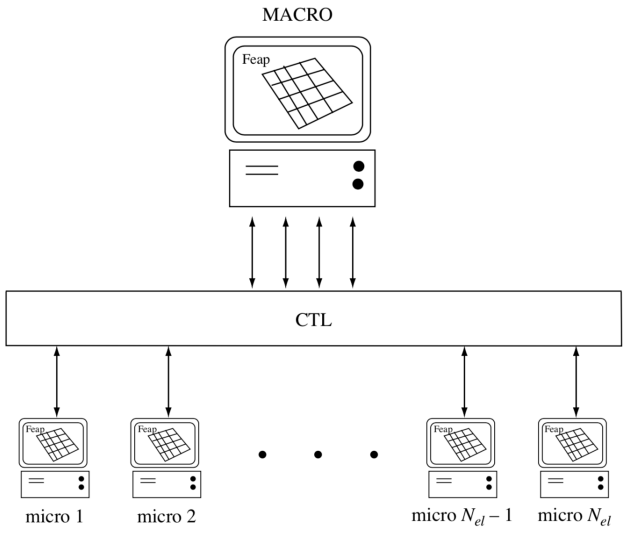


Figure 5.6: Parallel execution and code coupling with CTL (according to (Niekamp et al., 2009))

In each time step, the macroFEAP instance transfers the nodal displacement field and displacement discontinuity field for each macro-scale element to the corresponding microFEAP instance using CTL. Based on that input, microFEAP instances solve the imposed problem in parallel. After obtaining the final values of the stiffness matrix and residual, they transfer them back to macroFEAP. Only when all microFEAP instances finished their execution and the results are transfered, macroFEAP can continue with its execution.

The flowchart diagram of execution for a multi-scale problem is shown in Figure 5.7. Starting activities that initiate each instance of FEAP are colored in green. The activities that are a part of the standard FEAP code, and are slightly modified or not modified at all, are colored in yellow. Newly developed procedures are shown in blue. Everything that is related to the communication between the scales and is controlled by CTL is shown in gray. Finally, end points of each procedure is shown in red.

5.1.3 macroFEAP

macroFEAP is a modified version of FEAP code that is used for macro-scale analysis. After defining the macro-scale mesh, it initializes an instance of microFEAP for every macro-scale element. Instead of computing the tangent matrix and residual, it calls the micro-scale computations and obtains the values from there. After the micro-scale values are obtained, it continues the standard FEAP finite element procedure by solving the equations and computing displacement increments.

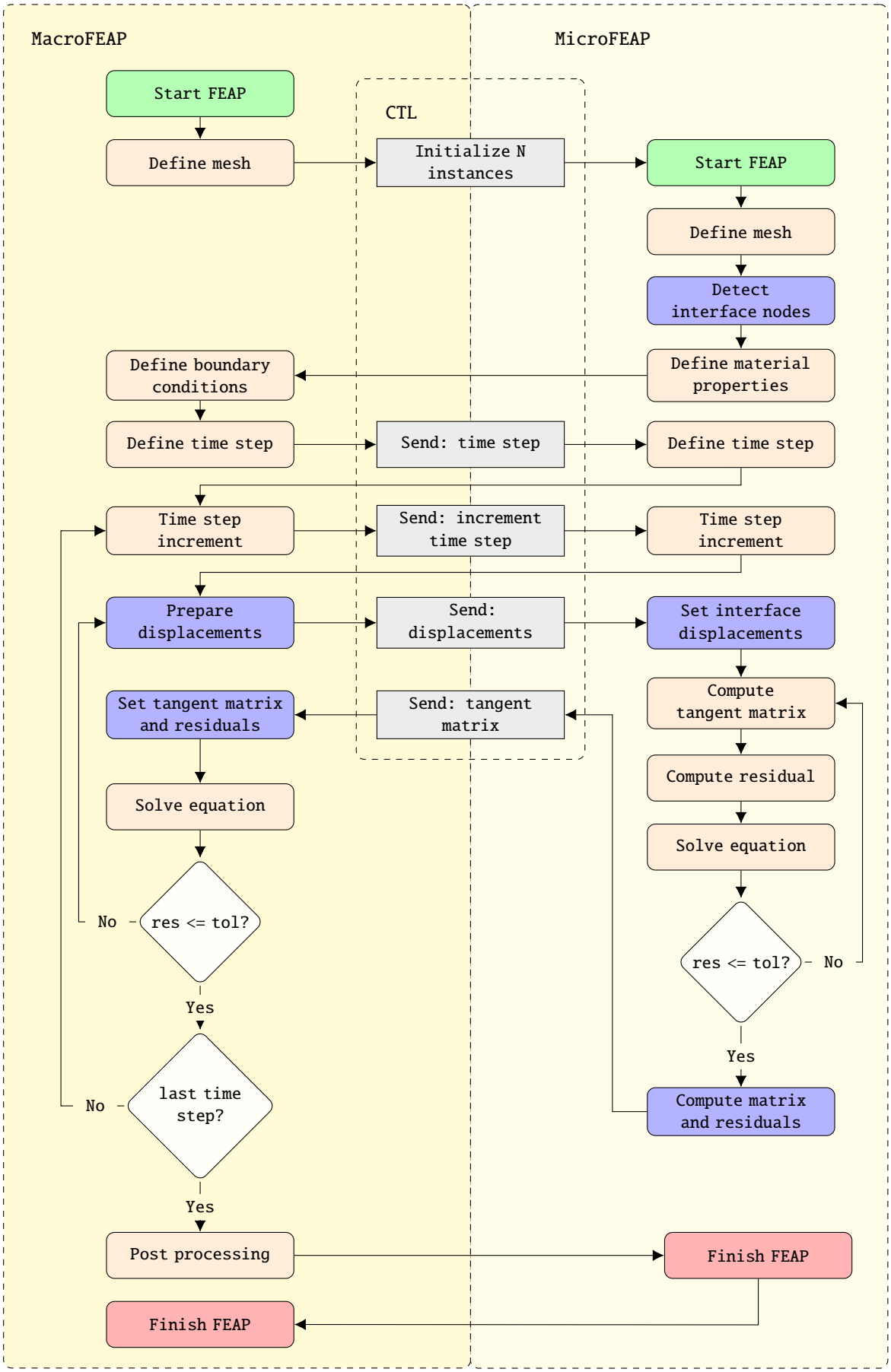


Figure 5.7: Flowchart for multi-scale FEAP problem execution

A few additional input file commands are developed to be able to control the course of execution. Namely, an input file commands for calling the procedures for defining the micro-scale time step (*sznj,tim,0.1*), to increment the micro-scale time step (*sznj,tim*), and to set the displacements and run the micro-scale computations (*sznj*), as it can be seen in detail in Figure 5.8.

```
FEAP
0, 0, 0, 2, 2, 4
! node num, elem num, mat num, dim, dof/node, node/elem

COORdinates  ! define nodal coordinates
...

ELEMent  ! define elements
...

BOUNdary  ! define boundary restraint conditions
...

DISPlacements  ! define nodal boundary displacements
...

! define user element that is going to be used
! instead of the material properties definition
MATErial 1
  USER 1

END

! define problem solution algorithm
BATCh
  dt,,0.1 ! define time step
  sznj,tim,0.1   ! define and set micro-scale time step
  loop,,100 ! time step loop
    time  ! time step increment
    sznj,tim    ! micro-scale time step increment
    loop,,20  ! iteration loop
      sznj  ! set the displacements and run the micro-scale
    computations
      tang  ! compute global tangent matrix (assembled from the micro-
    scale local tangent matrices)
      form  ! compute global residual (assembled from the micro-scale
    local residual)
      solv  ! solve equation
    next
  next
end

STOP  ! finish FEAP
```

Figure 5.8: Macro-scale FEAP input file example

5.1.4 microFEAP

microFEAP is a modified version of FEAP code that is used for micro-scale analysis. It can be run only programmatically and the only way to interact with it is through the CTL defined interface. In Figure 5.9, the interface function definitions are shown.

```
#ifndef _SIMURI_H_
#define _SIMURI_H_

#include <ctl.h>

# define CTL_ClassTmpl simuRI , (scalar1), 1
# include CTL_ClassBegin

// create simu with "filename" as starter file
# define CTL_Constructor1  (const string /*filename*/), 1
# define CTL_Method1 void, init, (const string /*filename*/), 1
// return parameter
# define CTL_Method2 array<scalar1> , getparam, (), 0
// set parameter p
# define CTL_Method3 void, setparam, (const array<scalar1> /*p*/),
1
// return primary variables
#   define CTL_Method4 array<scalar1>, getstate, () const, 0
// set primary variables to x
#   define CTL_Method5 void, setstate, (const array<scalar1> /*x*/)
, 1
// set load of system
#   define CTL_Method6 void, setload, (const array<scalar1> /*load
*/), 1
// get coupling indices i and values y (boundary cond. or load or
...)
#   define CTL_Method7 void, getcoupling, (array<int4> /*i*/, array
<scalar1> /*y*/) const, 2
// set coupling indices i
#   define CTL_Method8 void, setcoupling, (const array<int4> /*i*/)
, 1
// return the residuum at given state
#   define CTL_Method9 void, residual, (array<scalar1>/*r*/) const,
 1
// solve with given param, load, coupling values with at least
accuracy
// and set new state; write the new state into x, and additional
rhs
#   define CTL_Method10 array<scalar1>, solve, (const scalar1 /*
accuracy*/, const array<scalar1> /*rhs*/), 2
# include CTL_ClassEnd

#endif


```

Figure 5.9: microFEAP interface definition using CTL - *simuri.h*

To bind the interface to the function implementations, a generic CTL header function for Fortran is used, as shown in Figure 5.10. When the interface is bound in this way, a function names has to follow the specific template *simu_<function name>_impl* as it can be seen in a function implementation example in Figure 5.11.

```
#define CTL_ConnectF

#define CTL_ClassPrefix simu
typedef double scalar1;
#include <simuri.h>

void CTL_connect(){
  ctl::connectF<simuRI<double>, ctl::Extern::simuRI>();
}

```

Figure 5.10: microFEAP binding CTL interface to function implementation - *service.cpp*

An interface function implementation is written in Fortran code and it follows the function declaration (function name, return type and parameters). As this code is compile with FEAP source code, it has access to all internal values and functions, so it can be used, as in example from Figure 5.11, for setting the displacements of the micro-scale nodes on the interface.

```
subroutine simu_setstate_impl(s0,x0)
implicit   none
include   'cdata.h'
include   'comblk.h'
include   'iofile.h'
include   'pointer.h'
include   'sdata.h'
include   'inc/inout.h'

integer*8  s0, i,j
real*8     x0(s0)
real*8     sub(4)
integer    num

c.....setstate for 2D Q4 element
if (s0 .eq. 9) then
  stps  = 2 ! dofs
  stvoz = 4 ! nodes
  num = stps * stvoz + 1 ! 4nodes x 2dofs + alpha
  do i = 1,stps*stvoz
    makpom(i) = x0(i)
  end do
  makpom(9) = x0(9)  ! set alpha
  call pcontrbatch()

c.....setstate for 3D brick element with 3 dofs (or 3 dofs + alpha)
elseif (s0 .eq. 24 .or. s0 .eq. 26) then
  stps  = 3
  stvoz = 8 ! nodes for macro brick element
  num = stps * stvoz ! 8nodes x 3dofs

  do i = 1,stps*stvoz
    makpom(i) = x0(i)
  end do

  if (s0 .eq. 26) then
    makpom(25) = x0(25)  ! set alpha
  end if

  call pcontrbatch()

...

end


```

Figure 5.11: microFEAP function implementation - *simu_setstate_impl.f*

The boundary conditions for the micro-scale problem are set from the code, and not defined in the input file. For the displacement based interface, it sets all the displacements obtained on the macro-scale and defines boundary restraint conditions all over the interface nodes. Additionally, it computes all the tangent matrices (**K**, **H**, **F**) and residuals (**r**, **h**) needed for the macro-scale analysis and exports them back to the macro-scale. This can be seen in the multi-scale flowchart

diagram in Figure 5.7, and the micro-scale input file example shown in Figure 5.12.

```
FEAP
0, 0, 0, 2, 2, 3
! node num, elem num, mat num, dim, dof/node, node/elem

COORdinates  ! define nodal coordinates
...

ELEMent  ! define elements
...

MATErial 1  ! define material properties
  SOLId
    ELAStic ISOTropic 38e3 0.18
    PLANe STRAin

END

batch
  mesh vert.dat ! detect interface nodes
end

! define problem solution algorithm
BATCh
  zace  ! allocate memory for tangent matrix and residual
  inpu  ! get macro-scale displacement values
  loop,,20  ! iteration loop
    tang  ! compute tangent matrix
    form  ! compute residual
    solv  ! solve equation
  next
  expo  ! export tangent matrix and residual to macro-scale
end

STOP  ! finish FEAP
```

Figure 5.12: Micro-scale FEAP input file example

While computing the tangent matrices and micro-scale displacement field interpolations on the interface, it uses the transformation matrices $\mathbf{T}$ and $\mathbf{S}$. Generation and usage of transformation matrices are simplified not to store all the 0 values and full dimensions of the matrices. Only non-zero values are calculated in the software code and used to obtain the value of the micro-scale displacements, and stiffness matrix and residuals for transfer them back to the macro-scale.

5.2 OpenFOAM and FEAP software code coupling

For the software implementation of the fluid structure interaction problem, OpenFOAM (version 1.7.1) is used for fluid mechanics, and FEAP (version 8.2) for structural mechanics. The source code of both softwares is open, so they can be used and modified for the purpose of a partitioned solving of fluid-structure interaction problems. For the coupling of these softwares, Component Template Library (CTL) is used. It allows to use different programming languages, with FEAP written in Fortran and OpenFOAM written in C++. Implementation details of this coupling and the results obtained are presented in (Kassiotis et al., 2011a) and (Kassiotis et al., 2011b). Several software components were developed to implement the coupling:

- ofoam - interface component based on OpenFoam, that is used to solve fluid domain
- coFEAP - interface component based on FEAP, that is used to solve solid domain
- cops - software component for coupling between the ofoam and coFEAP and data exchange

and the diagram how they interact between each other is shown in Figure 5.13. Detail of the procedure solution algorithm is shown in Figure 5.15 and the implementation details for each component is outlined in the following sections.

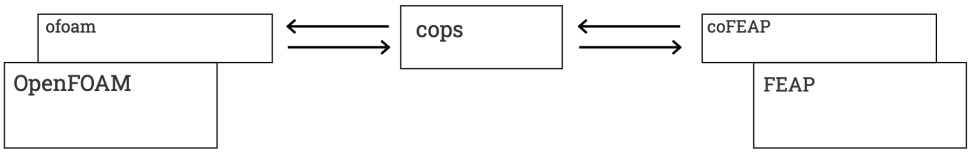


Figure 5.13: Software component architecture for code coupling (ofoam and coFEAP) with control component (cops) (Kassiotis et al., 2011b)

The fluid-structure interaction software coupling was tested on a lid-driven cavity flow problem with a flexible bottom in 2D in a square domain, as shown in Figure 5.14. This problem is often used as a validating example for computational fluid dynamics codes. A control component calls each of the software codes for solving the corresponding domain, exchanges the information between them and finds the final solution. It transfers the displacement field from the solid to the fluid domain and the forces from the fluid to the solid domain. Interpolations also need to be made over the values calculated on the interface, and for that an interpolation function based on the radial basis function is used.

The main challenge is to exchange the information consistently and accurately on the fluid-structure interface. In the FSI framework, a classical way to couple the fluid and structure domains consists in imposing the continuity of stress and displacement (or, equivalently, velocity) at the interface. In order to ensure

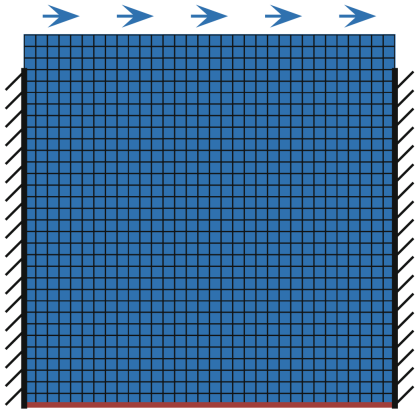


Figure 5.14: Lid-driven cavity flow problem with a flexible bottom (Kassiotis et al., 2011a)

stability of the coupling algorithm, it is important to ensure the correct energy transfer at the interface (Lombardi et al., 2013). This has been achieved in (Kassiotis et al., 2011b) by using radial basis function for mesh motion and interpolation of numerical values on the interface grid. With this approach, if a constant normal stress is acting on the fluid interface, the stress imposed on the structure side of the interface is not constant (Lombardi et al., 2013). In this thesis, the mortar element method was considered to achieve the conservation of stresses and displacements on the interface. With this approach, a better quality solution could be obtained with the main focus being to guarantee stability, computational efficiency and robustness (Rukavina and Ibrahimbegovic, 2017).

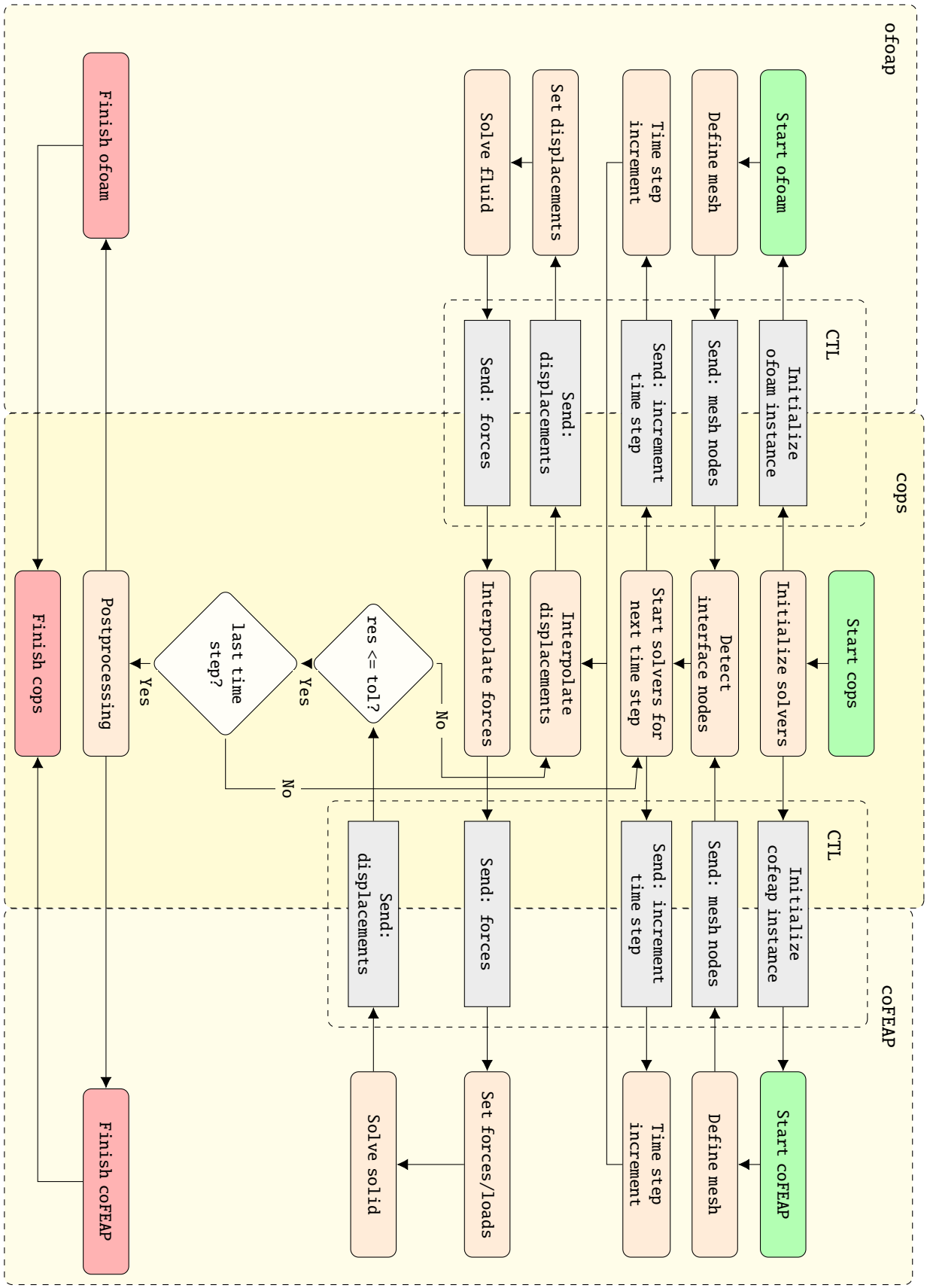


Figure 5.15: Flowchart for multi-physics FSI problem execution

5.2.1 OpenFOAM

OpenFOAM is an object-oriented software code and library for Computational Fluid Dynamics (CFD) and structural analysis (Jasak et al., 2007).

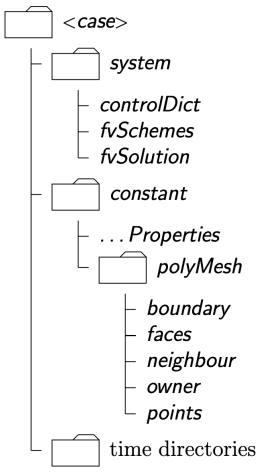


Figure 5.16: OpenFOAM input file directory (Jasak et al., 2007)

OpenFOAM input files for a specific problem are divided into several folders as shown in Figure 5.16. A problem is defined by its mesh and solution procedure. The folder *system* consists setting parameters associated with the solution procedure as

- *controlDict* - run control parameters are set including start/end time, time step and parameters for data output
- *fvSchemes* - discretisation schemes used in the solution may be selected at run-time
- *fvSolution* - the equation solvers, tolerances and other algorithm controls are set for the run

The folder *constant* contains a full description of the mesh in a subdirectory *polyMesh* and files specifying physical properties (Jasak et al., 2007). The *time directories* are a set of folders containing individual files of data for particular fields for each simulated time step, names as a time step value. The same input file format and directories are used for the multi-physics code example.

OpenFOAM implements many different solvers that are each designed to solve a specific problem in computational continuum mechanics. A transient solver for incompressible, turbulent flow of Newtonian fluids on a moving mesh, called pimpleDyMFoam, is used in the presented example.

5.2.2 ofoam

ofoam is developed by Martin Krosche and more details can be found in (Krosche, 2009). It is a software component based on OpenFOAM and can be used to run the fluid mechanics problems. It can communicate with other software components

through a well defined interface (Niekamp, 2005) defined in CTL. The basic methods for interaction with the execution flow of a solver are defined, like setting or getting the values of the fluid velocity, pressure or force, setting the time step or running the computations, as shown in Figure 5.17.

```
1   /* CI for a CFD simulator */
2
3   #ifndef __CFDSIMU_CI_
4   #define __CFDSIMU_CI_
5
6   #include <ctl.h>
7
8   #define CTL_Class CFDSimuCI
9   #include CTL_ClassBegin
10
11  #define CTL_Constructor1 ( const string /*case control file
    containing required data*/ ), 1
12  /*set the values of velocity, pressure or mesh*/
13  #define  CTL_Method1 void, set, ( const string /*fieldName*/, const
     array<real8> /*field*/ ), 2
14  /*get the values of veocity, pressure, force*/
15  #define  CTL_Method2 void, get, ( const string /*fieldName*/, array
    <real8> /*field*/ ), 2
16  /*run the computations for the current time step*/
17  #define  CTL_Method3 int4, solve, ( const real8 /*timeStep*/ ), 1
18  /*set the timestep to n-1*/
19  #define  CTL_Method4 int4, goback, (), 0
20  /*get coordinated of the mesh nodes*/
21  #define  CTL_Method5 void, getnodes, ( int4 /*dimension*/, array<
    real8> /*nodes*/ ) const, 2
22
23  #include CTL_ClassEnd
24  #endif
25
```

Figure 5.17: ofoam CTL interface definition - *cfdsimu.ci*

The interface is separated from the implementation, and each interface function has to be bound to specific implementation. To bind the interface to function declarations from a header file, CTL code shown in Figure 5.18 is used. It shows how each method from the interface *cfdsimu.ci* is bound to a function name from header file *pimpleDyMFoam.H*.

```
#define CTL_Connect

#include <cfdsimu.ci>
#include <pimpleDyMFoam.H>

struct connectDetailsCI {
CTL_Constructor( 1, ( const std::string& ), 1 );
/*bind function set to interface CTL method 1*/
CTL_Method( 1, void, ofoam::pimpleDyMFoam::set, ( const std::string
&, const std::vector<double>& ), 2);
/*bind function get to interface CTL method 2*/
CTL_Method( 2, void, ofoam::pimpleDyMFoam::get, ( const std::string
&, std::vector<double>& ), 2);
/*bind function solveLoop to interface CTL method 3*/
CTL_Method( 3, int, ofoam::pimpleDyMFoam::solveLoop, ( const double
& ), 1);
/*bind function goBack to interface CTL method 4*/
CTL_Method( 4, int, ofoam::pimpleDyMFoam::goBack, (), 0 );
/*bind function getNodes to interface CTL method 5*/
CTL_Method( 5, void, ofoam::pimpleDyMFoam::getNodes, ( int&, std::
vector<double>& ) const, 2 );
};

void CTL_connect() {
ctl::connect<CFDSimuCI, ofoam::pimpleDyMFoam, connectDetailsCI >();
}

```

Figure 5.18: ofoam binding CTL interface to function implementation - *connectpimpleDyMFoam.cc*

Finally, the functions declared in the interface binding has to be implemented in a C++ code. It has to follow the function declaration (function name, return type and parameters). This code is compiled together with OpenFOAM code so it has access to all internal details, so it can control the execution flow, get and set the values of variables. An example of such a function implementation (*goBack()*) is given in Figure 5.19.

```
int ofoam::pimpleDyMFoam::goBack(){

// pointers required
Time& runTime = runTime_();

// redirect output
std::ofstream Out( outputName_.c_str(), std::ios::app );
OldBuf_ = std::cout.rdbuf(Out.rdbuf());

if( !parallel_ )
Info << "<ofoam::pimpleDyMFoam::goBack> go back from "
<< runTime.timeName() << " to "  << timeOld_ << endl;

runTime.setTime( timeOld_, timeIndexOld_ );

// clean directories for time greater than _timeOld
if ( !parallel_ ){
string cmd = "cleanTimeDir -d \"" + dirCase_ + "\" -T \"" +
runTime.timeName() + "\"";
int ret = std::system( cmd.c_str() );
if( ret == -1 )
{
FatalErrorIn("ofoam::pimpleDyMFoam::goBack")
<< "Error in system call: " << cmd << nl << nl
<< exit(FatalError);
}
}

// go back to old value
imposeSavedMesh();
cumulativeContErr = cumulativeContErrOld_ ;
imposeSavedFields( surfaceScalarFieldGoBack_ );
imposeSavedFields( volScalarFieldGoBack_ );
imposeSavedFields( surfaceVectorFieldGoBack_ );
imposeSavedFields( volVectorFieldGoBack_ );

std::cout.rdbuf(OldBuf_);

return 1;
}

```

Figure 5.19: ofoam function implementation - *pimpleDyMFoam.C*

5.2.3 coFEAP

coFEAP is a software component developed by Martin Hautefeuille and Christophe Kassiotis, and more details can be found in (Kassiotis and Hautefeuille, 2008). It is an interface component based on FEAP (version 8.2), implemented using CTL and it can be used to run structural mechanics problems defined in FEAP. It implements all the basic methods for interaction with FEAP, like getting or setting the residuals, displacements or getting the values of the stiffness matrix.

The simulation interface is implemented in CTL and it can be used by other software components written in different programming languages in order to interact with coFEAP code. This software component has the same concept and a very similar implementation as microFEAP code defined in the previous section. In Figure 5.20, the interface function definitions are shown.

To bind the interface to function implementations, a generic CTL header function for Fortran is used, as shown in Figure 5.21. When the interface is bound in this way, a function name has to follow the specific template *simu_<function name>_impl* as it can be seen in Figure 5.22.

```
#ifndef __SIMU_CI_
#define __SIMU_CI_
#include <ctl.h>

#define CTL_ClassTmpl SimuCI, ( scalar1 ), 1
#include CTL_ClassBegin
# define CTL_Constructor1 (const string), 1
# define CTL_Constructor2 (const string, const array<scalar1>), 2
/*get and set parameters from control file*/
# define CTL_Method2 array<int4>, get_param,(const string) const, 1
# define CTL_Method3 void, set_param, (const array<int4>), 1
/*get and set the value of each selected dofs and nodes*/
# define CTL_Method4 void, get_state, (array<scalar1>) const, 1
# define CTL_Method5 void, set_state, (const array<scalar1>), 1
# define CTL_Method6 void, set_load, (const array<scalar1> ), 1
# define CTL_Method7 void, get_coupling, (array<int4>, array<scalar1
  >) const, 2
# define CTL_Method8 void, set_coupling, (const array<int4>), 1
/*get the residual values*/
# define CTL_Method9 void, get_residual, (array<scalar1>) const, 1
/*run the computations until convergence*/
# define CTL_Method10 int4, solve, (), 0
/*increment the time step*/
# define CTL_Method11 void, time_step, (const scalar1), 1
# define CTL_Method12 void, pre_cond, (const array<scalar1>, array<
  scalar1>) const, 2
# define CTL_Method13 void, dir_derivative, (const array<scalar1>,
  const array<scalar1>, array<scalar1>) const, 3
# define CTL_Method14 void, command, (const string) const, 1
# define CTL_Method15 void, get_nodes, (array<scalar1>, int4) const,
  2
/*get the mass matrix */
# define CTL_Method16 void, get_mass, (array<int4>, array<int4>,
  array<scalar1>) const, 3
/*get the stiffness tangent matrix */
# define CTL_Method17 void, get_stiff, (array<int4>, array<int4>,
  array<scalar1>), 3
# define CTL_Method18 void, set_activity, (const array<int4>), 1
# define CTL_Method19 void, get_activity, (array<int4>) const, 1
# define CTL_Method20 void, get_connectivity, (array<int4>, int4)
  const, 2
# define CTL_Method21 void, reaction, (array<scalar1>) const, 1
/*get Dirichlet and VonNeumann bound conditions*/
# define CTL_Method22 void, get_load, (array<scalar1>) const, 1
# define CTL_Method23 void, plot, (int4) const, 1
/*put a vector r in the residual array */
# define CTL_Method24 void, set_residual, (const array<scalar1>), 1
# define CTL_Method25 void, get_state2, (const string, array<scalar1
  >) const, 2
# define CTL_Method26 void, set_state2, (const string, const array<
  scalar1>), 2
#include CTL_ClassEnd
#endif

```

Figure 5.20: cofeap CTL interface definition - *simu.ci*

```
#define CTL_ConnectF

#define CTL_ClassPrefix simu
typedef double scalar1;

#include <simu.ci>

void CTL_connect()
{ ctl::connectF<SimuCI<double>, ctl::Extern::SimuCI>(); }

```

Figure 5.21: cofeap binding CTL interface to function implementation - *connectcofeap.cc*

```
      subroutine simu_set_residual_impl(s0,x0)

      implicit   none

      include   'iofile.h'
      include   'cdata.h'
      include   'pointer.h'
      include   'comblk.h'
      include   'control.h'


      integer  s0, i
      real*8   x0(s0)

      write(ioc,1000)
      if(s0.ne.neq) then
        write(ioc,2000) s0,neq
        write(*,2000) s0,neq
        call plstop()
      else
        do i =1,neq
          hr(np(26)+i-1) = - x0(i)
        enddo
      endif

1000  format(/3x,'M e t h o d   s e t - r e s i d u a l')

2000  format(6x,'*ERROR* [set-residual]: bad activ DOFs number'/
     .       6x, i10.10, ' > ', i10.10, ' = #activ DOFs'/
     .       6x, 'call PLSTOP')

      end

```

Figure 5.22: cofeap function implementation - *simu_set_residual_impl.f*

5.2.4 cops

The COupling COmponents by a Partitioned Strategy (cops) is a software component developed for the coupling between the domains and exchanging the data. Also, it provides a generic implementation of the explicit and implicit DFMT-BGS algorithms (Direct Force-Motion Transfer/Block-Gauss-Seidel) for fluid–structure interaction (Kassiotis et al., 2011b). The coupling component cops creates the instances of the two solvers - coFeap for the structure and ofoam for the fluid problem, and it stores the logic of execution order.

cops software component is used to communicate with ofoam and cofeap components and execute them in order to construct the final solution. It also implements a CTL interface that is used as an entry point of the simulation, as shown in Figure 5.23. It has only one method *solve* that is used to run the simulation.

The implementation details of each component that it is calling are unknown, it interacts with them through the interface *SimuCI* for coFEAP and *CFDSimuCI* for ofoam. By executing the methods available through each interface, it can construct the final solution.

```
#ifndef _COPS_CI_
#define _COPS_CI_

#include <ctl.h>

#define CTL_Class copsCI
# include CTL_ClassBegin

#define CTL_Constructor1 (const string /*controlFile*/), 1
#define CTL_Constructor1Throws ( std::string ), 1

#define CTL_Method1 int/*1 if success*/, solve, (), 0
#define CTL_Method2Throws ( std::string ), 1

# include CTL_ClassEnd

#endif
```

Figure 5.23: cofeap function implementation - *cops.ci*

To bind the cops interface to a function header, a CTL is used as shown in Figure 5.24.

cops implementation consists of running the ofoam and coFEAP solver iteratively for fluid and mechanical domain until the convergence occurs. In each time step it transfers the information about the displacement field from the mechanical domain to the fluid domain, and the stress field in the opposite direction. The implementation of function *solve* is given in Figure 5.25. It uses a template class *partitionedSolver* that consists of both fluid and solid solvers.

The final outcome of running the cops simulation is a set of *time directories* folders for each time step consisting of values of the physical fields for both domains, like displacement, stress and velocity, and also mesh points coordinates. It can be run and visualised using some post-processing software like paraFoam that is

using ParaView (Ahrens et al., 2005), an open source visualisation application.

```
#define CTL_Connect

#include "cops.h"
#include "cops.ci"

struct connectDetail
{
CTL_Constructor(1, (const std::string&), 1);
CTL_Method(1, int, cops<>::solve, (), 0);
};

void CTL_connect()
{
ctl::connect<copsCI, cops<>, connectDetail>();
}

```

Figure 5.24: cops binding CTL interface to function implementation - *connectcops.cc*

```
template<class VecVecReal, class VecInt>
inline typename cops<VecVecReal, VecInt>::integer
cops<VecVecReal, VecInt>::solve () {
  VecReal nodesF, nodesS;
  integer dim;
  _Solid.get_nodes( nodesS, dim );
  _Fluid.getnodes( dim, nodesF );

  field<VecReal> disp( "disp", nodesS.size(), dim, _caseDir );
  field<VecReal> velo( "velo", nodesS.size(), dim, _caseDir );
  field<VecReal> forc( "forc", nodesS.size(), dim, _caseDir );
  residual<VecReal> res( nodesS.size(), dim, _caseDir );

  if( _verbose > 1 )
    std::cout << "<cops::solve> init partitioned solver" << std::endl
  ;

  partitionedSolver<CFDSimuCI,SimuCI<real>,VecReal> solver (_Fluid,
  _Solid, res, _vm, _caseDir, (_verbose > 1));

  while( _time < _timeEnd ){
    _time += _timeStep;

    if( _verbose == 1 ) std::cout << "Time: " << _time << std::endl;
    else if( _verbose > 1) std::cout << "<cops::solve>: Time: " <<
  _time << std::endl;

    solver( disp, velo, forc, _timeStep );
    if( _verbose > 0 ) std::cout << solver.convergenceInfo() << std::
  endl;
  }

  return 1;
}
```

Figure 5.25: cops implementation - *cops.inl*

5.3 CTL

Component Template Library (CTL) is a C++ template library designed to create distributed component-based software systems. It can be used to create monolithic code by joining different existing codes at link-phase, even if each code is written and compiled in a different programming language (C, C++, Fortran, Java, Python). It supports different linkage types: library, threads, tcp/ip, pipes, mpi (Message Passing Interface), daemons and files. It was developed at the Institute of Scientific Computing of the University of Braunschweig (Niekamp, 2005).

Existing software components are connected through a communication channel and exchange data through a well defined interface. Software code does not have to be initially designed for parallel computing. Interaction between the codes depends only on the interface and not on the internal implementation. Due to all this, it can be used for both multi-physics and multi-scale problems,

coupling codes from the different domains.

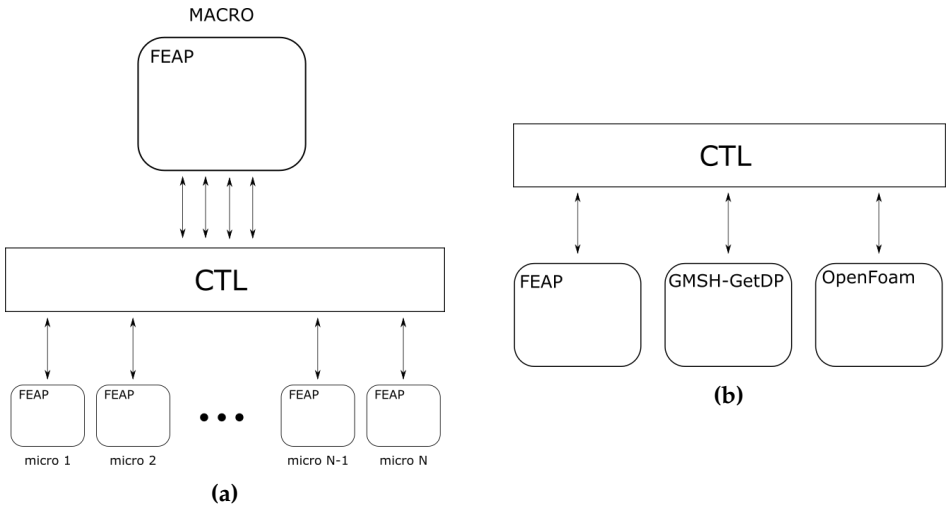


Figure 5.26: (a) CTL and FEAP used for multi-scale problems; (b) CTL with different software codes used for multi-physics problems

CTL allows to select which implementation and which linkage type are used at the run time. When running the application, the binding of the function signature to an implementation must be selected. The dependence graph for the classical monolithic linkage and the component linkage is given in Figure 5.27. The main difference is that in the monolithic linkage, the application depends directly on the used module. In the component linkage, the application depends only on the interface, so it allows flexibility of choosing the appropriate implementation and also allows the modification of the module without the need to compile again the whole code. At run-time the application can decide, which components and how many instances of these components on which hardware is to be used (Niekamp et al., 2009).

A static software component exists both as executable and as dynamic library that has implemented interface classes and functions. A component interface (CI) declares the functionality that a software component may implement. It defines all the necessary information that two components have to know to be able to communicate and exchange information. For example, using CTL, a modified FEAP code can implement a component interface and provide service to any other software component. The functions in this component interface could be: set time step, increment time step, set displacements or get tangent matrix. The interface specification must define which functions can be called, and what are the input and output data.

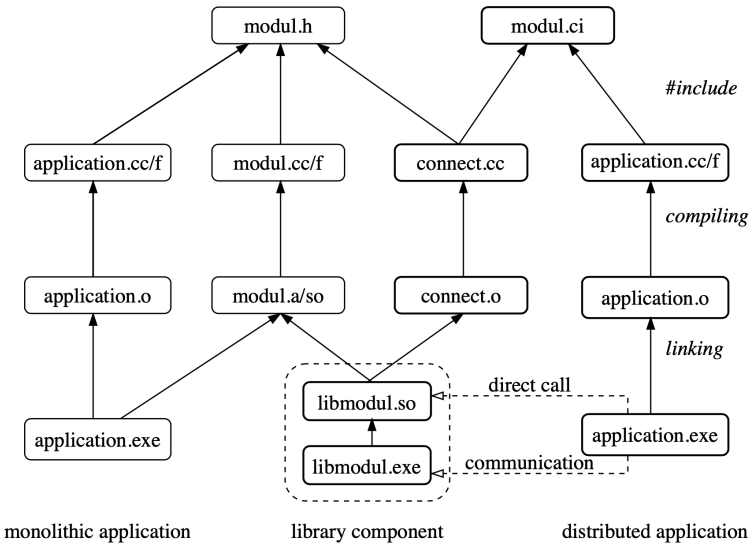


Figure 5.27: Monolithic and component linkage dependence graph (Niekamp et al., 2009)

Conclusion

In this chapter, the software implementation details for the multi-scale and multi-physics coupling have been shown. The multi-scale FEAP software architecture has been described, and together with the main differences compared to the standard FEAP code and execution procedure. Flowcharts were shown for both mono and multi-scale code procedure executions. The code is developed based on the previous work, adapted and rewritten for new versions of FEAP, CTL, and compilers. The original scientific contribution relates to the part of the implementation of the localized failure, both for 2D and 3D settings, as well as the introduction of an additional degree of freedom to be able to capture the micro-scale electro-mechanical coupling. The multi-scale code can be easily upgraded with new functionalities, micro-scale models or multi-physics coupling, to be finally able to simulate more complex behaviors of smart materials. Also, with the CTL interface for FEAP defined, microFEAP can be also used with some other software code that can call it and use the obtained results for its calculations. In the second section of the chapter, a fluid-structure interaction coupling implementation using OpenFOAM and FEAP software code has been presented. Furthermore, the implementation details for each of the components have been described.

6
Conclusion and perspectives

In this chapter, the final conclusion and comments on the previously presented work are stated. The original scientific contributions are underlined, and perspectives for future work and possible improvements are given.

In this thesis, an improved version of a multi-scale coupling procedure has been presented. First, the procedure has been extended with the embedded strong discontinuity on the macro-scale to take into account localized failure. With this addition, the crack opening that happens in the softening phase on the micro-scale can be successfully transfered and represented on the macro-scale. This is the main novelty and contribution of this work. It has been shown, on a simple 2D tension test, that the proposed method produces the same results as the mono-scale method, and shows a significant improvement for the softening phase compared to the previously developed multi-scale method.

Next, the multi-scale method has been extended to a 3D setting, where macro-scale tetrahedral and hexahedral finite elements have been developed. The biggest challenge in applying the multi-scale coupling procedure to a 3D setting was not developing a new theoretical formulation, as it stays mostly the same as for a 2D setting, but implementing it in a software code that needs to be changed to accommodate an additional spatial dimension. This increases the complexity of the implementation, like storing and handling the element arrays data, or detecting the nodes on the interface between the scales. This is also one of the original contributions of this work, as a 3D multi-scale coupling implementation did not exist in this form before. It has been shown that the multi-scale coupling procedure with localized failure with the hexahedral macro-scale element can produce the same results in the softening phase as the mono-scale method for this specific example.

Both in the 2D and the 3D multi-scale formulation, the localized failure is implemented in a way that the crack can only occur in mode I and parallel to the imposed displacement. This has been done to prove that the method can successfully enter into the softening phase and represent the localized failure on the macro-scale. To make the method more generalized, mode I and mode II should be added, so the crack would be described with both the normal and the tangential vector at the discontinuity. In this way, a crack could occur and propagate at a certain angle, and some more complex examples, like the three-point bending test for softening, could be executed.

The method could be additionally improved in several possible ways. One approach is to implement a dissipation based coupling, which would enforce that the sum of the micro-scale elements' dissipations is the same as the macro-scale element dissipation. An adequate way of calculating the dissipation on the micro-scale, which is a function of the displacement discontinuity and other internal variables, should be established. This would imply the introduction of internal variables on the macro-scale, for both hardening and softening phase, and differentiate the dissipation in the fracture process zone and the dissipation along the discontinuity surface. Relevant criteria for entering into the hardening and the softening phase on the macro-scale should also be defined.

Furthermore, in the scope of this thesis the multi-scale method has been successfully extended to a multi-physics setting, which allows to capture the electro-mechanical coupling behavior on the micro-scale. An additional degree of freedom for electric potential is added, accompanied by the respective changes in the multi-scale formulation. Numerical computations of an example with a micro-scale polarization model are performed, and the results obtained are shown

to correspond to the mono-scale computation results. Also, Hellinger-Reissner formulation for electrostatics has been applied in the multi-scale setting. In this formulation, two degrees of freedom are used to couple the scales on the interface – scalar electric potential and electric displacement. This is the first step in the application of the multi-scale computation procedure to describe the behavior of smart materials. In the future, additional degrees of freedom could be added and different fields introduced, such as temperature or magnetic field. Also, some more complex multi-physics coupling models can be developed based on the proposed approach.

Through the implementation of the multi-scale and multi-physics computation procedures listed above, a general software framework that is modular and able to handle different numerical models has been developed. This means that a new micro-scale element describing a different material behavior can be executed with this multi-scale code with little or no modifications at all. The only constraint is that the element has to be implemented in FEAP, allowing the macro-scale element to capture the behavior through the imposed interface conditions and obtained micro-scale element arrays.

The main advantage of the developed computation procedure and accompanying software code is that it can give a reliable representation of the behavior that is defined on the micro-scale, without having to go into its implementation details. Several different micro-scale behavior models can be used at the same time, and they can be defined for different parts of the heterogeneous macro-scale specimen. The trade-off is the higher execution time, as it takes additional steps for the scales to communicate, and to prepare and exchange data.

To summarize, some of the future work and possible improvements are listed here, grouped by the estimated time needed for its realization. A short term goal, that could be done in several months and would require a larger number of processors like in a computer cluster, is

- Perform the multi-scale code execution analysis, based on the number of distributed processors, mesh size, micro-scale processes, and compare it to the equivalent mono-scale code.

Medium term goals, that could be done as a part of a post-doctoral research requiring several months to one year for each goal, are:

- Develop a formulation to implement mode I and mode II crack opening, and random orientation of the discontinuity in the 2D multi-scale computation procedure;

- Develop a formulation to implement mode I, mode II and mode III crack opening, and random orientation of the discontinuity in the 3D multi-scale computation procedure;

- Implement and test the proposed multi-scale formulation with random orientation of the discontinuity on several examples, including a three-point bending test for the softening phase;

- Develop a multi-physics formulation with included temperature and magnetic field to be able to capture thermo-electro-magneto-elastoplastic coupling behavior on the micro-scale.

Long term goals, that could be a part of another doctoral thesis and the continuation of this work, are:

- Explore the possibilities of adding additional physics in a multi-physics coupling formulation (like fluids for fluid-structure interaction) to be able to model the behavior of smart materials for specific applications, like an aircraft wing with piezoelectric sensors and actuators;
- Explore the possibilities of coupling another software code to allow for additional multi-scale or multi-physics coupling.

Bibliography

▲ Ahrens, J., Geveci, B., and Law, C. (2005). Paraview: An end-user tool for large data visualization. *The visualization handbook*, 717.

▲ Anciaux, G., Coulaud, O., and Roman, J. (2006). High performance multiscale simulation or crack propagation. In *2006 International Conference on Parallel Processing Workshops (ICPPW'06)*, pages 8–pp. IEEE.

▲ Armstrong, P. J. and Frederick, C. (1966). *A mathematical representation of the multiaxial Bauschinger effect*, volume 731. Central Electricity Generating Board [and] Berkeley Nuclear Laboratories, Research & Development Department.

▲ Belgacem, F. B. (1999). The mortar finite element method with Lagrange multipliers. *Numerische Mathematik*, 84(2):173–197.

▲ Belgacem, F. B., Hild, P., and Laborde, P. (1998). The mortar finite element method for contact problems. *Mathematical and Computer Modelling*, 28(4-8):263–271.

▲ Belytschko, T., Loehnert, S., and Song, J.-H. (2008). Multiscale aggregating discontinuities: a method for circumventing loss of material stability. *International Journal for Numerical Methods in Engineering*, 73(6):869–894.

▲ Bengisu, M. and Ferrara, M. (2018). *Materials that Move: Smart Materials, Intelligent Design*. Springer.

▲ Bernholdt, D. E., Allan, B. A., Armstrong, R., Bertrand, F., Chiu, K., Dahlgren, T. L., Damevski, K., Elwasif, W. R., Epperly, T. G., Govindaraju, M., et al. (2006). A component architecture for high-performance scientific computing. *The International Journal of High Performance Computing Applications*, 20(2):163–202.

▲ Blanco, P. J., Sánchez, P. J., de Souza Neto, E. A., and Feijóo, R. A. (2016). Variational foundations and generalized unified theory of RVE-based multiscale models. *Archives of Computational Methods in Engineering*, 23(2):191–253.

▲ Borgdorff, J., Mamonski, M., Bosak, B., Kurowski, K., Belgacem, M. B., Chopard, B., Groen, D., Coveney, P. V., and Hoekstra, A. G. (2014). Distributed multiscale computing with MUSCLE 2, the multiscale coupling library and environment. *Journal of Computational Science*, 5(5):719–731.

▲ Brancherie, D. (2003). *Modéles continus et discrets pour les problémes de localisation et de rupture fragile et/ou ductile*. PhD thesis, École normale supérieure de Cachan-ENS Cachan.

▲ Brancherie, D. and Ibrahimbegovic, A. (2009). Novel anisotropic continuum-discrete damage model capable of representing localized failure of massive structures: Part I: theoretical formulation and numerical implementation. *Engineering Computations*, 26(1/2):100–127.

▲ Buhmann, M. D. (2003). *Radial basis functions: theory and implementations*, volume 12. Cambridge university press.

▲ Crosetto, P., Reymond, P., Deparis, S., Kontaxakis, D., Stergiopulos, N., and Quarteroni, A. (2011). Fluid–structure interaction simulation of aortic blood flow. *Computers & Fluids*, 43(1):46–57.

▲ De Souza Neto, E. A., Blanco, P. J., Sánchez, P. J., and Feijóo, R. A. (2015). An RVE-based multiscale theory of solids with micro-scale inertia and body force effects. *Mechanics of Materials*, 80:136–144.

▲ Dular, P. and Geuzaine, C. (2013). GetDP reference manual: the documentation for GetDP, a general environment for the treatment of discrete problems. *University of Liège*.

▲ Ferrari, A. and Mittica, A. (2013). Thermodynamic formulation of the constitutive equations for solids and fluids. *Energy conversion and management*, 66:77–86.

▲ Feyel, F. (2003). A multilevel finite element method (FE2) to describe the response of highly non-linear structures using generalized continua. *Computer Methods in applied Mechanics and engineering*, 192(28-30):3233–3244.

▲ Feyel, F. and Chaboche, J.-L. (2000). FE2 multiscale approach for modelling the elastoviscoplastic behaviour of long fibre SiC/Ti composite materials. *Computer methods in applied mechanics and engineering*, 183(3-4):309–330.

▲ Fish, J. and Yu, Q. (2001). Multiscale damage modelling for composite materials: theory and computational framework. *International Journal for Numerical Methods in Engineering*, 52(1-2):161–191.

▲ Fu, P., Liu, H., and Chu, X. (2017). An efficient multiscale computational formulation for geometric nonlinear analysis of heterogeneous piezoelectric composite. *Composite Structures*, 167:191–206.

▲ Gabriel, E., Fagg, G. E., Bosilca, G., Angskun, T., Dongarra, J. J., Squyres, J. M., Sahay, V., Kambadur, P., Barrett, B., Lumsdaine, A., et al. (2004). Open MPI: Goals, concept, and design of a next generation MPI implementation. In *European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting*, pages 97–104. Springer.

▲ Geers, M. G., Kouznetsova, V. G., and Brekelmans, W. (2010). Multi-scale computational homogenization: Trends and challenges. *Journal of computational and applied mathematics*, 234(7):2175–2182.

▲ Ghosh, S., Lee, K., and Raghavan, P. (2001). A multi-level computational model for multi-scale damage analysis in composite and porous materials. *International Journal of Solids and Structures*, 38(14):2335–2385.

▲ Gitman, I., Askes, H., and Sluys, L. (2008). Coupled-volume multi-scale modelling of quasi-brittle material. *European Journal of Mechanics-A/Solids*, 27(3):302–327.

▲ Gloria, A. (2012). Numerical homogenization: survey, new results, and perspectives. In *ESAIM: Proceedings*, volume 37, pages 50–116. EDP Sciences.

▲ Groen, D., Zasada, S. J., and Coveney, P. V. (2013). Survey of multiscale and multiphysics applications and communities. *Computing in Science & Engineering*, 16(2):34–43.

- Gropp, W. and Lusk, E. (1996). User's guide for mpich, a portable implementation of MPI.

- Gross, L., Bourgouin, L., Hale, A. J., and Mühlhaus, H.-B. (2007). Interface modeling in incompressible media using level sets in Escript. *Physics of the Earth and Planetary Interiors*, 163(1-4):23–34.

- Guidault, P.-A., Allix, O., Champaney, L., and Navarro, J.-P. (2007). A two-scale approach with homogenization for the computation of cracked structures. *Computers & structures*, 85(17-18):1360–1371.

- Guo, N. and Zhao, J. (2014). A coupled FEM/DEM approach for hierarchical multiscale modelling of granular media. *International Journal for Numerical Methods in Engineering*, 99(11):789–818.

- Hadzalic, E. (2019). *Analysis of pore pressure influence on failure mechanisms in structural systems*. PhD thesis, Université de Technologie de Compiègne / University of Sarajevo, Faculty of Civil Engineering.

- Hansbo, A. and Hansbo, P. (2004). A finite element method for the simulation of strong and weak discontinuities in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 193(33):3523–3540.

- Hautefeuille, M., Colliat, J.-B., Ibrahimbegovic, A., Matthies, H., and Villon, P. (2012). A multi-scale approach to model localized failure with softening. *Computers & Structures*, 94:83–95.

- Hwang, S., Lynch, C., and McMeeking, R. (1995). Ferroelectric/ferroelastic interactions and a polarization switching model. *Acta metallurgica et materialia*, 43(5):2073–2084.

- Ibrahimbegovic, A. (2009). *Nonlinear solid mechanics: theoretical formulations and finite element solution methods*, volume 160. Springer Science & Business Media.

- Ibrahimbegovic, A. and Markovic, D. (2003). Strong coupling methods in multi-phase and multi-scale modeling of inelastic behavior of heterogeneous structures. *Computer Methods in Applied Mechanics and Engineering*, 192(28-30):3089–3107.

- Ibrahimbegovic, A., Niekamp, R., Kassiotis, C., Markovic, D., and Matthies, H. G. (2014). Code-coupling strategy for efficient development of computer software in multiscale and multiphysics nonlinear evolution problems in computational mechanics. *Advances in Engineering Software*, 72:8–17.

- Ibrahimbegovic, A. and Wilson, E. (1991). A modified method of incompatible modes. *Communications in applied numerical methods*, 7(3):187–194.

- Jasak, H., Jemcov, A., Tukovic, Z., et al. (2007). OpenFOAM: A C++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, volume 1000, pages 1–20. IUC Dubrovnik Croatia.

- Kamlah, M. and Tsakmakis, C. (1999). Phenomenological modeling of the non-linear electro-mechanical coupling in ferroelectrics. *International journal of solids and structures*, 36(5):669–695.

- Karavelic, E., Nikolic, M., Ibrahimbegovic, A., and Kurtovic, A. (2017). Concrete meso-scale model with full set of 3D failure modes with random distribution of aggregate and cement phase. Part I: Formulation and numerical implementation. *Computer Methods in Applied Mechanics and Engineering*.

▲ Kassiotis, C. and Hautefeuille, M. (2008). cofeap's manual. *LMT-Cachan internal report*, 2.

▲ Kassiotis, C., Ibrahimbegovic, A., Niekamp, R., and Matthies, H. G. (2011a). Nonlinear fluid–structure interaction problem. Part I: implicit partitioned algorithm, nonlinear stability proof and validation examples. *Computational Mechanics*, 47(3):305–323.

▲ Kassiotis, C., Ibrahimbegovic, A., Niekamp, R., and Matthies, H. G. (2011b). Nonlinear fluid–structure interaction problem. Part II: space discretization, implementation aspects, nested parallelization and application examples. *Computational Mechanics*, 47(3):335–357.

▲ Keip, M.-A. and Schröder, J. (2011). A ferroelectric and ferroelastic microscopic switching criterion for tetragonal ferroelectrics. *PAMM*, 11(1):475–476.

▲ Kirk, B. S. and Peterson, J. (2003). libMesh-a C++ Finite Element Library. *CFDLab*.

▲ Klöppel, T., Popp, A., Küttler, U., and Wall, W. A. (2011). Fluid–structure interaction for non-conforming interfaces based on a dual mortar formulation. *Computer Methods in Applied Mechanics and Engineering*, 200(45-46):3111–3126.

▲ Kouznetsova, V., Brekelmans, W., and Baaijens, F. (2001). An approach to micro-macro modeling of heterogeneous materials. *Computational Mechanics*, 27(1):37–48.

▲ Kouznetsova, V. G. (2004). Computational homogenization for the multi-scale analysis of multi-phase materials.

▲ Kozar, I., Rukavina, T., and Ibrahimbegovic, A. (2018). Method of incompatible modes–overview and application. *Građevinar*, 70(1):19–29.

▲ Krosche, M. (2009). Ofoam's manual. informatikbericht. *Institute for Scientific Computing, Braunschweig, Germany (in preparation)*.

▲ Kucerova, A., Brancherie, D., Ibrahimbegovic, A., Zeman, J., and Bittnar, Z. (2009). Novel anisotropic continuum-discrete damage model capable of representing localized failure of massive structures: Part II: identification from tests under heterogeneous stress field. *Engineering Computations*, 26(1/2):128–144.

▲ Lacroix, S., Vassilevski, Y. V., and Wheeler, M. F. (2001). Decoupling preconditioners in the implicit parallel accurate reservoir simulator (PARS). *Numerical linear algebra with applications*, 8(8):537–549.

▲ Ladevèze, P., Loiseau, O., and Dureisseix, D. (2001). A micro–macro and parallel computational strategy for highly heterogeneous structures. *International Journal for Numerical Methods in Engineering*, 52(1-2):121–138.

▲ Ladeveze, P. and Nouy, A. (2003). On a multiscale computational strategy with time and space homogenization for structural mechanics. *Computer Methods in Applied Mechanics and Engineering*, 192(28-30):3061–3087.

▲ Larson, J., Jacob, R., and Ong, E. (2005). The model coupling toolkit: a new fortran90 toolkit for building multiphysics parallel coupled models. *The International Journal of High Performance Computing Applications*, 19(3):277–292.

▲ Loehnert, S. and Belytschko, T. (2007). A multiscale projection method for macro/microcrack simulations. *International Journal for Numerical Methods in Engineering*, 71(12):1466–1482.

▲ Lombardi, M., Parolini, N., and Quarteroni, A. (2013). Radial basis functions for inter-grid interpolation and mesh motion in FSI problems. *Computer Methods in Applied Mechanics and Engineering*, 256:117–131.

▲ Lv, J., Yang, K., Zhang, H., Yang, D., and Huang, Y. (2014). A hierarchical multiscale approach for predicting thermo-electro-mechanical behavior of heterogeneous piezoelectric smart materials. *Computational Materials Science*, 87:88–99.

▲ Markovic, D. (2004). *Modélisation multi-échelles de structures hétérogènes aux comportements anélastiques non-linéaires*. PhD thesis.

▲ Markovic, D. and Ibrahimbegovic, A. (2004). On micro–macro interface conditions for micro scale based FEM for inelastic behavior of heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, 193(48-51):5503–5523.

▲ Markovic, D., Niekamp, R., Ibrahimbegovic, A., Matthies, H. G., and Taylor, R. L. (2005). Multi-scale modeling of heterogeneous structures with inelastic constitutive behaviour: Part I–physical and mathematical aspects. *Engineering Computations*, 22(5/6):664–683.

▲ Matthies, H. G., Niekamp, R., and Steindorf, J. (2006). Algorithms for strong coupling procedures. *Computer methods in applied mechanics and engineering*, 195(17-18):2028–2049.

▲ Mei, C. C. and Vernescu, B. (2010). *Homogenization methods for multiscale mechanics*. World scientific.

▲ Morel, T., Duchaine, F., Thévenin, A., Piacentini, A., Kirmse, M., and Quémerais, E. (2019). Coupleur OpenPALM version 4.3. 0 manuel utilisateur et de formation.

▲ Moreno-Navarro, P. (2019). *Multiphysics formulation and multiscale finite element discretizations of thermo-electro-magneto-mechanic coupling for smart materials design*. PhD thesis, Université de Technologie de Compiègne.

▲ Moreno-Navarro, P., Ibrahimbegovic, A., and Ospina, A. (2020). Multi-field variational formulations and mixed finite element approximations for electrostatics and magnetostatics. *Computational Mechanics*, 65(1):41–59.

▲ Moreno-Navarro, P., Ibrahimbegovic, A., and Pérez-Aparicio, J. (2018). Linear elastic mechanical system interacting with coupled thermo-electro-magnetic fields. *Coupled systems mechanics*, 7:5–25.

▲ Niekamp, D. R. (2005). CTL Manual for Linux. *Unix for the Usage with C+*.

▲ Niekamp, R., Ibrahimbegovic, A., and Matthies, H. (2014). Formulation, solution and CTL software for coupled thermomechanics systems. *Coupled systems mechanics*, 3(1):1–25.

▲ Niekamp, R., Markovic, D., Ibrahimbegovic, A., Matthies, H. G., and Taylor, R. L. (2009). Multi-scale modelling of heterogeneous structures with inelastic constitutive behavior: Part II–software coupling implementation aspects. *Engineering computations*, 26(1/2):6–28.

▲ Nikolic, M. (2015). *Rock Mechanics, Failure Phenomena with Pre-Existing Cracks and Internal Fluid Flow through Cracks: Doctoral Dissertation*. PhD thesis, University of Split. Faculty of Civil Engineering, Architecture and Geodesy. Department of Geotechnical Engineering.

▲ Nikolic, M. and Ibrahimbegovic, A. (2015). Rock mechanics model capable of representing initial heterogeneities and full set of 3D failure mechanisms. *Computer Methods in Applied Mechanics and Engineering*, 290:209–227.

▲ Özdemir, I., Brekelmans, W., and Geers, M. G. (2008). FE2 computational homogenization for the thermo-mechanical analysis of heterogeneous solids. *Computer Methods in Applied Mechanics and Engineering*, 198(3-4):602–613.

▲ Park, K., Felippa, C., and Rebel, G. (2002). A simple algorithm for localized construction of non-matching structural interfaces. *International Journal for Numerical Methods in Engineering*, 53(9):2117–2142.

▲ Peszynska, M., Lu, Q., and Wheeler, M. F. (2000). Multiphysics coupling of codes. In *Computational Methods in Water Resources*. Citeseer.

▲ Plimpton, S., Thompson, A., Crozier, P., and Kohlmeyer, A. (2011). LAMMPS molecular dynamics simulator.

▲ Raghavan, P. and Ghosh, S. (2004). Adaptive multi-scale computational modeling of composite materials. *Computer Modeling in Engineering and Sciences*, 5(2):151–170.

▲ Ramirez, F., Heyliger, P. R., and Pan, E. (2006). Free vibration response of two-dimensional magneto-electro-elastic laminated plates. *Journal of Sound and Vibration*, 292(3-5):626–644.

▲ Rochus, V., Rixen, D. J., and Golinval, J.-C. (2006). Monolithic modelling of electro-mechanical coupling in micro-structures. *International journal for numerical methods in engineering*, 65(4):461–493.

▲ Rukavina, I. and Ibrahimbegovic, A. (2017). Enhanced interface and operator-split software-coupling based approach for fluid-structure interaction. In *ECCOMAS MSF 2017: 3rd International Conference on Multiscale Computational Methods for Solids and Fluids*.

▲ Rukavina, I., Ibrahimbegovic, A., Do, X. N., and Markovic, D. (2019). ED-FEM multi-scale computation procedure for localized failure. *Coupled systems mechanics*, 8(2):111–127.

▲ Rukavina, I., Ibrahimbegovic, A., Moreno-Navarro, P., and Ospina Vargas, L. A. (2017). Comparison of alternative discretization and solution procedures for multiphysics problems. In *ECCOMAS MSF 2017: 3rd International Conference on Multiscale Computational Methods for Solids and Fluids*.

▲ Rukavina, T. (2018). *Multi-scale damage model of fiber-reinforced concrete with parameter identification*. PhD thesis, Université de Technologie de Compiègne / University of Rijeka, Faculty of Civil Engineering.

▲ Said, S., Sabri, M., and Salleh, F. (2017). Ferroelectrics and their applications.

▲ Schwartz, M. (2002). *Encyclopedia of smart materials*. John Wiley & Sons.

▲ Sengupta, A., Papadopoulos, P., and Taylor, R. L. (2012). A multiscale finite element method for modeling fully coupled thermomechanical problems in solids. *International Journal for Numerical Methods in Engineering*, 91(13):1386–1405.

▲ Simo, J. and Taylor, R. (1985). Consistent tangent operators for rate-independent elastoplasticity. *Computer Methods in Applied Mechanics and Engineering*, 48:101–118.

- Šmilauer, V., Catalano, E., Chareyre, B., Dorofeenko, S., Duriez, J., Gladky, A., Kozicki, J., Modenese, C., Scholtès, L., Sibille, L., et al. (2010). Yade reference documentation. *Yade Documentation*, 474(1).
- Sommerville, I. (2011). Software engineering 9th edition. *Addison-Wesley*.
- Taylor, R. L. (2014). FEAP - Finite Element Analysis Program, manual.
- Zienkiewicz, O. C., Taylor, R. L., and Zhu, J. Z. (2005). *The finite element method: its basis and fundamentals*. Elsevier.

List of Figures